Apostolos Antonacopoulos ·
Subhasis Chaudhuri · Rama Chellappa ·
Cheng-Lin Liu · Saumik Bhattacharya ·
Umapada Pal (Eds.)

LNCS 15307

Pattern Recognition

27th International Conference, ICPR 2024
Kolkata, India, December 1–5, 2024
Proceedings, Part VII

Springer

MOREMEDIA

Lecture Notes in Computer Science **15307**

The series Lecture Notes in Computer Science (LNCS), including its subseries Lecture Notes in Artificial Intelligence (LNAI) and Lecture Notes in Bioinformatics (LNBI), has established itself as a medium for the publication of new developments in computer science and information technology research, teaching, and education.

LNCS enjoys close cooperation with the computer science R & D community, the series counts many renowned academics among its volume editors and paper authors, and collaborates with prestigious societies. Its mission is to serve this international community by providing an invaluable service, mainly focused on the publication of conference and workshop proceedings and postproceedings. LNCS commenced publication in 1973.

Apostolos Antonacopoulos ·
Subhasis Chaudhuri · Rama Chellappa ·
Cheng-Lin Liu · Saumik Bhattacharya ·
Umapada Pal
Editors

Pattern Recognition

27th International Conference, ICPR 2024
Kolkata, India, December 1–5, 2024
Proceedings, Part VII

Springer

Editors
Apostolos Antonacopoulos
University of Salford
Salford, Lancashire, UK

Subhasis Chaudhuri
Indian Institute of Technology Bombay
Mumbai, Maharashtra, India

Rama Chellappa
Johns Hopkins University
Baltimore, MD, USA

Cheng-Lin Liu
Chinese Academy of Sciences
Beijing, China

Saumik Bhattacharya
IIT Kharagpur
Kharagpur, West Bengal, India

Umapada Pal
Indian Statistical Institute Kolkata
Kolkata, West Bengal, India

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-031-78182-7 ISBN 978-3-031-78183-4 (eBook)
https://doi.org/10.1007/978-3-031-78183-4

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

President's Address

On behalf of the Executive Committee of the International Association for Pattern Recognition (IAPR), I am pleased to welcome you to the 27th International Conference on Pattern Recognition (ICPR 2024), the main scientific event of the IAPR.

After a completely digital ICPR in the middle of the COVID pandemic and the first hybrid version in 2022, we can now enjoy a fully back-to-normal ICPR this year. I look forward to hearing inspirational talks and keynotes, catching up with colleagues during the breaks and making new contacts in an informal way. At the same time, the conference landscape has changed. Hybrid meetings have made their entrance and will continue. It is exciting to experience how this will influence the conference. Planning for a major event like ICPR must take place over a period of several years. This means many decisions had to be made under a cloud of uncertainty, adding to the already large effort needed to produce a successful conference. It is with enormous gratitude, then, that we must thank the team of organizers for their hard work, flexibility, and creativity in organizing this ICPR. ICPR always provides a wonderful opportunity for the community to gather together. I can think of no better location than Kolkata to renew the bonds of our international research community.

Each ICPR is a bit different owing to the vision of its organizing committee. For 2024, the conference has six different tracks reflecting major themes in pattern recognition: Artificial Intelligence, Pattern Recognition and Machine Learning; Computer and Robot Vision; Image, Speech, Signal and Video Processing; Biometrics and Human Computer Interaction; Document Analysis and Recognition; and Biomedical Imaging and Bioinformatics. This reflects the richness of our field. ICPR 2024 also features two dozen workshops, seven tutorials, and 15 competitions; there is something for everyone. Many thanks to those who are leading these activities, which together add significant value to attending ICPR, whether in person or virtually. Because it is important for ICPR to be as accessible as possible to colleagues from all around the world, we are pleased that the IAPR, working with the ICPR organizers, is continuing our practice of awarding travel stipends to a number of early-career authors who demonstrate financial need. Last but not least, we are thankful to the Springer LNCS team for their effort to publish these proceedings.

Among the presentations from distinguished keynote speakers, we are looking forward to the three IAPR Prize Lectures at ICPR 2024. This year we honor the achievements of Tin Kam Ho (IBM Research) with the IAPR's most prestigious King-Sun Fu Prize "for pioneering contributions to multi-classifier systems, random decision forests, and data complexity analysis". The King-Sun Fu Prize is given in recognition of an outstanding technical contribution to the field of pattern recognition. It honors the memory of Professor King-Sun Fu who was instrumental in the founding of IAPR, served as its first president, and is widely recognized for his extensive contributions to the field of pattern recognition.

The Maria Petrou Prize is given to a living female scientist/engineer who has made substantial contributions to the field of Pattern Recognition and whose past contributions, current research activity and future potential may be regarded as a model to both aspiring and established researchers. It honours the memory of Professor Maria Petrou as a scientist of the first rank, and particularly her role as a pioneer for women researchers. This year, the Maria Petrou Prize is given to Guoying Zhao (University of Oulu), "for contributions to video analysis for facial micro-behavior recognition and remote biosignal reading (RPPG) for heart rate analysis and face anti-spoofing".

The J.K. Aggarwal Prize is given to a young scientist who has brought a substantial contribution to a field that is relevant to the IAPR community and whose research work has had a major impact on the field. Professor Aggarwal is widely recognized for his extensive contributions to the field of pattern recognition and for his participation in IAPR's activities. This year, the J.K. Aggarwal Prize goes to Xiaolong Wang (UC San Diego) "for groundbreaking contributions to advancing visual representation learning, utilizing self-supervised and attention-based models to establish fundamental frameworks for creating versatile, general-purpose pattern recognition systems".

During the conference we will also recognize 21 new IAPR Fellows selected from a field of very strong candidates. In addition, a number of Best Scientific Paper and Best Student Paper awards will be presented, along with the Best Industry Related Paper Award and the Piero Zamperoni Best Student Paper Award. Congratulations to the recipients of these very well-deserved awards!

I would like to close by again thanking everyone involved in making ICPR 2024 a tremendous success; your hard work is deeply appreciated. These thanks extend to all who chaired the various aspects of the conference and the associated workshops, my ExCo colleagues, and the IAPR Standing and Technical Committees. Linda O'Gorman, the IAPR Secretariat, deserves special recognition for her experience, historical perspective, and attention to detail when it comes to supporting many of the IAPR's most important activities. Her tasks became so numerous that she recently got support from Carolyn Buckley (layout, newsletter), Ugur Halici (ICPR matters), and Rosemary Stramka (secretariat). The IAPR website got a completely new design. Ed Sobczak has taken care of our web presence for so many years already. A big thank you to all of you!

This is, of course, the 27th ICPR conference. Knowing that ICPR is organized every two years, and that the first conference in the series (1973!) pre-dated the formal founding of the IAPR by a few years, it is also exciting to consider that we are celebrating over 50 years of ICPR and at the same time approaching the official IAPR 50th anniversary in 2028: you'll get all information you need at ICPR 2024. In the meantime, I offer my thanks and my best wishes to all who are involved in supporting the IAPR throughout the world.

September 2024

Arjan Kuijper
President of the IAPR

Preface

It is our great pleasure to welcome you to the proceedings of the 27th International Conference on Pattern Recognition (ICPR 2024), held in Kolkata, India. The city, formerly known as 'Calcutta', is the home of the fabled Indian Statistical Institute (ISI), which has been at the forefront of statistical pattern recognition for almost a century. Concepts like the Mahalanobis distance, Bhattacharyya bound, Cramer–Rao bound, and Fisher–Rao metric were invented by pioneers associated with ISI. The first ICPR (called IJCPR then) was held in 1973, and the second in 1974. Subsequently, ICPR has been held every other year. The International Association for Pattern Recognition (IAPR) was founded in 1978 and became the sponsor of the ICPR series. Over the past 50 years, ICPR has attracted huge numbers of scientists, engineers and students from all over the world and contributed to advancing research, development and applications in pattern recognition technology.

ICPR 2024 was held at the Biswa Bangla Convention Centre, one of the largest such facilities in South Asia, situated just 7 kilometers from Kolkata Airport (CCU). According to ChatGPT "Kolkata is often called the 'Cultural Capital of India'. The city has a deep connection to literature, music, theater, and art. It was home to Nobel laureate Rabindranath Tagore, and the Bengali film industry has produced globally renowned filmmakers like Satyajit Ray. The city boasts remarkable colonial architecture, with landmarks like Victoria Memorial, Howrah Bridge, and the Indian Museum (the oldest and largest museum in India). Kolkata's streets are dotted with old mansions and buildings that tell stories of its colonial past. Walking through the city can feel like stepping back into a different era. Finally, Kolkata is also known for its street food."

ICPR 2024 followed a two-round paper submission format. We received a total of 2135 papers (1501 papers in round-1 submissions, and 634 papers in round-2 submissions). Each paper, on average, received 2.84 reviews, in single-blind mode. For the first-round papers we had a rebuttal option available to authors.

In total, 945 papers (669 from round-1 and 276 from round-2) were accepted for presentation, resulting in an acceptance rate of 44.26%, which is consistent with previous ICPR events. At ICPR 2024 the papers were categorized into six tracks: Artificial Intelligence, Machine Learning for Pattern Analysis; Computer Vision and Robotic Perception; Image, Video, Speech, and Signal Analysis; Biometrics and Human-Machine Interaction; Document and Media Analysis; and Biomedical Image Analysis and Informatics.

The main conference ran over December 2–5, 2024. The main program included the presentation of 188 oral papers (19.89% of the accepted papers), 757 poster papers and 12 competition papers (out of 15 submitted). A total 10 oral sessions were held concurrently in four meeting rooms with a total of 40 oral sessions. In total 24 workshops and 7 tutorials were held on December 1, 2024.

The plenary sessions included three prize lectures and three invited presentations. The prize lectures were delivered by Tin Kam Ho (IBM Research, USA; King Sun

Fu Prize winner), Xiaolong Wang (University of California, San Diego, USA; J.K. Aggarwal Prize winner), and Guoying Zhao (University of Oulu, Finland; Maria Petrou Prize winner). The invited speakers were Timothy Hospedales (University of Edinburgh, UK), Venu Govindaraju (University at Buffalo, USA), and Shuicheng Yan (Skywork AI, Singapore).

Several best paper awards were presented in ICPR: the Piero Zamperoni Award for the best paper authored by a student, the BIRPA Best Industry Related Paper Award, and the Best Paper Awards and Best Student Paper Awards for each of the six tracks of ICPR 2024.

The organization of such a large conference would not be possible without the help of many volunteers. Our special gratitude goes to the Program Chairs (Apostolos Antonacopoulos, Subhasis Chaudhuri, Rama Chellappa and Cheng-Lin Liu), for their leadership in organizing the program. Thanks to our Publication Chairs (Ananda S. Chowdhury and Wataru Ohyama) for handling the overwhelming workload of publishing the conference proceedings. We also thank our Competition Chairs (Richard Zanibbi, Lianwen Jin and Laurence Likforman-Sulem) for arranging 12 important competitions as part of ICPR 2024. We are thankful to our Workshop Chairs (P. Shivakumara, Stephanie Schuckers, Jean-Marc Ogier and Prabir Bhattacharya) and Tutorial Chairs (B.B. Chaudhuri, Michael R. Jenkin and Guoying Zhao) for arranging the workshops and tutorials on emerging topics. ICPR 2024, for the first time, held a Doctoral Consortium. We would like to thank our Doctoral Consortium Chairs (Véronique Eglin, Dan Lopresti and Mayank Vatsa) for organizing it.

Thanks go to the Track Chairs and the meta reviewers who devoted significant time to the review process and preparation of the program. We also sincerely thank the reviewers who provided valuable feedback to the authors.

Finally, we acknowledge the work of other conference committee members, like the Organizing Chairs and Organizing Committee Members, Finance Chairs, Award Chair, Sponsorship Chairs, and Exhibition and Demonstration Chairs, Visa Chair, Publicity Chairs, and Women in ICPR Chairs, whose efforts made this event successful. We also thank our event manager Alpcord Network for their help.

We hope that all the participants found the technical program informative and enjoyed the sights, culture and cuisine of Kolkata.

October 2024

Umapada Pal
Josef Kittler
Anil Jain

Organization

General Chairs

Umapada Pal	Indian Statistical Institute, Kolkata, India
Josef Kittler	University of Surrey, UK
Anil Jain	Michigan State University, USA

Program Chairs

Apostolos Antonacopoulos	University of Salford, UK
Subhasis Chaudhuri	Indian Institute of Technology, Bombay, India
Rama Chellappa	Johns Hopkins University, USA
Cheng-Lin Liu	Institute of Automation, Chinese Academy of Sciences, China

Publication Chairs

Ananda S. Chowdhury	Jadavpur University, India
Wataru Ohyama	Tokyo Denki University, Japan

Competition Chairs

Richard Zanibbi	Rochester Institute of Technology, USA
Lianwen Jin	South China University of Technology, China
Laurence Likforman-Sulem	Télécom Paris, France

Workshop Chairs

P. Shivakumara	University of Salford, UK
Stephanie Schuckers	Clarkson University, USA
Jean-Marc Ogier	Université de la Rochelle, France
Prabir Bhattacharya	Concordia University, Canada

Tutorial Chairs

B. B. Chaudhuri	Indian Statistical Institute, Kolkata, India
Michael R. Jenkin	York University, Canada
Guoying Zhao	University of Oulu, Finland

Doctoral Consortium Chairs

Véronique Eglin	CNRS, France
Daniel P. Lopresti	Lehigh University, USA
Mayank Vatsa	Indian Institute of Technology, Jodhpur, India

Organizing Chairs

Saumik Bhattacharya	Indian Institute of Technology, Kharagpur, India
Palash Ghosal	Sikkim Manipal University, India

Organizing Committee

Santanu Phadikar	West Bengal University of Technology, India
SK Md Obaidullah	Aliah University, India
Sayantari Ghosh	National Institute of Technology Durgapur, India
Himadri Mukherjee	West Bengal State University, India
Nilamadhaba Tripathy	Clarivate Analytics, USA
Chayan Halder	West Bengal State University, India
Shibaprasad Sen	Techno Main Salt Lake, India

Finance Chairs

Kaushik Roy	West Bengal State University, India
Michael Blumenstein	University of Technology Sydney, Australia

Awards Committee Chair

Arpan Pal	Tata Consultancy Services, India

Sponsorship Chairs

P. J. Narayanan	Indian Institute of Technology, Hyderabad, India
Yasushi Yagi	Osaka University, Japan
Venu Govindaraju	University at Buffalo, USA
Alberto Bel Bimbo	Università di Firenze, Italy

Exhibition and Demonstration Chairs

Arjun Jain	FastCode AI, India
Agnimitra Biswas	National Institute of Technology, Silchar, India

International Liaison, Visa Chair

Balasubramanian Raman	Indian Institute of Technology, Roorkee, India

Publicity Chairs

Dipti Prasad Mukherjee	Indian Statistical Institute, Kolkata, India
Bob Fisher	University of Edinburgh, UK
Xiaojun Wu	Jiangnan University, China

Women in ICPR Chairs

Ingela Nystrom	Uppsala University, Sweden
Alexandra B. Albu	University of Victoria, Canada
Jing Dong	Institute of Automation, Chinese Academy of Sciences, China
Sarbani Palit	Indian Statistical Institute, Kolkata, India

Event Manager

Alpcord Network

Track Chairs – Artificial Intelligence, Machine Learning for Pattern Analysis

Larry O'Gorman	Nokia Bell Labs, USA
Dacheng Tao	University of Sydney, Australia
Petia Radeva	University of Barcelona, Spain
Susmita Mitra	Indian Statistical Institute, Kolkata, India
Jiliang Tang	Michigan State University, USA

Track Chairs – Computer and Robot Vision

C. V. Jawahar	International Institute of Information Technology (IIIT), Hyderabad, India
João Paulo Papa	São Paulo State University, Brazil
Maja Pantic	Imperial College London, UK
Gang Hua	Dolby Laboratories, USA
Junwei Han	Northwestern Polytechnical University, China

Track Chairs – Image, Speech, Signal and Video Processing

P. K. Biswas	Indian Institute of Technology, Kharagpur, India
Shang-Hong Lai	National Tsing Hua University, Taiwan
Hugo Jair Escalante	INAOE, CINVESTAV, Mexico
Sergio Escalera	Universitat de Barcelona, Spain
Prem Natarajan	University of Southern California, USA

Track Chairs – Biometrics and Human Computer Interaction

Richa Singh	Indian Institute of Technology, Jodhpur, India
Massimo Tistarelli	University of Sassari, Italy
Vishal Patel	Johns Hopkins University, USA
Wei-Shi Zheng	Sun Yat-sen University, China
Jian Wang	Snap, USA

Track Chairs – Document Analysis and Recognition

Xiang Bai	Huazhong University of Science and Technology, China
David Doermann	University at Buffalo, USA
Josep Llados	Universitat Autònoma de Barcelona, Spain
Mita Nasipuri	Jadavpur University, India

Track Chairs – Biomedical Imaging and Bioinformatics

Jayanta Mukhopadhyay	Indian Institute of Technology, Kharagpur, India
Xiaoyi Jiang	Universität Münster, Germany
Seong-Whan Lee	Korea University, Korea

Metareviewers (Conference Papers and Competition Papers)

Wael Abd-Almageed	University of Southern California, USA
Maya Aghaei	NHL Stenden University, Netherlands
Alireza Alaei	Southern Cross University, Australia
Rajagopalan N. Ambasamudram	Indian Institute of Technology, Madras, India
Suyash P. Awate	Indian Institute of Technology, Bombay, India
Inci M. Baytas	Bogazici University, Turkey
Aparna Bharati	Lehigh University, USA
Brojeshwar Bhowmick	Tata Consultancy Services, India
Jean-Christophe Burie	University of La Rochelle, France
Gustavo Carneiro	University of Surrey, UK
Chee Seng Chan	Universiti Malaya, Malaysia
Sumohana S. Channappayya	Indian Institute of Technology, Hyderabad, India
Dongdong Chen	Microsoft, USA
Shengyong Chen	Tianjin University of Technology, China
Jun Cheng	Institute for Infocomm Research, A*STAR, Singapore
Albert Clapés	University of Barcelona, Spain
Oscar Dalmau	Center for Research in Mathematics, Mexico

Tyler Derr	Vanderbilt University, USA
Abhinav Dhall	Indian Institute of Technology, Ropar, India
Bo Du	Wuhan University, China
Yuxuan Du	University of Sydney, Australia
Ayman S. El-Baz	University of Louisville, USA
Francisco Escolano	University of Alicante, Spain
Siamac Fazli	Nazarbayev University, Kazakhstan
Jianjiang Feng	Tsinghua University, China
Gernot A. Fink	TU Dortmund University, Germany
Alicia Fornes	CVC, Spain
Junbin Gao	University of Sydney, Australia
Yan Gao	Amazon, USA
Yongsheng Gao	Griffith University, Australia
Caren Han	University of Melbourne, Australia
Ran He	Institute of Automation, Chinese Academy of Sciences, China
Tin Kam Ho	IBM, USA
Di Huang	Beihang University, China
Kaizhu Huang	Duke Kunshan University, China
Donato Impedovo	University of Bari, Italy
Julio Jacques	University of Barcelona and Computer Vision Center, Spain
Lianwen Jin	South China University of Technology, China
Wei Jin	Emory University, USA
Danilo Samuel Jodas	São Paulo State University, Brazil
Manjunath V. Joshi	DA-IICT, India
Jayashree Kalpathy-Cramer	Massachusetts General Hospital, USA
Dimosthenis Karatzas	Computer Vision Centre, Spain
Hamid Karimi	Utah State University, USA
Baiying Lei	Shenzhen University, China
Guoqi Li	Chinese Academy of Sciences, and Peng Cheng Lab, China
Laurence Likforman-Sulem	Institut Polytechnique de Paris/Télécom Paris, France
Aishan Liu	Beihang University, China
Bo Liu	Bytedance, USA
Chen Liu	Clarkson University, USA
Cheng-Lin Liu	Institute of Automation, Chinese Academy of Sciences, China
Hongmin Liu	University of Science and Technology Beijing, China
Hui Liu	Michigan State University, USA

Jing Liu	Institute of Automation, Chinese Academy of Sciences, China
Li Liu	University of Oulu, Finland
Qingshan Liu	Nanjing University of Posts and Telecommunications, China
Adrian P. Lopez-Monroy	Centro de Investigacion en Matematicas AC, Mexico
Daniel P. Lopresti	Lehigh University, USA
Shijian Lu	Nanyang Technological University, Singapore
Yong Luo	Wuhan University, China
Andreas K. Maier	FAU Erlangen-Nuremberg, Germany
Davide Maltoni	University of Bologna, Italy
Hong Man	Stevens Institute of Technology, USA
Lingtong Min	Northwestern Polytechnical University, China
Paolo Napoletano	University of Milano-Bicocca, Italy
Kamal Nasrollahi	Milestone Systems, Aalborg University, Denmark
Marcos Ortega	University of A Coruña, Spain
Shivakumara Palaiahnakote	University of Salford, UK
P. Jonathon Phillips	NIST, USA
Filiberto Pla	University Jaume I, Spain
Ajit Rajwade	Indian Institute of Technology, Bombay, India
Shanmuganathan Raman	Indian Institute of Technology, Gandhinagar, India
Imran Razzak	UNSW, Australia
Beatriz Remeseiro	University of Oviedo, Spain
Gustavo Rohde	University of Virginia, USA
Partha Pratim Roy	Indian Institute of Technology, Roorkee, India
Sanjoy K. Saha	Jadavpur University, India
Joan Andreu Sánchez	Universitat Politècnica de València, Spain
Claudio F. Santos	UFSCar, Brazil
Shin'ichi Satoh	National Institute of Informatics, Japan
Stephanie Schuckers	Clarkson University, USA
Srirangaraj Setlur	University at Buffalo, SUNY, USA
Debdoot Sheet	Indian Institute of Technology, Kharagpur, India
Jun Shen	University of Wollongong, Australia
Li Shen	JD Explore Academy, China
Chen Shengyong	Zhejiang University of technology and Tianjin University of Technology, China
Andy Song	RMIT University, Australia
Akihiro Sugimoto	National Institute of Informatics, Japan
Qianru Sun	Singapore Management University, Singapore
Arijit Sur	Indian Institute of Technology, Guwahati, India
Estefania Talavera	University of Twente, Netherlands

Wei Tang	University of Illinois at Chicago, USA
Joao M. Tavares	Universidade do Porto, Portugal
Jun Wan	NLPR, CASIA, China
Le Wang	Xi'an Jiaotong University, China
Lei Wang	Australian National University, Australia
Xiaoyang Wang	Tencent AI Lab, USA
Xinggang Wang	Huazhong University of Science and Technology, China
Xiao-Jun Wu	Jiangnan University, China
Yiding Yang	Bytedance, China
Xiwen Yao	Northwestern Polytechnical University, China
Xu-Cheng Yin	University of Science and Technology Beijing, China
Baosheng Yu	University of Sydney, Australia
Shiqi Yu	Southern University of Science and Technology, China
Xin Yuan	Westlake University, China
Yibing Zhan	JD Explore Academy, China
Jing Zhang	University of Sydney, Australia
Lefei Zhang	Wuhan University, China
Min-Ling Zhang	Southeast University, China
Wenbin Zhang	Florida International University, USA
Jiahuan Zhou	Peking University, China
Sanping Zhou	Xi'an Jiaotong University, China
Tianyi Zhou	University of Maryland, USA
Lei Zhu	Shandong Normal University, China
Pengfei Zhu	Tianjin University, China
Wangmeng Zuo	Harbin Institute of Technology, China

Reviewers (Competition Papers)

Liangcai Gao	Da-Han Wang
Mingxin Huang	Yang Xue
Lei Kang	Wentao Yang
Wenhui Liao	Jiaxin Zhang
Yuliang Liu	Yiwu Zhong
Yongxin Shi	

Reviewers (Conference Papers)

Aakanksha Aakanksha
Aayush Singla
Abdul Muqeet
Abhay Yadav
Abhijeet Vijay Nandedkar
Abhimanyu Sahu
Abhinav Rajvanshi
Abhisek Ray
Abhishek Shrivastava
Abhra Chaudhuri
Aditi Roy
Adriano Simonetto
Adrien Maglo
Ahmed Abdulkadir
Ahmed Boudissa
Ahmed Hamdi
Ahmed Rida Sekkat
Ahmed Sharafeldeen
Aiman Farooq
Aishwarya Venkataramanan
Ajay Kumar
Ajay Kumar Reddy Poreddy
Ajita Rattani
Ajoy Mondal
Akbar K.
Akbar Telikani
Akshay Agarwal
Akshit Jindal
Al Zadid Sultan Bin Habib
Albert Clapés
Alceu Britto
Alejandro Peña
Alessandro Ortis
Alessia Auriemma Citarella
Alexandre Stenger
Alexandros Sopasakis
Alexia Toumpa
Ali Khan
Alik Pramanick
Alireza Alaei
Alper Yilmaz
Aman Verma
Amit Bhardwaj
Amit More
Amit Nandedkar
Amitava Chatterjee
Amos L. Abbott
Amrita Mohan
Anand Mishra
Ananda S. Chowdhury
Anastasia Zakharova
Anastasios L. Kesidis
Andras Horvath
Andre Gustavo Hochuli
André P. Kelm
Andre Wyzykowski
Andrea Bottino
Andrea Lagorio
Andrea Torsello
Andreas Fischer
Andreas K. Maier
Andreu Girbau Xalabarder
Andrew Beng Jin Teoh
Andrew Shin
Andy J. Ma
Aneesh S. Chivukula
Ángela Casado-García
Anh Quoc Nguyen
Anindya Sen
Anirban Saha
Anjali Gautam
Ankan Bhattacharyya
Ankit Jha
Anna Scius-Bertrand
Annalisa Franco
Antoine Doucet
Antonino Staiano
Antonio Fernández
Antonio Parziale
Anu Singha
Anustup Choudhury
Anwesan Pal
Anwesha Sengupta
Archisman Adhikary
Arjan Kuijper
Arnab Kumar Das

Arnav Bhavsar
Arnav Varma
Arpita Dutta
Arshad Jamal
Artur Jordao
Arunkumar Chinnaswamy
Aryan Jadon
Aryaz Baradarani
Ashima Anand
Ashis Dhara
Ashish Phophalia
Ashok K. Bhateja
Ashutosh Vaish
Ashwani Kumar
Asifuzzaman Lasker
Atefeh Khoshkhahtinat
Athira Nambiar
Attilio Fiandrotti
Avandra S. Hemachandra
Avik Hati
Avinash Sharma
B. H. Shekar
B. Uma Shankar
Bala Krishna Thunakala
Balaji Tk
Balázs Pálffy
Banafsheh Adami
Bang-Dang Pham
Baochang Zhang
Baodi Liu
Bashirul Azam Biswas
Beiduo Chen
Benedikt Kottler
Beomseok Oh
Berkay Aydin
Berlin S. Shaheema
Bertrand Kerautret
Bettina Finzel
Bhavana Singh
Bibhas C. Dhara
Bilge Gunsel
Bin Chen
Bin Li
Bin Liu
Bin Yao
Bin-Bin Jia
Binbin Yong
Bindita Chaudhuri
Bindu Madhavi Tummala
Binh M. Le
Bi-Ru Dai
Bo Huang
Bo Jiang
Bob Zhang
Bowen Liu
Bowen Zhang
Boyang Zhang
Boyu Diao
Boyun Li
Brian M. Sadler
Bruce A. Maxwell
Bryan Bo Cao
Buddhika L. Semage
Bushra Jalil
Byeong-Seok Shin
Byung-Gyu Kim
Caihua Liu
Cairong Zhao
Camille Kurtz
Carlos A. Caetano
Carlos D. Martã-Nez-Hinarejos
Ce Wang
Cevahir Cigla
Chakravarthy Bhagvati
Chandrakanth Vipparla
Changchun Zhang
Changde Du
Changkun Ye
Changxu Cheng
Chao Fan
Chao Guo
Chao Qu
Chao Wen
Chayan Halder
Che-Jui Chang
Chen Feng
Chenan Wang
Cheng Yu
Chenghao Qian
Cheng-Lin Liu

Chengxu Liu
Chenru Jiang
Chensheng Peng
Chetan Ralekar
Chih-Wei Lin
Chih-Yi Chiu
Chinmay Sahu
Chintan Patel
Chintan Shah
Chiranjoy Chattopadhyay
Chong Wang
Choudhary Shyam Prakash
Christophe Charrier
Christos Smailis
Chuanwei Zhou
Chun-Ming Tsai
Chunpeng Wang
Ciro Russo
Claudio De Stefano
Claudio F. Santos
Claudio Marrocco
Connor Levenson
Constantine Dovrolis
Constantine Kotropoulos
Dai Shi
Dakshina Ranjan Kisku
Dan Anitei
Dandan Zhu
Daniela Pamplona
Danli Wang
Danqing Huang
Daoan Zhang
Daqing Hou
David A. Clausi
David Freire Obregon
David Münch
David Pujol Perich
Davide Marelli
De Zhang
Debalina Barik
Debapriya Roy (Kundu)
Debashis Das
Debashis Das Chakladar
Debi Prosad Dogra
Debraj D. Basu
Decheng Liu
Deen Dayal Mohan
Deep A. Patel
Deepak Kumar
Dengpan Liu
Denis Coquenet
Désiré Sidibé
Devesh Walawalkar
Dewan Md. Farid
Di Ming
Di Qiu
Di Yuan
Dian Jia
Dianmo Sheng
Diego Thomas
Diganta Saha
Dimitri Bulatov
Dimpy Varshni
Dingcheng Yang
Dipanjan Das
Dipanjyoti Paul
Divya Biligere Shivanna
Divya Saxena
Divya Sharma
Dmitrii Matveichev
Dmitry Minskiy
Dmitry V. Sorokin
Dong Zhang
Donghua Wang
Donglin Zhang
Dongming Wu
Dongqiangzi Ye
Dongqing Zou
Dongrui Liu
Dongyang Zhang
Dongzhan Zhou
Douglas Rodrigues
Duarte Folgado
Duc Minh Vo
Duoxuan Pei
Durai Arun Pannir Selvam
Durga Bhavani S.
Eckart Michaelsen
Elena Goyanes
Élodie Puybareau

Emanuele Vivoli
Emna Ghorbel
Enrique Naredo
Enyu Cai
Eric Patterson
Ernest Valveny
Eva Blanco-Mallo
Eva Breznik
Evangelos Sartinas
Fabio Solari
Fabiola De Marco
Fan Wang
Fangda Li
Fangyuan Lei
Fangzhou Lin
Fangzhou Luo
Fares Bougourzi
Farman Ali
Fatiha Mokdad
Fei Shen
Fei Teng
Fei Zhu
Feiyan Hu
Felipe Gomes Oliveira
Feng Li
Fengbei Liu
Fenghua Zhu
Fillipe D. M. De Souza
Flavio Piccoli
Flavio Prieto
Florian Kleber
Francesc Serratosa
Francesco Bianconi
Francesco Castro
Francesco Ponzio
Francisco Javier Hernández López
Frédéric Rayar
Furkan Osman Kar
Fushuo Huo
Fuxiao Liu
Fu-Zhao Ou
Gabriel Turinici
Gabrielle Flood
Gajjala Viswanatha Reddy
Gaku Nakano
Galal Binamakhashen
Ganesh Krishnasamy
Gang Pan
Gangyan Zeng
Gani Rahmon
Gaurav Harit
Gennaro Vessio
Genoveffa Tortora
George Azzopardi
Gerard Ortega
Gerardo E. Altamirano-Gomez
Gernot A. Fink
Gibran Benitez-Garcia
Gil Ben-Artzi
Gilbert Lim
Giorgia Minello
Giorgio Fumera
Giovanna Castellano
Giovanni Puglisi
Giulia Orrù
Giuliana Ramella
Gökçe Uludoğan
Gopi Ramena
Gorthi Rama Krishna Sai Subrahmanyam
Gourav Datta
Gowri Srinivasa
Gozde Sahin
Gregory Randall
Guanjie Huang
Guanjun Li
Guanwen Zhang
Guanyu Xu
Guanyu Yang
Guanzhou Ke
Guhnoo Yun
Guido Borghi
Guilherme Brandão Martins
Guillaume Caron
Guillaume Tochon
Guocai Du
Guohao Li
Guoqiang Zhong
Guorong Li
Guotao Li
Gurman Gill

Haechang Lee
Haichao Zhang
Haidong Xie
Haifeng Zhao
Haimei Zhao
Hainan Cui
Haixia Wang
Haiyan Guo
Hakime Ozturk
Hamid Kazemi
Han Gao
Hang Zou
Hanjia Lyu
Hanjoo Cho
Hanqing Zhao
Hanyuan Liu
Hanzhou Wu
Hao Li
Hao Meng
Hao Sun
Hao Wang
Hao Xing
Hao Zhao
Haoan Feng
Haodi Feng
Haofeng Li
Haoji Hu
Haojie Hao
Haojun Ai
Haopeng Zhang
Haoran Li
Haoran Wang
Haorui Ji
Haoxiang Ma
Haoyu Chen
Haoyue Shi
Harald Koestler
Harbinder Singh
Harris V. Georgiou
Hasan F. Ates
Hasan S. M. Al-Khaffaf
Hatef Otroshi Shahreza
Hebeizi Li
Heng Zhang
Hengli Wang
Hengyue Liu
Hertog Nugroho
Hieyong Jeong
Himadri Mukherjee
Hoai Ngo
Hoda Mohaghegh
Hong Liu
Hong Man
Hongcheng Wang
Hongjian Zhan
Hongxi Wei
Hongyu Hu
Hoseong Kim
Hossein Ebrahimnezhad
Hossein Malekmohamadi
Hrishav Bakul Barua
Hsueh-Yi Sean Lin
Hua Wei
Huafeng Li
Huali Xu
Huaming Chen
Huan Wang
Huang Chen
Huanran Chen
Hua-Wen Chang
Huawen Liu
Huayi Zhan
Hugo Jair Escalante
Hui Chen
Hui Li
Huichen Yang
Huiqiang Jiang
Huiyuan Yang
Huizi Yu
Hung T. Nguyen
Hyeongyu Kim
Hyeonjeong Park
Hyeonjun Lee
Hymalai Bello
Hyung-Gun Chi
Hyunsoo Kim
I-Chen Lin
Ik Hyun Lee
Ilan Shimshoni
Imad Eddine Toubal

Imran Sarker
Inderjot Singh Saggu
Indrani Mukherjee
Indranil Sur
Ines Rieger
Ioannis Pierros
Irina Rabaev
Ivan V. Medri
J. Rafid Siddiqui
Jacek Komorowski
Jacopo Bonato
Jacson Rodrigues Correia-Silva
Jaekoo Lee
Jaime Cardoso
Jakob Gawlikowski
Jakub Nalepa
James L. Wayman
Jan Čech
Jangho Lee
Jani Boutellier
Javier Gurrola-Ramos
Javier Lorenzo-Navarro
Jayasree Saha
Jean Lee
Jean Paul Barddal
Jean-Bernard Hayet
Jean-Philippe G. Tarel
Jean-Yves Ramel
Jenny Benois-Pineau
Jens Bayer
Jerin Geo James
Jesús Miguel García-Gorrostieta
Jia Qu
Jiahong Chen
Jiaji Wang
Jian Hou
Jian Liang
Jian Xu
Jian Zhu
Jianfeng Lu
Jianfeng Ren
Jiangfan Liu
Jianguo Wang
Jiangyan Yi
Jiangyong Duan
Jianhua Yang
Jianhua Zhang
Jianhui Chen
Jianjia Wang
Jianli Xiao
Jianqiang Xiao
Jianwu Wang
Jianxin Zhang
Jianxiong Gao
Jianxiong Zhou
Jianyu Wang
Jianzhong Wang
Jiaru Zhang
Jiashu Liao
Jiaxin Chen
Jiaxin Lu
Jiaxing Ye
Jiaxuan Chen
Jiaxuan Li
Jiayi He
Jiayin Lin
Jie Ou
Jiehua Zhang
Jiejie Zhao
Jignesh S. Bhatt
Jin Gao
Jin Hou
Jin Hu
Jin Shang
Jing Tian
Jing Yu Chen
Jingfeng Yao
Jinglun Feng
Jingtong Yue
Jingwei Guo
Jingwen Xu
Jingyuan Xia
Jingzhe Ma
Jinhong Wang
Jinjia Wang
Jinlai Zhang
Jinlong Fan
Jinming Su
Jinrong He
Jintao Huang

Jinwoo Ahn
Jinwoo Choi
Jinyang Liu
Jinyu Tian
Jionghao Lin
Jiuding Duan
Jiwei Shen
Jiyan Pan
Jiyoun Kim
João Papa
Johan Debayle
John Atanbori
John Wilson
John Zhang
Jónathan Heras
Joohi Chauhan
Jorge Calvo-Zaragoza
Jorge Figueroa
Jorma Laaksonen
José Joaquim De Moura Ramos
Jose Vicent
Joseph Damilola Akinyemi
Josiane Zerubia
Juan Wen
Judit Szücs
Juepeng Zheng
Juha Roning
Jumana H. Alsubhi
Jun Cheng
Jun Ni
Jun Wan
Junghyun Cho
Junjie Liang
Junjie Ye
Junlin Hu
Juntong Ni
Junxin Lu
Junxuan Li
Junyaup Kim
Junyeong Kim
Jürgen Seiler
Jushang Qiu
Juyang Weng
Jyostna Devi Bodapati
Jyoti Singh Kirar
Kai Jiang
Kaiqiang Song
Kalidas Yeturu
Kalle Åström
Kamalakar Vijay Thakare
Kang Gu
Kang Ma
Kanji Tanaka
Karthik Seemakurthy
Kaushik Roy
Kavisha Jayathunge
Kazuki Uehara
Ke Shi
Keigo Kimura
Keiji Yanai
Kelton A. P. Costa
Kenneth Camilleri
Kenny Davila
Ketan Atul Bapat
Ketan Kotwal
Kevin Desai
Keyu Long
Khadiga Mohamed Ali
Khakon Das
Khan Muhammad
Kilho Son
Kim-Ngan Nguyen
Kishan Kc
Kishor P. Upla
Klaas Dijkstra
Komal Bharti
Konstantinos Triaridis
Kostas Ioannidis
Koyel Ghosh
Kripabandhu Ghosh
Krishnendu Ghosh
Kshitij S. Jadhav
Kuan Yan
Kun Ding
Kun Xia
Kun Zeng
Kunal Banerjee
Kunal Biswas
Kunchi Li
Kurban Ubul

Lahiru N. Wijayasingha
Laines Schmalwasser
Lakshman Mahto
Lala Shakti Swarup Ray
Lale Akarun
Lan Yan
Lawrence Amadi
Lee Kang Il
Lei Fan
Lei Shi
Lei Wang
Leonardo Rossi
Lequan Lin
Levente Tamas
Li Bing
Li Li
Li Ma
Li Song
Lia Morra
Liang Xie
Liang Zhao
Lianwen Jin
Libing Zeng
Lidia Sánchez-González
Lidong Zeng
Lijun Li
Likang Wang
Lili Zhao
Lin Chen
Lin Huang
Linfei Wang
Ling Lo
Lingchen Meng
Lingheng Meng
Lingxiao Li
Lingzhong Fan
Liqi Yan
Liqiang Jing
Lisa Gutzeit
Liu Ziyi
Liushuai Shi
Liviu-Daniel Stefan
Liyuan Ma
Liyun Zhu
Lizuo Jin
Longteng Guo
Lorena Álvarez Rodríguez
Lorenzo Putzu
Lu Leng
Lu Pang
Lu Wang
Luan Pham
Luc Brun
Luca Guarnera
Luca Piano
Lucas Alexandre Ramos
Lucas Goncalves
Lucas M. Gago
Luigi Celona
Luis C. S. Afonso
Luis Gerardo De La Fraga
Luis S. Luevano
Luis Teixeira
Lunke Fei
M. Hassaballah
Maddimsetti Srinivas
Mahendran N.
Mahesh Mohan M. R.
Maiko Lie
Mainak Singha
Makoto Hirose
Malay Bhattacharyya
Mamadou Dian Bah
Man Yao
Manali J. Patel
Manav Prabhakar
Manikandan V. M.
Manish Bhatt
Manjunath Shantharamu
Manuel Curado
Manuel Günther
Manuel Marques
Marc A. Kastner
Marc Chaumont
Marc Cheong
Marc Lalonde
Marco Cotogni
Marcos C. Santana
Mario Molinara
Mariofanna Milanova

Markus Bauer
Marlon Becker
Mårten Wadenbäck
Martin G. Ljungqvist
Martin Kampel
Martina Pastorino
Marwan Torki
Masashi Nishiyama
Masayuki Tanaka
Massimo O. Spata
Matteo Ferrara
Matthew D. Dawkins
Matthew Gadd
Matthew S. Watson
Maura Pintor
Max Ehrlich
Maxim Popov
Mayukh Das
Md Baharul Islam
Md Sajid
Meghna Kapoor
Meghna P. Ayyar
Mei Wang
Meiqi Wu
Melissa L. Tijink
Meng Li
Meng Liu
Meng-Luen Wu
Mengnan Liu
Mengxi China Guo
Mengya Han
Michaël Clément
Michal Kawulok
Mickael Coustaty
Miguel Domingo
Milind G. Padalkar
Ming Liu
Ming Ma
Mingchen Feng
Mingde Yao
Minghao Li
Mingjie Sun
Ming-Kuang Daniel Wu
Mingle Xu
Mingyong Li
Mingyuan Jiu
Minh P. Nguyen
Minh Q. Tran
Minheng Ni
Minsu Kim
Minyi Zhao
Mirko Paolo Barbato
Mo Zhou
Modesto Castrillón-Santana
Mohamed Amine Mezghich
Mohamed Dahmane
Mohamed Elsharkawy
Mohamed Yousuf
Mohammad Hashemi
Mohammad Khalooei
Mohammad Khateri
Mohammad Mahdi Dehshibi
Mohammad Sadil Khan
Mohammed Mahmoud
Moises Diaz
Monalisha Mahapatra
Monidipa Das
Mostafa Kamali Tabrizi
Mridul Ghosh
Mrinal Kanti Bhowmik
Muchao Ye
Mugalodi Ramesha Rakesh
Muhammad Rameez Ur Rahman
Muhammad Suhaib Kanroo
Muming Zhao
Munender Varshney
Munsif Ali
Na Lv
Nader Karimi
Nagabhushan Somraj
Nakkwan Choi
Nakul Agarwal
Nan Pu
Nan Zhou
Nancy Mehta
Nand Kumar Yadav
Nandakishor Nandakishor
Nandyala Hemachandra
Nanfeng Jiang
Narayan Hegde

Narayan Ji Mishra
Narayan Vetrekar
Narendra D. Londhe
Nathalie Girard
Nati Ofir
Naval Kishore Mehta
Nazmul Shahadat
Neeti Narayan
Neha Bhargava
Nemanja Djuric
Newlin Shebiah R.
Ngo Ba Hung
Nhat-Tan Bui
Niaz Ahmad
Nick Theisen
Nicolas Passat
Nicolas Ragot
Nicolas Sidere
Nikolaos Mitianoudis
Nikolas Ebert
Nilah Ravi Nair
Nilesh A. Ahuja
Nilkanta Sahu
Nils Murrugarra-Llerena
Nina S. T. Hirata
Ninad Aithal
Ning Xu
Ningzhi Wang
Niraj Kumar
Nirmal S. Punjabi
Nisha Varghese
Norio Tagawa
Obaidullah Md Sk
Oguzhan Ulucan
Olfa Mechi
Oliver Tüselmann
Orazio Pontorno
Oriol Ramos Terrades
Osman Akin
Ouadi Beya
Ozge Mercanoglu Sincan
Pabitra Mitra
Padmanabha Reddy Y. C. A.
Palaash Agrawal
Palaiahnakote Shivakumara
Palash Ghosal
Pallav Dutta
Paolo Rota
Paramanand Chandramouli
Paria Mehrani
Parth Agrawal
Partha Basuchowdhuri
Patrick Horain
Pavan Kumar
Pavan Kumar Anasosalu Vasu
Pedro Castro
Peipei Li
Peipei Yang
Peisong Shen
Peiyu Li
Peng Li
Pengfei He
Pengrui Quan
Pengxin Zeng
Pengyu Yan
Peter Eisert
Petra Gomez-Krämer
Pierrick Bruneau
Ping Cao
Pingping Zhang
Pintu Kumar
Pooja Kumari
Pooja Sahani
Prabhu Prasad Dev
Pradeep Kumar
Pradeep Singh
Pranjal Sahu
Prasun Roy
Prateek Keserwani
Prateek Mittal
Praveen Kumar Chandaliya
Praveen Tirupattur
Pravin Nair
Preeti Gopal
Preety Singh
Prem Shanker Yadav
Prerana Mukherjee
Prerna A. Mishra
Prianka Dey
Priyanka Mudgal

Qc Kha Ng
Qi Li
Qi Ming
Qi Wang
Qi Zuo
Qian Li
Qiang Gan
Qiang He
Qiang Wu
Qiangqiang Zhou
Qianli Zhao
Qiansen Hong
Qiao Wang
Qidong Huang
Qihua Dong
Qin Yuke
Qing Guo
Qingbei Guo
Qingchao Zhang
Qingjie Liu
Qinhong Yang
Qiushi Shi
Qixiang Chen
Quan Gan
Quanlong Guan
Rachit Chhaya
Radu Tudor Ionescu
Rafal Zdunek
Raghavendra Ramachandra
Rahimul I. Mazumdar
Rahul Kumar Ray
Rajib Dutta
Rajib Ghosh
Rakesh Kumar
Rakesh Paul
Rama Chellappa
Rami O. Skaik
Ramon Aranda
Ran Wei
Ranga Raju Vatsavai
Ranganath Krishnan
Rasha Friji
Rashmi S.
Razaib Tariq
Rémi Giraud
René Schuster
Renlong Hang
Renrong Shao
Renu Sharma
Reza Sadeghian
Richard Zanibbi
Rimon Elias
Rishabh Shukla
Rita Delussu
Riya Verma
Robert J. Ravier
Robert Sablatnig
Robin Strand
Rocco Pietrini
Rocio Diaz Martin
Rocio Gonzalez-Diaz
Rohit Venkata Sai Dulam
Romain Giot
Romi Banerjee
Ru Wang
Ruben Machucho
Ruddy Théodose
Ruggero Pintus
Rui Deng
Rui P. Paiva
Rui Zhao
Ruifan Li
Ruigang Fu
Ruikun Li
Ruirui Li
Ruixiang Jiang
Ruowei Jiang
Rushi Lan
Rustam Zhumagambetov
S. Amutha
S. Divakar Bhat
Sagar Goyal
Sahar Siddiqui
Sahbi Bahroun
Sai Karthikeya Vemuri
Saibal Dutta
Saihui Hou
Sajad Ahmad Rather
Saksham Aggarwal
Sakthi U.

Salimeh Sekeh
Samar Bouazizi
Samia Boukir
Samir F. Harb
Samit Biswas
Samrat Mukhopadhyay
Samriddha Sanyal
Sandika Biswas
Sandip Purnapatra
Sanghyun Jo
Sangwoo Cho
Sanjay Kumar
Sankaran Iyer
Sanket Biswas
Santanu Roy
Santosh D. Pandure
Santosh Ku Behera
Santosh Nanabhau Palaskar
Santosh Prakash Chouhan
Sarah S. Alotaibi
Sasanka Katreddi
Sathyanarayanan N. Aakur
Saurabh Yadav
Sayan Rakshit
Scott McCloskey
Sebastian Bunda
Sejuti Rahman
Selim Aksoy
Sen Wang
Seraj A. Mostafa
Shanmuganathan Raman
Shao-Yuan Lo
Shaoyuan Xu
Sharia Arfin Tanim
Shehreen Azad
Sheng Wan
Shengdong Zhang
Shengwei Qin
Shenyuan Gao
Sherry X. Chen
Shibaprasad Sen
Shigeaki Namiki
Shiguang Liu
Shijie Ma
Shikun Li
Shinichiro Omachi
Shirley David
Shishir Shah
Shiv Ram Dubey
Shiva Baghel
Shivanand S. Gornale
Shogo Sato
Shotaro Miwa
Shreya Ghosh
Shreya Goyal
Shuai Su
Shuai Wang
Shuai Zheng
Shuaifeng Zhi
Shuang Qiu
Shuhei Tarashima
Shujing Lyu
Shuliang Wang
Shun Zhang
Shunming Li
Shunxin Wang
Shuping Zhao
Shuquan Ye
Shuwei Huo
Shuyue Lan
Shyi-Chyi Cheng
Si Chen
Siddarth Ravichandran
Sihan Chen
Siladittya Manna
Silambarasan Elkana Ebinazer
Simon Benaïchouche
Simon S. Woo
Simone Caldarella
Simone Milani
Simone Zini
Sina Lotfian
Sitao Luan
Sivaselvan B.
Siwei Li
Siwei Wang
Siwen Luo
Siyu Chen
Sk Aziz Ali
Sk Md Obaidullah

Sneha Shukla
Snehasis Banerjee
Snehasis Mukherjee
Snigdha Sen
Sofia Casarin
Soheila Farokhi
Soma Bandyopadhyay
Son Minh Nguyen
Son Xuan Ha
Sonal Kumar
Sonam Gupta
Sonam Nahar
Song Ouyang
Sotiris Kotsiantis
Souhaila Djaffal
Soumen Biswas
Soumen Sinha
Soumitri Chattopadhyay
Souvik Sengupta
Spiros Kostopoulos
Sreeraj Ramachandran
Sreya Banerjee
Srikanta Pal
Srinivas Arukonda
Stephane A. Guinard
Su O. Ruan
Subhadip Basu
Subhajit Paul
Subhankar Ghosh
Subhankar Mishra
Subhankar Roy
Subhash Chandra Pal
Subhayu Ghosh
Sudip Das
Sudipta Banerjee
Suhas Pillai
Sujit Das
Sukalpa Chanda
Sukhendu Das
Suklav Ghosh
Suman K. Ghosh
Suman Samui
Sumit Mishra
Sungho Suh
Sunny Gupta
Suraj Kumar Pandey
Surendrabikram Thapa
Suresh Sundaram
Sushil Bhattacharjee
Susmita Ghosh
Swakkhar Shatabda
Syed Ms Islam
Syed Tousiful Haque
Taegyeong Lee
Taihui Li
Takashi Shibata
Takeshi Oishi
Talha Ahmad Siddiqui
Tanguy Gernot
Tangwen Qian
Tanima Bhowmik
Tanpia Tasnim
Tao Dai
Tao Hu
Tao Sun
Taoran Yi
Tapan Shah
Taveena Lotey
Teng Huang
Tengqi Ye
Teresa Alarcon
Tetsuji Ogawa
Thanh Phuong Nguyen
Thanh Tuan Nguyen
Thattapon Surasak
Thibault Napolãon
Thierry Bouwmans
Thinh Truong Huynh Nguyen
Thomas De Min
Thomas E. K. Zielke
Thomas Swearingen
Tianatahina Jimmy Francky Randrianasoa
Tianheng Cheng
Tianjiao He
Tianyi Wei
Tianyuan Zhang
Tianyue Zheng
Tiecheng Song
Tilottama Goswami
Tim Büchner

Tim H. Langer
Tim Raven
Tingkai Liu
Tingting Yao
Tobias Meisen
Toby P. Breckon
Tong Chen
Tonghua Su
Tran Tuan Anh
Tri-Cong Pham
Trishna Saikia
Trung Quang Truong
Tuan T. Nguyen
Tuan Vo Van
Tushar Shinde
Ujjwal Karn
Ukrit Watchareeruetai
Uma Mudenagudi
Umarani Jayaraman
V. S. Malemath
Vallidevi Krishnamurthy
Ved Prakash
Venkata Krishna Kishore Kolli
Venkata R. Vavilthota
Venkatesh Thirugnana Sambandham
Verónica Maria Vasconcelos
Véronique Ve Eglin
Víctor E. Alonso-Pérez
Vinay Palakkode
Vinayak S. Nageli
Vincent J. Whannou De Dravo
Vincenzo Conti
Vincenzo Gattulli
Vineet Padmanabhan
Vishakha Pareek
Viswanath Gopalakrishnan
Vivek Singh Baghel
Vivekraj K.
Vladimir V. Arlazarov
Vu-Hoang Tran
W. Sylvia Lilly Jebarani
Wachirawit Ponghiran
Wafa Khlif
Wang An-Zhi
Wanli Xue
Wataru Ohyama
Wee Kheng Leow
Wei Chen
Wei Cheng
Wei Hua
Wei Lu
Wei Pan
Wei Tian
Wei Wang
Wei Wei
Wei Zhou
Weidi Liu
Weidong Yang
Weijun Tan
Weimin Lyu
Weinan Guan
Weining Wang
Weiqiang Wang
Weiwei Guo
Weixia Zhang
Wei-Xuan Bao
Weizhong Jiang
Wen Xie
Wenbin Qian
Wenbin Tian
Wenbin Wang
Wenbo Zheng
Wenhan Luo
Wenhao Wang
Wen-Hung Liao
Wenjie Li
Wenkui Yang
Wenwen Si
Wenwen Yu
Wenwen Zhang
Wenwu Yang
Wenxi Li
Wenxi Yue
Wenxue Cui
Wenzhuo Liu
Widhiyo Sudiyono
Willem Dijkstra
Wolfgang Fuhl
Xi Zhang
Xia Yuan

Xianda Zhang
Xiang Zhang
Xiangdong Su
Xiang-Ru Yu
Xiangtai Li
Xiangyu Xu
Xiao Guo
Xiao Hu
Xiao Wu
Xiao Yang
Xiaofeng Zhang
Xiaogang Du
Xiaoguang Zhao
Xiaoheng Jiang
Xiaohong Zhang
Xiaohua Huang
Xiaohua Li
Xiao-Hui Li
Xiaolong Sun
Xiaosong Li
Xiaotian Li
Xiaoting Wu
Xiaotong Luo
Xiaoyan Li
Xiaoyang Kang
Xiaoyi Dong
Xin Guo
Xin Lin
Xin Ma
Xinchi Zhou
Xingguang Zhang
Xingjian Leng
Xingpeng Zhang
Xingzheng Lyu
Xinjian Huang
Xinqi Fan
Xinqi Liu
Xinqiao Zhang
Xinrui Cui
Xizhan Gao
Xu Cao
Xu Ouyang
Xu Zhao
Xuan Shen
Xuan Zhou
Xuchen Li
Xuejing Lei
Xuelu Feng
Xueting Liu
Xuewei Li
Xueyi X. Wang
Xugong Qin
Xu-Qian Fan
Xuxu Liu
Xu-Yao Zhang
Yan Huang
Yan Li
Yan Wang
Yan Xia
Yan Zhuang
Yanan Li
Yanan Zhang
Yang Hou
Yang Jiao
Yang Liping
Yang Liu
Yang Qian
Yang Yang
Yang Zhao
Yangbin Chen
Yangfan Zhou
Yanhui Guo
Yanjia Huang
Yanjun Zhu
Yanming Zhang
Yanqing Shen
Yaoming Cai
Yaoxin Zhuo
Yaoyan Zheng
Yaping Zhang
Yaqian Liang
Yarong Feng
Yasmina Benmabrouk
Yasufumi Sakai
Yasutomo Kawanishi
Yazeed Alzahrani
Ye Du
Ye Duan
Yechao Zhang
Yeong-Jun Cho

Yi Huo
Yi Shi
Yi Yu
Yi Zhang
Yibo Liu
Yibo Wang
Yi-Chieh Wu
Yifan Chen
Yifei Huang
Yihao Ding
Yijie Tang
Yikun Bai
Yimin Wen
Yinan Yang
Yin-Dong Zheng
Yinfeng Yu
Ying Dai
Yingbo Li
Yiqiao Li
Yiqing Huang
Yisheng Lv
Yisong Xiao
Yite Wang
Yizhe Li
Yong Wang
Yonghao Dong
Yong-Hyuk Moon
Yongjie Li
Yongqian Li
Yongqiang Mao
Yongxu Liu
Yongyu Wang
Yongzhi Li
Youngha Hwang
Yousri Kessentini
Yu Wang
Yu Zhou
Yuan Tian
Yuan Zhang
Yuanbo Wen
Yuanxin Wang
Yubin Hu
Yubo Huang
Yuchen Ren
Yucheng Xing
Yuchong Yao
Yuecong Min
Yuewei Yang
Yufei Zhang
Yufeng Yin
Yugen Yi
Yuhang Ming
Yujia Zhang
Yujun Ma
Yukiko Kenmochi
Yun Hoyeoung
Yun Liu
Yunhe Feng
Yunxiao Shi
Yuru Wang
Yushun Tang
Yusuf Osmanlioglu
Yusuke Fujita
Yuta Nakashima
Yuwei Yang
Yuwu Lu
Yuxi Liu
Yuya Obinata
Yuyao Yan
Yuzhi Guo
Zaipeng Xie
Zander W. Blasingame
Zedong Wang
Zeliang Zhang
Zexin Ji
Zhanxiang Feng
Zhaofei Yu
Zhe Chen
Zhe Cui
Zhe Liu
Zhe Wang
Zhekun Luo
Zhen Yang
Zhenbo Li
Zhenchun Lei
Zhenfei Zhang
Zheng Liu
Zheng Wang
Zhengming Yu
Zhengyin Du

Zhengyun Cheng
Zhenshen Qu
Zhenwei Shi
Zhenzhong Kuang
Zhi Cai
Zhi Chen
Zhibo Chu
Zhicun Yin
Zhida Huang
Zhida Zhang
Zhifan Gao
Zhihang Ren
Zhihang Yuan
Zhihao Wang
Zhihua Xie
Zhihui Wang
Zhikang Zhang
Zhiming Zou
Zhiqi Shao
Zhiwei Dong
Zhiwei Qi
Zhixiang Wang
Zhixuan Li
Zhiyu Jiang
Zhiyuan Yan
Zhiyuan Yu
Zhiyuan Zhang
Zhong Chen
Zhongwei Teng
Zhongzhan Huang
Zhongzhi Yu
Zhuan Han
Zhuangzhuang Chen
Zhuo Liu
Zhuo Su
Zhuojun Zou
Zhuoyue Wang
Ziang Song
Zicheng Zhang
Zied Mnasri
Zifan Chen
Žiga Babnik
Zijing Chen
Zikai Zhang
Ziling Huang
Zilong Du
Ziqi Cai
Ziqi Zhou
Zi-Rui Wang
Zirui Zhou
Ziwen He
Ziyao Zeng
Ziyi Zhang
Ziyue Xiang
Zonglei Jing
Zongyi Xu

Contents – Part VII

Graph Matching Networks Meet Optimum-Path Forest: How to Prune Ensembles Efficiently

Danilo Jodas(✉), Leandro A. Passos, Douglas Rodrigues, Kelton Costa, and João Paulo Papa

São Paulo State University (UNESP), School of Sciences, Bauru, Brazil
{danilo.jodas,leandro.passos,d.rodrigues,kelton.costa, joao.papa}@unesp.br

Abstract. Ensemble pruning techniques are widely used to enhance a set of classifiers' efficiency and predictive performance by selecting a subset of representative models, preventing redundancy, and ensuring diversity in classification tasks. The Optimum-Path Forest (OPF), a stable and efficient graph-based framework, offers versatile supervised and unsupervised capabilities in various machine-learning applications. The supervised version provides remarkable results with a simple graph-based structure produced by a training process conducted over a single dataset. However, one can notice little effort in OPF-based ensemble learning. This paper introduces an innovative approach to pruning OPF classifiers using meta-descriptions learned by Graph-Matching Networks, which are further employed to cluster similar OPF instances. The strategy selectively chooses representative models that excel in predictive tasks from groups generated by unsupervised OPF. Results demonstrate competitive performance to state-of-the-art pruning algorithms, with experiments conducted over fifteen public datasets, encouraging further exploration of Graph Matching Networks applied to ensemble pruning.

Keywords: Graph Matching · Graph Matching Networks · Optimum-Path Forest · Ensemble Pruning

1 Introduction

A graph is a mathematical tool that models the relationship of objects by representing each instance as nodes or vertices whose connections are determined through an adjacency relation. Its first use remits to the eighteenth century's topological problem of Königsberg bridge [1], and has numerous applications on today's issues, ranging from routing [16], network flow [8], and social network modelling [25] to machine learning [9,19,20].

Graph representations help encode complex and abstract structural information, including DNA sequences, documents, and images. In this sense, a graph represents general knowledge of a given domain. Finding similar graphs corresponds to uncovering patterns in such complex structures. The area that studies

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 1–18, 2025.
https://doi.org/10.1007/978-3-031-78183-4_1

such similarities is called graph matching (GM) and regards the problem of finding correspondences between sets of nodes while preserving complex relational information among them [2].

In the context of graph matching, one can find a broad spectrum of related research involving robust and efficient approaches through spectral embeddings [14], Lagrangean relaxations optimized via sub-gradient and message passing mechanisms [24], and strategies based on non-convex and non-separable optimization problems, employing matching constraint decomposition involving pairwise and higher-order constraints. Additionally, a Gromov-Wasserstein learning framework has been proposed by Xu et al. [27] for joint graph matching and embedding vector learning based on dissimilarity measurements between graph pairs.

Usually, graph matching is conducted through embedding-based approaches [26]. Nevertheless, recent works tackled the problem through a network-based method, which employs a cross-graph attention mechanism to compute a similarity score, namely Graph Matching Networks (GMNs) [12]. The method calculates the similarity score jointly, producing more robust and accurate results.

Apart from GM, a graph-based framework for supervised and unsupervised learning called Optimum-Path Forest [17,18,23] obtained notorious popularity in past years due to its successful implementation over a wide range of applications. Among such implementations, Jodas et al. [10] recently proposed the OPFsemble, an OPF-based mechanism to tackle ensemble pruning, with significant results compared to state-of-the-art approaches. In such context, OPF revealed itself as a feasible solution applied to the pruning of classifiers constituting an ensemble model, thus becoming an interesting and promising approach to obtain a performance similar to that of the complete set when selecting a subset of models.

Ensemble pruning techniques can be applied in several ways, such as in classifier selection experiments. Approaches include Reduce-Error Pruning and Kappa Pruning, Complementarity Measure, Margin Distance Minimization models, and Orientation Ordering and Boosting-Based Ordering [15]. Besides, Zyblewski et al. [28] tackled the problem via a clustering-based approach to capturing the models with similar performance and selecting the ones from each produced cluster. Apart from the well-known and established performance in grouping similar data, most clustering algorithms still suffer from the need to define the number of clusters while grouping the input data.

This paper offers a novel ensemble pruning strategy that combines the strength of Optimum-Path Forest and Graph Matching Networks to create a robust solution. Firstly, the method performs a weak classification of several data subsets using an ensemble of supervised OPF-based classifiers whose predictive performance similarities are computed by GMNs. Further, such graphs (classifiers) are clustered according to their similarities using the unsupervised OPF, and the well-known Gaussian Mixture Model learning algorithm determines the importance of each group, which will be used in the prediction phase. Lastly, we select the most representative model per cluster and prune the remaining ones

with similar predictive capabilities. In short, the motivation behind this paper lies in the successful application of three main concepts: (i) using the OPF framework for ensemble and classification in general, (ii) the employment of GMN to expose the similarity among graph-based models, and (iii) the performance of ensemble pruning to extinguish redundancy and improve performance. In this context, the main contributions of this paper are described as follows:

- To propose a novel ensemble pruning method combining Optimum-Path Forest and Graph Matching Networks;
- To evaluate GMNs in the context of OPF graph similarities computation; and
- To foster the literature regarding graph-based approaches and ensemble pruning.

The remainder of this paper is organized as follows. Section 2 presents a literature review concerning the topic of ensemble pruning, and Sect. 3 provides a theoretical background concerning supervised and unsupervised OPF classifiers. Section 4 describes the proposed approach, followed by the methodology provided in Sect. 5. Last but no least, Sects. 6 and 7 state the results and conclusions, respectively.

2 Related Works

To our knowledge, no papers have modeled the problem of ensemble pruning as a graph-matching task. This section briefly reviews some papers on graph matching that we believe are interesting to this work.

In 2015, Lyzinski et al. [14] proposed a graph-matching approach efficient for very large graphs that combine spectral graph embeddings with seeded graph matching using a divide-and-conquer procedure. A couple of years later, a study conducted by Swoboda et al. [24] obtained state-of-the-art results by introducing several Lagrangean relaxations on the graph matching problem optimized using sub-gradient and message passing (dual ascent) mechanisms.

Later on, Lê-Huu et al. [11] formulated the graph matching problem as a non-convex non-separable optimization problem using decomposition of matching constraints, providing a modular and scalable framework that can be instantiated into different variants, namely pair-wise and higher-order constraints. Further, Xu et al. [27] proposed a Gromov-Wasserstein learning framework to jointly match graphs and learn embedding vectors by measuring dissimilarities between pairs of graphs.

Yujia et al. [12] presented a cross-attention-based matching mechanism composed of two main steps: training a Graph Neural Network to learn graph representations in a vector space and using these representations to calculate the similarity between pairs of graphs to detect software vulnerabilities. Similarly, Fey et al. [5] provided a two-stage neural architecture for graph matching that employs localized node embeddings to compute the correspondences between

nodes and uses synchronous message-passing networks to iteratively reach a matching consensus in such likenesses by the nodes' local neighborhoods.

More recently, Guo et al. [7] proposed a semantic histogram-based graph matching method to tackle the problem of visual multi-robot localization. In contrast, Qu et al. [22] introduced the Edge Attention-adaptive Graph Matching network to solve the problem of outliers in practical scenarios. Finally, Lin et al. [13] presented a contrastive learning-based technique called Contrastive Matching with Momentum Distillation to address the problem of noisy correspondence in graphs.

To the best of our knowledge, no additional study has employed the concept of graph-matching networks in the context of ensemble pruning applications. The proposed method is inherently suited for graph-structured data, such as adjacency matrices, which are fundamental to the OPF model, thereby supporting the proposed method. In addition, the method is scalable to larger datasets since the OPF framework has been applied to a wide range of research comprising a variety of data representations. A notable study by Papa et al. [17] revealed the OPF capabilities in large datasets from different domains, including an image dataset with magnetic resonance images of the human brain. This aspect demonstrates the model's ability to handle larger datasets effectively, thus showing the adaptability of the proposed ensemble pruning to larger datasets.

3 Theoretical Background

This section provides the theoretical background regarding Optimum-Path Forest and the main concepts related to graph matching and Graph Matching Networks.

3.1 Optimum-Path Forest

The Optimum-Path Forest framework [17,18,23] encodes input patterns as graph nodes in a high-dimensional space and creates optimum-path trees over these points, which are generated from the training instances and are built to optimize the separation between classes/clusters. Further sections detail the supervised and unsupervised OPF versions.

Supervised Learning with Complete Graph: Let $G = (\mathcal{V}, \mathcal{E}, w)$ be a complete and weighted graph where $\mathcal{V} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ corresponds to the set of vertices (samples), such that $\boldsymbol{x}_i \in \mathbb{R}^D$, and $(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{E}$ denotes an arc from the set of edges $\mathcal{E}$, such that $i \neq j$. Moreover, arcs are weighted by the function $w : \mathcal{E} \rightarrow \Re^+$, which can be a similarity, matching, or distance measure.

Let $C \in \Re^{n \times n}$ be a matrix such that C_{ij} represents the cost of the optimum path between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. The cost C_{ij} is determined by function $f(C_{ij})$, designed to capture the expected separation between classes. Papa et al. [17,18,23] proposed to use the maximum arc-weight along a path, but other functions can also be employed, provided some restrictions are satisfied.

The goal is to find the optimum path $\mathcal{O}_{ij}^*$ between two training instances $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ that minimizes the cost function $f(C_{ij})$ as follows:

$$\mathcal{O}_{ij}^* = \arg\min_{\mathcal{O}_{ij}} f(C_{ij}),\ \forall i, j \in \{1, 2, \ldots, n\}, \tag{1}$$

where $\mathcal{O}_{ij}$ represents a path between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. Notice that several approaches, e.g., Dijkstra's algorithm, dynamic programming, or branch and bound techniques, can be used for this purpose. The optimum-path O^* is selected based on the algorithm and the defined cost function, representing the most efficient route between the instances in terms of class separation. Costs are initialized to 0 for the prototype samples, i.e., connected samples with different labels in a Minimum Spanning Tree computed over $\mathcal{V}$. A large cost is assigned to the remaining samples.

Once the optimum paths are determined, an optimum-path forest is built by finding the minimum-cost path between all pairs of instances within the same class, thus forming trees in the forest. This structure provides an efficient framework for classification purposes.

To classify a new pattern $\boldsymbol{x}_{new}$, the distances between $\boldsymbol{x}_{new}$ and all training instances are calculated. Let $\gamma_{i,new}$ be a cost between samples $\boldsymbol{x}_i$ and $\boldsymbol{x}_{new}$. The classification step aims at solving the following optimization problem:

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}_i} \gamma_{i,new},\ \forall i, j \in \{1, 2, \ldots, n\}. \tag{2}$$

In other words, $\boldsymbol{x}^* \in \mathcal{V}$ denotes the node that offered the minimum cost to $\boldsymbol{x}_{new}$, also assigning its label to the new sample.

Unsupervised Learning: OPF groups training samples in its unsupervised form, connecting them to their k-nearest neighbors with weighted arcs. In this version, nodes are also weighted by a probability density function (PDF) as follows:

$$\rho(\boldsymbol{x}_i) = \frac{1}{\sqrt{2\pi\psi^2}k} \sum_{\forall \boldsymbol{x}_j \in \mathcal{A}_k(\boldsymbol{x}_i)} \exp\left(\frac{-d^2(\boldsymbol{x}_i, \boldsymbol{x}_j)}{2\psi^2}\right), \tag{3}$$

where $\mathcal{A}_k(\boldsymbol{x}_i)$ is the k-neighborhood of sample $\boldsymbol{x}_i$, $\psi = \frac{m_w}{3}$, and m_w is the maximum weight among graph edges. The optimum number of nearest neighbours, k^*, is determined by minimizing the graph cut over $\mathcal{V}$ within the range $1 \leq k\text{max} \leq |\mathcal{V}|$ [23]. The model elects prototypes $\mathcal{P}$ based on maxima of the PDF, and each $\boldsymbol{x}_j$ is assigned the path with the highest minimum density value as follows:

$$\begin{aligned} f_{min}(\langle \boldsymbol{x}_j \rangle) &= \begin{cases} \rho(\boldsymbol{x}_j) & \text{if } \boldsymbol{x}_j \in \mathcal{P} \\ \rho(\boldsymbol{x}_j) - \delta & \text{otherwise,} \end{cases} \\ f_{min}(\langle \phi_{\boldsymbol{x}_i} \cdot \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle \rangle) &= \min\{f_{min}(\phi_{\boldsymbol{x}_i}), \rho(\boldsymbol{x}_j)\}, \end{aligned} \tag{4}$$

where $\phi_{\boldsymbol{x}_i}$ denotes a path starting from a prototype in $\mathcal{P}$ with terminus at sample $\boldsymbol{x}_i$, and δ is a small constant.

3.2 Graph Matching Network

Graph Matching is an existing graph theory and computer science issue. It addresses the analysis and comparison of graphs to identify similarities or correspondences between them. In short, GM aims to check the equivalence between two graphs or discover correspondences between the vertices of different graphs. According to Pavithra et al. [21], GM refers to the correspondence of a collection of edges that do not share common vertices in a graph. this problem is tackled by modelling a pair of input graphs in a vector embedding space that facilitates the subsequent similarity assessment. Usually, such approaches involve taking the graph embedding representations independently for the graphs' equivalency estimation. However, Li et al. [12] proposed graph-matching networks for simultaneous graph similarity learning, introducing a cross-graph attention mechanism. The model takes pairs of graphs as input, using an encoder network with a multilayer perceptron architecture to obtain initial embedding representations for nodes and edges. Additionally, GMN adjusts the node update module in each propagation layer, considering aggregated messages and a cross-graph correspondence vector for enhanced node association between graphs. Unlike traditional methods that compute similarity scores separately, their approach calculates scores jointly for graph pairs, promoting robustness at a slightly higher computational cost.

Formally speaking, let (G_1, G_2) be the pair of two graphs G_1 and G_2, and $\boldsymbol{h}_{G_1}$ and $\boldsymbol{h}_{G_2}$ the graph-level representation after applying the propagation and aggregation functions on both graphs. Both functions represent Multilayer Perceptron (MLP) networks. Their purpose is to map the nodes and edges of the graph into vector representations using the graph's adjacency matrix. During the learning process, the similarity for the graph pair is computed using a pairwise loss function represented as follows:

$$L_{pair} = \mathbb{E}_{(G_1,G_2,t)}[max\{0, \gamma - t(1 - d(G_1, G_2))\}], \tag{5}$$

where $t \in \{-1, 1\}$ represents the label indicating similar (1) or dissimilar (-1) graphs, γ is the margin parameter, and $d(G_1, G_2) = \|\boldsymbol{h}_{G_1} - \boldsymbol{h}_{G_2}\|^2$, i.e., the Euclidean distance between the graphs in the vector space. This loss function promotes $d(G_1, G_2) < 1-\gamma$ when the pair is similar ($t = 1$), and $d(G_1, G_2) > 1+\gamma$ when $t = -1$. Subsequently, the loss is optimized using gradient descent-based methods.

4 Proposed Approach

The proposed methodology addresses classifier pruning into an eight-step model, illustrated in Fig. 1. The first five steps comprise the training phase, while the remaining three compose the test (prediction) phase. Such steps are detailed below:

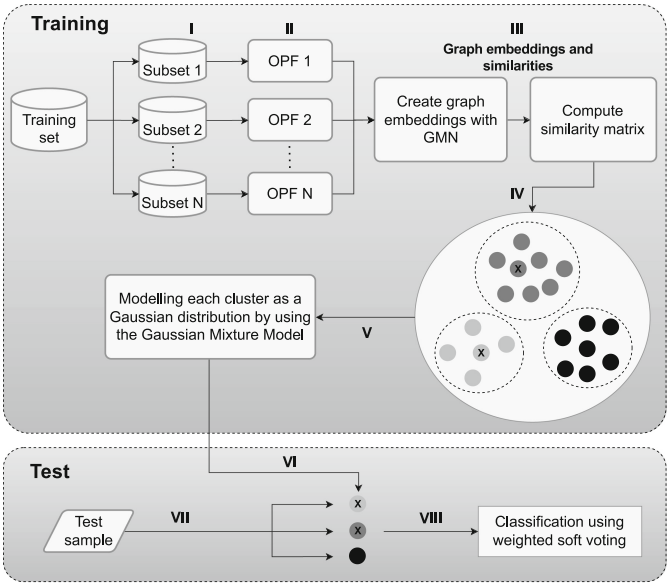


Fig. 1. Pipeline of the proposed ensemble pruning based on Graph Matching Networks and Optimum-Path Forest.

I. The first step splits the training set into N subsets using bootstrap aggregation (bagging) from random sampling with replacement. This approach is similar to the one employed by the Random Forest model, i.e., it provides each classifier with a different subset of the training set to allow for variation reduction, generalization, and robustness.
II. The second step trains N supervised OPF classifiers to initialize the ensemble. In this stage, we allocate each training subset to its corresponding OPF instance, setting the stage for the subsequent training process. This deliberate assignment ensures that each OPF instance receives the appropriate training data, facilitating the development of distinct classifiers that collectively contribute to the overall ensemble model diversity. The tailored nature of this step emphasizes the precision and individualization in producing each OPF classifier, an essential preparation for the subsequent stages of our methodology.
III. The third step applies GMNs to compute the graph similarities among the trained OPF instances within the ensemble model. The approach entails creating a graph representation after training the OPF model with the training data. First, we apply the supervised OPF to establish the graph structure using the features of the training data. This process is an inherent aspect of the OPF model, which produces a graph with the edges representing the distance of the adjacent training samples. Next, we create an adjacency matrix based on this graph structure and then represent the nodes and edges as feature vectors. Similar to the study of Li et al. [12],

we set the corresponding nodes' and edges' features to a constant of 1 s so that we can represent only the graph's adjacency structure and node distribution. Then, the nodes and edges features are provided in pairs to the GMN model to compute their embeddings and similarity. We create a pair between each OPF classifier generated in the previous step to feed a GMN model to capture their embedding representations. The processed embeddings are then compared via a pairwise similarity function, producing a similarity value for the analyzed graph pair. Formally speaking, let $\mathcal{S}$ be the set of supervised OPF instances composing the ensemble model. The process to yield a set of graph pairs from $\mathcal{S}$ is achieved as follows:

$$\mathcal{G}_p = \{(G_i, G_j) \mid G_i, G_j \in \mathcal{S} \wedge i \neq j \text{ for } i = 1, 2, \ldots, N\}, \tag{6}$$

where $\mathcal{G}_p$ stands for the resulting set of graph pairs which is subsequently fed into the GMN model. This process yields a similarity matrix $M \in \Re^{N \times N}$, where a cell m_{ij} corresponds to the similarity value between graphs G_i and G_j.

IV. This step involves clustering similar OPF instances based on the similarities computed by GMN. We used the unsupervised OPF to group the supervised OPF instances using the similarity matrix $\mathcal{G}_p$ obtained in the previous step.

V. The contribution of each cluster is derived from a probability density function modeled from a Gaussian distribution computed from each group. This process involves a Gaussian Mixture Model (GMM) initialized with the mean vector and covariance matrix of the clusters calculated in the previous step. The GMM is trained through an iterative process on the initial mean and covariance matrix with a specified number of iterations to find the weights of each distribution. As we have a reasonable estimative of both the mean and covariance matrix, we modified the Expectation-Maximization algorithm to learn the weights of each Gaussian distribution only. Notice these weights are used in the prediction phase only.

VI. As a critical measure to enhance diversity, refine generalization capabilities, and facilitate accurate predictions during inference, the proposed pruning strategy removes all classifiers from each group but the prototypes (nodes marked with 'X' in Fig. 1). By isolating these representative instances, we eliminate redundant or less informative classifiers.

VII. In the prediction step, each individually selected model is presented with a new test sample. This involves providing the input test data to each supervised OPF model, allowing them to generate individual predictions based on their learned patterns and characteristics. The prediction step not only involves obtaining individual model predictions for a given test sample but also considers the weight of each classifier (step V) in the ensemble, resulting in a refined final prediction that leverages the strengths of each model.

VIII. The final step combines the provided individual predictions into a final classification result using soft voting. The weights associated with the selected classifiers are assigned to the respective individual predictions, representing each classifier's relative importance or contribution within the ensemble model.

Let $\mathcal{Y} = \{\omega_1, \omega_2, \ldots, \omega_c\}$ be a set of labels and $\boldsymbol{z}$ be a test sample. Besides, let $\lambda : \mathcal{V} \rightarrow \mathcal{Y}$ be labeling function and $P(\boldsymbol{z}) = \{\boldsymbol{p}_1(\boldsymbol{z}), \boldsymbol{p}_2(\boldsymbol{z}), \ldots, \boldsymbol{p}_c(\boldsymbol{z})\}$ be a set of class probabilities concerning $\boldsymbol{z}$ computed by the classifers comprising the final ensemble, such that $\boldsymbol{p}_i(\boldsymbol{z}) \in \Re^m$ and $m < N$ stands for the number of selected classifiers after pruning. Our soft voting strategy calculates the class probabilities assigned to sample $\boldsymbol{x}$ as follows:

$$p_i(\boldsymbol{z}) = \sum_{j=1}^{m} w_j p_{ij}(\boldsymbol{z}), \forall i \in \{1, 2, \ldots, k\}, \tag{7}$$

where w_j stands for the weight of the j-th classifier (step V), and p_{ij} denotes the probability of the i-th class computed by the j-th supervised OPF classifier.

It is worth noting the standard OPF does not cope with soft classification. Although past works attempted to deal with such a shortcoming [3,4], they either work with binary classification problems or require fine-tuning additional parameters. We propose the following strategy to compute the j-th classifier's probability concerning the i-th class:

$$p_{ij}(\boldsymbol{z}) = softmax\{-\phi(\boldsymbol{b}, \boldsymbol{z}) | \boldsymbol{b} \in Q(\boldsymbol{z}) \wedge \lambda(b) = i\}, \tag{8}$$

where $\phi(\boldsymbol{b}, \boldsymbol{z})$ stands for the cost node $\boldsymbol{b}$ offered to testing sample $\boldsymbol{z}$, $Q(\boldsymbol{z})$ outputs the training sample that conquered $\boldsymbol{z}$. In a nutshell, $p_{ij}(\boldsymbol{z})$ stands for the set of class probabilities assigned by a training sample $\boldsymbol{b}$ to $\boldsymbol{z}$, which are computed using a $softmax$ over the optimum-path costs assigned by a sample $\boldsymbol{b}$.

Before applying the $softmax$ function, we first normalize all costs $\phi(\boldsymbol{b}, \boldsymbol{z})$ using a minimum-maximum normalization to restrict them within the range of 0 and 1. Then, we take the negative of the normalized costs, for higher probabilities encode smaller normalized costs. This procedure generates probabilities for each class $\omega_i \in \mathcal{Y}$ while ensuring their sum equals 1, $i = 1, 2, \ldots, c$.

The final class prediction is formulated as follows:

$$y_{\boldsymbol{z}} = \arg\max_i (p_i(\boldsymbol{z})), \forall i \in \{1, 2, ..., c\}, \tag{9}$$

where $y_{\boldsymbol{z}}$ represents the label for which the weighted probability is the highest.

We also employed two straightforward variants when applying the prediction step by the supervised OPF classifiers selected from the pruning procedure. Such additional strategies involve hard and soft voting computed by applying equal weights to the class labels' probabilities computed by Eq. 7. In the latter variant, the classifiers' weights are equally assigned by taking the ratio of the number of classifiers selected after pruning, thus ensuring that each classifier contributes similarly to the final prediction.

5 Methodology

This section describes the methodology concerning the employed datasets and experimental setup.

5.1 Dataset

The experiments were conducted over a set of 15 datasets available in the UCI machine learning repository[1]: Speaker Accent Recognition (D_1), Cervical Cancer (D_2), Breast Cancer Wisconsin (D_3), Breast Cancer Wisconsin - Original (D_4), Breast Cancer Wisconsin - Prognostic (D_5), Forest Type Mapping (D_6), Glass Identification (D_7), Indian Liver Patient Dataset (D_8), Iris (D_9), Mammographic Mass (D_{10}), Diabetic Retinopathy Debrecen (D_{11}), Primary Tumor (D_{12}), Seismic Bumps (D_{13}), Statlog - Vehicle Silhouettes (D_{14}), and Vertebral Column (D_{15}). The datasets include binary and multiclass labels, exhibiting diversity in features. The datasets underwent a preprocessing step to address missing values, with each feature's missing data provided with the average value. Subsequently, standardization was applied by scaling the data assuming a Gaussian distribution.

5.2 Experimental Setup

Each dataset was split into 20 folds by employing proportions of 70%, 15%, and 15% for training, validation, and testing of the models, respectively, assuming a cross-validation strategy for performance assessment.

Unsupervised OPF for graph matching-based pruning was compared to similar clustering-based algorithms, namely k-Means and agglomerative clustering, and the example margin and ensemble diversity proposed by Guo et al. [6], thus reaching a fair comparison with the state-of-the-art techniques for ensemble pruning. The baseline algorithms employed in our study are provided by PyPruning, a comprehensive framework offering diverse algorithms that serve as foundational methods for our comparative and experimental analyses. The framework is accessible through the GitHub repository[2].

All pruning strategies, including our proposed method, were assessed across three ensemble setups initialized with 10, 30, and 50 supervised OPF instances. Such diversity assessment allowed for a detailed exploration of each pruning method's impact, enhancing the robustness of the comparative analysis. Additionally, we limited the iterations to 100 concerning the Expectation-Maximization approach to determine classifiers' weights using the Gaussian Mixture Model.

In the baseline setup, we used a pruning strategy from the PyPruning framework, selecting a subset of representative classifiers based on each pruning algorithm function. The chosen number of classifiers, denoted as r, follows the criteria set by each baseline pruning algorithm. Table 1 outlines the selection criteria

[1] https://archive.ics.uci.edu.
[2] https://github.com/sbuschjaeger/PyPruning.

for each ensemble arrangement. The values of r increase proportionally but are capped at half of the classifiers used to initialize the ensemble model.

Table 1. Selection criteria for each ensemble configuration.

Ensemble setup	Subset for r
10 classifiers	5
30 classifiers	[5,10,15]
50 classifiers	[10,15,25]

However, the above hyperparameter optimization is partially dispensable when operating the unsupervised OPF classifier[3]. Nevertheless, as outlined in Sect. 3.1, selecting the appropriate k-max value becomes essential to ensure the accurate grouping of analogous samples when employing the unsupervised OPF algorithm. Therefore, the k-max parameter was properly optimized to select the best value whose F1-score is the highest when employing the validation set through each clustering attempt using the unsupervised OPF algorithm. This process is repeated in each of the 20 data split configurations.

GMN training followed Li et al.'s [12] setup using their GitHub code[4]. Initially, a synthetic training set with diverse graph setups was created. Each graph was initialized with $n = 20$ nodes and edge probability $p = 0.2$. For each produced graph G, a positive sample G_1 was then formed by replacing k_p edges with new edges in G, while a negative sample G_2 was created by substituting k_n edges from G, with $k_p < k_n$. This resulted in pairs (G_1, G_2) and (G_1, G_3). Following Li et al.'s strategy [12], k_p and k_n were both set to 1 and 2, respectively. Subsequently, the model underwent 500,000 training steps with a batch size of 20 samples and a learning rate set to 0.0001. In addition, a 128-sized vector for OPF graph embedding was utilized. The learning process aimed to minimize graph similarity errors using a pairwise loss function based on Euclidean distance in the graph's vector space.

Regarding the comparison analysis, we employed the F1-score to assess the efficacy of each pruning method used in this work. In addition, we evaluated the optimal value for r, i.e., the number of selected classifiers that showed the highest average F1-score for each baseline pruning algorithm over the testing set. Furthermore, we employed the Wilcoxon signed-rank test with a significance of 0.05 (5%) to inspect statistical differences between the proposed pruning strategy's outcomes and the ensemble model's results obtained without applying any pruning approach. Similarly, statistical differences among the pruning algorithms' results were assessed using the Friedman test with the same significance level of 5%.

[3] Although unsupervised OPF figures one hyperparameter only, i.e., k-max, it can learn the number of clusters on-the-fly.

[4] https://github.com/Lin-Yijie/Graph-Matching-Networks.

6 Results and Discussions

This section discusses the comparative analysis involving the F1-score values yielded by each pruning algorithm. The highest score values are presented in bold, while statistical differences are underscored when assessing the comparative analysis between the pruning results and the outcomes when no pruning strategy is applied to the ensemble model.

Table 2 compares the GMN pruning's results with those achieved when no pruning strategy is applied to the ensemble model. Notice the statistically similar scores yielded by most of the tested datasets, particularly for the 50 classifiers configuration. Such comparative analysis implies that the proposed strategy attained similar behavior as if the ensemble model were employed with all OPF instances during the inference step.

Table 2. Average F1-scores against the baseline, i.e., no pruning.

	10 classifiers		30 classifiers		50 classifiers	
	No pruning	Pruning	No pruning	Pruning	No pruning	Pruning
D_1	**0.806±0.059**	0.777±0.057	**0.819±0.040**	0.799±0.041	**0.819±0.041**	0.811±0.043
D_2	**0.944±0.020**	0.936±0.026	0.943±0.023	**0.944±0.027**	**0.944±0.020**	0.943±0.021
D_3	0.952±0.017	**0.954±0.015**	**0.954±0.018**	0.947±0.021	**0.954±0.017**	0.950±0.020
D_4	0.823±0.046	**0.823±0.049**	**0.824±0.045**	0.810±0.042	**0.827±0.044**	0.816±0.046
D_5	0.652±0.088	**0.664±0.102**	**0.671±0.089**	0.660±0.093	**0.665±0.084**	0.663±0.089
D_6	0.673±0.047	**0.675±0.046**	**0.674±0.045**	0.669±0.058	**0.676±0.047**	0.670±0.047
D_7	**0.937±0.050**	0.935±0.052	0.941±0.046	**0.946±0.048**	0.943±0.046	**0.946±0.043**
D_8	**0.762±0.038**	0.734±0.043	**0.772±0.036**	0.760±0.039	**0.773±0.034**	0.766±0.037
D_9	**0.600±0.028**	0.588±0.032	**0.606±0.026**	0.600±0.022	0.607±0.028	**0.613±0.032**
D_{10}	**0.949±0.021**	0.946±0.020	**0.951±0.017**	0.949±0.018	0.951±0.018	**0.951±0.017**
D_{11}	**0.312±0.052**	0.310±0.054	**0.320±0.050**	0.306±0.048	**0.328±0.060**	0.325±0.053
D_{12}	**0.707±0.113**	0.681±0.109	**0.721±0.111**	0.712±0.088	**0.722±0.108**	0.704±0.099
D_{13}	**0.906±0.019**	0.900±0.016	**0.909±0.018**	0.903±0.015	**0.908±0.019**	0.904±0.016
D_{14}	**0.694±0.036**	0.694±0.035	**0.696±0.041**	0.685±0.033	**0.697±0.043**	0.693±0.034
D_{15}	**0.819±0.046**	0.811±0.047	**0.827±0.044**	0.820±0.040	**0.829±0.044**	0.816±0.039

Underscored values stand for statistical difference at $p = 0.05$.

Table 3 shows the average F1-score values computed from the 20 splits applied to each dataset. The pruning strategy by the GMM version attained comparable results with the counterpart versions applying hard voting (HV) and soft voting (SV) with equally distributed weights. In most cases, the soft voting approach improves the prediction capability by providing higher average scores than the hard voting of the class label predictions. Notice the average scores provided by the Accent, Forest types, and Iris datasets, to cite a few, which achieved the highest average score values with the soft voting strategies compared to the hard voting approach. Regarding the comparative analysis with the baseline algorithms, one can notice the competitive results achieved by the proposed approach regarding the comparative assessment with corresponding average values

provided by k-Means, agglomerative clustering, and the pruning with the margin and diversity strategy. OPF-GMN-based pruning provided results similar to those of the baseline algorithms, although the latter approaches outperformed the OPF-GMN in most cases. However, since the pruning approach relies on the prototype selection from the clusters produced by the unsupervised OPF algorithm, it is worth noting that there is no need to determine the expected number of classifiers after applying the pruning procedure with the proposed method.

In addition, we conducted a further analysis to determine the minimum number of OPF classifiers required to achieve maximum accuracy in specific classification scenarios. We assessed the minimum, maximum, and average amount of OPF instances selected from running the 20 folds on each employed dataset. Table 4 provides such results for the pruning strategy harnessed by the GMN model. Regarding the minimum selection, all ensemble configurations provided a minimum set of one OPF instance for further inference of the sample class label. Moreover, an interesting point pertains to the maximum number of classifiers each ensemble configuration provides, constituting nearly half of the instances initialized for the ensemble model.

Following the same analysis, we assessed the number of classifiers that showcase the highest average scores for each baseline pruning algorithm across the 20-fold execution. Table 5 provides such comparative assessment. Similar to the GMN-based pruning strategy, the methods lead to almost half the number of OPF instances assembled for the ensemble setup. Regardless of which pruning method is used, such results reveal the trend toward selecting more classifiers as the number of models increases in the ensemble initialization.

The experiments conducted in this study utilized datasets from the UCI repository, which are relatively small in size. While these datasets have demonstrated our method's effectiveness, the proposed ensemble pruning can be adapted to larger, more structured datasets. Our method is inherently scalable and adaptable to larger datasets and structured data, such as those defined by adjacency matrices, an intrinsic aspect of the OPF model, thus favoring the proposed approach.

As a last discussion, the pruning technique is widely used in machine learning applications with the main objective of reducing the computational cost of models, making them more efficient in processing time and resource use. It is worth noting that in some cases, pruning offers statistically similar results with fewer classifiers in the ensemble, as shown by our results. Although pruning may result in better performance for specific techniques and methods, it is not guaranteed in all cases. In fact, such a problem can occur when the underlying classifiers perform poorly during the classification process. However, this issue is not exclusive to the OPF classifier; it can arise with any classifier. The source of this problem lies in the pruning process itself, making it an inherent challenge across various classification models, thus affecting the ensemble pruning performance. This behavior can be explained by the particular nature of the data and models used, where the additional complexity without the pruning technique allows for better capture of the nuances present in the data, resulting in superior performance.

Table 3. Average F1-Score provided by each baseline.

10 classifiers						
	Ours HV	Ours SV	Ours GMM	***k*-Means**	Agglom.	Rank I.M.
D_1	0.775±0.058	0.776±0.056	0.777±0.057	**0.792±0.059**	0.785±0.062	0.775±0.046
D_2	0.934±0.025	0.937±0.027	0.936±0.026	0.940±0.024	0.941±0.025	**0.942±0.023**
D_3	0.951±0.019	0.954±0.016	**0.954±0.015**	0.949±0.014	0.949±0.014	0.951±0.018
D_4	0.800±0.046	0.821±0.049	**0.823±0.049**	0.817±0.040	0.822±0.045	0.815±0.049
D_5	**0.672±0.098**	0.662±0.099	0.664±0.102	0.641±0.083	0.647±0.093	0.649±0.095
D_6	0.665±0.042	0.675±0.043	0.675±0.046	0.669±0.052	0.673±0.055	**0.677±0.049**
D_7	0.932±0.057	0.935±0.050	0.935±0.052	0.937±0.048	**0.939±0.046**	0.932±0.051
D_8	0.736±0.033	0.737±0.045	0.734±0.043	**0.739±0.047**	0.738±0.047	0.723±0.041
D_9	0.587±0.031	0.589±0.029	0.588±0.032	0.604±0.026	0.604±0.028	**0.604±0.030**
D_{10}	0.941±0.020	0.946±0.019	0.946±0.020	0.947±0.019	**0.949±0.020**	0.945±0.019
D_{11}	0.293±0.055	0.314±0.057	0.310±0.054	0.310±0.053	**0.319±0.051**	0.302±0.050
D_{12}	0.689±0.101	0.684±0.112	0.681±0.109	**0.696±0.110**	0.690±0.109	0.688±0.093
D_{13}	0.902±0.016	0.899±0.017	0.900±0.016	0.904±0.019	0.905±0.019	**0.905±0.018**
D_{14}	0.694±0.038	**0.695±0.037**	0.694±0.035	0.693±0.037	0.694±0.038	0.691±0.038
D_{15}	0.803±0.045	0.810±0.045	0.811±0.047	0.807±0.052	0.811±0.052	**0.817±0.052**
30 classifiers						
	Ours HV	Ours SV	Ours GMM	***k*-Means**	Agglom.	Rank I.M.
D_1	0.797±0.041	0.810±0.042	0.799±0.041	**0.819±0.041**	0.809±0.043	0.802±0.047
D_2	0.939±0.021	0.944±0.025	0.944±0.027	0.943±0.025	0.944±0.022	**0.946±0.021**
D_3	0.944±0.021	0.945±0.020	0.947±0.021	0.955±0.018	0.955±0.017	**0.956±0.014**
D_4	0.799±0.044	0.810±0.042	0.810±0.042	0.823±0.044	**0.826±0.039**	0.813±0.048
D_5	0.654±0.079	0.654±0.090	0.660±0.093	0.667±0.086	**0.667±0.102**	0.659±0.088
D_6	0.660±0.052	0.671±0.056	0.669±0.058	0.676±0.047	**0.681±0.044**	0.674±0.046
D_7	0.939±0.049	**0.946±0.048**	0.946±0.048	0.939±0.046	0.941±0.046	0.946±0.044
D_8	0.751±0.037	0.758±0.041	0.760±0.039	**0.763±0.033**	0.760±0.035	0.757±0.032
D_9	0.592±0.028	0.599±0.023	0.600±0.022	0.604±0.030	0.608±0.031	**0.609±0.028**
D_{10}	0.944±0.019	0.950±0.017	0.949±0.018	0.952±0.018	**0.952±0.019**	0.948±0.019
D_{11}	0.292±0.050	0.301±0.050	0.306±0.048	0.318±0.057	**0.322±0.050**	0.312±0.050
D_{12}	**0.721±0.099**	0.708±0.090	0.712±0.088	0.716±0.105	0.716±0.115	0.693±0.096
D_{13}	0.903±0.017	0.903±0.016	0.903±0.015	**0.909±0.017**	0.907±0.019	0.907±0.018
D_{14}	0.692±0.032	0.684±0.034	0.685±0.033	**0.700±0.042**	0.699±0.032	0.694±0.042
D_{15}	0.806±0.038	0.821±0.042	0.820±0.040	0.823±0.041	0.823±0.042	**0.824±0.049**
50 classifiers						
	Ours HV	Ours SV	Ours GMM	***k*-Means**	Agglom.	Rank I.M.
D_1	0.798±0.051	0.808±0.041	0.811±0.043	**0.819±0.045**	0.817±0.046	0.806±0.045
D_2	0.936±0.023	0.943±0.021	0.943±0.021	0.942±0.025	**0.945±0.025**	0.945±0.020
D_3	0.946±0.018	0.950±0.020	0.950±0.020	0.954±0.017	**0.956±0.015**	0.953±0.017
D_4	0.795±0.044	0.817±0.039	0.816±0.046	0.826±0.040	**0.828±0.042**	0.818±0.045
D_5	**0.679±0.096**	0.666±0.088	0.663±0.089	0.663±0.092	0.671±0.095	0.672±0.105
D_6	0.665±0.039	0.672±0.044	0.670±0.047	0.678±0.052	0.677±0.048	**0.680±0.055**
D_7	0.936±0.051	0.946±0.043	0.946±0.043	0.939±0.047	0.943±0.046	**0.950±0.040**
D_8	0.748±0.030	0.763±0.038	0.766±0.037	**0.770±0.039**	0.768±0.031	0.760±0.034
D_9	0.602±0.041	**0.614±0.032**	0.613±0.032	0.610±0.032	0.608±0.029	0.611±0.029
D_{10}	0.942±0.018	0.951±0.017	0.951±0.017	**0.952±0.018**	0.952±0.018	0.948±0.016
D_{11}	0.304±0.052	**0.328±0.051**	0.325±0.053	0.325±0.052	0.328±0.055	0.326±0.054
D_{12}	0.715±0.105	0.703±0.098	0.704±0.099	0.709±0.113	**0.720±0.106**	0.709±0.095
D_{13}	0.902±0.017	0.904±0.017	0.904±0.016	0.907±0.018	**0.908±0.018**	0.908±0.019
D_{14}	0.699±0.033	0.694±0.035	0.693±0.034	0.700±0.036	**0.701±0.032**	0.700±0.041
D_{15}	0.810±0.038	0.816±0.039	0.816±0.039	**0.829±0.038**	0.824±0.038	0.826±0.043

† Underscored values stand for statistical difference computed by the Friedman test with $p = 0.05$.
*HV = Hard Voting; SV = Soft Voting; GMM = Gaussian Mixture Model; Agglom. = Agglomerative Clustering; I.M. = Individual Margin.

Table 4. Minimum, maximum and average number of classifiers selected after the pruning procedure.

	10 classifiers			30 classifiers			50 classifiers		
	Min.	Max.	Avg.	Min.	Max.	Avg.	Min.	Max.	Avg.
D_1	1	6	3.7±1.6	1	15	7.3±4.6	2	23	12.3±7.2
D_2	1	5	2.7±1.5	1	15	6.5±5.0	3	25	16.4±8.1
D_3	1	6	3.4±1.7	1	16	7.8±5.8	1	24	14.3±7.4
D_4	1	6	3.5±1.6	1	16	5.8±5.7	1	25	12.2±8.8
D_5	1	6	3.1±1.6	1	15	5.7±5.3	1	26	10.1±8.0
D_6	1	7	3.5±1.8	1	15	5.5±4.4	2	25	12.7±9.0
D_7	1	6	3.6±1.2	1	18	10.7±5.0	1	25	15.1±8.4
D_8	1	6	3.8±1.4	1	15	8.7±4.6	1	26	14.2±8.1
D_9	1	6	2.7±1.6	1	13	4.8±4.7	1	25	11.4±8.2
D_{10}	1	5	3.0±1.6	1	16	8.6±5.5	1	26	13.7±8.5
D_{11}	1	6	2.9±1.5	1	14	7.0±4.9	2	24	12.8±7.8
D_{12}	1	6	3.0±1.7	1	15	7.2±4.8	1	24	11.0±8.8
D_{13}	1	5	2.9±1.3	1	16	5.8±4.5	2	24	8.2±6.8
D_{14}	1	7	3.1±1.9	1	15	6.2±4.7	1	25	12.5±7.5
D_{15}	1	5	3.6±1.1	1	16	8.8±5.2	3	24	15.4±8.2

Table 5. Number of estimators selected for each baseline.

	10 classifiers			30 classifiers			50 classifiers		
	k-Means	Agglom.	Rank I.M.	k-Means	Agglom.	Rank I.M.	k-Means	Agglom.	Rank I.M.
D_1	$(r=5)$	$(r=5)$	$(r=5)$	$(r=15)$	$(r=15)$	$(r=15)$	$(r=25)$	$(r=25)$	$(r=25)$
D_2	$(r=5)$	$(r=5)$	$(r=5)$	$(r=15)$	$(r=5)$	$(r=15)$	$(r=25)$	$(r=10)$	$(r=25)$
D_3	$(r=5)$	$(r=5)$	$(r=5)$	$(r=15)$	$(r=15)$	$(r=10)$	$(r=25)$	$(r=15)$	$(r=10)$
D_4	$(r=5)$	$(r=5)$	$(r=5)$	$(r=15)$	$(r=10)$	$(r=15)$	$(r=25)$	$(r=25)$	$(r=25)$
D_5	$(r=5)$	$(r=5)$	$(r=5)$	$(r=15)$	$(r=10)$	$(r=15)$	$(r=10)$	$(r=10)$	$(r=15)$
D_6	$(r=5)$	$(r=5)$	$(r=5)$	$(r=15)$	$(r=5)$	$(r=15)$	$(r=10)$	$(r=25)$	$(r=10)$
D_7	$(r=5)$	$(r=5)$	$(r=5)$	$(r=5)$	$(r=10)$	$(r=10)$	$(r=15)$	$(r=10)$	$(r=10)$
D_8	$(r=5)$	$(r=5)$	$(r=5)$	$(r=15)$	$(r=15)$	$(r=15)$	$(r=15)$	$(r=25)$	$(r=25)$
D_9	$(r=5)$	$(r=5)$	$(r=5)$	$(r=5)$	$(r=15)$	$(r=5)$	$(r=15)$	$(r=25)$	$(r=25)$
D_{10}	$(r=5)$	$(r=5)$	$(r=5)$	$(r=10)$	$(r=10)$	$(r=5)$	$(r=25)$	$(r=10)$	$(r=25)$
D_{11}	$(r=5)$	$(r=5)$	$(r=5)$	$(r=5)$	$(r=15)$	$(r=15)$	$(r=25)$	$(r=25)$	$(r=15)$
D_{12}	$(r=5)$	$(r=5)$	$(r=5)$	$(r=15)$	$(r=15)$	$(r=10)$	$(r=25)$	$(r=25)$	$(r=25)$
D_{13}	$(r=5)$	$(r=5)$	$(r=5)$	$(r=15)$	$(r=10)$	$(r=15)$	$(r=25)$	$(r=25)$	$(r=25)$
D_{14}	$(r=5)$	$(r=5)$	$(r=5)$	$(r=15)$	$(r=5)$	$(r=10)$	$(r=25)$	$(r=10)$	$(r=15)$
D_{15}	$(r=5)$	$(r=5)$	$(r=5)$	$(r=10)$	$(r=10)$	$(r=10)$	$(r=25)$	$(r=15)$	$(r=25)$

However, applying the pruning technique was mainly motivated by reducing the computational cost and not necessarily by the expectation of achieving state-of-the-art results (SOTA). Thus, the variation in observed results reflects the complexity and adaptability of machine learning techniques to different scenarios and data sets, highlighting that the decision to use pruning must be based on a careful analysis of the specific needs of each application.

7 Conclusions

This paper presented a novel ensemble pruning approach integrating Graph Matching Networks and Optimum-Path Forest classifiers. Such a design allows a suitable application to graph matching problems owing to the model's intrinsic graph-based structure. Combining a reduced and diverse ensemble of OPF classifiers was the initial hypothesis to achieve similar predictive performance with comparable results to the entire ensemble model but with a lower computational cost. In addition, the proposed method was designed to prevent the necessity of specifying the required number of models to compose the final ensemble. The results revealed comparable performance with statistical similarities when no pruning is applied to the ensemble model and competitive predictive capabilities with state-of-the-art ensemble pruning methods. Furthermore, it is important to note that this work focuses only on classic machine learning models, particularly the Optimal Path-Forest model, rather than deep learning techniques.

Regarding future work, we aim to extend this work by replacing the softmax-based strategy used to perform the soft classification and evaluate the Probabilistic OPF for the task. Although our study employed small datasets, which are valuable for benchmarking and pose unique challenges like overfitting and data sparsity, future work will involve extensive testing on larger datasets for further validation and enhancing our method's performance. In addition, OPF was not designed for regression tasks, so we did not address regression in this study. However, we plan to explore regression tasks once an OPF regressor becomes available.

Acknowledgements. The authors are grateful to the Brazilian National Council for Scientific and Technological Development (CNPq) grants 308529/2021-9 and 400756/2024-2, to the São Paulo Research Foundation (FAPESP) grants 2013/07375-0, 2018/25225-9, 2019/07665- 4, 2023/14427-8, 2023/10823-6, 2023/03726-4, 2023/01374-3, and 2023/14354-0, to Unesp-IEPe-RC-#06/2023 PROPe grant, and the Petrobrás Brazil grant #2017/00285-6 for their financial support.

References

1. Biggs, N., Lloyd, E.K., Wilson, R.J.: Graph Theory, pp. 1736–1936. Oxford University Press, Oxford (1986)
2. Caetano, T.S., McAuley, J.J., Cheng, L., Le, Q.V., Smola, A.J.: Learning graph matching. IEEE Trans. Pattern Anal. Mach. Intell. **31**(6), 1048–1058 (2009)
3. Fernandes, S.E.N., Passos, L.A., Jodas, D.S., Akio, M., de Souza, A.N., Papa, J.P.: A multi-class probabilistic optimum-path forest. In: Proceedings of the 18th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2023), vol. 5: VISAPP, pp. 361–368. INSTICC, SciTePress (2023)
4. Fernandes, S.E., Pereira, D.R., Ramos, C.C., Souza, A.N., Gastaldello, D.S., Papa, J.P.: A probabilistic optimum-path forest classifier for non-technical losses detection. IEEE Trans. Smart Grid **10**(3), 3226–3235 (2018)

5. Fey, M., Lenssen, J.E., Morris, C., Masci, J., Kriege, N.M.: Deep graph matching consensus. arXiv preprint arXiv:2001.09621 (2020)
6. Guo, H., Liu, H., Li, R., Wu, C., Guo, Y., Xu, M.: Margin & diversity based ordering ensemble pruning. Neurocomputing **275**, 237–246 (2018)
7. Guo, X., Hu, J., Chen, J., Deng, F., Lam, T.L.: Semantic histogram based graph matching for real-time multi-robot global localization in large scale environment. IEEE Rob. Autom. Lett. **6**(4), 8349–8356 (2021)
8. Jiang, Z., Hu, X., Gao, S.: A parallel ford-fulkerson algorithm for maximum flow problem. In: Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA), p. 70. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp) (2013)
9. Jodas, D.S., Passos, L.A., Adeel, A., Papa, J.P.: PL-kNN: a parameterless nearest neighbors classifier. In: 2022 29th International Conference on Systems, Signals and Image Processing (IWSSIP), pp. 1–4. IEEE (2022)
10. Jodas, D.S., Passos, L.A., Rodrigues, D., Lucas, T.J., Da Costa, K.A.P., Papa, J.P.: OPFsemble: an ensemble pruning approach via optimum-path forest. In: 2023 30th International Conference on Systems, Signals and Image Processing (IWSSIP), pp. 1–5. IEEE (2023)
11. Lê-Huu, D.K., Paragios, N.: Alternating direction graph matching. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4914–4922. IEEE (2017)
12. Li, Y., Gu, C., Dullien, T., Vinyals, O., Kohli, P.: Graph matching networks for learning the similarity of graph structured objects. In: International Conference on Machine Learning, pp. 3835–3845. PMLR (2019)
13. Lin, Y., Yang, M., Yu, J., Hu, P., Zhang, C., Peng, X.: Graph matching with bi-level noisy correspondence. In: 2023 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 23305–23314. IEEE Computer Society, Los Alamitos (2023)
14. Lyzinski, V., et al.: Spectral clustering for divide-and-conquer graph matching. Parallel Comput. **47**, 70–87 (2015)
15. Martínez-Muñoz, G., Hernández-Lobato, D., Suárez, A.: An analysis of ensemble pruning techniques based on ordered aggregation. IEEE Trans. Pattern Anal. Mach. Intell. **31**(2), 245–259 (2008)
16. Mei-Ko, K.: Graphic programming using odd or even points. Chin. Math. **1**, 237–277 (1962)
17. Papa, J.P., Falcão, A.X., Albuquerque, V.H.C., Tavares, J.M.R.S.: Efficient supervised optimum-path forest classification for large datasets. Pattern Recogn. **45**(1), 512–520 (2012)
18. Papa, J.P., Falcão, A.X., Suzuki, C.T.N.: Supervised pattern classification based on optimum-path forest. Int. J. Imaging Syst. Technol. **19**(2), 120–131 (2009)
19. Passos, L.A., Papa, J.P., Hussain, A., Adeel, A.: Canonical cortical graph neural networks and its application for speech enhancement in audio-visual hearing aids. Neurocomputing **527**, 196–203 (2023)
20. Passos, L.A., Jodas, D.S., Ribeiro, L.C., Akio, M., De Souza, A.N., Papa, J.P.: Handling imbalanced datasets through optimum-path forest. Knowl.-Based Syst. **242**, 108445 (2022)
21. Pavithra, R., Priyadharshini, S., Hemanandhini, G.: Image matching using weighted graph matching algorithm. In: 7th International Conference on Advanced Computing and Communication Systems (ICACCS), vol. 1, pp. 1–5 (2021)

22. Qu, J., Ling, H., Zhang, C., Lyu, X., Tang, Z.: Adaptive edge attention for graph matching with outliers. In: Zhou, Z.H. (ed.) Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21, pp. 966–972. International Joint Conferences on Artificial Intelligence Organization (8 2021)
23. Rocha, L.M., Cappabianco, F.A.M., Falcão, A.X.: Data clustering as an optimum-path forest problem with applications in image analysis. Int. J. Imaging Syst. Technol. **19**(2), 50–68 (2009)
24. Swoboda, P., Rother, C., Alhaija, H., Kainmuller, D., Savchynskyy, B.: A study of lagrangean decompositions and dual ascent solvers for graph matching. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7062–7071. IEEE Computer Society, Los Alamitos (2017)
25. Ugander, J., Karrer, B., Backstrom, L., Marlow, C.: The anatomy of the facebook social graph. arXiv preprint arXiv:1111.4503 (2011)
26. Wang, R., Yan, J., Yang, X.: Learning combinatorial embedding networks for deep graph matching. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 3056–3065. IEEE Computer Society, Los Alamitos (2019)
27. Xu, H., Luo, D., Zha, H., Duke, L.C.: Gromov-Wasserstein learning for graph matching and node embedding. In: Proceedings of the 36th International Conference on Machine Learning, pp. 6932–6941. PMLR (2019)
28. Zyblewski, P., Woźniak, M.: Novel clustering-based pruning algorithms. Pattern Anal. Appl. **23**(3), 1049–1058 (2020)

Understanding the Influence of Extremely High-Degree Nodes on Graph Anomaly Detection

Xun Sun[1,2], Xi Xiao[1,2], Zhaoguo Wang[3(✉)], Guangwu Hu[4], Xuhui Jiang[5], Bin Zhang[6], and Hao Li[7]

[1] Shenzhen International Graduate School, Tsinghua University, Shenzhen, China
x-sun21@mails.tsinghua.edu.cn, xiaox@sz.tsinghua.edu.cn
[2] Key Laboratory of Data Protection and Intelligent Management (Sichuan University), Ministry of Education, Chengdu, China
[3] Harbin Institute of Technology (Shenzhen), Shenzhen, China
wangzhaoguo@hit.edu.cn
[4] Shenzhen Institute of Information Technology, Shenzhen, China
hugw@sziit.edu.cn
[5] Institute of Computing Technology, CAS, Beijing, China
jiangxuhui19g@ict.ac.cn
[6] Peng Cheng Laboratory, Shenzhen, China
bin.zhang@pcl.ac.cn
[7] National Key Laboratory of Advanced Communication Network, Shijiazhuang, China

Abstract. Graph Anomaly Detection (GAD) has attracted considerable attention for its potential in detecting anomalies. However, an overlooked issue in prior research is the presence of extremely high-degree node, which can introduce noise into GAD, escalate computational costs, and intensify the problem of over-smoothing. To tackle this issue, this paper first presents a novel graph anomaly dataset, NFTGraph, characterized by a notable presence of extremely high-degree nodes. A series of experiments on this dataset sheds light on the influence of such nodes on GAD. Moreover, we introduce a novel model, the Super Node-Aware Graph Neural Network (SNGNN), designed to mitigate the noise emanating from extremely high-degree nodes. Experimental results demonstrate that SNGNN outperforms extant models, achieving an average improvement of over 2% in the Area Under the ROC Curve (AUROC), and effectively reducing noise.

Keywords: Graph Anomaly Detection · Extremely High-degree Nodes

1 Introduction

Graph, a data structure with nodes and edges, has been widely used to model real-world scenarios, such as social networks [12], financial trading networks [30], and paper citing networks [10]. Since graph structures can capture relationships between entities, many anomaly detection methods are also based on graphs

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 19–34, 2025.
https://doi.org/10.1007/978-3-031-78183-4_2

[2,38], aiming to identify anomalies that are distinct from the majority in the graph. Historically, numerous models for graph anomaly detection (GAD) have been put forth, such as CONAD [38] and PCGNN [20]. These models have contributed to the advancement of GAD.

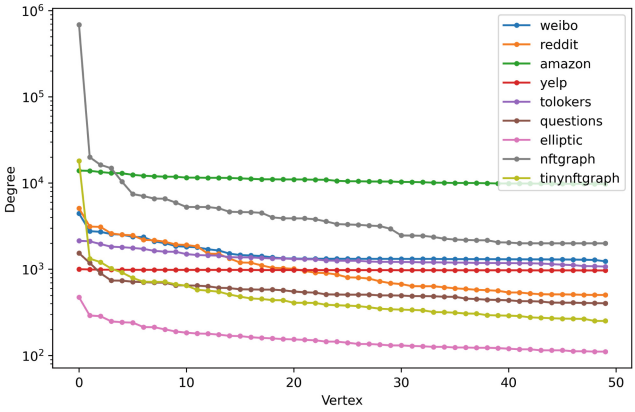


Fig. 1. The distribution of node degrees among the foremost 50 nodes.

However, a critical aspect overlooked by prior GAD studies pertains to the presence of extremely high-degree nodes, which can influence GAD models significantly. Firstly, since extreme high-degree nodes results in the formation of a tightly connected component within the graph, anomalies may inadvertently assimilate features from normal nodes through neighbor aggregation. This process complicates the delineation of anomalous nodes, introducing noise into the learning process. Moreover, the extensive connectivity associated with high-degree nodes could lead to elevated computational costs and exacerbate the issue of over-smoothing.

To elucidate the influence of extremely high-degree nodes, we first introduce a novel graph anomaly dataset termed NFTGraph. As depicted through the yellow and gray lines in Fig. 1, both NFTGraph and its variant, NFTGraph-Tiny, manifest pronounced traits of extremely high-degree nodes. Additionally, utilizing these datasets, we investigate the influence of such nodes, including their role in introducing noise to GAD, amplifying computational costs, and intensifying the issue of over-smoothing. In response, we propose the Super Node-Aware Graph Neural Network (SNGNN), a novel GAD model that incorporates a Dummy Node and Link Predictor to mitigate the noise stemming from extremely high-degree nodes. Experimental results indicate that SNGNN surpasses current models, marked by an average increase of over 2% in the detection Area Under the ROC Curve (AUROC), alongside a reduction in noise. In essence, our contributions are multifaceted:

- We provide critical insights into the influence of extremely high-degree nodes, emphasizing their potential to disrupt GAD, amplify computational costs, and exacerbate the issue of over-smoothing.
- SNGNN is designed to mitigate the noise generated by extreme high-degree nodes. Experimental results indicate that SNGNN surpasses current methods across four datasets, registering an average enhancement of more than 2% in detection AUROC, while effectively reducing noise.

Moreover, we make the dataset and code publicly available on Github to facilitate further research.

2 Related Works

Graph Anomaly Datasets: Numerous graph anomaly datasets are widely employed in previous studies. For example, Weibo [17] and Reddit [17] are derived from social networks. Questions [26] is a question-answering dataset. Moreover, Amazon [24], Yelpchi [27], Tolokers [26] and Elliptic [34] are also famous GAD datasets. While these traditional datasets have played a pivotal role in advancing GAD, they fall short in accurately representing real-world networks by not encompassing the distinct attributes of extremely high-degree nodes. They obstruct a comprehensive understanding of the influence of extremely high-degree nodes on GAD.

Degree-Related GNNs: Historically, several GNNs with a focus on degree-related considerations have been introduced to rectify node degree distribution biases. Notable examples include DEMO-Net [36] and SL-DSGCN [31], which implement degree-specific node transformations, and DegFairGNN [23] employs a function for generating debiasing contexts. Other GNNs addressing degree-related performance differences include Tail-GNN [22] and RawlsGCN [15], etc. However, these models have primarily been explored within the context of node or graph classification tasks. Thus far, a scarcity of research has addressed the ramifications of extremely high-degree nodes in anomaly detection tasks.

3 Data Collection and Properties

To understand the influence of extremely high-degree nodes, we initially gather data from NFT transactions on the blockchain and organize it into a graph structure. (1) **Raw data:** We extract certain fields of ERC-1155 NFT transaction on the Ethereum blockchain to compose the format of raw data. (Table 1). (2) **Graph Structure:** The *From* and *To* addresses, acting as the sending and receiving parties of a transaction, serve as the source and target nodes in the graph. An edge is established between the source and target nodes if tokens are transferred between them. Each node possesses 50-dimensional attributes. (3) **Labeling Suspicious Node:** We label nodes that exhibit interactions with the *ground-truth fraudulent nodes* (encompassing Ponzi schemes [6] and phishing

scams [5]) exceeding a count of three instances as *suspicious nodes*. Suspicious nodes aim to alleviate the notable imbalance of ground-truth fraudulent nodes aligning with the NFTGraph's node set. (4) **Variant Dataset:** By extracting 20,000 of the most active nodes while excluding isolated nodes, we form NFTGraph-Tiny, leading to a substantial size reduction. This is executed with the recognition that certain GNNs may encounter challenges in handling extensive graphs within resource-constrained environments. More details are described in the supplemental materials.

Table 1. Format of raw NFT transaction data.

TxHash	From	To	Token	Timestamp	Amount	Value($)	TxFee($)
0xb5...b420	0x94...7293	0x6e...b7d3	0xd0...2430	20220730055230	1	78.52	2.23
0xa5...aeea	0x00...0000	0xd8...ac95	0xd0...2430	20220730055230	14	0.0	0.98
0xa2...bdf1	0x5b...1abb	0x4f...6580	0xd0...2430	20220730055138	1	0.0	0.33
...	...	...	...	...	...	...	...

Properties: Table 2 illustrates a comparison of statistical properties between NFTGraph and several other graph anomaly datasets [29]. The anomaly ratio of NFTGraph is only 0.39%, making it the lowest in the dataset. Moreover, in NFTGraph, the highest degree (No.1 deg) is 789,782, significantly surpassing No.2 deg. This pattern is consistent in NFTGraph-Tiny, but in other graph datasets, the discrepancy between No.1 deg and No.2 deg is less pronounced.

Table 2. Statistics of NFTGraph and some common graph anomaly datasets.

Dataset	#Nodes	#Edges	Anomaly	Avg deg	No.1 deg	No.2 deg	q_1	q_2
Weibo	8,405	416,368	10.3%	99.08	4,447	2,769	44.88	27.95
Reddit	10,984	168,016	3.3%	30.59	5,112	3,134	167.10	102.44
Amazon	11,944	8,847,096	9.5%	1481.43	13,964	13,874	9.43	9.37
YelpChi	45,954	7,739,912	14.5%	336.85	1,004	996	2.98	2.96
Tolokers	11,758	530,758	21.8%	90.28	2,140	2,113	23.70	23.40
Questions	48,921	202,461	3.0%	8.28	1,541	1,186	186.18	143.29
Elliptic	203,769	438,124	9.8%	4.30	475	291	110.46	67.67
NFTGraph-Tiny	20,000	245,221	1.30%	24.52	18,104	1,330	**738.27**	54.24
NFTGraph	1,161,847	2,851,407	0.39%	4.91	789,782	20,000	**160904.05**	4074.64

4 Exploring the Influence of Extremely High-Degree Node

4.1 Definition of Extremely High-Degree Node

In the past, scholars have proposed the concepts of *influential nodes* and *central nodes* [3,11], which differ from extremely high-degree nodes. While extremely

high-degree nodes primarily focus on node degree, influential nodes and central nodes can be defined in various ways, such as through K-shell value [14] or betweenness centrality [3], among others.

Define a high-degree node in a graph as a node with a degree greater than or equal to the average degree (avg_deg). An extremely high-degree node is defined as a node whose degree/avg_deg$\geq q$ ($q \geq 1$), indicating that the node's degree exceeds the average degree by q times. q is a hyperparameter that varies depending on the dataset. Let q_1 denote the hyperparameter selectively elevating No. 1 node to an extremely high-degree node, while q_2 signifies the hyperparameter concurrently elevating both No. 1 and No. 2 to extremely high-degree nodes. As seen from Table 2, for NFTGraph, q_1 exceeds 16,0000, and q_2 also exceeds 4,000. Furthermore, there is a considerable discrepancy between the values of q_1 and q_2, whereas for graph datasets including Weibo, Reddit, and Questions, q_1 and q_2 are closer. This demonstrates significant characteristics of extremely high-degree nodes. NFTGraph-Tiny exhibits similar features. The threshold q is determined through hyperparameter tuning on the dataset's validation set to optimize the detection AUROC. For simplicity and clarity in illustrating the influence of extremely high-degree nodes, we set q equal to q_1 in subsequent discussions. Therefore, only the node with the highest degree is considered as the extremely high-degree node (abbreviated as SN).

4.2 Experimental Settings

Datasets: Due to the similarity properties between NFTGraph and NFTGraph-Tiny, and the challenges faced by certain GNNs in handling large graphs, the proposed NFTGraph-Tiny is chosen as the foundational dataset. To assess the influence of SN, a variant dataset is introduced by removing SN and the edges connected to it. These two graphs are respectively denoted as w/ SN and w/o SN. From Table 3, it can be observed that without SN, No.1 degree decreased from 18,104 to 1,330, bringing it closer to the degrees of its immediate neighbors. AnomalyAvgDeg represents the average degree of all abnormal nodes. The average abnormal node degree of NFTGraph-Tiny is 27.66, indicating that abnormal nodes generally have lower degrees than SN. Moreover, to demonstrate the advantage of the proposed dataset, several commonly used and well-known graph anomaly datasets, namely Weibo [17,19], Reddit [19,39], and Questions [26], are selected for comparison, as shown in Table 3.

Task Description: This section outlines a task aimed at identifying suspicious nodes. Formally, the objective is to train a model $f : f(u) \rightarrow \{0, 1\}$, where $\forall u \in \mathcal{V}$, $\mathcal{V}$ is node set, 1 denotes anomaly nodes and 0 denotes normal nodes.

Models and Evaluation Metrics: To comprehensively evaluate the influnce of SN, this section selects 34 anomaly detection models, including both supervised and unsupervised models, based on GNN and non-GNN models. Specifically, the unsupervised and non-GNN models [40] include OCSVM, LOF, CBLOF, COF, HBOS, SOD, COPOD, ECOD, LODA, and IForest; unsupervised and

Table 3. Datases for exploring the influence of SN.

Datasets	#Nodes	#Edges	#Feature	#Anomaly	No.1–5 Deg	AnomalyAvgDeg
NFTGraph-Tiny w/ SN	20,000	245,221	50	259	[18104,1330,1212,1020,917]	27.66
NFTGraph-Tiny w/o SN	19,999	227,118	50	259	[1330,1211,1020,916,793]	27.66
Weibo w/ SN	8,405	416,368	400	868	[4447,2769,2723,2558,2523]	54.82
Weibo w/o SN	8,404	411,922	400	868	[2767,2721,2556,2521,2376]	54.82
Reddit w/ SN	10,984	168,016	64	366	[5112,3134,3106,2608,2518]	24.75
Reddit w/o SN	10,983	162,905	64	366	[3134,3106,2608,741,2476]	24.75
Questions w/ SN	48,921	202,461	301	1460	[1541,1186,901,741,739]	20.93
Questions w/o SN	48,920	200,921	301	1460	[1185,900,740,738,717]	20.93

GNN-based models [19] include ANOMALOUS [25], ONE [1], OCGNN [33], CoLA [21], DONE [2], AnomalyDAE [8], CONAD [38], and DOMINANT [7]; supervised and non-GNN models [30] include MLP, KNN, SVM, RF; supervised and GNN-based models [30] include GCN [16], SGC [35], GIN [37], GraphSAGE [12], GAT [32], GT [28], GAS [18], BernNet [13], AMNet [4], GHRN [9], GAT-Sep [42], PCGNN [20]. Due to the severe class imbalance between suspicious and non-suspicious nodes, the Area Under the ROC Curve (AUROC) is chosen for evaluation. Other settings are in the supplemental materials.

4.3 Influence of SN on GNN-Based and Non-GNN-Based Models

Table 4 presents the AUROC of models on NFTGraph-Tiny, Weibo, Reddit, and Questions datasets, along with their corresponding graphs without SN. Refining Table 4, the **significant change rate** is defined as the proportion of models with AUROC changes exceeding 2% (±2%) after removing SN, while the **positive significant change rate** indicates an augmentation in AUROC (+2%) after SN removal.

Table 5 illustrates the significant change rates for both non-GNN and GNN models. Remarkably, the significant change rate for NFTGraph-Tiny surpasses that of Weibo, Reddit, and Questions, with datasets such as Weibo and Questions showing a minimal 0% significant change rate. This highlights the distinct advantage of employing NFTGraph-Tiny for exploring the influence of extremely high-degree nodes, thereby suggesting the limited utility of other datasets in this context.

Across the four datasets, the significant change rates of GNN-based models are substantially higher than those of non-GNN-based models. Specifically, within the NFTGraph-Tiny dataset, GNN-based models show a remarkable significant change rate of 70.00%, in contrast to the 50.00% observed for non-GNN-based models. Furthermore, Table 5 highlights that the positive significant change rate for GNN-based models surpasses 50%. This outcome appears counterintuitive since the inclusion of SN is theoretically expected to enhance the informational content, suggesting that graphs incorporating SN should uniformly exhibit superior AUROC performance. Nonetheless, the empirical data reveals that over half of the GNN-based models achieve higher AUROC scores

Table 4. AUROC of anomaly detection models. Bold for significant change of AUROC (Higher AUROC).

Models	Datasets	NFTGraph-Tiny		Weibo		Reddit		Questions	
		w/ SN	w/o SN	w/ SN	w/o SN	w/ SN	w/o SN	w/ SN	w/o SN
Unsupervised & non-GNN-based	OCSVM	0.4763	**0.5018**	0.8001	0.8017	0.5702	0.5703	0.5995	0.5995
	LOF	**0.5658**	0.5332	0.5756	0.5756	0.5369	0.5372	0.5680	0.5679
	CBLOF	0.5134	0.5106	0.8003	0.8084	0.5809	0.5827	0.6016	0.6003
	COF	**0.5662**	0.5430	0.4877	0.4885	0.5755	0.5756	0.5591	0.5591
	HBOS	0.4998	0.5041	0.4038	0.4034	0.5338	0.5338	0.5951	0.5951
	SOD	0.6590	0.6405	0.4258	0.4249	0.5495	0.5402	0.5526	0.5553
	COPOD	0.5977	0.5977	0.4736	0.4738	0.4974	0.4975	0.6059	0.6059
	ECOD	0.5240	0.5241	0.4774	0.4775	0.4999	0.4999	0.6015	0.6015
	LODA	**0.5749**	0.5412	0.7139	0.7096	0.5630	0.5633	0.5745	0.5745
	IForest	0.6016	0.6009	0.5500	0.5502	**0.5942**	0.5514	0.6057	0.6020
Unsupervised GNN-based	ANOMALOUS	0.6159	**0.6818**	0.9876	0.9876	0.5688	0.5629	0.5527	0.5530
	ONE	**0.5445**	0.4992	0.6637	0.6518	**0.5356**	0.5157	0.4867	**0.5102**
	OCGNN	**0.6327**	0.5389	0.8251	0.8257	**0.6308**	0.6139	0.5590	**0.5745**
	CoLA	**0.4943**	0.4021	0.4254	**0.4464**	0.4963	**0.5409**	0.5306	0.5465
	DONE	0.5734	0.5858	0.5536	**0.6569**	0.5518	0.5556	0.6644	0.6639
	AnomalyDAE	0.5555	**0.5803**	0.8256	0.8268	0.5805	0.5709	0.4771	**0.4995**
	CONAD	0.5382	0.5424	0.6311	**0.7050**	0.4680	**0.5174**	0.6019	0.6021
	DOMINANT	0.6026	**0.6251**	**0.7015**	0.6290	0.5129	0.5138	0.6036	0.6028
Supervised non-GNN-based	MLP	0.5645	**0.6730**	0.9738	0.9669	0.6771	0.6765	0.6753	0.6785
	KNN	0.5994	**0.6204**	0.9672	0.9674	0.6067	**0.6301**	0.6760	0.6789
	SVM	0.5756	0.5773	0.9536	0.9539	0.6622	0.6659	0.6359	0.6410
	RF	**0.6539**	0.6314	0.9864	0.9865	0.6290	0.6312	0.5621	0.5512
Supervised GNN-based	GCN	0.6580	0.6401	0.9830	0.9867	0.7172	0.7122	0.7018	0.7011
	SGC	0.5968	**0.6179**	0.9892	0.9893	0.6842	0.6885	0.6911	0.6921
	GIN	**0.6688**	0.6164	0.9881	0.9901	**0.7028**	0.6574	0.7185	0.7185
	GraphSAGE	0.5777	**0.6437**	0.9934	0.9932	0.6949	0.7130	0.7197	0.7179
	GAT	0.6510	0.6405	0.9800	0.9816	0.6866	0.6724	0.7037	0.7093
	GT	0.6163	**0.6518**	0.9899	0.9897	0.6444	**0.6682**	0.6949	0.7134
	GAS	0.6663	0.6636	0.9828	0.9824	**0.6858**	0.6627	**0.7118**	0.6913
	BernNet	0.6230	**0.6628**	0.9783	0.9853	0.6868	0.6763	0.6951	0.7095
	AMNet	**0.6970**	0.6601	0.9808	0.9858	0.6445	0.6371	0.6990	0.6989
	GHRN	0.6734	0.6656	0.9792	0.9892	0.6894	**0.7180**	0.7204	0.7210
	GAT-Sep	**0.6775**	0.6534	0.9846	0.9863	0.6665	0.6739	0.6913	0.6892
	PCGNN	**0.6895**	0.6377	0.9848	0.9846	0.6779	0.6785	**0.6929**	0.6692

upon the removal of SN, with some models registering a remarkable AUROC increase of up to 7% (e.g., ANOMALOUS). This phenomenon indicates that SN and its connected edges might introduce noise that, via neighbor aggregation, obscures the distinction between normal and anomalous nodes, complicating their differentiation.

4.4 Impact of SN on Unsupervised and Supervised GADs

Given that GNN-based models generally exhibit higher significant change rates compared to non-GNN-based models, the experimental results of GNN-based models are further analyzed to assess the influence of SN on unsupervised and supervised settings.

From Table 6, it can be observed that, regardless of the supervised or unsupervised setting, the significant change rate of NFTGraph-Tiny is not lower than

Table 5. Significant change rate for non-GNN and GNN models.

	NFTGraph-Tiny	Weibo	Reddit	Questions
non-GNN-based	**50.00%**	0.00%	14.29%	0.00%
GNN-based	**70.00%**	20.00%	40.00%	25.00%
non-GNN-based w/o SN +	42.86%	–	50.00%	–
GNN-based w/o SN +	50.00%	75.00%	50.00%	60.00%

that of the other three datasets, which also shows the advantages of NFTGraph-Tiny. Additionally, the positive significant change rates of unsupervised GNN models are higher than those of supervised GNN models across the four datasets. This suggests that, after removing SN, unsupervised GNN models achieve a higher proportion of models with increased AUROC. This phenomenon may be attributed to the absence of training labels in unsupervised GNN models, making the noise introduced by SN edges more impactful for anomaly detection. Consequently, after removing SN, unsupervised GNN models may learn better, highlighting the importance of considering the noise introduced by SN.

Table 6. Significant change rate for unsupervised and supervised GADs.

	NFTGraph-Tiny	Weibo	Reddit	Questions
Unsupervised GAD	**50.00%**	50.00%	50.00%	37.50%
Supervised GAD	**66.67%**	0.00%	33.33%	16.67%
Unsupervised GAD w/o SN +	75.00%	75.00%	50.00%	100.00%
Supervised GAD w/o SN +	50.00%	–	50.00%	0.00%

4.5 Computational Cost

Considering the highlighted benefits of NFTGraph-Tiny, it will be the primary dataset employed for further investigation in subsequent sections. Additionally, to more effectively demonstrate the impact on computational costs, this subsection will also incorporate NFTGraph to provide statistics on the average number of node neighbors and the execution time for both 1-layer and 2-layer Graph Attention Networks (GAT).

Table 7 shows the average number of 1-hop and 2-hop neighbors of GAT in NFTGraph-Tiny is 11.26 and 2386.39. Upon removing SN, the average number of 1-hop neighbors remains relatively unchanged, while the average number of 2-hop neighbors sharply decreases to 123.88. This is due to the fact that the degree of SN in NFTGraph-Tiny is 18,104, indicating that the majority of the whole 20,000 nodes in the graph are connected to SN. More pronounced disparities are observed in NFTGraph and NFTGraph w/o SN. Table 8 demonstrates how the execution time of GAT varies with different numbers of layers. Notably,

on NFTGraph-Tiny and NFTGraph, removing SN leads to a nonlinear decrease in execution time, with a more significant reduction observed for 2-layer GAT compared to a single layer. The presence of SN significantly impacts the computational cost, leading to a substantial increase in both the average number of node neighbors and the execution time.

Table 7. Average number of node neighbors for GAT at different hops.

Dataset/Hops	1-hop	2-hop
NFTGraph-Tiny	11.26	**2386.39**
NFTGraph-Tiny w/o SN	10.36	123.88
NFTGraph	2.45	**27647.54**
NFTGraph w/o SN	1.86	20.42

Table 8. Execution time (s) of GAT with different numbers of layers.

Dataset/Layer Number	1-layer	2-layer
NFTGraph-Tiny	4.89	**5.84**
NFTGraph-Tiny w/o SN	4.58	5.31
NFTGraph	49.26	**91.0**6
NFTGraph w/o SN	20.03	33.73

4.6 Over-Smoothing

To investigate the influence of extremely high-degree nodes on the issue of over-smoothing, we compute two over-smoothing metrics [41]: Instance Information Gain (G_{Ins}) and Group Distance Ratio (R_{Group}). These calculations are performed across different layer numbers of GAT applied to both the NFTGraph-Tiny dataset and its variant (w/o SN). Generally, lower values of these metrics indicate a higher level of over-smoothing.

Figure 2 illustrates the changes in the over-smoothing metrics G_{Ins} and R_{Group} for different layer numbers of GAT. It is evident that, in the majority of cases, as the number GAT's layers increases, the metrics decrease, indicating a progressive over-smoothing of node representations. Upon removal of SN, both G_{Ins} and R_{Group} metrics exhibit an increase compared to the original, thereby alleviating the over-smoothing phenomenon. Therefore, the results demonstrate that with the same layer number of GNN, the presence of extremely high-degree nodes increases the likelihood of over-smoothing.

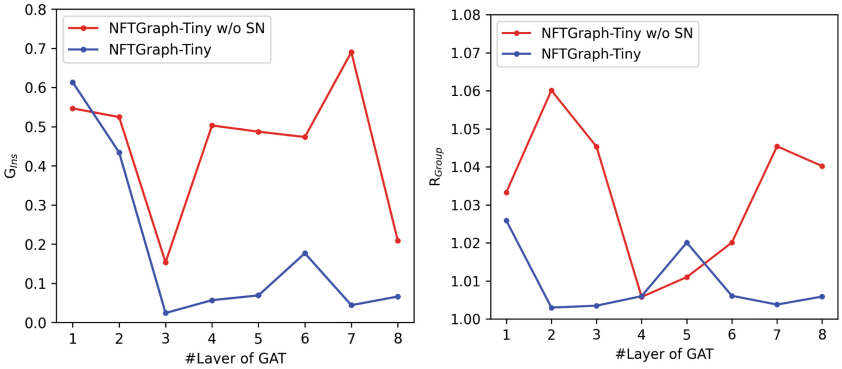
5 Method and Experiments

5.1 SNGNN

In this section, we introduce a novel graph anomaly detection model, named Super Node-Aware Graph Neural Network (SNGNN), aimed at mitigating the noise generated by nodes with extremely high degrees. The conceptual framework of SNGNN is depicted in Fig. 3.

In SNGNN, we first introduce a dummy node (DN) as a new node, and establish edges between DN and all anomaly nodes in G_0, as follows:

$$\mathcal{V} = \mathcal{V} \cup \{DN\}, \mathcal{E} = \mathcal{E} \cup \{e'\}, \tag{1}$$



(a) Over-smoothing metric G_{Ins} (b) Over-smoothing metric R_{Group}

Fig. 2. Over-smoothing metrics G_{Ins} and R_{Group} for different layer numbers of GAT on NFTGraph-Tiny and NFTGraph-Tiny w/o SN. Lower values of these metrics indicate a higher level of over-smoothing. On average, the presence of extremely high-degree nodes increases the over-smoothing problem.

where,

$$e' = (DN, s), \forall s \in \mathcal{N}_{anomaly}, \tag{2}$$

where $\mathcal{V}$ represents the set of nodes, and $\mathcal{E}$ denotes the set of all edges. Edge $e = (u, v)$ connects u and v ($\{u, v\} \in \mathcal{V}$). $\mathcal{N}_{anomaly}$ denotes the set of anomaly nodes.

Subsequently, we undertake the task of link prediction for the SN. This component inputs the SN and its neighboring nodes into a Link Predictor (LP), yielding a probability vector ($\boldsymbol{c}$) that represents the likelihood of an edge existing between SN and its neighbors. Here, we employ a straightforward dot product as the LP, which is articulated as follows:

$$c_n =< h_{SN}, h_n >, \forall n \in \mathcal{N}_{neigbhour} \tag{3}$$

where h_n is the hidden vector of node n, $\mathcal{N}_{neighbour}$ is the set of the original neighbors of SN, and $<,>$ indicates the dot product between two vectors. Then, the p_1-quantile (c_{p_1}) and p_2-quantile (c_{p_2}) are derived from the probability vector $\mathbf{c}$. Nodes with probabilities below c_{p_1} are severed from SN, whereas those with probabilities above c_{p_2} are linked to SN. Formally:

$$\mathbf{A}_{SN,n} = \begin{cases} 0, c_n \leq c_{p_1} \\ 1, c_n \geq c_{p_2} \end{cases}, \forall n \in \mathcal{N}_{neigbhour}, \tag{4}$$

where $\mathbf{A}_{SN,n} = 0$ signifies the absence of an edge between the SN and node n in the adjacency matrix $\mathbf{A}$.

Finally, we update the node representations by aggregating neighbor information according to the updated graph topology at each iteration. Utilizing the

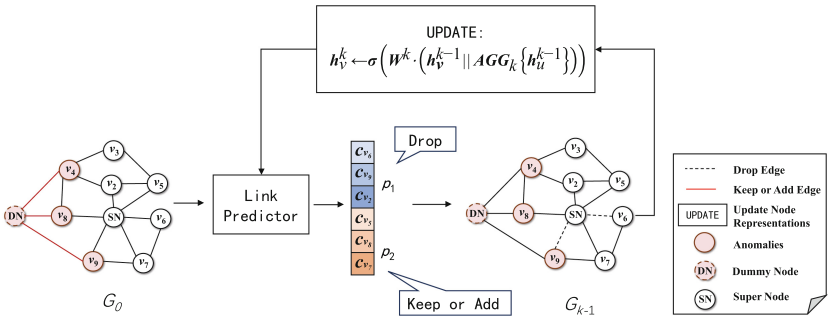


Fig. 3. Schema of SNGNN.

node representations from the final layer, we then ascertain whether nodes are anomalous.

The rationale behind the development of SNGNN encompasses several key aspects: Firstly, considering the typically lower degree of anomalous nodes themselves (as indicated by AnomalyAvgDeg in Table 3), the integration of a dummy node (DN) connected to all identified anomalous nodes serves to mitigate the imbalance between low-degree anomalous nodes and extremely high-degree nodes to some extent. Moreover, the inclusion of DN facilitates the acquisition of a more "pure" representation of anomalous nodes during the propagation process. This is crucial as the connection of anomalous nodes to their normal counterparts results in the amalgamation of information from normal nodes during the neighbor aggregation phase by GNN, which is counterproductive for accurately modeling the representations of anomalous nodes.

Secondly, the implementation of thresholds p_1 and p_2 allows for the disconnection of edges with probabilities below p_1 and the maintenance or addition of edges with probabilities above p_2. This approach is designed to evaluate the reliability of edges associated with the SN. By learning from anomaly labels, the model is capable of autonomously determining the optimal thresholds to either facilitate or inhibit message transmission, thereby diminishing noise in the connections to SN and enhancing the model's resilience.

5.2 Experiments

Setup: NFTGraph-Tiny, Weibo, Reddit, and Questions datasets are used, and the task is anomaly detection. The baseline models include three basic GNNs: GCN, GAT, and GraphSAGE, as well as several GNNs that achieve better performance in Table 4: PCGNN, GAS, GIN, AMNet and GHRN. Settings are different from Sect. 4.2, which can be found in the supplemental materials.

Results: Table 9 shows the comparison of AUROC between SNGNN and other GADs. Across all four datasets, SNGNN consistently achieves the highest AUROC, with an average increase of over 2% compared to other models. Notably,

on the NFTGraph-Tiny dataset, SNGNN achieves an AUROC of 0.6980, surpassing GHRN by 5%. Similarly, on the Weibo dataset, SNGNN's AUROC is 0.9926, higher than the second-best model GraphSAGE. Likewise, on Reddit and Questions, SNGNN outperforms GCN (0.7189) and GraphSAGE (0.7259) models, resulting in a 1% AUROC improvement. It is noteworthy that SNGNN exhibits at least a 5% AUROC improvement on NFTGraph-Tiny, significantly higher than the approximately 1% improvement observed on other datasets. This performance disparity stems from the notably higher degree values of social networks in NFTGraph-Tiny compared to Weibo, Reddit, and Questions, as evidenced in Table 3, underscoring SNGNN's superior efficacy in this context.

Table 9. Comparison of AUROC between SNGNN and other GADs. The best performance is shown in bold, while the second-best performance is underlined.

Model/Dataset	NFTGraph-Tiny	Weibo	Reddit	Questions
GCN	0.5953	0.9875	<u>0.7189</u>	0.6819
GAT	0.6226	0.9902	0.6733	0.7167
GraphSAGE	0.6428	<u>0.9917</u>	0.6800	<u>0.7259</u>
PCGNN	0.5832	0.9848	0.7079	0.6784
GAS	0.5552	0.9915	0.6996	0.7111
GIN	0.5929	0.9908	0.6872	0.7160
AMNet	0.6263	0.9764	0.6731	0.7064
GHRN	<u>0.6479</u>	0.9860	0.6963	0.7164
SNGNN	**0.6980**	**0.9926**	**0.7272**	**0.7325**

5.3 Ablation Study

To validate the effectiveness of SNGNN, we design several ablation tests. Specifically, while keeping the other parts and hyperparameters unchanged, Dummy Node (referred to as w/o DN) and Link Predictor (referred to as w/o LP) are removed separately, and then the performance is observed.

Table 10 shows the results of ablation tests. Notably, the removal of the Dummy Node (DN) results in a diminished detection AUROC for SNGNN. This effect is particularly pronounced on the NFTGraph-Tiny and Questions datasets, where the AUROC for SNGNN drops by over 5% and 3%, respectively. Similarly, the elimination of the Link Predictor (LP) also leads to a reduction in AUROC, with a significant decrease of more than 6% on the NFTGraph-Tiny dataset. Consequently, the incorporation of both Dummy Node and Link Predictor is essential for the effectiveness of SNGNN.

In addition, we visualize the changes in the number of edges of graphs caused by the LP of SNGNN in the training stage, and simultaneously match the changes in the AUROC of the validation set, as shown in Fig. 4. As can be seen,

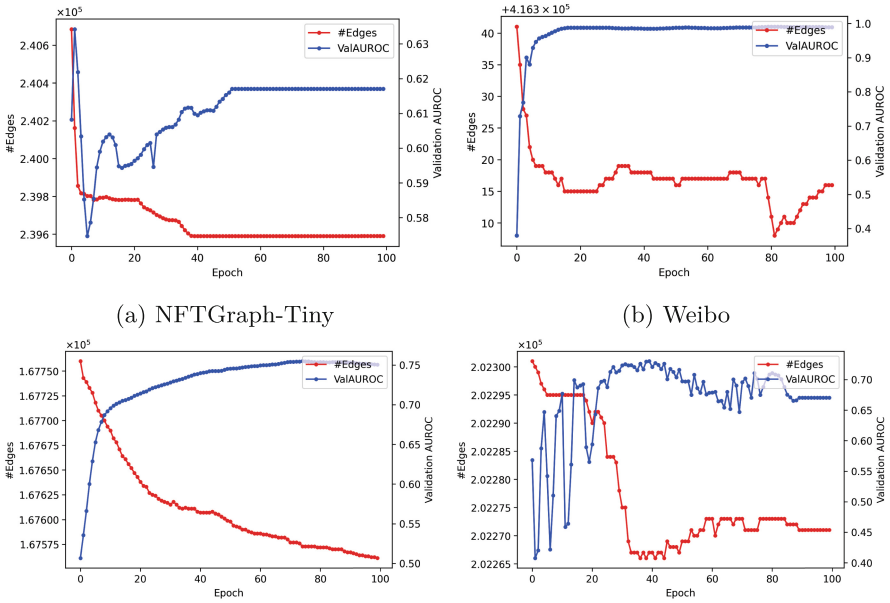
Table 10. Results of ablation study for SNGNN.

Dataset/Model	w/o DN	w/o LP	SNGNN
NFTGraph-Tiny	0.6414	0.6351	**0.6980**
Weibo	0.9914	0.9919	**0.9926**
Reddit	0.7188	0.7239	**0.7272**
Questions	0.7086	0.727	**0.7325**

Table 11. Performance of noise reducing.

Dataset/Model	w/o DN&LP	SNGNN
NFTGraph-Tiny	0.5807	**1.9401**
Weibo	1.6492	**2.9917**
Reddit	0.8738	**1.4031**
Questions	1.1473	**1.7393**

on the four datasets, the operation of LP deleting edges is dominant, the total number of edges in graphs is gradually decreasing, while the AUROC on the validation set is generally increasing, which may be due to the noise caused by LP eliminating SN connecting edges, thus improving the performance of SNGNN.



(a) NFTGraph-Tiny (b) Weibo

(c) Reddit (d) Questions

Fig. 4. Changes in the number of edges caused by the Link Predictor of SNGNN and changes in the AUROC on the validation set during training.

Additionally, we compute the average inter-group distance (DisInter) [41] between anomalies and normal nodes. A higher DisInter metric indicates that SNGNN more effectively distances the embeddings of anomalous nodes from those of normal ones, signifying more precise outcomes. This metric serves as an indicator of the model's efficiency in noise reduction. In Table 11, the baseline

represents the SNGNN model without the Dummy Node (DN) and Link Predictor (LP), essentially constituting the GNN backbone of SNGNN (corresponding to the *Update* stage in Fig. 3). For the Reddit dataset, the baseline is GCN, while for other datasets, it is GraphSAGE. Table 11 demonstrates that SNGNN achieves a greater DisInter value compared to its baseline, indicating that the incorporation of DN and LP enhances the model's ability to reduce noise.

Furthermore, parameter sensitivity analysis of SNGNN is provided in the supplemental materials.

6 Conclusion and Limitation

In this paper, our focus is on exploring the influence of extremely high-degree nodes on graph anomaly detection (GAD). To address this objective, we first introduce a novel graph dataset, NFTGraph, and conduct a comprehensive analysis of the influence, including the introduction of noise to GAD, the escalation of computational costs, and the exacerbation of over-smoothing phenomena. Additionally, we propose a novel model called Super Node-Aware Graph Neural Network (SNGNN) to mitigate the noise introduced by extremely high-degree nodes. SNGNN demonstrates superior performance compared to existing models, achieving an average improvement in detection AUROC of over 2% while efficiently reducing noise.

Although SNGNN specifically targets the SN, it can be extended and applied to all extremely high-degree nodes. In addition, in the future, we will further analyze the relationships between SNGNN's performance and other issues, such as computational costs, over-smoothing and heterophily, and further improve SNGNN to mitigate these influences. Additionally, extremely high-degree nodes are prevalent in various real-world networks, such as influencers in social networks, banks in financial trading networks, and super-spreaders in disease transmission networks. In the future, our aim is to investigate these aspects beyond the scope of blockchain transaction networks.

Acknowledgement. The study was supported by the Key Laboratory of Data Protection and Intelligent Management, Ministry of Education, Sichuan University and also the Fundamental Research Funds for the Central Universities under Grant SCU2023D008.

References

1. Bandyopadhyay, S., Lokesh, N., Murty, M.N.: Outlier aware network embedding for attributed networks. In: AAAI, vol. 33, pp. 12–19 (2019)
2. Bandyopadhyay, S., Vivek, S.V., Murty, M.: Outlier resistant unsupervised deep architectures for attributed network embedding. In: Proceedings of the 13th International Conference on Web Search and Data Mining, pp. 25–33 (2020)
3. Bhattacharya, R., Nagwani, N.K., Tripathi, S.: Detecting influential nodes with topological structure via graph neural network approach in social networks. Int. J. Inf. Technol. **15**(4), 2233–2246 (2023)

4. Chai, Z., et al.: Can abnormality be detected by graph neural networks? In: IJCAI (2022)
5. Chen, L., Peng, J., Liu, Y., Li, J., Xie, F., Zheng, Z.: Phishing scams detection in ethereum transaction network. ACM Trans. Internet Technol. (TOIT) **21**(1), 1–16 (2020)
6. Chen, W., Zheng, Z., Cui, J., Ngai, E., Zheng, P., Zhou, Y.: Detecting ponzi schemes on ethereum: towards healthier blockchain technology. In: WWW, pp. 1409–1418 (2018)
7. Ding, K., Li, J., Bhanushali, R., Liu, H.: Deep anomaly detection on attributed networks. In: SIAM International Conference on Data Mining, pp. 594–602 (2019)
8. Fan, H., Zhang, F., Li, Z.: Anomalydae: dual autoencoder for anomaly detection on attributed networks. In: ICASSP, pp. 5685–5689. IEEE (2020)
9. Gao, Y., Wang, X., He, X., Liu, Z., Feng, H., Zhang, Y.: Addressing heterophily in graph anomaly detection: a perspective of graph spectrum. In: Proceedings of the ACM Web Conference (2023)
10. Giles, C.L., Bollacker, K.D., Lawrence, S.: Citeseer: an automatic citation indexing system. In: Proceedings of the Third Conference on Digital Libraries, pp. 89–98 (1998)
11. Hafiene, N., Karoui, W., Romdhane, L.B.: Influential nodes detection in dynamic social networks: a survey. Expert Syst. Appl. **159**, 113642 (2020)
12. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: NeurIPS, vol. 30 (2017)
13. He, M., Wei, Z., Huang, Z., Xu, H.: Bernnet: Learning arbitrary graph spectral filters via bernstein approximation. NeurIPS (2021)
14. Hu, H., Sun, Z., Wang, F., Zhang, L., Wang, G.: Exploring influential nodes using global and local information. Sci. Rep. **12**(1), 22506 (2022)
15. Kang, J., Zhu, Y., Xia, Y., Luo, J., Tong, H.: Rawlsgcn: towards rawlsian difference principle on graph convolutional network. In: WWW, pp. 1214–1225 (2022)
16. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: ICLR (2017)
17. Kumar, S., Zhang, X., Leskovec, J.: Predicting dynamic embedding trajectory in temporal interaction networks. In: 27th SIGKDD, pp. 1269–1278 (2019)
18. Li, A., Qin, Z., Liu, R., Yang, Y., Li, D.: Spam review detection with graph convolutional networks. In: the 28th CIKM, pp. 2703–2711 (2019)
19. Liu, K., et al.: Bond: benchmarking unsupervised outlier node detection on static attributed graphs. In: NeurIPS, vol. 35, pp. 27021–27035 (2022)
20. Liu, Y., et al.: Pick and choose: a gnn-based imbalanced learning approach for fraud detection. In: WWW, pp. 3168–3177 (2021)
21. Liu, Y., Li, Z., Pan, S., Gong, C., Zhou, C., Karypis, G.: Anomaly detection on attributed networks via contrastive self-supervised learning. IEEE Trans. Neural Netw. Learn. Syst. **33**(6), 2378–2392 (2021)
22. Liu, Z., Nguyen, T.K., Fang, Y.: Tail-gnn: tail-node graph neural networks. In: The 27th SIGKDD, pp. 1109–1119 (2021)
23. Liu, Z., Nguyen, T.K., Fang, Y.: On generalized degree fairness in graph neural networks. In: 37th AAAI, pp. 7–14 (2023)
24. McAuley, J.J., Leskovec, J.: From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In: Proceedings of the 22nd International Conference on Blockchain, pp. 897–908 (2013)
25. Peng, Z., Luo, M., Li, J., Liu, H., Zheng, Q., et al.: Anomalous: a joint modeling approach for anomaly detection on attributed networks. In: IJCAI, vol. 18, pp. 3513–3519 (2018)

26. Platonov, O., Kuznedelev, D., Diskin, M., Babenko, A., Prokhorenkova, L.: A critical look at the evaluation of gnns under heterophily: are we really making progress? arXiv preprint arXiv:2302.11640 (2023)
27. Rayana, S., Akoglu, L.: Collective opinion spam detection: bridging review networks and metadata. In: 21th SIGKDD, pp. 985–994 (2015)
28. Shi, Y., Huang, Z., Feng, S., Zhong, H., Wang, W.: Masked label prediction: unified message passing model for semi-supervised classification. In: IJCAI (2021)
29. Tang, J., Hua, F., Gao, Z., Zhao, P., Li, J.: Gadbench: revisiting and benchmarking supervised graph anomaly detection. In: NeurIPS, vol. 36 (2024)
30. Tang, J., Li, J., Gao, Z., Li, J.: Rethinking graph neural networks for anomaly detection. In: ICML, pp. 21076–21089 (2022)
31. Tang, X., et al.: Investigating and mitigating degree-related biases in graph convolutional networks. In: the 29th CIKM, pp. 1435–1444 (2020)
32. Velickovic, P., et al.: Graph attention networks. arXiv preprint (2017)
33. Wang, X., Jin, B., Du, Y., Cui, P., Tan, Y., Yang, Y.: One-class graph neural networks for anomaly detection in attributed networks. Neural Comput. Appl. **33**(18), 12073–12085 (2021)
34. Weber, M., et al.: Anti-money laundering in bitcoin: experimenting with graph convolutional networks for financial forensics. arXiv preprint (2019)
35. Wu, F., Jr., A.H.S., Zhang, T., Fifty, C., Yu, T., Weinberger, K.Q.: Simplifying graph convolutional networks. In: The 36th ICML, vol. 97, pp. 6861–6871 (2019)
36. Wu, J., He, J., Xu, J.: Demo-net: degree-specific graph neural networks for node and graph classification. In: 25th SIGKDD, pp. 406–415 (2019)
37. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? In: ICLR (2019)
38. Xu, Z., Huang, X., Zhao, Y., Dong, Y., Li, J.: Contrastive attributed network anomaly detection with data augmentation. In: Advances in Knowledge Discovery and Data Mining: 26th Pacific-Asia Conference (PAKDD), pp. 444–457 (2022)
39. Zhao, T., Deng, C., Yu, K., Jiang, T., Wang, D., Jiang, M.: Error-bounded graph anomaly loss for gnns. In: The 28th CIKM, pp. 1873–1882 (2020)
40. Zhao, Y., Nasrullah, Z., Li, Z.: Pyod: a python toolbox for scalable outlier detection. JMLR **20**(96), 1–7 (2019)
41. Zhou, K., Huang, X., Li, Y., Zha, D., Chen, R.: Towards deeper graph neural networks with differentiable group normalization. In: NeurIPS, vol. 33, pp. 4917–4928 (2020)
42. Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L.: Beyond homophily in graph neural networks: current limitations and effective designs. In: NeurIPS, vol. 33 (2020)

Spatio-Temporal Heterogeneous Graph Neural Network With Multi-view Learning For Traffic Prediction

Liting Song[1], Qianqian Ren[1(✉)], and Ying Zhou[2(✉)]

[1] Department of Computer Science and Technology, Heilongjiang University, Harbin 150080, China
renqianqian@hlju.edu.cn

[2] Heilongjiang Network Space Research Center, Harbin 150090, China
zhouying9666@126.com

Abstract. Among various traffic data modeling and predicting methods, graph learning-based models attract more attention, because of their powerful representation ability for modeling spatial and temporal dependencies with graph neural networks. Despite their promising performance, several key problems have not been well addressed: 1) Sensed data are often noisy in many real transportation scenarios. 2) The spatio-temporal correlations of traffic data are complex and dynamic, especially for long-term modeling and predicting. In such cases, existing methods may not lead to satisfactory prediction results. In this paper, we aim at the above problems by exploring a Spatio-Temporal Heterogeneous Graph Neural Network With Multi-View Learning Framework(MVJGL) for traffic prediction. In particular, we first model different types of traffic features and construct multiple graph structures. Then, we design two parallel heterogeneous gated temporal convolution modules to extract long and short-term temporal dependencies from different traffic features, respectively. Moreover, we introduce parallel graph convolutions to cross-characterize the time-varying spatial dependencies of each view. Extensive experiment results on four real traffic datasets show the superior performance grain obtained by the proposed model.

Keywords: Traffic prediction · multi-view · graph convolution · graph learning

1 Introduction

As an essential issue of intelligent transportation systems (ITS), traffic prediction has attracted much attention in recent years [1]. Robust and accurate traffic prediction helps improve the operational efficiency and reliability of the traffic system, thus guiding traffic resource allocation, developing traffic routines, and relieving traffic congestion. In addition, traffic prediction is also meaningful in road safety assurance and traffic accident prevention [2]. However, the dynamic and complex spatial-temporal traffic features propose challenges for accurate traffic prediction.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 35–52, 2025.
https://doi.org/10.1007/978-3-031-78183-4_3

Many traffic forecasting approaches have been proposed. Deep learning-based methods are efficient in improving forecasting accuracy via modeling the temporal and spatial correlations in traffic data [3]. In particular, convolution neural networks are often utilized to model the spatial dependencies, recurrent neural networks (e.g., LSTM [4], GRU [5]) and temporal convolution networks (TCN) [6] are employed to extract temporal dependencies. Recently, graph convolution networks (GCNs) have been introduced in modeling spatial correlations and graph representation. Among these methods, predefined static graph-based methods construct graph structures by linking road networks, while graph learning-based methods [7] automatically learn dynamic spatial dependencies. Graph-WaveNet [8] modeled spatial dependencies by constructing an adaptive adjacency matrix, MTGNN [9] automatically extracted single dependencies between variables to construct dynamic graphs. The dynamic graph construction method employed by DMSTGCN [10] learned the specific multiple spatial-temporal correlations of road segments. PGCN [11] captured the spatial correlations over time by generating progressive graph structures. The above methods have effectively improved the accuracy of traffic forecasting, but it is far from being well addressed, which faces the following challenges.

- **Challenge 1: Data noise and incompleteness**. As factors such as traffic flow, traffic crowd, and weather conditions change, data noise and incompleteness are inevitable in collected sensed data. Therefore, constructing a spatial graph based on a single traffic feature may not reveal the real topology of the road network. It motivates us to model spatial dependencies from multiple traffic features.
- **Challenge 2: Complexity and dynamicity**. Traffic conditions are complex and dynamic. The spatial correlations among different nodes of the road network vary at different time steps, existing methods based on pre-defined or stationary graphs may not obtain satisfactory forecasting results. Therefore, it is important to model the varying spatial-temporal dependencies, especially in the application of long-term prediction.
- **Challenge 3: Multi-view fusion**. Constructed multiple views are needed to effectively fuse to comprehensively exploit the multiple aspects of the complex spatial correlations. It is meaningful to enhance prediction performance.

In light of these challenges, we propose a Spatio-Temporal Heterogeneous Graph Neural Network With Multi-View Learning For Traffic Prediction framework (MVJGL) for improving traffic prediction accuracy. MVJGL aims to learn multiple continuous graph structures from multiple traffic features to characterize complex spatial and temporal correlations. Specifically, we solve the first challenge by mining the potential correlations among multiple traffic features. In response to the second challenge, we propose a dynamic graph learning module for representing continuous long-range spatial dependencies between nodes with multiple views that vary dynamically over time. Furthermore, two heterogeneous temporal convolution modules are proposed to jointly learn the temporal correlations from historical observations. Finally, a multi-view fusion module is

proposed to interact and integrate the multiple features. Our contributions are highlighted as follows:

- In this paper, a novel Spatio-Temporal Heterogeneous Graph Neural Network (MVJGL) framework for traffic prediction is proposed, which is featured by the two streams of graph learning, temporal networks, and GCNs.
- The proposed MVJGL extracts the dynamic dependencies from multiple traffic features, which contributes to exploit the complex latent relations of the traffic system and reduce the influence of data noise.
- Extensive experiments are conducted on four real-world traffic datasets. The experimental results demonstrate that MVJGL outperforms the state-of-the-art baseline methods by up to 2.3% in forecasting accuracy.

The paper is organized as follows. In Sect. 2 we summarise related work in the field of traffic prediction, and in Sect. 3 we formulate the research question. In Sect. 4 we describe the model, in Sect. 5 an extensive experimental evaluation is performed and Sect. 6 concludes the paper.

2 Related Work

In this section, we summarize the related study in three aspects: traffic forecasting, graph neural networks and dynamic graph learning approaches for traffic forecasting.

2.1 Traffic Forecasting

Existing works address the problem of traffic prediction in the following categories: Traditional statistically based methods usually use historical data to build models. Such methods use historical traffic data to analyze traffic states and trends to predict future traffic speeds [12]. These methods have the advantages of better explainability and easy implementation. Still, they cannot consider complex traffic scenarios and changes in traffic mobility leading to poor prediction results. Machine learning-based methods use historical traffic data to train models that can forecast future traffic speeds by analyzing and learning the patterns of the data [13,14]. Machine learning methods are relatively small in terms of data requirements, and the data needed mainly include historical observation and traffic environment factors. For example, weather and road conditions can be obtained using sensors, GPS and other devices. Machine learning methods are less computationally complex, typically use fewer hardware resource requirements and have relatively better predictive results, but they are influenced by factors such as data attributes and model selection. Recently, deep learning methods have obtained better performance in traffic speed forecasting, especially for time series data of traffic speeds [9,15]. Deep learning methods usually need to use a large amount of observations to train the model. The computational complexity is higher and requires the use of higher-performance

hardware resources to train deep neural networks. In terms of forecasting accuracy, deep learning methods can usually obtain better results, but over-fitting may occur for smaller amounts of data. In general, all of the above methods can be used for traffic prediction, and the model selection depends on the specific application scenario and requirements. Machine learning methods can be chosen if the data volume is small and the computation speed is fast. If the data volume is large and high forecast accuracy is required, the deep learning method can be chosen.

2.2 Graph Neural Networks

To extract the complex spatial correlations in road networks, recent studies have used deep learning models integrated with graph neural networks. DCRNN [3] used diffusion convolution to capture spatial-temporal features in spatial graphs and temporal correlations are extracted by gated recurrent units to encode temporal information. In addition, Graph-WaveNet [8] used adaptive graph convolution to model spatial correlations to improve the limitations of spatial relationships in predefined traffic networks. STSGCN [16] introduced a spatial-temporal synchronous paradigm to model the spatial and temporal correlations. MRA-BGCN [17] designed a two-component graph convolution method to represent nodes and edges separately for learning complex nonlinear relationships. In summary, graph neural networks became a popular trend for the extraction of spatial-temporal attributes.

2.3 Dynamic Graph Learning

To better describe complex traffic conditions, models used dynamic graph learning methods to represent the road space relationships. GMAN [18] used a node2vec approach to learn node representations and the embedding of nodes into vectors to preserve graph structure information. Graph-WaveNet [8] learned spatial dependencies by constructing an adaptive adjacency matrix. AGCRN [19] used an adaptive parametric learning (NAPL) module to capture node-specific patterns. PGCN [11] captured spatial correlations over time by generating a progressive graph approach, which adapted to the data. MTGNN [9] automatically extracted one-way relationships between variables to construct dynamic graphs. Several recent studies have proved that the use of dynamic graphs showed great effectiveness in the field of traffic prediction, and traditional predefined static graph methods are no longer able to be satisfied with complex traffic prediction needs.

3 Problem Definition

Definition 1 (Spatial-Temporal Graph). The traffic network is represented as a graph $G_t = (V, E, A_t)$, where $V = \{v_1, v_2, \cdots, v_N\}$ represents the set of N nodes, E is the set of edges among nodes, and $A_t \in R^{N \times N}$ denotes the potential correlations at a certain time step t.

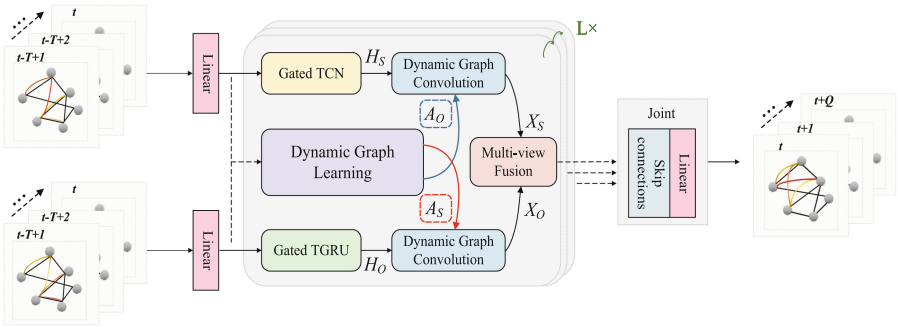


Fig. 1. The architecture of MVJGL. The model can be roughly divided into five parts: the Gated Temporal Convolution Networks(Gated TCN), the Gated temporal recurrent unit(Gated TGRU), the Dynamic Graph Learning module, the Dynamic Graph Convolution module, and the multi-view fusion module.

Definition 2 (Multiple Traffic Features). $X_t \in R^{N \times C}$ represents the traffic characteristics observed by all nodes at time t, such as traffic speed, traffic volume, etc. In this paper, we take two traffic features as an example. In particular, traffic speed is treated as the primary feature, and traffic volume is treated as the auxiliary feature. Traffic observations of the primary feature and auxiliary feature collected at time step t by all nodes in G_t are denoted as $X_t^S, X_t^O \in R^{N \times C}$.

Problem Description (Traffic Prediction). Our objective is to learn a nonlinear function, denoted as $f(\cdot)$, which can forecast the primary feature for the future Q time steps $\widehat{Y}_{t:t+Q}^S$ based on historical primary feature and auxiliary features for T time steps. We formulate the problem as follows:

$$[X_{S_{t-T+1:t}}; X_{O_{t-T+1:t}}] \xrightarrow{f(\cdot)} [\widehat{Y}_{S_{t:t+Q}}] \tag{1}$$

4 Methodology

Our model learns graph structure from multiple traffic features followed by dynamic graph convolution modules and temporal convolution modules, which jointly exploit the spatial and temporal dependencies for long and short-term traffic prediction. The overall model architecture is illustrated in Fig. 1, which stacks L layers. Each layer consists of the following four components:

- A dynamic graph learning module that learns multiple views from different traffic features. The learned views are fed into the consequent graph convolution modules to extract the dynamic spatial dependencies. In this paper, we consider traffic features including traffic speed and traffic volume.
- Two heterogeneous temporal convolution modules, Gated Temporal Convolution Network (GTCN) and Gated Temporal Recurrent Unit (TGRU) capture multiple temporal correlations from the patterns of two traffic features, respectively.

– Two dynamic graph convolution modules are used to cross-model the spatial dependencies of multiple features, one for the primary traffic feature view and the other for the auxiliary traffic feature view.
– A multi-view fusion module weights the spatio-temporal dependencies of multiple features.

4.1 Heterogeneous Temporal Convolution Module

To comprehensively model the temporal dependencies from multiple features, such as traffic speed and traffic volume, we correspondingly design two heterogeneous temporal convolution networks to model each traffic feature. In particular, we integrate the gated mechanism with the temporal convolution network (GTCN) to extract the temporal correlations from the primary feature, while the gated temporal recurrent unit (TGRU) to extract correlations from the auxiliary feature. Inspired by the gating mechanism such as GRU, we integrate gate structure with temporal convolution networks to adjust the flow of information passing to the next module.

Gated Temporal Convolution Networks. We design gated temporal convolution networks (GTCN) to capture dynamic behaviors of primary features in the temporal dimension. As shown in Fig. 2(a), GTCN is composed of two temporal convolution layers and a gated mechanism.

We will next describe the details of GTCN. The input is $X^S = \{x_1^S, x_2^S, \cdots, x_N^S\}$, where $x_i^S \in R^{N\times C}$(C=1 for GTCN). Next, the input is passed through two dilated convolution modules separately for the convolutional results Z_f and Z_s. Then, *sigmoid* and tanh function are used to Z_f and Z_s, respectively. They map the outputs of the first TCN to (0,1) and the results of the second TCN to the interval (-1,1). Finally, the element-wise product of $\tanh(Z_f)$ and $\sigma(Z_s)$ is output.

In particular, GTCN is composed of two dilated convolution modules to exploit the multiple features, and the dilation factor is introduced to process long-term traffic data. Thus, GTCN is defined as follows:

$$H^{S(l)} = \tanh(\Theta_1 \star X^{S(l-1)}) \odot \sigma(\Theta_2 \star X^{S(l-1)}) \tag{2}$$

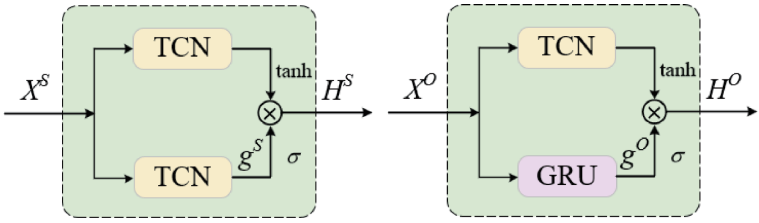
where $X^{S(l-1)} \in R^{N\times T\times C}$ denotes the input sequence from the primary feature at l-th layer ($l \in [1, 2, ..., L]$), $H^{S(l)} \in R^{N\times T\times C}$ represents the output of GTCN module. Θ_1 and Θ_2 are two convolution kernels parameters. $\star$ denotes the dilated convolution, and $\odot$ denotes the element-wise multiplication operation.

Gated Temporal Recurrent Unit. The gated temporal recurrent unit (TGRU) module aims to extract temporal dependencies from the auxiliary feature. Given the input patterns $X^O = \{x_1^O, x_2^O, \cdots, x_N^O\}$, where $x_i^O \in R^{N\times C}$(C=1 for TGRU) at the l-th layer. Gated TGRU is mathematically defined as follows:

$$H^{O(l)} = \tanh(\Theta_3 \star X^{O(l-1)}) \odot \sigma(g^{O(l-1)}) \tag{3}$$

where $H^{O(l)}$ is the hidden state at the $l-$ th layer, and $g^{O(l-1)}$ denotes the output of GRU unit. Θ_3 represents the convolution kernel parameter. Formally, $g^{O(l)}$ is updated as:

$$\begin{aligned} z^{O(l)} &= \sigma(W_{x_1} X^{O(l-1)} + U_{x_1} h^{O(l-1)})) \\ r^{O(l)} &= \sigma(W_{x_2} X^{O(l-1)} + U_{x_2} h^{O(l-1)})) \\ \widetilde{h}^{O(l)} &= \tanh(W_h X^{O(l-1)} + U_h(r^l \odot h^{O(l-1)})) \\ g^{O(l)} &= (z^{O(l)} \odot \widetilde{h}^{O(l)} + (1 - z^{O(l)}) \odot h^{O(l-1)}) \end{aligned} \tag{4}$$



(a) GTCN (b) TGRU

Fig. 2. The detail of our proposed heterogeneous temporal convolutional networks. (a) GTCN for primary feature extraction. (b) TGRU for auxiliary feature extraction.

The GRU method merges the information transmitted from the previous layer with the input of the subsequent layer and dynamically updates the input data weights. It filters and prioritizes key information within patterns, which facilitates the learning of vital dependencies between various time steps in the pattern. Regarding traffic speed attributes, gated TCN enhances the model's capability to capture long-distance dependencies and dynamically adjust the information flow. For traffic volume attributes, considering the characteristics of time series data, we combine the short-term information processing advantages of recurrent neural networks (GRU) with the long-term dependent capture capability of TCN. The gating mechanism further enhances the flexibility and efficiency of the model.

4.2 Dynamic Graph Learning Module

We solve the spatial features from two aspects, namely primary feature and auxiliary feature to enhance the graph learning results. The spatial correlations between nodes in the graphs are constantly changing due to the complexity and dynamics of traffic observations. The methods based on pre-defined graphs or static graphs cannot capture dynamically varying spatial dependencies, which influences prediction results. To address this problem, we present a dynamic

graph learning module for modeling dynamic and continuous spatial dependencies from multiple traffic features. For each iteration, two feature matrices $A^{S(l)}$ and $A^{O(l)}$ are generated at the same time step, which is fed into the two dynamic graph convolution modules. In this paper, the design for the primary feature is taken as an example, and the generated tensor $A^{S(l)}$ represents the neighborhood relationship between the source node and the target node at the time step t, which is represented by the following formula:

$$\begin{aligned} A_1^l &= \sum_{i=1}^{d}\sum_{j=1}^{d} M_{1(i)}^l M_{2(j)}^l M_{h(i,j)}^l \\ A_2^l &= \sum_{\tau=1}^{d} A_1^l M_{3_{(\tau)}}^l \\ A^{S(l)} &= ReLU(tanh(A_1^l(A_2^l)^T - A_2^l(A_1^l)^T)) \end{aligned} \tag{5}$$

Time nodes are data points with different time steps, and each time node represents the traffic state with a specific time step. MVJGL uses learnable parameters to generate a dynamic spatial dependency tensor and constructs a dynamic graph by randomly initializing the node embeddings, i.e., initializing at the beginning of the training period by random values to generate three learnable matrices M_1^l, M_2^l, and M_3^l, and a core tensor M_h^l to represent the relationship between different nodes and the relationships between time steps. As the model is trained, these matrices and tensors are continuously updated and optimized by a back-propagation algorithm to capture the dynamics of the traffic network. $M_1^l \in R^{N_1 \times d}$ represents the matrix of embedded time nodes, $M_2^l \in R^{N_2 \times d}$ represents the matrix of embedded source nodes, and $M_3^l \in R^{N_3 \times d}$ represents the matrix of embedded target nodes. The core tensor $M_h^l \in R^{d \times d \times d}$ is intended to represent the dynamics in the graph, which captures the complex relationships between nodes and time steps. N_1, N_2, N_3 represent the number of time intervals, the number of source nodes, and the number of target nodes, respectively, and d is the dimension of node-embedding. Through the above methods, MVJGL captures the heterogeneous relationship between source and target nodes by transposing the source and target node matrices at moment t, strengthens the difference signals between source and target nodes, helps the model learn and capture the complex relationship between source and target nodes at the same time flexibly and generates the dynamic neighborhood tensor $A^{S(l)}$, which improves the prediction performance and the dynamic response to the changes of the traffic network.

4.3 Dynamic Graph Convolution Module

Taking into account the interactions between multiple traffic features and the spatial interactions between neighboring nodes, we use dynamic graph convolutional crossover to extract spatial features in the road network. The dynamic graph convolution module can update the node features by aggregating the information between each node and its neighboring nodes to efficiently process data

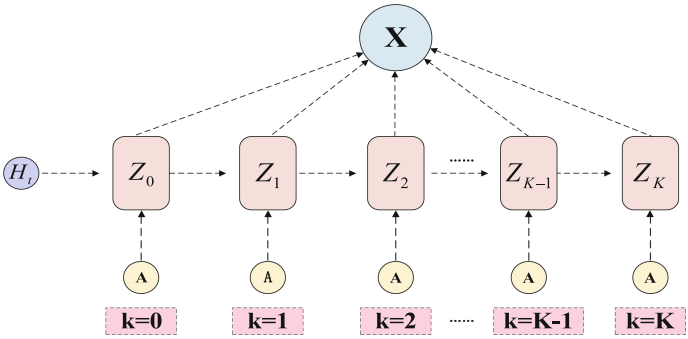


Fig. 3. The detailed architecture of dynamic graph convolution module.

with a dynamic graph structure. In our model, we design two parallel GCNs to process primary and auxiliary features, respectively, after a feature complementation strategy to enhance the accuracy of traffic prediction. This cross-strategy approach not only reveals the potential correlation between speed and volume but also increases the model's knowledge of complex traffic patterns, especially during peak hours and traffic congestion. Figure 3 shows the architecture of this module. We cross-pass H^S and H^O outputs from the temporal convolutional network and $A^{S(l)}$ and $A^{O(l)}$ generated by the dynamical graph learning module into the two dynamical graph convolutional modules. The GCNs are defined as follows:

$$\begin{aligned} Z_k^{S(l)} &= A^{O(l)} Z_{(k-1)}^{S(l)} (k > 0) \\ X^{S(l)} &= concat(Z_0^{S(l)}, Z_1^{S(l)}, \cdots, Z_K^{S(l)}) \end{aligned} \tag{6}$$

where $k \in [0, K]$, $Z_0^{S(l)} = H^{S(l)}$, and $X^{S(l)}$ is the output of the GCN for the primary feature. Specifically, the outputs of the dynamic graph convolution modules for the primary feature and auxiliary feature are denoted as $X^{S(l)}$ and $X^{O(l)}$, which will be fed into the multi-view fusion module.

4.4 Multi-view Fusion Module

To realize the interaction between the spatiotemporal correlation of different features and enhance the global correlation of a single node in the network, a multi-view weight fusion module is proposed to fuse the traffic feature information extracted from two dynamic graph convolution modules. By marking different weights on various features, highlighting the importance of primary features and the auxiliary role of auxiliary features, and improving the utilization rate of various data. The operation $l-$th layer is formulated as follows:

$$X_S^l = X^{S(l)} \odot \alpha + X^{O(l)} \odot \beta \tag{7}$$

where α, and β are the corresponding weights for the primary feature and auxiliary feature. $\alpha+\beta = 1$. The weight fusion module fuses multiple spatial-temporal features to get the final result.

4.5 Output Module

Our output module consists of two standard 1×1 convolution layers. The skip connection layer is partly a $1 \times L_i$ convolution that connects the hidden states of the primary feature of each module after the temporal convolution module. L_i is the length of the input sequence corresponding to i-th jump-connected layer, which normalizes the feature to have the same sequence length 1.

$$X = concat(X_S^0, X_S^1, \cdots, X_S^L) \tag{8}$$

Mean absolute error (MAE) is used to train the model:

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{n=1}^{N} \mid Y_n - \hat{Y}_n \mid \tag{9}$$

where θ denotes all learnable parameters in MVJGL, N is the number of samples, $\hat{Y}_n$ and Y_n are predicted values and ground truth of the n-th sample.

5 Experimental Studies

We conduct extensive experiments on four real work traffic datasets described in detail in Table.1. We aim to validate the performance of MVJGL against baseline methods toward answering the following questions:

- **RQ1.** Does the proposed MVJGL outperform existing baseline methods on traffic prediction problems?
- **RQ2.** Do all modules of the model benefit the overall performance of MVJGL? How does each module affect the model performance?
- **RQ3.** Does the proposed multiple-view scheme contribute to graph learning and improving prediction accuracy?

5.1 Datasets and Evaluation Metrics

Table 1 reports the detailed information of four datasets, PEMS-BAY [8], METR-LA [8], PEMSD4 [20] and PEMSD8 [20]. In particular, we choose traffic speed as the primary feature and traffic volume as the auxiliary feature. For PEMS-BAY and METR-LA datasets, 70% of data are set for training, 20% of data are used for testing, and 10% are used for validation. For PEMSD4 and PEMSD8, 60% of data are selected for training, 20% of data are used for testing, and the rest 20% are used for validation.

In the experiments, we adopt Absolute error (MAE), mean absolute percentage error (MAPE), and root mean square error (RMSE) to evaluate the

performance, which is formulated as follows:

$$
\begin{aligned}
MAE(Y,\hat{Y}) &= \frac{1}{N}\sum_{n=1}^{N} \mid Y_n - \hat{Y}_n \mid, \\
MAPE(Y,\hat{Y}) &= \frac{1}{N}\sum_{n=1}^{N} \frac{\mid Y_n - \hat{Y}_n \mid}{\mid Y_n \mid} \times 100\%, \\
RMSE(Y,\hat{Y}) &= \sqrt{\frac{1}{N}\sum_{n=1}^{N}(Y_n - \hat{Y}_n)^2},
\end{aligned}
\tag{10}
$$

where $\widehat{Y}$ and Y denote the predicted value and ground truth,respectively.

Table 1. Statistics of Datasets.

Dataset	#Nodes	#Time Steps	#Time Span
METR-LA	207	34272	4 months
PEMS-BAY	325	52116	5 months
PEMSD4	307	16969	2 months
PEMSD8	170	17833	2 months

5.2 Experimental Settings

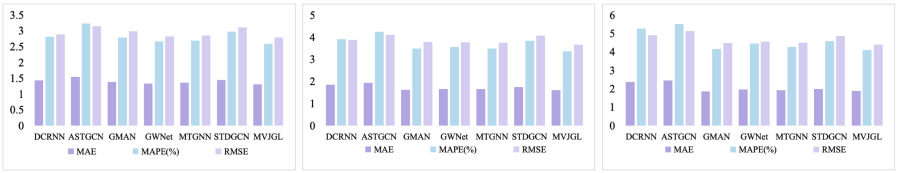
The number of stacked layers L is 4, and the dilated rates of the temporal convolution at each layer are [1,2,1,2,1,2,1,2]. The maximum depth of the dynamic graph convolution module is $k = 2$. We set weights in the fusion model as $\alpha = 0.7$, $\beta = 0.3$. The dimension of node-embedding $d = 32$. The channel size of the temporal dilated convolution is set to 32, and the hidden dimension of the graph learning module is set to 16. The Batch size is set to 64. Epochs are set to 200, early-stop scheme with a patience of 30 is adopted. Our MVJGL is trained by Adam optimizer with a learning rate of 0.001.

5.3 Baselines

To provide a comprehensive evaluation of the MVJGL method, we compare the performance with many representative baseline methods. We carefully choose the compared statistic approaches, and the representative and deep learning approaches.

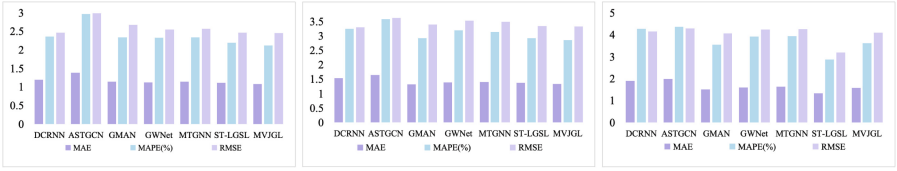
- **HA**: It refers to the historical average approach.
- **FC-LSTM** [21]: It integrates fully connected and LSTM structures.
- **ASTGCN** [20]: It is a model that combines attention mechanisms with convolutions.

- **STFGNN** [22]: It designs a spatial-temporal fusion graph by introducing a gated convolutional neural network module in order to capture local and global spatial and temporal correlations simultaneously.
- **AGCRN** [19]: It designs a powerful method for traffic prediction that employs graph convolution networks and recurrent networks to adaptively learn the graph structure and capture temporal dependencies in dynamic spatial-temporal data.
- **STGODE** [23]: It employs continuous graph neural network for traffic prediction.
- **DMSTGCN** [10]: It designs a multiple features-based graph convolution neural network for traffic forecasting.
- **DCRNN** [3]: It utilizes diffusion graph convolution and recursive neural networks.
- **GMAN** [18]: It adopts attention mechanism and transformer for prediction.
- **Graph WaveNet** [8]: It introduces a generation adaptive graph method and diffusion graph convolution.
- **MTGNN** [9]: It leverages graph neural networks to forecast multivariate time series.
- **STDGCN** [24]: It combines new graph generation methods and fusion strategies in spatial-temporal network architecture to capture the deep dynamic dependencies in the historical information of dynamic graphs.
- **ST-LGSL** [25]: It utilizes multi-layer perceptron and KNN graph generator to explore spatial-temporal features, a prediction module combining gated temporal convolution and diffusion graph convolution to optimize traffic prediction performance.



(a) 15min (b) 30min (c) 60min

Fig. 4. Performance comparing of baselines on the PEMSD4 dataset.



(a) 15min (b) 30min (c) 60min

Fig. 5. Performance comparing of baselines on the PEMSD8 dataset.

5.4 Comparison Results

Table 2 report the prediction results of MVJGL against multiple baseline methods for 15, 30, and 60-min traffic forecasting. We observe that our MVJGL obtains the best results in terms of MAE, RMSE, and MAPE, respectively. This significant performance improvement validates the effectiveness of our model in addressing challenges posed by traffic prediction problems. Based on the experimental results, the following observations are obtained.

(1) Deep learning-based models perform better than traditional methods because they can exploit complex and non-linear traffic features. As FC-LSTM is a recurrent neural network, it only takes temporal correlations and ignores spatial dependencies resulting in worse performance.
(2) Gnn-based models such as ASTGCN, AGCRN, STFGNN and DMSTGCN use graph neural networks to extract temporal and spatial dependencies in road networks, thus achieving better prediction performance. However, since the convolution operations of these models only aggregate the information of the adjacent nodes of each layer, they are not good at extracting the spatial-temporal dependencies of remote changes.
(3) STGODE combines graph embedding with ordinary differential equation method to improve the prediction accuracy. Both AGCRN and DMSTGCN make significant improvements when trying to learn dynamic graphs. However, because they rely on the global shared graph structure, none of them achieves satisfactory results.
(4) Our proposed model consistently outperforms all baselines on different datasets. For example, for the METR-LA, the best MAE of baselines is

Table 2. The comparative results on METR-LA and PEMS-BAY.

	15min			30min			60min		
	MAE	MAPE(%)	RMSE	MAE	MAPE(%)	RMSE	MAE	MAPE(%)	RMSE
METR-LA									
HA	4.16	13.00%	7.80	4.16	13.00%	7.80	4.16	13.00%	7.80
FC-LSTM	3.44	9.60%	6.30	3.77	10.09%	7.23	4.37	13.20%	8.69
ASTGCN	4.86	7.81%	9.27	5.43	10.13%	10.61	6.51	11.64%	12.52
STFGNN	3.26	8.04%	7.43	4.03	10.22%	9.44	5.02	13.03%	11.62
AGCRN	2.87	7.70%	5.58	3.23	9.00%	6.58	3.62	10.38%	7.51
STGODE	3.47	8.76%	6.76	4.36	11.14%	8.47	5.50	14.32%	10.33
DMSTGCN	2.85	7.54%	5.54	3.26	9.19%	6.56	3.72	10.96%	7.55
MVJGL(ours)	**2.80**	**7.40%**	**5.41**	**3.16**	**8.92%**	**6.42**	**3.53**	**10.21%**	**7.37**
PEMS-BAY									
HA	2.88	6.80%	5.59	2.88	6.80%	5.59	2.88	6.80%	5.59
FC-LSTM	2.05	4.80%	4.19	2.20	5.20%	4.55	2.37	5.70%	4.96
ASTGCN	1.52	3.22%	3.13	2.01	4.28%	4.27	2.61	6.00%	5.42
STFGNN	1.36	2.83%	2.81	1.67	**3.78%**	3.79	1.97	4.64%	4.52
AGCRN	1.35	2.87%	2.83	1.69	3.84%	3.81	1.96	4.67%	4.52
STGODE	1.43	2.99%	2.88	1.84	3.84%	3.90	2.30	4.61%	4.89
DMSTGCN	1.33	2.80%	2.83	1.67	3.81%	3.79	1.99	4.78%	4.54
MVJGL(ours)	**1.30**	**2.80%**	**2.76**	**1.63**	3.79%	**3.70**	**1.92**	**4.55%**	**4.40**

3.62 while MVJGL is 3.53 for 60-minute prediction results, the improvement is up to 2.48%. Our approach achieves these results as it concerns both primary feature and auxiliary features, and learns multiple views for capturing spatial-temporal dependencies. Furthermore, heterogeneous temporal convolution networks model multiple temporal dependencies from different features, which can be susceptible to data noise existing in sensed data and achieve the best performance. MVJGL further improves the prediction performance by learning dynamic graphs at different time steps, generating a continuous sequence of graphs.

Figures 4 and 5 visually show the comparison of different methods at different prediction intervals across the four publicly available data sets. We observe that when the prediction intervals increase, traffic forecasting tasks become more difficult, and thus the prediction accuracy of all models decreases. While the accuracy of MVJGL decreases the least in most cases, it implies that MVJGL gets the advantage of solving long-term forecast tasks. On the PEMSD8 dataset, the ST-LGSL model has a better long-term prediction effect than our model, but the ST-LGSL model is weaker than the MVJGL model in short and medium-term prediction tasks and is limited by the dataset. Furthermore, the prediction results of dynamic graph-based methods are significantly better than that of predefined graph-based ones (e.g., DCRNN, ASTGCN, etc.). Our model adopts multi-view-based dynamic graphs learning, its performance is significantly better than that of other global shared graphs-based models. We conclude that the use of a multi-view module is meaningful for improving traffic prediction results.

5.5 Ablation Study

Table 3. Ablation Study on METR-LA Dataset

Dataset	Models	(15 min/30 min/60 min)								
		MAE	MAPE	RMSE	MAE	MAPE	RMSE	MAE	MAPE	RMSE
METR-LA	MVJGL-HT	3.08	8.36%	6.00	3.73	10.72%	7.46	4.75	14.54%	9.26
	MVJGL-GT	2.86	7.59%	5.55	3.28	9.20%	6.61	3.72	10.85%	7.63
	MVJGL-TG	2.81	7.56%	5.47	3.19	9.10%	6.53	3.58	10.55%	7.47
	MVJGL-M	2.83	7.66%	5.51	3.21	9.10%	6.54	3.60	10.50%	7.45
	MVJGL-A	3.00	8.09%	5.87	3.62	10.42%	7.24	4.53	13.97%	8.93
	MVJGL-G	2.98	7.90%	5.89	3.59	10.22%	7.27	4.44	13.71%	8.97
	MVJGL-F	2.82	7.42%	5.46	3.20	8.95%	6.46	3.59	10.24%	7.39
	MVJGL(ours)	**2.80**	**7.40%**	**5.41**	**3.16**	**8.92%**	**6.42**	**3.53**	**10.21%**	**7.37**

This section validates the effectiveness of the key components in MVJGL and implements ablation experiments on the METR-LA dataset. We conduct the ablation study from two aspects, temporal dimension and spatial dimension.

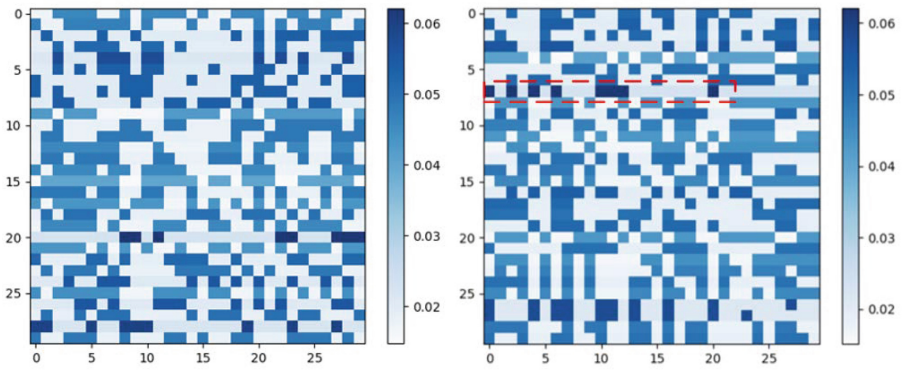
Temporal Dimension. For the temporal dimension, three variants of MVJGL are introduced.

- MVJGL-HT: It removes the temporal convolution module.
- MVJGL-GT: It removes the gated temporal convolution module.
- MVJGL-TG: It removes the gated temporal recurrent unit module.

Spatial Dimension. For the spatial dimension, four variants of MVJGL are presented

- MVJGL-M: It removes the multi-view learning module and uses a global shared graph.
- MVJGL-A: It removes the auxiliary feature module.
- MVJGL-G: It removes the dynamic graph convolution module.
- MVJGL-F: It removes the fusion module and adds the outputs of two GCNs.

All the variant models adopt the same settings as MVJGL except for the differences introduced above. The MAE, RMSE, and MAPE results for the METR-LA are presented in Table 3. The results demonstrate that all key components of MVJGL contribute significantly to traffic prediction. In particular, the temporal convolution module(MVJGL-HT) has the greatest influence on the results, it is because the parallel heterogeneous convolution structure is efficient in capturing multiple-scale temporal dependencies, especially for long-term prediction. The design of the auxiliary feature (MVJGL-A) in the model also has a great influence on the prediction performance. This is because the function of solving the data noise problem through multiple features fails. It is found that the multi-view dynamic convolution module design (MVJGL-G) also has a significant impact on the model performance, especially on the medium and long-term prediction results. This verifies the effectiveness of multiple views in spatial dependency



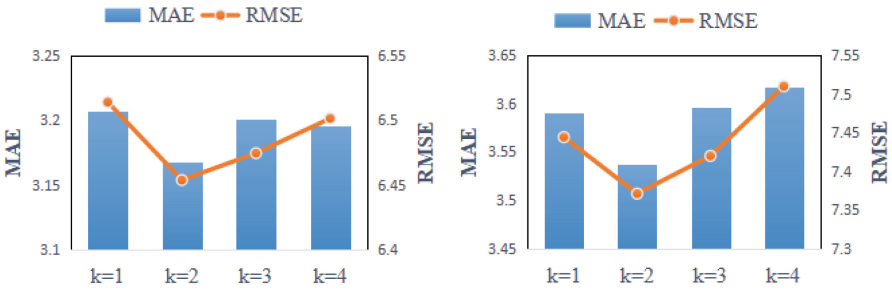
(a) PEMSD4 (b) PEMSD8

Fig. 6. Visualization of the graph structure for the first 30 nodes.

extraction. The fusion module fuses the primary and auxiliary features with different weights to improve the utilization of various data and improve the performance of the model.

5.6 Effects of Multi-view and Dynamic Graph Learning

We further investigate the robustness of multi-view design in improving prediction results. MVJGL can effectively extract spatial features from noisy traffic data even if it only relies on the primary features. The heatmap analysis of dynamic adjacency matrices from Fig. 6 further indicates that the graph learning module proficiently captures spatial dependencies sensitive to temporal changes, thereby improving graph representation. Figure 7 shows the variation of the hyperparameter, with a convolution kernel size of 2 producing the best performance, deviations from this value can lead to reduced accuracy and excessive smoothing problems.



(a) 30 min (b) 60 min

Fig. 7. Hyperparameter effects of graph convolution sizes k on METR-LA dataset.

6 Conclusion

This paper introduces a spatio-temporal heterogeneous graph neural network with multi-view learning for traffic prediction. Unlike traditional approaches based on predefined graphs or models built on globally shared graphs, we utilize multiple traffic features to learn multiple continuous dynamic graphs for the complex spatial correlation representation. Meanwhile, MVJGL models and fuses information from multiple traffic features to capture potential hidden connections among nodes of traffic networks, thus further improving the performance of the traffic prediction task. Extensive experimental results on various traffic datasets demonstrate the superior prediction results of our proposed model.

In our future work, we will focus on the following two aspects:

– Apply our model to solve spatio-temporal prediction problems in other domains such as energy, stock market, etc.
– Consider incorporating more traffic features (e.g., pedestrian flow, traffic signals, major festivals, traffic accidents, etc.) into the model to further improve the accuracy of the model's prediction results.

References

1. Qureshi, K.N., Abdullah, A.H.: A survey on intelligent transportation systems. Middle-East J. Sci. Res. **15**(5), 629–642 (2013)
2. Yuan, Z., Zhou, X., Yang, T.: Hetero-convlstm: a deep learning approach to traffic accident prediction on heterogeneous spatio-temporal data. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 984–992 (2018)
3. Li, Y., Yu, R., Shahabi, C., Liu, Y.: Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. arXiv preprint arXiv:1707.01926 (2017)
4. Zhao, Z., Chen, W., Wu, X., Chen, P.C., Liu, J.: Lstm network: a deep learning approach for short-term traffic forecast. IET Intel. Transp. Syst. **11**(2), 68–75 (2017)
5. Gao, Y., Zhao, J., Qin, Z., Feng, Y., Yang, Z., Jia, B.: Traffic speed forecast in adjacent region between highway and urban expressway: based on mfd and gru model. J. Adv. Transp. **2020**, 1–18 (2020)
6. Zhang, K., He, F., Zhang, Z., Lin, X., Li, M.: Graph attention temporal convolutional network for traffic speed forecasting on road networks. Transportmetrica B: Transp. Dyn. **9**(1), 153–171 (2021)
7. Peng, H., et al.: Dynamic graph convolutional network for long-term traffic flow prediction with reinforcement learning. Inf. Sci. **578**, 401–416 (2021)
8. Wu, Z., Pan, S., Long, G., Jiang, J., Zhang, C.: Graph wavenet for deep spatial-temporal graph modeling. arXiv preprint arXiv:1906.00121 (2019)
9. Wu, Z., et al.: Connecting the dots: multivariate time series forecasting with graph neural networks. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 753–763 (2020)
10. Han, L., Du, B., Sun, L., Fu, Y., Lv, Y., Xiong, H.: Dynamic and multi-faceted spatio-temporal deep learning for traffic speed forecasting. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 547–555 (2021)
11. Shin, Y., Yoon, Y.: Pgcn: progressive graph convolutional networks for spatial-temporal traffic forecasting. arXiv preprint arXiv:2202.08982 (2022)
12. Williams, B.M., Hoel, L.A.: Modeling and forecasting vehicular traffic flow as a seasonal arima process: theoretical basis and empirical results. J. Transp. Eng. **129**(6), 664–672 (2003)
13. Quach, K.N.D., et al.: Short-term traffic speed prediction using hybrid lstm-svr model. In: International Conference on Robot Intelligence Technology and Applications, pp. 438–450. Springer, Heidelberg (2022)
14. Lütkepohl, H.: Vector autoregressive models. In: Handbook of Research Methods and Applications in Empirical Macroeconomics, pp. 139–164. Edward Elgar Publishing (2013)
15. Han, H., et al.: Stgcn: a spatial-temporal aware graph learning method for poi recommendation. In: 2020 IEEE International Conference on Data Mining (ICDM), pp. 1052–1057. IEEE (2020)

16. Song, C., Lin, Y., Guo, S., Wan, H.: Spatial-temporal synchronous graph convolutional networks: a new framework for spatial-temporal network data forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 01, pp. 914–921 (2020)
17. Chen, W., Chen, L., Xie, Y., Cao, W., Gao, Y., Feng, X.: Multi-range attentive bicomponent graph convolutional network for traffic forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 04, pp. 3529–3536 (2020)
18. Zheng, C., Fan, X., Wang, C., Qi, J.: Gman: a graph multi-attention network for traffic prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 01, pp. 1234–1241 (2020)
19. Bai, L., Yao, L., Li, C., Wang, X., Wang, C.: Adaptive graph convolutional recurrent network for traffic forecasting. Adv. Neural Inf. Process. Syst. **33**, 17804–17815 (2020)
20. Guo, S., Lin, Y., Feng, N., Song, C., Wan, H.: Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01, pp. 922–929 (2019)
21. Shi, X., Chen, Z., Wang, H., Yeung, D.Y., Wong, W.K., Woo, W.C.: Convolutional LSTM network: a machine learning approach for precipitation nowcasting. Adv. Neural Inf. Process. Syst. **28** (2015)
22. Li, M., Zhu, Z.: Spatial-temporal fusion graph neural networks for traffic flow forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, no. 5, pp. 4189–4196 (2021)
23. Fang, Z., Long, Q., Song, G., Xie, K.: Spatial-temporal graph ode networks for traffic flow forecasting. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 364–373 (2021)
24. Yin, X., Zhang, W., Zhang, S.: Spatiotemporal dynamic graph convolutional network for traffic speed forecasting. Inf. Sci. **641**, 119056 (2023)
25. Tang, J., Qian, T., Liu, S., Du, S., Hu, J., Li, T.: Spatio-temporal latent graph structure learning for traffic forecasting. In: 2022 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2022)

BotSCL: Heterophily-Aware Social Bot Detection with Supervised Contrastive Learning

Qi Wu[1], Yingguang Yang[1], Buyun He[1], Hao Liu[1], Renyu Yang[2], and Yong Liao[1(✉)]

[1] University of Science and Technology of China, Hefei 230026, Anhui, China
qiwu4512@mail.ustc.edu.cn, yliao@ustc.edu.cn
[2] Beihang University, Beijing, China

Abstract. Detecting social bots, which continuously evolve, presents an escalating challenge. Although graph-based detection techniques utilize various relationships within social networks to model node behavior, they often fail to account for inherent heterophily–connections between different types of accounts. When message passing occurs across heterophilous edges, it can cause feature blending between bots and legitimate users, leading to indistinct representations. To address this issue, we propose BotSCL, a contrastive learning framework that is aware of heterophily. BotSCL adapts by differentiating between representations of heterophilous neighbors while aligning representations of homophilous ones. Our approach employs two graph augmentation strategies to create varied graph views and introduces a channel-wise, attention-free encoder to address the limitations of traditional neighbor information aggregation. Supervised contrastive learning then helps the encoder focus on aggregating information specific to each class. Extensive experiments on two real-world social bot detection datasets reveal that BotSCL outperforms existing baseline models, including advanced bot detection methods, as well as techniques based on partial heterophily and graph contrastive learning.

Keywords: homophily and heterophily · social bot detection · supervised contrastive learning

1 Introduction

Social bots, which are automated accounts often used for malicious activities like spreading misinformation [9], promoting extremism [16], and interfering in elections [10], continue to present a major challenge on social networks. These bots are highly adaptable, constantly evolving to bypass new detection methods. In response, numerous bot detection technologies have been developed.

Twitter account analysis typically involves gathering a wide range of data types, such as tweet text [19], metadata [3,34], and temporal activity patterns [6]. This holistic approach provides a comprehensive understanding of

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 53–68, 2025.
https://doi.org/10.1007/978-3-031-78183-4_4

account behavior. The extracted features are then integrated into various classifiers, utilizing deep neural networks with different architectures to enhance model generalization [32]. Recent advancements in graph neural network-based methods [2,11] have driven progress in understanding semantic relationships by constructing heterogeneous graphs and employing relational graph transformers that aggregate insights from both intra-relational and inter-relational contexts.

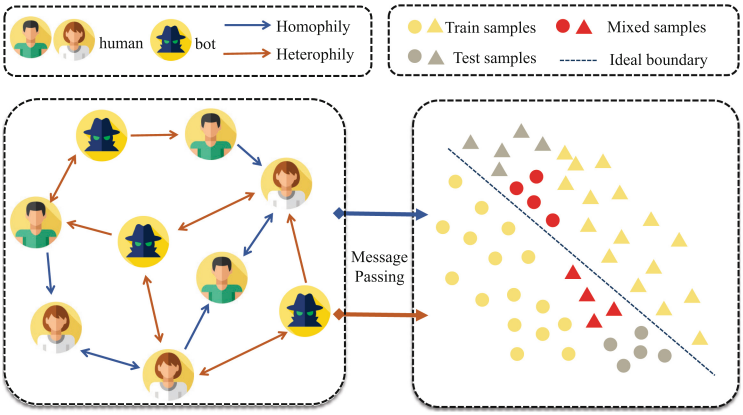


Fig. 1. Illustration of heterophily influence. Message passing on heterophilous edges leads to feature mixing and classification boundary shift in social bot detection.

Despite the success of graph-based detection methods, many fail to account for heterophily, which refers to connections between different classes of accounts. Recent studies [23,31] show that Twitter bots often engage with human users more frequently, challenging the conventional view that bots primarily interact with each other to boost their influence [9]. As illustrated in Fig. 1, when both homophilous and heterophilous interactions are present, message passing along heterophilous edges can cause node representations to blend with those of the opposite class. Therefore, an effective graph-based detection approach should prevent inter-class feature blending to produce more distinct representations. Our method's key insight is to use supervised contrastive learning [17] to improve inter-class discrimination. By treating nodes of the same class as positive pairs and those of different classes as negative pairs, we can promote the aggregation of class-specific information while avoiding feature mixing.

Building on these insights, we present a novel framework called Social Bot Detection with Supervised Contrastive Learning (BotSCL). First, we develop two graph augmentation strategies–feature augmentation and topological structure augmentation–to create different graph views. Then, we design an encoder to aggregate similar and distinct information across feature channels. Finally, supervised contrastive learning is applied in a cross-view manner to generate class-consistent representations.

This paper makes the following key contributions:

- We introduce and analyze the negative impact of heterophily on social bot detection, supported by experimental data.
- We propose the BotSCL framework, which mitigates the effects of heterophily on social bots in the feature space through supervised contrastive learning.
- Extensive experiments on two real-world datasets demonstrate that BotSCL consistently outperforms all baseline methods.

2 Related Work

In this section, we will explore related research in three key areas: graph-based social bot detection, graph neural networks (GNNs) for heterophilous graphs, and contrastive learning.

2.1 Graph-Based Social Bot Detection

Graph-based techniques for social bot detection have gained significant traction in recent research. [2] pioneered the use of graph convolutional networks (GCN), combining account features with the structural properties of relation graphs to enhance detection. Satar [13] further refined this approach by employing GCNs for feature engineering and integrating self-supervision to detect bots. Building on this, Relational Graph Convolutional Networks (R-GCNs) [26] were adopted by [15] to aggregate information from multiple relationships, and this was later enhanced by [11], who introduced additional relations and applied graph transformers for more adaptive neighbor aggregation. RoSGAS [36] leverages heterogeneous information networks to model diverse entities and relationships in social networks effectively, using adaptive reinforcement learning to improve performance. These methods generally assume that bots and humans interact more within their respective classes, and the smoothing of node representations is advantageous for classification. However, in reality, advanced bots can evade detection by actively engaging with human users, making this assumption less reliable.

2.2 GNNs for Graphs with Heterophily

Given the widespread presence of heterophily, GNNs designed for heterophilous graphs have become a significant focus of research. Broadly, there are two primary approaches: Aggregation of non-local neighbor information: Techniques in this category gather information from higher-order neighbors [1,38] or from potential same-class neighbors [25,30] to enhance intra-class information aggregation. Adaptive Message Passing: Approaches like FAGCN [5] combine high- and low-frequency information, while GPRGNN [8] utilizes learnable weights to aggregate information from different hop distances. In social bot detection, bots tend to display more heterophilous behavior, while humans generally exhibit homophilous tendencies. Consequently, directly applying GNNs designed for heterophilous graphs to bot detection may not deliver optimal outcomes due to the differing behaviors of bots and humans.

2.3 Contrastive Learning

Contrastive learning focuses on training an encoder that produces consistent representations across different views of the data. Graph contrastive learning (GCL) extends this approach to the graph domain, allowing nodes to learn representations in a self-supervised manner. For instance, GRACE [39] generates augmented graph views by removing edges and masking features, treating the same node in different views as a positive pair. DGI [29] optimizes node representations by maximizing mutual information between local and global embeddings. Supervised contrastive learning, first applied in the field of computer vision [17], treats instances of the same class as positive pairs and those of different classes as negative pairs. This approach ensures that embeddings from the same class are pulled closer together while those from different classes are pushed apart.

3 Methodology

The complete pipeline of BotSCL is illustrated in Fig. 2. The process begins with the application of two graph augmentation techniques to produce two distinct graph views. Next, nodes aggregate similar information from their homophilous neighbors while adaptively distinguishing the representations of heterophilous neighbors for each relation. Finally, node representations are optimized through supervised contrastive learning.

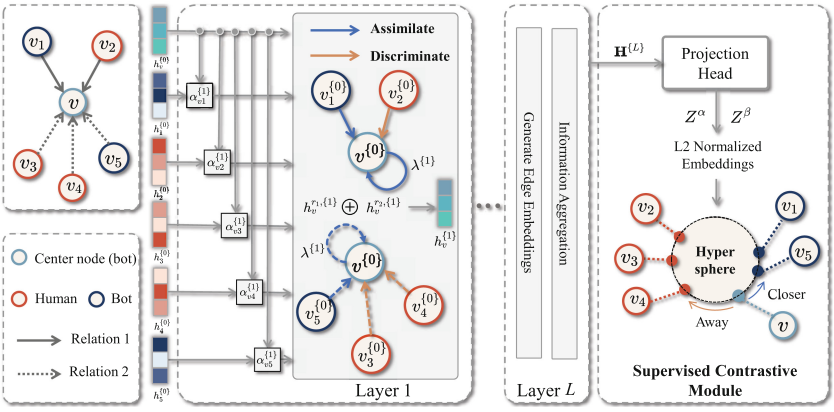


Fig. 2. The proposed BotSCL framework.

3.1 Graph Augmentation

Graph contrastive learning (GCL) leverages various graph augmentation techniques, such as edge addition, feature masking, and personalized PageRank diffusion, to create diverse graph views [37]. However, it's important to recognize

that not all augmentation methods are suitable for graphs with heterophily. For instance, [35] utilizes low-pass and high-pass filters to generate graph views for self-supervised contrastive learning, while [21] classifies edges as either within-class or between-class, then constructs homophilous and heterophilous views accordingly. Despite their effectiveness, both methods face challenges, including potential information loss and the introduction of noise.

To address these limitations and enable the simultaneous handling of both homophilous and heterophilous edges without the need for explicit differentiation, we propose a new graph augmentation technique called class-aware node shuffling (CNS). This method involves randomly swapping nodes within the same class. By doing so, we can obtain representations that remain stable despite changes in the neighborhood, without significantly disrupting the underlying graph structure. This is achieved through intra-class feature swapping:

$$\widetilde{\mathcal{X}} = \mathcal{C}(\mathcal{X}, \mathrm{Y}_{train}). \tag{1}$$

Additionally, we employed a traditional graph augmentation method, edge removing (ER), to augment the graph structure. The edge removal method can be formulated as follows:

$$\tilde{\mathcal{E}} = \mathcal{E} \odot M^{\mathcal{E}}, \tag{2}$$

where $\mathcal{E}$ is total edge set, each element of $M^{\mathcal{E}}$, $m_{ij}^{\mathcal{E}}$ stands for the mask value of edge e_{ij} and $m_{ij}^{\mathcal{E}} \sim \mathcal{B}(1 - pe)$, pe is the probability to be removed. Through the aforementioned two graph augmentation methods, both the features and topological structure of the original graph are augmented. Note that graph augmentation is optional for BotSCL, as supervised contrastive learning can be employed in a single-pass manner.

3.2 Aggregation Strategy

Multi Modal Feature Fusion. Multiple modalities, such as tweets, metadata, and account descriptions, have been shown to significantly improve social bot detection [11,15]. Unlike traditional node classification tasks, social bot detection relies on features that span diverse types and dimensions of social content. Therefore, it is essential to use a multi-layer perceptron (MLP) to align the dimensions of these heterogeneous feature vectors. Following prior work [11,15], we first extract numerical features x_i^{cat} and boolean features x_i^{bool} for each node v_i. Additionally, RoBERTa [20] is employed to extract descriptive features x_i^{des} from the account descriptions. Furthermore, RoBERTa is also used to obtain average feature representations x_i^{tweet} from up to 20 tweets per account.

In this study, we further use TransformerEncoder [27] for feature fusion adaptively by treating the feature vectors of different types as token embeddings:

$$x_i^0 = TransformerEncoder([x_i^{cat}, x_i^{bool}, x_i^{des}, x_i^{tweet}]), \tag{3}$$

where x_i^0 is the concatenation of the TransformerEnocder outputs. In this way, benefiting from the self-attention and residual mechanisms within Transformer,

a better fusion of information from different modalities is achieved, effectively countering feature camouflage of social bots. On top of that, we apply another MLP to x_i^0 and obtain the input of the graph convolutional layer $h_i^{\{0\}}$:

$$h_i^{\{0\}} = \sigma(W_I x_i^0 + b_I), \tag{4}$$

where W_I, b_I are learnable parameters and $\sigma(\cdot)$ is a nonlinear activation function.

Information Aggregation Beyond Homophily. After generating augmented views, BotSCL employs an encoder capable of aggregating information from both similar and dissimilar neighbors in a distinguishable way to produce effective node representations. In the spectral domain, GNNs that operate under the homophily assumption function as low-pass filters. However, previous research on graphs with heterophily [22] has demonstrated that high-pass filters, which capture differential information, are more suitable for modeling heterophilous connections.

In the spatial domain, low-pass filtering is achieved by aggregating information from neighboring nodes, while high-pass filtering is performed by distinguishing between the representations of different neighbors. These two filtering operations can be mathematically formulated as follows:

$$\begin{aligned} \left(h_i^{\{l\}}\right)_{Low} &= W^{\{l\}} \sum_{j \in \mathcal{N}_{(i)}} \left(h_i^{\{l-1\}} + h_j^{\{l-1\}}\right), \\ \left(h_i^{\{l\}}\right)_{High} &= W^{\{l\}} \sum_{j \in \mathcal{N}_{(i)}} \left(h_i^{\{l-1\}} - h_j^{\{l-1\}}\right). \end{aligned} \tag{5}$$

In addition to these two, there are other filters that retain different frequency band information, such as band-pass filters. Although [22] employs a channel-wise attention mechanism to fuse low-pass, high-pass filters, and self-information, there still exists information loss from other frequencies. Inspired by the above, we design a channel-wise frequency-adaptive mechanism to aggregate similar information from homophilous neighbors and differential information from heterophilous neighbors. Specifically, given a central node $v_i \in V$ and its arbitrary neighbor $v_j \in N_r(v_i)$, we first use a linear transformation and a separate element-wise multiplication across channels to obtain the query and key:

$$\begin{aligned} q_i^{\{l\}} &= W_A^{\{l\}} h_i^{\{l-1\}} \odot Q^{\{l\}}, \\ k_j^{\{l\}} &= W_A^{\{l\}} h_j^{\{l-1\}} \odot K^{\{l\}}, \end{aligned} \tag{6}$$

where $W_A^{\{l\}} \in \mathbb{R}^{d_{l-1} \times d_{l-1}}$ is the weight martix of layer l, $Q^{\{l\}}, K^{\{l\}} \in \mathbb{R}^{1 \times d_{l-1}}$ are weight vectors for query and key and $\odot$ denotes the Hadamard product operation. $q_j^{\{l\}}$ and $k_i^{\{l\}}$ can also be calculated in the same way.

Then we calculate the channel-wise and pass-free attention coefficient $\alpha_{ij}^{\{l\}}$ for edge e_{ij}:

$$\alpha_{ij}^{\{l\}} = tanh\left(\frac{q_i^{\{l\}} \odot k_j^{\{l\}} + q_j^{\{l\}} \odot k_i^{\{l\}}}{2}\right). \tag{7}$$

It is worth noting that the obtained $\alpha_{ij}^{\{l\}}$ using the above calculation method is direction- and relation-agnostic and can be also seen as the embedding of edge e_{ij}. Furthermore, due to the use of the tanh activation function, any element in $\alpha_{ij}^{\{l\}}$ is in the range of [–1, 1], which breaks the previous restriction on the sum of neighbor information.

Finally, we aggregate information from the neighbors using the generated channel-wise weights $\alpha_{ij}^{\{l\}}$ to obtain l layer node representation $h_i^{\{l\}}$:

$$h_i^{r,\{l\}} = W_r^{\{l\}}(\lambda^{\{l\}} h_i^{\{l-1\}} + \sum_{j \in N_r(i)} \frac{\alpha_{ij}^{\{l\}}}{|N_r(i)|} \odot h_j^{\{l-1\}}), \tag{8}$$

$$h_i^{\{l\}} = \frac{1}{R} \sum_{r=1}^{R} h_i^{r,\{l\}}, \tag{9}$$

where $W_r^{\{l\}} \in \mathbb{R}^{d_{l-1} \times d_l}$ is the weight matrix for relation r, and $|N_r(v_i)|$ is the number of neighbor nodes on relation r. Following the graph filter proposed by [5], we apply a hyperparameter $\lambda^{\{l\}}$ to preserve the information of the node itself. RGT [11] uses an attention mechanism to fuse information from different relations, but here we trivially take the average of information from different relations to avoid information missing.

3.3 Supervised Contrastive Optimization

Following the previous contrastive learning framework [29,37], we use a projection head consisting of two MLP layers to obtain z_i:

$$z_i = W_2 \sigma(W_1 h_i^{\{L\}} + b_1) + b_2, \tag{10}$$

where $h_i^{\{L\}}$ is the output of last layer L. Thus, we can obtain projections z_i^{α} and z_i^{β} of node v_i in two graph views $\mathcal{G}^{\alpha}$ and $\mathcal{G}^{\beta}$ in respect.

Next, we employ supervised contrastive learning as the loss function for training. However, because this approach considers all nodes of the same class across different views as positive pairs, it can lead to overfitting. In this case, the representations of nodes from the same class become overly similar, impairing the model's ability to generalize to unseen nodes. To mitigate overfitting, we implement supervised contrastive learning in a cross-view manner. For a randomly chosen set of N samples from the training nodes, we first project each node through the graph augmentation, encoder, and projection head described earlier. Taking node v_i from graph view $\mathcal{G}^{\alpha}$ as an example, we consider its projection and the projections of same-class nodes from other views as positive pairs, while projections from different-class nodes are treated as negative pairs, to compute the contrastive loss.

$$\mathcal{L}_i^{\alpha} = -\frac{1}{N_{y_i}} \sum_{j=1}^{N} 1_{y_i = y_j} \cdot log \frac{e^{cos(z_i^{\alpha}, z_j^{\beta})/\tau}}{\sum_{k=1}^{N} e^{cos(z_i^{\alpha}, z_k^{\beta})/\tau}}, \tag{11}$$

Algorithm 1: The first training process of BotSCL

Input : a directed and multi-relation graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{X}\}$, the labels of train nodes $\mathrm{Y}_{\mathrm{train}}$, Training epochs N_{epochs}, the number of layers in the encoder L

Output: node representations $\mathbf{H}$

initialization;
generate two graph views $\mathcal{G}^{\alpha}$ and $\mathcal{G}^{\beta} \leftarrow$ Eq. (1–2);
for $e = 1, \cdots, N_{epochs}$ **do**
  obtain $\mathbf{H}^{\{0\}} \leftarrow$ Eq. (3–4);
  for *each graph view* **do**
    for $l = 1, \cdots, L$ **do**
      $q_i^{\{l\}}, q_j^{\{l\}}, k_i^{\{l\}}, k_j^{\{l\}} \leftarrow$ Eq. (6);
      $\alpha_{ij}^{\{l\}} \leftarrow$ Eq. (7);
      for $r = 1, \cdots, R$ **do**
        $h_i^{r,\{l\}} \leftarrow$ Eq. (8);
      end
      $h_i^{\{l\}} \leftarrow$ Eq. (9);
    end
    obtain node representations $\mathbf{H}^{\{L\}}$;
  end
  $z_i^{\alpha}, z_i^{\beta} \leftarrow$ Eq. (10);
  $\mathcal{L} \leftarrow$ Eq. (11–12);
  update parameters through backpropagation;
end
return $\mathbf{H} = \left[\mathbf{H}^{\{0\}} \| \mathbf{H}^{\{L\}}\right]$

where N_{y_i} represents the number of samples in the same class as node v_i among N samples, $cos(\cdot, \cdot)$ function is used to calculate the cosine similarity, and τ is the temperature coefficient which can regulate the degree of distribution uniformity.

Finally, we calculate the loss for all nodes in the sampled set of two views in the same way, and take the average:

$$\mathcal{L} = \frac{1}{2N} \sum_{i=1}^{N} (\mathcal{L}_i^{\alpha} + \mathcal{L}_i^{\beta}). \tag{12}$$

Training Strategy. In this study, we adopt a two-stage training approach similar to previous contrastive methods [29,37,39]. In the first stage, we use the previously described method to generate node representations and update the model parameters by minimizing $\mathcal{L}$. Given that the original features preserve important class information [7], we proceed to the second stage by concatenating the encoder input $\mathbf{H}^{\{0\}}$ with the output $\mathbf{H}^{\{L\}}$: $\mathbf{H} = \left[\mathbf{H}^{\{0\}} \| \mathbf{H}^{\{L\}}\right] \in \mathbb{R}^{n \times (d_0 + d_L)}$. In this stage, we utilize a straightforward machine learning classifier, namely Logistic Regression (LR), for both training and evaluation using $\mathbf{H}$.

4 Experiments

4.1 Experiment Setup

Datasets. TwiBot-20 [14] and **TwiBot-22** [12], which include user followers and following relations, are well-suited for our method and subsequent experiments. TwiBot-20 consists of 229,580 nodes and 227,979 edges, while TwiBot-22 includes one million nodes and 3,743,634 edges. We partition the datasets into training, validation, and test sets using a 7:2:1 ratio.

Baselines. To evaluate the effectiveness of BotSCL, we compare it with several different methods, described briefly below:

Homophilous GNNs: GCN [18] and **GAT** [28] are representative GNNs based on the homophily assumption, where information is aggregated by summing up neighbor representations.

Heterophilous GNNs: H2GCN [38], **FAGCN** [5], and **GPRGNN** [8] are models specifically designed to address heterophilous edges by employing various information aggregation techniques.

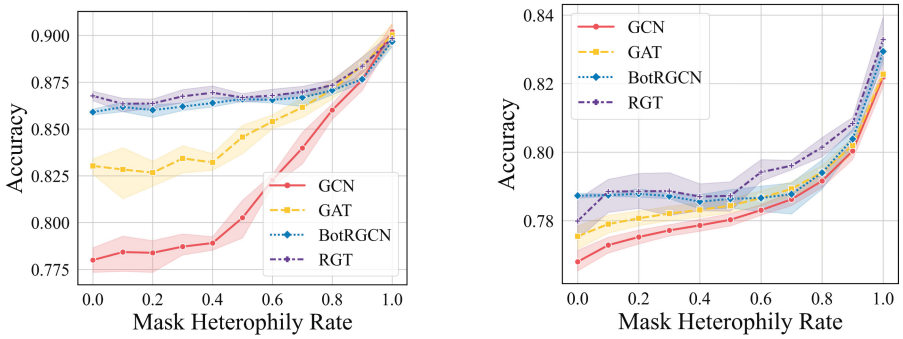
Graph-Based Social Bot Detection: Alhosseini et al. [2], **EvolveBot** [33], **Moghaddam et al.** [24], **BotRGCN** [15], and **RGT** [11] leverage graph structures to improve social bot detection.

Graph Contrastive Learning: DGI [29], **GRACE** [39], and **GBT** [4] are prominent self-supervised graph contrastive learning frameworks that learn node representations without labels. **SupCon** [17] refers to supervised contrastive loss, which we implement by adapting the loss function of GRACE.

Hyperparameter Setting. We use a GNN with two layers, each having a hidden dimension of 32. To mitigate overfitting, we apply a dropout mechanism with a rate of 0.5. The temperature coefficient is set to 0.07 to enhance the distribution of the representation space.

4.2 Heterophily Evidence and Influence

Before investigating the impact of heterophily on social bot detection, we first analyze the levels of homophily and heterophily in the datasets. In the TwiBot-22 dataset, the homophily ratios for follower and following relationships among human nodes are 88.05% and 96.20%, respectively, whereas for bot nodes, these ratios are 16.55% and 6.25%, respectively. This disparity suggests a clear pattern where social bots are more inclined to interact with human nodes.



(a) Mask heterophily on TwiBot-20 (b) Mask heterophily on TwiBot-22

Fig. 3. Heterophily influence on previous graph-based methods.

To illustrate the adverse effects of heterophily, we plot the accuracy changes as heterophilous edges are progressively removed in increments of 0.1, as shown in Fig. 3. We evaluate four different models: GCN, GAT, BotRGCN, and RGT. The results demonstrate a significant increase in accuracy for all models as heterophily decreases. This underscores the importance of considering both homophilous and heterophilous edges in graph-based social bot detection.

4.3 Performance Comparison

Table 1 provides a summary of the detection results for all baselines and BotSCL on the TwiBot-20 and TwiBot-22 datasets. The performance metrics for homophilous GNNs and graph-based social bot detection methods are derived from an empirical survey of social bot detection [12] and additional experiments conducted by us. BotSCL outperforms all 14 baseline methods in terms of both Accuracy and F1-score.

As shown in Table 1, heterophilous GNNs generally outperform homophilous GNNs, suggesting that accounting for heterophilous edges can help identify sophisticated bots with extensive human connections. Compared to heterophilous GNNs, BotSCL not only incorporates various types of relations but also employs supervised contrastive loss as the optimization objective, resulting in improvements in both Accuracy and F1-score across both datasets. Self-supervised contrastive learning methods perform less effectively on TwiBot-20, which has a large number of unlabeled nodes, but show better results on TwiBot-22. This discrepancy is likely due to uneven training distributions, as fewer training nodes are used for TwiBot-20 in the second training stage. Our method significantly outperforms these self-supervised approaches, highlighting the crucial role of supervised signals in effectively handling both homophilous and heterophilous edges.

Table 1. Performance comparison on TwiBot-20 and TwiBot-22 in terms of accuracy and F1-score. The best and second-best results are highlighted with **bold** and underline.

Methods	TwiBot-20		TwiBot-22	
	Accuracy	F1-score	Accuracy	F1-score
GCN	77.53 ± 1.73	80.86 ± 0.86	80.07 ± 0.77	51.71 ± 4.05
GAT	83.27 ± 0.56	85.25 ± 0.38	85.07 ± 1.19	69.32 ± 4.02
FAGCN	85.43 ± 0.40	87.36 ± 0.32	88.11 ± 1.43	77.43 ± 3.20
H2GCN	85.84 ± 0.34	87.57 ± 0.15	89.09 ± 1.16	79.99 ± 1.53
GPRGNN	86.05 ± 0.34	87.50 ± 0.30	89.07 ± 1.20	80.48 ± 1.62
Alhosseini et al.	59.88 ± 0.59	72.07 ± 0.48	47.72 ± 8.71	38.10 ± 5.93
EvolveBot	65.83 ± 0.64	69.75 ± 0.51	71.09 ± 0.04	14.09 ± 0.09
Moghaddam et al.	74.05 ± 0.80	77.87 ± 0.71	73.78 ± 0.01	32.07 ± 0.03
BotRGCN	85.75 ± 0.69	87.25 ± 0.74	79.66 ± 0.14	57.50 ± 1.42
RGT	86.57 ± 0.42	88.01 ± 0.42	76.47 ± 0.45	42.94 ± 0.49
DGI	84.93 ± 0.31	87.09 ± 0.36	79.61 ± 0.13	44.06 ± 1.52
GRACE	84.74 ± 0.88	86.90 ± 0.84	80.02 ± 0.91	46.17 ± 4.48
GBT	84.74 ± 0.92	86.87 ± 0.79	79.75 ± 0.76	47.27 ± 3.08
SupCon	86.10 ± 0.14	87.67 ± 0.16	80.00 ± 0.24	44.41 ± 3.83
BotSCL	$\mathbf{87.26 \pm 0.31}$	$\mathbf{88.79 \pm 0.27}$	$\mathbf{82.39 \pm 0.50}$	$\mathbf{61.53 \pm 1.45}$

4.4 Ablation Study

We conduct an ablation study to assess the effects of various modules and graph augmentation techniques on social bot detection. We develop three ablation models: one without self-supervised learning (w/o Sup), one lacking negative information aggregation (w/o Neg), and one utilizing cross-entropy loss. The results of these ablation models on TwiBot-20 and TwiBot-22 are summarized in Table 2.

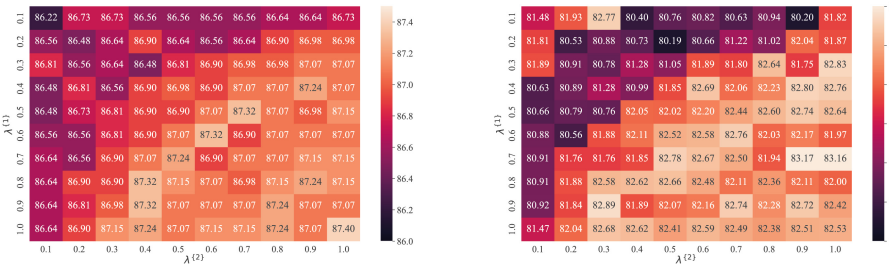
The results indicate a significant decline in performance when self-supervised learning is omitted, highlighting its essential role in effective information aggregation during the message-passing process. Although the model without negative information aggregation still performs relatively well due to the strength of supervised contrastive learning, its effectiveness is reduced compared to the full model. In contrast, the model using cross-entropy loss performs noticeably worse, as it mainly focuses on the commonalities among training samples while neglecting exceptional cases.

Additionally, different graph augmentation methods affect model performance in varying degrees. Notably, CNS and edge removal have substantial impacts, with CNS being crucial for generating category-invariant representations. On the other hand, augmentation methods such as feature masking and

Table 2. Ablation Study in terms of Different Modules and Graph Augmentation Methods.

Settings	TwiBot-20		TwiBot-22	
	Accuracy	F1-score	Accuracy	F1-score
full model	$\mathbf{87.26 \pm 0.31}$	$\mathbf{88.79 \pm 0.27}$	$\mathbf{82.39 \pm 0.50}$	$\mathbf{61.53 \pm 1.45}$
w/o Sup	86.45 ± 0.21	88.45 ± 0.19	82.00 ± 0.61	56.20 ± 2.88
w/o Neg	86.90 ± 0.45	88.48 ± 0.35	82.12 ± 0.46	53.43 ± 1.16
cross entropy	84.50 ± 0.53	87.19 ± 0.36	80.13 ± 0.86	46.59 ± 0.89
CNS	87.13 ± 0.17	88.70 ± 0.20	81.90 ± 0.26	58.78 ± 0.84
edge adding	86.96 ± 0.14	88.55 ± 0.11	82.02 ± 0.64	58.93 ± 2.17
edge removing	87.11 ± 0.15	88.67 ± 0.14	82.17 ± 0.37	61.46 ± 1.46
featuren masking	86.81 ± 0.21	88.50 ± 0.14	81.99 ± 0.40	61.78 ± 1.29

edge addition have minimal effects or can introduce noise, potentially disrupting node feature information.



(a) $\lambda^{\{1\}}$ and $\lambda^{\{2\}}$ on TwiBot-20 (b) $\lambda^{\{1\}}$ and $\lambda^{\{2\}}$ on TwiBot-22

Fig. 4. Sensitive analysis of hyperparameter $\lambda^{\{1\}}$ and $\lambda^{\{2\}}$ on two datasets.

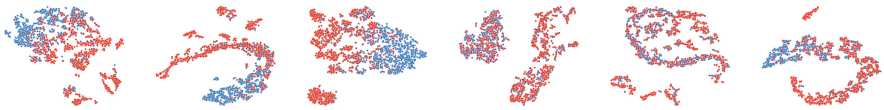
4.5 Sensitive Analysis

We assess the performance of BotSCL with respect to the hyperparameters $\lambda^{\{1\}}$ and $\lambda^{\{2\}}$, as our model incorporates two layers of information aggregation. We vary these parameters from 0.1 to 1.0 in increments of 0.1, keeping all other model parameters constant. The results on TwiBot-20 and TwiBot-22 are presented in Fig. 4a and Fig. 4b, respectively.

Figure 4 shows that increasing the hyperparameters $\lambda^{\{1\}}$ and $\lambda^{\{2\}}$ leads to a gradual improvement in accuracy for both TwiBot-20 and TwiBot-22. Specifically, when both hyperparameters are set to 1, the model achieves its highest classification performance. This highlights the importance of preserving self-information. Figure 4b reveals a noticeable accuracy drop for TwiBot-22 when

the hyperparameter is reduced to 0.8. In contrast, for TwiBot-20, the accuracy decline is more gradual, with a sharp drop occurring only when the hyperparameter approaches 0.4. This discrepancy can be attributed to TwiBot-22 having labels for all nodes and facing class imbalance issues. When the weights for original information are too low, the representation of the central node can be dominated by neighboring information.

Overall, despite variations in hyperparameters, the accuracy changes are within 0.04 for TwiBot-20 and within 0.01 for TwiBot-22. This indicates that while the hyperparameters $\lambda^{\{1\}}$ and $\lambda^{\{2\}}$ do influence the performance of BotSCL, the model is not highly sensitive to their values.



(a) GCN (b) FAGCN (c) BotRGCN (d) RGT (e) DGI (f) BotSCL

Fig. 5. User representations visualization. Red represents bots, while blue represents humans. (Color figure online)

4.6 Visualization

We visualize node representations obtained from six different models—GCN, FAGCN, BotRGCN, RGT, DGI, and our method BotSCL—on the TwiBot-22 dataset. Node representations from each model are first computed and then reduced to 2D using t-SNE for visualization. For clarity, we randomly select 1000 humans and 1000 bots from the test set for this visualization.

As shown in Fig. 5, node representations from homophilous GNNs are more scattered compared to those from FAGCN. These homophilous models, which operate on the assumption of homophily, tend to smoothen the representations of neighboring nodes without distinguishing between them, resulting in a distribution that is heavily influenced by local information. In contrast, the representations produced by BotSCL exhibit more pronounced clustering with fewer local clusters. While FAGCN accounts for both homophilous and heterophilous edges, its use of cross-entropy loss may overlook samples from less frequent distributions, leading to less distinct clustering. DGI, which lacks supervised signals during training, shows poor discriminative power and exhibits significant class overlap, where the representations of bots and humans are nearly indistinguishable.

5 Conclusion

In this paper, we reveal that social bots can evade graph-based detection methods by actively engaging with human users. To counter this, we propose BotSCL,

which incorporates both homophilous and heterophilous edges into the detection framework. The core of BotSCL is its encoder, which leverages supervised contrastive learning to integrate or differentiate neighbor representations effectively. Our extensive experiments on two real-world social bot datasets highlight the detrimental effect of heterophily on detection performance and demonstrate the superior effectiveness of the proposed BotSCL method.

Acknowledgements. This work was supported by the National Key Research and Development Program of China through the grants 2022YFB3105405 and 2021YFC3300502, NSFC through grants 62322202 and 61932002, Beijing Natural Science Foundation through grant 4222030, Guangdong Basic and Applied Basic Research Foundation through grant 2023B1515120020, Shijiazhuang Science and Technology Plan Project through grant 231130459A.

References

1. Abu-El-Haija, S., et al.: Mixhop: higher-order graph convolutional architectures via sparsified neighborhood mixing. In: ICML, pp. 21–29. PMLR (2019)
2. Ali Alhosseini, S., Bin Tareaf, R., Najafi, P., Meinel, C.: Detect me if you can: spam bot detection using inductive representation learning. In: WWW, pp. 148–153 (2019)
3. Beskow, D.M., Carley, K.M.: Its all in a name: detecting and labeling bots by their name. Comput. Math. Organ. Theory **25**(1), 24–35 (2019)
4. Bielak, P., Kajdanowicz, T., Chawla, N.V.: Graph barlow twins: a self-supervised representation learning framework for graphs. Knowl.-Based Syst. **256**, 109631 (2022)
5. Bo, D., Wang, X., Shi, C., Shen, H.: Beyond low-frequency information in graph convolutional networks. In: AAAI, pp. 3950–3957 (2021)
6. Chavoshi, N., Hamooni, H., Mueen, A.: Temporal patterns in bot activities. In: WWW, pp. 1601–1606 (2017)
7. Chen, J., Zhu, G., Qi, Y., Yuan, C., Huang, Y.: Towards self-supervised learning on graphs with heterophily. In: CIKM, pp. 201–211 (2022)
8. Chien, E., Peng, J., Li, P., Milenkovic, O.: Adaptive universal generalized pagerank graph neural network. arXiv preprint arXiv:2006.07988 (2020)
9. Cresci, S.: A decade of social bot detection. Commun. ACM **63**(10), 72–83 (2020)
10. Deb, A., Luceri, L., Badaway, A., Ferrara, E.: Perils and challenges of social media and election manipulation analysis: The 2018 us midterms. In: WWW, pp. 237–247 (2019)
11. Feng, S., Tan, Z., Li, R., Luo, M.: Heterogeneity-aware twitter bot detection with relational graph transformers. In: AAAI, vol. 36, pp. 3977–3985 (2022)
12. Feng, S., et al.: Twibot-22: towards graph-based twitter bot detection. arXiv preprint arXiv:2206.04564 (2022)
13. Feng, S., Wan, H., Wang, N., Li, J., Luo, M.: Satar: a self-supervised approach to twitter account representation learning and its application in bot detection. In: CIKM, pp. 3808–3817 (2021)
14. Feng, S., Wan, H., Wang, N., Li, J., Luo, M.: Twibot-20: a comprehensive twitter bot detection benchmark. In: CIKM, pp. 4485–4494 (2021)
15. Feng, S., Wan, H., Wang, N., Luo, M.: Botrgcn: twitter bot detection with relational graph convolutional networks. In: SNAM, pp. 236–239 (2021)

16. Hamdi, S.A.: Mining ideological discourse on twitter: the case of extremism in Arabic. Discourse Commun. **16**(1), 76–92 (2022)
17. Khosla, P., et al.: Supervised contrastive learning. Adv. Neural. Inf. Process. Syst. **33**, 18661–18673 (2020)
18. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
19. Kudugunta, S., Ferrara, E.: Deep neural networks for bot detection. Inf. Sci. **467**, 312–322 (2018)
20. Liu, Y., et al.: Roberta: a robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
21. Liu, Y., Zheng, Y., Zhang, D., Lee, V., Pan, S.: Beyond smoothing: unsupervised graph representation learning with edge heterophily discriminating. arXiv preprint arXiv:2211.14065 (2022)
22. Luan, S., et al.: Is heterophily a real nightmare for graph neural networks to do node classification? arXiv preprint arXiv:2109.05641 (2021)
23. des Mesnards, N.G., Hunter, D.S., el Hjouji, Z., Zaman, T.: Detecting bots and assessing their impact in social networks. Oper. Res. **70**(1), 1–22 (2022)
24. Moghaddam, S.H., Abbaspour, M.: Friendship preference: scalable and robust category of features for social bot detection. IEEE Trans. Depend. Secure Comput. **20**, 1516–1528 (2022)
25. Pei, H., Wei, B., Chang, K.C.C., Lei, Y., Yang, B.: Geom-gcn: geometric graph convolutional networks. arXiv preprint arXiv:2002.05287 (2020)
26. Schlichtkrull, M., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: Gangemi, A., et al. (eds.) ESWC 2018. LNCS, vol. 10843, pp. 593–607. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93417-4_38
27. Vaswani, A., et al.: Attention is all you need. Adv. Neural Inf. Process. Syst. **30** (2017)
28. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
29. Velickovic, P., Fedus, W., Hamilton, W.L., Liò, P., Bengio, Y., Hjelm, R.D.: Deep graph infomax. In: ICLR (Poster), vol. 2, no. 3, p. 4 (2019)
30. Wang, T., Jin, D., Wang, R., He, D., Huang, Y.: Powerful graph convolutional networks with adaptive propagation mechanism for homophily and heterophily. In: AAAI, pp. 4210–4218 (2022)
31. Williams, E.M., Novak, V., Blackwell, D., Platzman, P., McCulloh, I., Phillips, N.E.: Homophily and transitivity in bot disinformation networks. In: SNAMS, pp. 1–7. IEEE (2020)
32. Wu, Y., Fang, Y., Shang, S., Jin, J., Wei, L., Wang, H.: A novel framework for detecting social bots with deep neural networks and active learning. Knowl.-Based Syst. **211**, 106525 (2021)
33. Yang, C., Harkreader, R., Gu, G.: Empirical evaluation and new design for fighting evolving twitter spammers. IEEE Trans. Inf. Forensics Secur. **8**(8), 1280–1293 (2013)
34. Yang, K.C., Varol, O., Hui, P.M., Menczer, F.: Scalable and generalizable social bot detection through data selection. In: AAAI, vol. 34, pp. 1096–1103 (2020)
35. Yang, W., Mirzasoleiman, B.: Contrastive learning under heterophily. arXiv preprint arXiv:2303.06344 (2023)
36. Yang, Y., et al.: Rosgas: adaptive social bot detection with reinforced self-supervised gnn architecture search. ACM Trans. Web (2022)

37. You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., Shen, Y.: Graph contrastive learning with augmentations. Adv. Neural. Inf. Process. Syst. **33**, 5812–5823 (2020)
38. Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L., Koutra, D.: Beyond homophily in graph neural networks: current limitations and effective designs. Adv. Neural. Inf. Process. Syst. **33**, 7793–7804 (2020)
39. Zhu, Y., Xu, Y., Yu, F., Liu, Q., Wu, S., Wang, L.: Deep graph contrastive representation learning. arXiv preprint arXiv:2006.04131 (2020)

SimDrop: Towards Deep Graph Convolutional Networks

Gongjunjian Zhang and Lu Bai(✉)

Beijng Normal University, Beijng, China

Abstract. Recently, graph neural networks have achieved impressive results in areas like graph representation learning. However, as the number of layers in graph neural networks continues to increase, the performance of the network does not improve as expected. The primary obstacle to building deep graph neural networks is considered to be the over-smoothing problem. In this paper, we first theoretically analyze the relationship between over-smoothing problem and the algebraic connectivity of graphs, and establish the relationship between algebraic connectivity and edge dropping operations based on the degrees of nodes in the graph. We propose SimDrop, it consists of two parts: a sampler based on algebraic connectivity, and a sampler based on node feature similarity. Our method can perceive both the structural information and node feature information of the graph, aiming to alleviate the over-smoothing problem in deep graph neural networks. Experimental evaluations on various datasets demonstrate that our method achieves better performance in both full-supervised and semi-supervised tasks.

Keywords: graph convolutional networks · over smoothing · algebraic connectivity

1 Introduction

Graph neural networks [10,16,27,29,30,36] have been shown to be quite successful in learning representations of nodes and graphs on graph structured data [1,7,11], as well as in downstream tasks such as node classification [10,16,29], graph classification [35], and link prediction [18]. However, most graph neural networks are designed with only one or two layers. This is different from convolutional neural networks [20], where stacking more layers can learn more features of samples and improve the network's expressive power. Stacking layers on graph neural networks [16] fails to learn the rich topological information and node features in graph data [15,36] because these models suffer from severe over-smoothing [23]. As the number of layers in a graph neural network approaches infinity, the output will exponentially approach a certain subspace. Regardless of the input node features, deep graph neural networks can only learn the degree information of nodes and information about the connected components. They cannot leverage higher-order topological information [36] and the original node features [15] from graph data.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 69–84, 2025.
https://doi.org/10.1007/978-3-031-78183-4_5

On aware of this, several studies have focused on addressing the over-smoothing issue. The residual connection inspired by deep CNNs [17,29,31] from the computer vision domain is shown to be an effective strategy for addressing the over-smoothing issue for various graph convolution networks. Recently, data augmentation strategies proposed from works likes DropEdge, PTDNet [22], DSpar [19] and NeuralSparse [37] from the perspectives of graph structure and expressive power can be used as general techniques for improving robustness and GNN training. DSpar [19] sparsify the graph before training using node degree information to train more efficiently. NeuralSparse [37] remove task-irrelevant edges and simultaneously optimize graph sparsification and representation for more robust performence. However, these models achieve their best results in training and robustness tasks, but they cannot solve over-smoothing problem and make networks become deeper. Among the various methods proposed so far, Dropedge [24] involves randomly dropping a portion of the graph edges with a probability p at each layer of network training. This method has been theoretically and experimentally proven to effectively alleviate over-smoothing and can improve the performance of various deep graph neural networks.

However, Dropedge cannot fully exploit the higher-order structural information and node feature information in graph within network design. And previous works based on subgraph sampling and edge-dropping methods can be associated with the simple node-level random walk on unweighted graphs, hence they are only able to learn pure structure information but no feature information.

In this paper, we advance this by introducing SimDrop, an improved version of DropEdge. SimDrop consists of two components: a sampler based on algebraic connectivity [5], and a sampler based on node feature similarity [15]. Unlike previous approaches that analyze the eigenvalues of the adjacency matrix to improve network expressive power, we introduce algebraic connectivity from the Laplacian matrix to measure over-smoothing through changes in algebraic connectivity and design a better dropping method to reduce the convergence speed of over-smoothing. According to the variation of algebraic connectivity and the close relationship with node degrees, we assigned different weights to the adjacency matrix based on node degrees. Secondly we introduce a node feature sampling method, using cosine similarity as a distance measure between node features to adjust the edge dropping probability. Our node feature sampling method not only leverages all the benefits of DropEdge in mitigating over-smoothing but also jointly learns feature and structure information from the input graph data.

In summary, this paper contributes the following:

1. We theoretically analyzed the relationship between over-smoothing and the algebraic connectivity of the graph Laplacian matrix. To our knowledge, this is the first study to incorporate algebraic connectivity into deep GCNs to address the over-smoothing problem.
2. Unlike the unbiased sampling approach in DropEdge, we propose SimDrop including an edge sampling method based on node degrees and node feature sampling method to learn the feature and structural information from

graph data, that directly alleviates over-smoothing and enhances the expressive power of graph neural networks.
3. Our method SimDrop can be incorporated into various existing graph neural network frameworks. Experimental results show that SimDrop exceeds DropEdge and the vanilla graph neural network without sampling in mitigating over-smoothing problem.

2 Related Work

Graph Neural Network. Inspired by the extensive application of convolutional neural networks [12,20] in computer vision, [30] begins exploring the use of convolutional neural networks for non-Euclidean data, like graphs. Bruna et al. [2] builds upon the theoretical foundation of graph signals by introducing the Laplacian matrix and Fourier transform. They incorporate convolution operations from signal processing into tasks involving graph data. Then [16] based on the theory of first-order Chebyshev polynomials and Laplacian regularization matrices, they propose the graph convolutional neural network with first-order local features. This laid the foundation for various graph convolutional neural network models [7,10,11,27,29]. SimP-GCN [15] discovers that the information aggregation process of graph neural networks can disrupt node similarity in the original feature space, limiting model performance. An adaptive aggregation method is proposed to combine graph structure with node features, preserving feature similarity. Although SimP-GCN also focuses on the representation capability of nodes on graphs, our method SimDrop applies node similarity as a measure to alleviate over-smoothing in deep GNNs.

Methods to Alleviate over-Smoothing. Recently, some work have demonstrated the effectiveness of training graph convolutional networks with an increasing number of layers. We categorize existing methods into two types. The first kind of solutions is motivated by the deep CNN strategy in the area of computer vision research, such as Skip connection [12], DenseNet [14], and dilated aggregation [32]. DeepGCN [17] analogizes optimization methods from CNN, employing three modules: GCN, ResGCN, and DenseGCN. These models use different ways to incorporate skip connections. JKNet [31] based on the random walk extension connects the hidden embeddings of each layer output to the final layer. DRGCN [34] uses a dynamic block for each node to adaptively extract information from its initial representation and an evolving block to capture developing residual patterns.

The second kind of solution originates from the perspective of the graph structure, combining deep aggregation strategies with shallow graph neural networks. GDC [6] extends personalized PageRank to graph diffusion processes. DropEdge [24] implicitly increases graph diversity and reduces message passing by using a stochastic edge dropout strategy. DropMessage [4] introduces a message matrix to delete propagated messages instead of the adjacency matrix, allowing a node to send different messages to different neighbors. pathGCN [3] develop expressive

spatial operators for GCNs by learning from random paths within the graph. A-DGN [8] stemming from an ODE formulation preserves long-term dependencies between nodes and prevents gradient explosion or vanishing.

Unfortunately, these methods do not focus on the real causes of over-smoothing and can only mitigate a portion of the model degradation from a graph data input perspective and cannot theoretically justify the effectiveness of the methods. By exploiting our method SimDrop, we are able to alleviate over-smoothing and boost the expressive power of GNNs on node classification tasks.

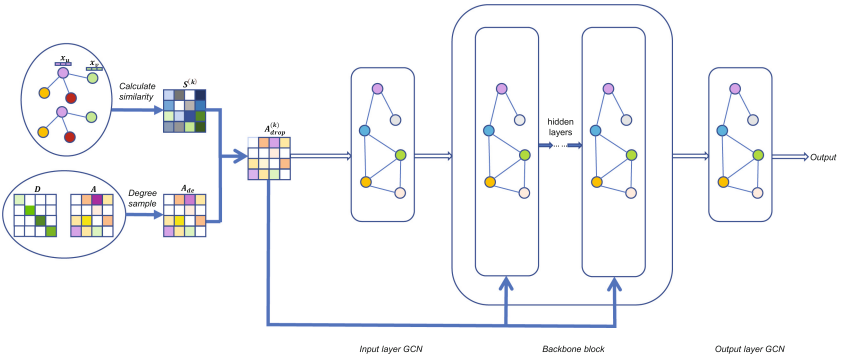


Fig. 1. Illustrations of SimDrop,backbone can be ResGCN,DenseGCN,IncepGCN.

3 Preliminaries

Notations. Considering a connected undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with n nodes and m edges. $\mathcal{V} = \{\nu_1, \nu_2, \ldots, \nu_n\}$ denotes the group of nodes, where n denotes the number of nodes. $\mathcal{E}$ is the set of connected edges, where the connection between ν_i and ν_j is e_{ij}. We use $\mathbf{A}$ to denote the adjacency matrix of the graph $\mathcal{G}$, let $\mathbf{D}$ denote the degree of node in the degree matrix of the graph $\mathcal{G}$. We use $\mathbf{X}$ to denote the features of the graph $\mathcal{G}$, with x_i being the feature of node i. The symbol "$\sim$" denotes random sampling.

GCN. Let $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, let $\tilde{\mathbf{D}}$ denote the degree matrix corresponding to $\tilde{\mathbf{A}}$. The feature propagation formula for adding hidden layers and weight matrices is as follows:

$$\mathbf{H}^{(l+1)} = \sigma(\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{H}^{l}\mathbf{W}^{l}) \tag{1}$$

where $\mathbf{H}^{(l+1)} = \{h_1, h_2, \ldots, h_n\}$ are the node features of the l-th layer, with $\mathbf{H}^{(0)} = \mathbf{X}$ as the initial feature matrix. $\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{H}^{l}$ belongs to the feature propagation process. $\mathbf{W}^{l}$ are trainable parameters optimized by the loss

function, and σ denotes the activation function which constitute the feature transformation process. In vanilla GCN, nodes collect feature vectors from their neighbors and combine them with their own features to update their representations, resulting in interacted nodes having very similar representations.

Over-Smoothing. The following is a definition of over-smoothing from [23,24] : let $w_1 \leq w_2 \leq \cdots \leq w_N$ be the eigenvalues of $\tilde{\mathbf{A}}$. Assume the set of eigenvalues in $\tilde{\mathbf{A}}$ are sorted in ascending order, let the count of the largest eigenvalue w_N being M. i.e. $(w_{(N-M)} < w_{(N-M+1)} = \cdots = w_N)$. The eigenvalue which is the second largest in the adjacency matrix $\tilde{\mathbf{A}}$ is defined as $w := \max_{n=1}^{N-M} |w_n|$. Let $\mathbf{O} \in \mathbb{R}^{N \times M}$ be the feature space associated with w_{N-M+1}, w_N, and let s_l denote the maximum value in $\mathbf{W}^l$, then we have $w_{N-M} \leq w_{N-M+1} = \cdots = w_N = 1$ and

$$d_{\mathcal{M}}(\mathbf{H}^l) \leq s_l w d_{\mathcal{M}}(\mathbf{H}^{(l-1)}) \tag{2}$$

where $\mathcal{M} := \{\mathbf{OC} \,|\, \mathbf{C} \in \mathbb{R}^{M \times C}\}$, and d is Frobenius-norminduced distance. This inequality indicates that as the number of network layers l tends to infinity, the output of the graph neural network on graph $\mathcal{G}$ will approach a low-dimensional subspace $\mathcal{M}$ at an exponential rate. This subspace is independent of the node features $\mathbf{X}$ of the network input, leading to the problem of over-smoothing. This equation indicates that the eigenvalues of the modified adjacency matrix $\tilde{\mathbf{A}}$ have a crucial impact on the over-smoothing problem. It is easy to see that increasing the eigenvalue of the adjacency matrix that is second in magnitude can alleviate over-smoothing, which is the aim of the DropEdge method.

Dropedge. [24] have analyzed from both theoretical and experimental perspectives that randomly dropping a fixed number of edges from graph can increase the value of w, reducing the speed of node feature information propagation and alleviating over-smoothing problem. Additionally, the randomly dropping can increase the diversity of graph, preventing the occurrence of overfitting. At each training epoch, DropEdge uniformly drops a certain number of edges from the graph. From the perspective of the adjacency matrix, this means converting a fixed portion of positive values in the adjacency matrix to 0. It can be expressed in formula as follows:

$$\mathbf{A}_{\text{drop}} = \mathbf{A} - \hat{\mathbf{A}} \tag{3}$$

Algebraic Connectivity. The algebraic connectivity [5] λ_2 of graph $\mathcal{G}$ is the second smallest eigenvalue in the Laplacian matrix of $\mathcal{G}$. The magnitude of algebraic connectivity reflects the overall connectivity of the entire graph. In addition to being related to the number of nodes in the graph, the value of algebraic connectivity also depends on the way nodes are connected to each other. For random graphs, the algebraic connectivity decreases as the number of nodes increases and increases with the increase of the average degrees. The importance of algebraic connectivity [21] lies in: 1. Measuring the robustness of a graph. 2. Being closely related to random walks on the graph.

4 Method

Relationship Between Algebraic Connectivity and over-Smoothing. [23,24] have shown that many methods for mitigating over-smoothing can slow down the convergence speed of node feature similarity to 0. However, employing these methods fails to improve the expressive power of deep graph neural network. In many tasks, there is not much difference in performance between these deep models and shallow models, which means they fail to fully exploit the advantages of deep graph neural networks.

From previous analysis [21,31], The over-smoothing in graph neural networks is equivalent to node-level lazy random walk on the graph. So we can analyze over-smoothing from the perspective of random walk. And the convergence rate of the lazy random walk to the stationary distribution is determined by the second smallest eigenvalue of the adjacency matrix: a smaller one implies faster convergence. The relationship between the eigenvalues of the adjacency matrix and the Laplacian matrix is as follows: $\lambda_i = 2 - 2w_i$. We analyze the relationship between oversmoothing and algebraic connectivity using the eigenvalues λ_2 and the corresponding eigenvectors of the Laplacian matrix. We need to adopt some concepts from [9,21]. If a random walk mixes in one step, then for any node g and h, we have $|p_t(h) - \pi(h)| \leq \sqrt{\frac{d(h)}{d(g)}} w_2^t$, where $p_t(h)$ is the probability that the random walk of node h at time t. $\pi(h)$ is the long-term probability of being at node h in the stationary distribution. The degree vector d is the Perron vector in the graph adjacency matrix. w_2^t denotes the second smallest eigenvalue of the graph's Laplacian matrix used for the random walk in the t step. Using $1 - x \approx exp(-x)$, we can prove the following.

$$
\begin{aligned}
&|p_t(h) - \pi(h)| \leq \sqrt{\frac{d(h)}{d(g)}} \left(1 - \frac{1}{2}\lambda_2\right)^t && \Leftrightarrow \\
&\sqrt{\frac{d(h)}{d(g)}} \left(1 - \frac{1}{2}\lambda_2\right)^t \leq \frac{d(h)}{2d(V)} && \Leftrightarrow \\
&\left(1 - \frac{1}{2}\lambda_2\right)^t \leq \frac{\sqrt{d(h)\,d(g)}}{2d(V)} && \Leftrightarrow \\
&\exp\left(-\frac{t\lambda_2}{2}\right) \leq \frac{\sqrt{d(h)\,d(g)}}{2d(V)} && \Leftrightarrow \\
&-\frac{t\lambda_2}{2} \leq \ln\left(\frac{\sqrt{d(h)\,d(g)}}{2d(V)}\right) && \Leftrightarrow \\
&t \geq 2\ln\left(\frac{2d(V)}{\sqrt{d(h)\,d(g)}}\right)/\lambda_2
\end{aligned}
$$

where V is the number of nodes in graph. For all graphs with approximately constant degrees, the upper bound on the mixing time is $ln(n)/\lambda_2$, which means that the smaller the algebraic connectivity, the slower the convergence speed of

over-smoothing. To slow down the speed, we first analyze the impact of edge dropping [9] on algebraic connectivity from a theoretical perspective. Dropping an edge in a given undirected graph does not increase any of its eigenvalues. The eigenvalues before and after the topological change satisfy the following interlacing Lemma 1 [9].

Lemma 1. *Assuming $\mathcal{G}$ is an undirected graph with n vertices, and v_i,v_j are two vertices of graph $\mathcal{G}$. $e_{ij} \subset \mathcal{E}(\mathcal{G})$. Let $\mathcal{G}_{drop} = \mathcal{G} - e_{ij}$. We have $\lambda_1(\mathcal{G}_{drop}) \leq \lambda_1(\mathcal{G}) \leq \lambda_1(\mathcal{G}_{drop}) \leq \cdots \leq \lambda_n(\mathcal{G})$.*

The trace of the Laplacian matrix $tr(L) = \sum_{i=1}^{N} \lambda_i(\mathcal{G}) = 2E$, where E is the number of edges. Lemma 1 implies that there is at least one strict inequality: $0 \leq \lambda_2(\mathcal{G}) - \lambda_2(\mathcal{G}_{\text{drop}}) \leq 2$. The inequality indicates that reducing the edges will decrease the algebraic connectivity.

To demonstrate the impact of different edge dropping strategies on algebraic connectivity, we first consider adding an edge e_{ij} to the graph, where $e_{ij} \notin \mathcal{E}$. Let q_i be a one-hot vector with the i-th position as 1 and all other positions as 0. We obtain the Laplacian matrix L_{add} after adding an edge. We have $L_{\text{add}} = L + \Delta L$ and $\Delta L = \Delta D - \Delta A$, where ΔD is the augmented degree matrix and ΔA is the adjacency matrix. We have $\Delta D = q_i q_i^T + q_j q_j^T$ and $\Delta A = q_i q_j^T + q_j q_i^T$. Then we obtain $L_{\text{add}} = L + (q_i - q_j)(q_i - q_j)^T$ which indicates that after adding an edge, L_{add} is composed of the original Laplacian matrix L and a rank one matrix. Similarly, we can obtain that after dropping an edge, the Laplacian matrix of graph $\mathcal{G}$ becomes $L_{\text{remove}} = L - (q_i - q_j)(q_i - q_j)^T$. By combining the above equation [21] with the previous part $\lambda_2(\mathcal{G}) = min_{\|x\|_2=1} x^T L x$ and letting u represent the eigenvector corresponding to the algebraic connectivity of L, we can compute the upper bound of the algebraic connectivity after dropping an edge. $\lambda_2(\mathcal{G}_{\text{drop}}) \leq x^T L x = \lambda_2(\mathcal{G}) - (u_i - u_j)^2$. When we drop a set of edges $\mathcal{E}_m \subset \mathcal{E}$, the upper bound of the algebraic connectivity becomes $\lambda_2(\mathcal{G}_{\text{drop}}) \leq x^T L x = \lambda_2(\mathcal{G}) - \sum_{k=(i,j)\in\mathcal{E}_m} (u_i - u_j)^2$. It can be observed that dropping the top edges m based on the eigenvector corresponding to the algebraic connectivity result in the most significant decrease in algebraic connectivity and slowing the convergence speed of over-smoothing.

Degree-Based Sampling. Following Wang et al. [28], we propose a simple and computationally less complex strategy to reduce algebraic connectivity: removing edges connected to high degrees. This results in a larger change in $\lambda_2(\mathcal{G})$. Calculating eigenvector corresponding to algebraic connectivity and finding the optimal edge dropping strategy computationally is infeasible. Balancing between not disrupting important substructures in the graph during the edge dropping, which can lead to a decrease in the performance of graph neural networks, and slowing down the convergence rate of over-smoothing as much as possible is desirable. Meanwhile, nodes with low degrees or nodes connected to nodes with high degrees can only receive limited information or receive a lot of misleading information. Therefore, we use node degree as a metric to drop edges. We perform node-degree-based sampling on the adjacency matrix, where edges are

randomly dropped based on the size of node degrees. The higher the degree of a node, the greater the probability that the edges connected to it will be dropped. The formula for node degree-based sampling is as follows:

$$\mathbf{A}_{\mathbf{de}}^{(\mathbf{k})} \sim \text{DegreeSample}(\mathbf{A}, \mathbf{D}, \mathbf{p}) \tag{4}$$

Here, p denotes the dropping probability, which varies with the change in graph neural network layers.

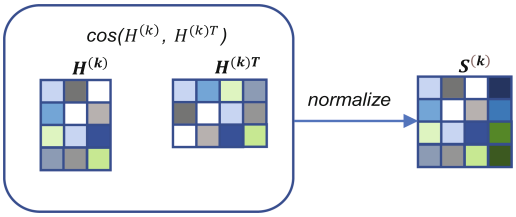


Fig. 2. Using cosine similarity to calculate node feature similarity.

Node Feature Similarity Sampling. We propose a method of edge drop based on node feature similarity, called node feature similarity sampling. It uses cosine similarity as the distance metric for node features between layers of the network to dynamically adjust the probability of dropping edges, thus learning both feature and structural information from the original graph data. We define the feature vectors of node i and node j as $h_i^{(k)}$ and $h_j^{(k)}$. The cosine similarity of node features is linearly correlated with message passing between nodes. The more dissimilar the feature similarity, the lower the probability of exchanging information between them. The similarity calculation schematic diagram is depicted in Fig. 2. We define the similarity of the nodes between node i and node j in the k-th layer of the graph neural network as $s_{ij}^{(k)}$

$$s_{ij}^{(k)} = \frac{\|\mathbf{h}_i^{(k)} \cdot \mathbf{h}_j^{(k)}\|}{\|\mathbf{h}_i^{(k)}\| \cdot \|\mathbf{h}_j^{(k)}\|} \tag{5}$$

In homogeneous graphs, it can measure the similarity between node features. Nodes with more similar features are more likely to belong to the same category. To facilitate subsequent network processing, we normalize the node feature similarity. After passing the normalized node feature similarity matrix and the adjacency matrix sampled based on node degrees obtained in the previous section are passed through an indicator function $I(s_{ij}^{(k)} > \gamma)$. The adjacency matrix obtained after sampling with our node feature similarity is

$$\mathbf{A}_{\text{drop}(i,j)}^{(k)} = I(s_{ij}^{(k)} > \gamma) = \begin{cases} 1, & \text{if } s_{ij}^{(k)} > \gamma \text{ and } \mathbf{A}_{\deg(i,j)} > 0 \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

Here, γ is a threshold value varying from datasets. This equation implies that edges between nodes with different characteristics are more likely to be removed.

The node feature sampling method can alleviate the over-smoothing problem by slowing the rate of information exchange between nodes of different types. Compared to graph neural networks using subgraph sampling [36], our node feature sampling can enhance the expressive power of graph neural networks while reducing computational complexity. It also does not require manual construction of the corresponding subgraphs, which offers better versatility. Compared to DropEdge, our method not only mitigates the over-smoothing problem but also further slows down the convergence speed of over-smoothing. Furthermore, it retains more information on the characteristics of the input layer nodes, thus enhancing the performance of graph neural networks.

In terms of heterogeneous graphs, when the adjacent nodes belong to different types, the features of the two nodes propagate misleading information, which makes the nodes cannot distinguish between them. In graph representation learning, the meaning of node representations in each dataset is different, and there are no unified tokens similar to those in computer vision and natural language processing. Our node feature sampling method can still enhance the performance of graph neural networks.

Overall Framework. After the analysis of the three parts above, the final version of SimDrop is the integration of the Degree-based sampling and Node feature similarity sampling. The detailed algorithm design is shown in Algorithm 1.

Algorithm 1 SimDrop

Input: The adjacency matrix $\mathbf{A}$;The degree matrix $\mathbf{D}$;The feature matrix $\mathbf{H}$;The number of layers $\mathbf{K}$;The drop rate p,the threshold γ;
Output: The set of Dropped adjacency matrices $\mathcal{A} = \{\mathbf{A}_{drop}^{(1)} \dots \mathbf{A}_{drop}^{(k)}\}$;

```
Initialize A = {};
for layer K = 1, 2, ··· , k do
    Compute S^(k) = cos(H^(k), H^(k)T);
    A_de^(k) ~ DegreeSample(A, D, p);
    A_drop^(k) = I(S^(k), A_de^(k), γ);
    A ⋃ A_drop^(k);
end for
return A
```

5 Experiment

5.1 Experiment Setting

In this section, we apply our SimDrop on node classification tasks using five datasets [13,25,33], including supervised and semi-supervised learning tasks. Additionally, we perform an ablation study to examine the different components of our method and analyze the loss throughout the training process.

Table 1. Datasets Statistics.

Dataset	Node	Edge	Feature	Class	Training/Validation/Testing
Cora(full)	2708	5429	1433	7	1624/542/542
Citeseer(full)	3327	4732	3703	6	1995/666/666
Pubmed(full)	19717	44338	500	3	11829/3944/3944
Cora(semi)	2708	5429	1433	7	140/500/1000
Citeseer(semi)	3327	4732	3703	6	120/500/1000
Pubmed(semi)	19717	44338	500	3	60/500/1000
ogbn-arxiv	169343	1166243	128	40	90941/29799/48603
Flickr	89250	899756	500	7	50%/25%/25%

Dataset. First, we use three publicly available datasets: three standard citation network datasets, Cora, CiteSeer, and PubMed. For fully supervised datasets, we divide the dataset into training, validation, and testing sets at a ratio of 60%, 20%, and 20%. Then, we conduct the experiments on the Node Property Prediction of Open Graph Benchmark and Flickr. In semi-supervised experiments, for Cora, Citeseer, Pubmed and ogbn-arxiv, we evaluate the performance with accuracy, and for Flickr, we evaluate with micro-F1 scores. The statistics of overall datasets are summarized in Table 1.

Configurations. In our experiments, the standard GCN model is combined with backbone layers to construct our architecture. The overall experimental design diagram is depicted in Fig. 1. Batch normalization and l_2 normalization are applied in the training process. Additionally, we use ReLU as the activation function, and the proposed architecture is optimized using the Adam algorithm. For every dataset and different layers, the number of learning rate, dropout rate and the number of epochs is selected based on the highest cross-validation accuracy.

Baseline Methods. We compare our method SimDrop with other alleviating over-smoothing methods, including GCN [16], Dropedge [24], Jknet [31], ResGCN [17], DenseGCN [17], Inception GCN [26]. We apply these methods to various GCNs as baseline models and compare their performance across different datasets in both fully-supervised and semi-supervised tasks.

Backbone Models. We apply the SimDrop method to three popular network architectures: ResGCN, Inception GCN, and DenseGCN. For ResGCN, residual connections at each layer are used to capture effective information from the early layers of the network. For DenseGCN, dense connectivity between layers is leveraged to improve information flow in the network and capture node feature information effectively. For InceptionGCN, an Inception structure is employed to decompose convolutional layers and regularize to enhance computational efficiency. We adopt the DropEdge method and our SimDrop method to three network architectures: ResGCN, Inception GCN, and DenseGCN. And we compare and analyze their performance with GCN and JKnet.

5.2 Experimental Results and Analysis

A Node Classification Analysis. Table 2 summarizes the results for the deep models with varying layer counts in fully-supervised tasks. For the node classification task, performance is evaluated by accuracy on three public datasets: Cora, CiteSeer, and PubMed. We can see that the impact of SimDrop methods varies across different datasets, backbone models, and layer configurations. For example, SimDrop methods on DenseGCN obtain an average accuracy improvement of 1.9% on Pubmed in 32 layers, while 0.1% on Citeseer in 32 layers. It is evident that SimDrop consistently outperforms NoDrop and DropEdge in enhancing performance across all backbones and varying layer configurations. When using DenseGCN or IncepGCN as the backbone, SimDrop generally maintains stability with increasing layers and nearly achieves the best performance.

Table 3 summarizes the results in semi-supervised tasks. We can see the classification performances of our SimDrop method outperform the baseline methods on semi-supervised tasks. Especially with a 1% improvement on the Cora dataset comparing with NoDrop method and DropEdge. A reasonable explanation is that our SimDrop method can learn more feature and structural information from graph data and slow down the convergence speed of over-smoothing.

B Training Process Analysis. We conduct experiments on various graph neural networks to analyze the loss throughout the training process. Figure 3 illustrates the variation of loss throughout the training process of using different graph neural network strategies in Cora, Citeseer, and Pubmed dataset. The experimental results indicate that our SimDrop strategy achieves the fastest convergence and the most stable performance among all methods.

Table 2. Testing accuracy(%) on full-supervised tasks.

Dataset	Model	2 layers	4 layers	8 layers	16 layers	32 layers	64 layers
Cora	GCN	85.6	85.1	78.6	82.4	72.5	51.6
	ResGCN	–	84.3	84.6	84.2	85.5	66.8
	DenseGCN	–	85.6	84.2	85.4	84.3	84.2
	Incep GCN	–	82.6	83.0	84.4	84.1	84.3
	ResGCN Drop	–	86.1	86.6	84.1	61.9	31.8
	DenseGCN Drop	–	87.2	86.6	86.2	**87.6**	86.8
	incep Drop	–	86.7	85.9	87.6	86.3	87.2
	JKNet	–	84.3	83.5	84.1	84.2	85.2
	Res SimDrop	–	86.8	**87.1**	85.0	62.2	31.9
	Dense SimDrop	–	**87.6**	86	86.2	87.1	87
	Incep SimDrop	–	87.1	86.6	**87.6**	86.9	**87.8**
Citeseer	GCN	75.8	76.2	74.8	65.3	58.4	43.2
	ResGCN	–	75.1	74.4	74.7	75.8	70.9
	DenseGCN	–	75.6	76.1	76.1	74.8	76.2
	Incep GCN	–	75.7	75.5	74.7	75.3	75.2
	ResGCN Drop	–	77	76.9	**78.7**	60.9	23.1
	DenseGCN Drop	–	**77.6**	77.4	78.2	77.4	78.2
	Incep Drop	–	76.5	77.7	77.5	77.6	78.6
	JKNet	–	72.3	73.4	72.2	72.4	72.3
	Res SimDrop	–	77.1	77.7	78.1	61.9	23.4
	Dense SimDrop	–	76.8	**78.4**	78.4	77.5	**78.9**
	Incep SimDrop	–	77.4	77.4	77.5	**77.8**	78.0
Pubmed	GCN	86.5	86.8	85.4	85.2	85.3	78.4
	ResGCN	–	85.9	86.8	87.1	85.5	84.6
	DenseGCN	–	86.6	86.5	86.5	86.6	OOM
	Incep GCN	–	86.2	86.7	86.9	86.6	OOM
	ResGCN Drop	–	87.4	87	87.1	82.6	50.6
	DenseGCN Drop	–	87.5	86.6	87.5	85.3	OOM
	Incep Drop	–	87.1	87.2	87.6	86.3	OOM
	JKNet	–	87.5	86.6	86.4	86.6	**86.5**
	Res SimDrop	–	**87.6**	87.1	87	83.2	69.1
	Dense SimDrop	–	87.5	87.5	87.7	**87.2**	OOM
	Incep SimDrop	–	86.9	**88.0**	**87.8**	86.7	OOM

C Ablation Study. In this section, we explore different components of our SimDrop. Figure 4 presents the results of an ablation study that assesses the contributions of our two techniques based on the ResGCN model: degree-based sampling and node feature similarity sampling. The ND results indicate that

Table 3. Testing accuracy(%) on semi-supervised tasks. micro-F1 scores for Flickr dataset.

Model	Dataset				
	Cora	Citeseer	Pubmed	ogbn-arxiv	Flickr
GCN	80.7	69.6	77.5	70.46	50.10
ResGCN	78.7	64.3	75.2	70.55	50.09
DenseGCN	78.9	66.4	76.2	70.93	50.07
IncepGCN	68.2	66.3	76.4	70.74	51.64
ResGCN Drop	79.4	69.5	76.2	70.66	51.78
DenseGCN Drop	81.4	66.9	76.4	70.83	51.66
Incep Drop	80.4	67.7	68.3	70.77	51.65
ResGCN SimDrop	81.1	**70.0**	77.2	70.89	51.90
DenseGCN SimDrop	**82.4**	69.1	76.7	71.01	**52.07**
Incep SimDrop	81.9	68.3	**78.0**	**71.23**	51.94

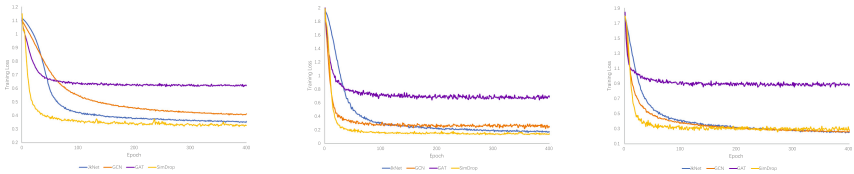


Fig. 3. Training loss on different datasets. Left: Cora; Middle: Citeseer; Right: Pubmed.

ResGCN helps mitigate the over-smoothing issue. However, compared to ND, the DD results reveal that degree-based sampling benefits shallow layers but falls short for deeper layers. Additionally, when combined with node feature similarity sampling, the model shows significant improvement, highlighting the importance of learning node feature information for deep GCNs.

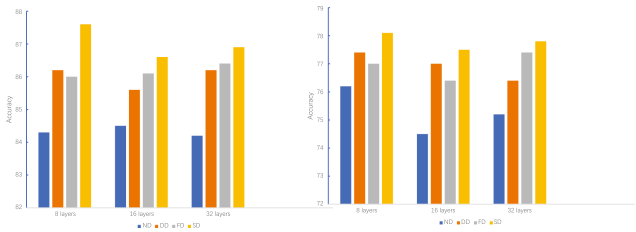


Fig. 4. Accuracy comparison of different components. Left: Cora; Right: Citseer (ND: NoDrop. DD: Degree Drop. FD: Feature Drop. SD: SimDrop).

6 Conclusion

In this paper, we theoretically analyse the impact of algebraic connectivity and edge dropping on alleviating the over-smoothing problem and propose two effective edge dropping methods inspired by theoretical analysis: edge dropping based on node degree and edge dropping based on node feature similarity. Edge dropping based on node degree can slow down the convergence rate of over-smoothing, while edge dropping based on node feature similarity can enhance the expressiveness of deep graph neural networks. Our overall framework, SimDrop, combines these two methods and applies them to node classification tasks in both full-supervised and semi-supervised learning settings. Experimental results demonstrate that our method achieves better performance in both shallow and deep models.

Acknowledgements. This work is supported by the National Natural Science Foundation of China under Grants T2122020.

References

1. Bhagat, S., Cormode, G., Muthukrishnan, S.: Node classification in social networks. Soc. Netw. Data Anal. 115–148 (2011)
2. Bruna, J., Zaremba, W., Szlam, A., LeCun, Y.: Spectral networks and locally connected networks on graphs. arXiv preprint arXiv:1312.6203 (2013)
3. Eliasof, M., Haber, E., Treister, E.: pathgcn: Learning general graph spatial operators from paths. In: International Conference on Machine Learning, pp. 5878–5891. PMLR (2022)
4. Fang, T., Xiao, Z., Wang, C., Xu, J., Yang, X., Yang, Y.: Dropmessage: unifying random dropping for graph neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, pp. 4267–4275 (2023)
5. Fiedler, M.: Algebraic connectivity of graphs. Czechoslov. Math. J. **23**(2), 298–305 (1973)
6. Gasteiger, J., Bojchevski, A., Günnemann, S.: Predict then propagate: graph neural networks meet personalized pagerank. arXiv preprint arXiv:1810.05997 (2018)
7. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: International Conference on Machine Learning, pp. 1263–1272. PMLR (2017)
8. Gravina, A., Bacciu, D., Gallicchio, C.: Anti-symmetric dgn: a stable architecture for deep graph networks. arXiv preprint arXiv:2210.09789 (2022)
9. Grone, R., Merris, R., Sunder, V.S.: The laplacian spectrum of a graph. SIAM J. Matrix Anal. Appl. **11**(2), 218–238 (1990)
10. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. Adv. Neural Inf. Process. Syst. **30** (2017)
11. Hao, Z., et al.: Asgn: an active semi-supervised graph neural network for molecular property prediction. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 731–752 (2020)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)

13. Hu, W., et al.: Open graph benchmark: datasets for machine learning on graphs. Adv. Neural. Inf. Process. Syst. **33**, 22118–22133 (2020)
14. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708 (2017)
15. Jin, W., Derr, T., Wang, Y., Ma, Y., Liu, Z., Tang, J.: Node similarity preserving graph convolutional networks. In: Proceedings of the 14th ACM International Conference on Web Search and Data Mining, pp. 148–156 (2021)
16. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
17. Li, G., Muller, M., Thabet, A., Ghanem, B.: Deepgcns: Can gcns go as deep as cnns? In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9267–9276 (2019)
18. Liben-Nowell, D., Kleinberg, J.: The link prediction problem for social networks. In: Proceedings of the Twelfth International Conference on Information and Knowledge Management, pp. 556–559 (2003)
19. Liu, Z., et al.: Dspar: an embarrassingly simple strategy for efficient gnn training and inference via degree-based sparsification. Trans. Mach. Learn. Res. (2023)
20. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440 (2015)
21. Lovász, L.: Random walks on graphs. Combinatorics, Paul erdos is eighty **2**(1–46), 4 (1993)
22. Luo, D., et al.: Learning to drop: robust graph neural network via topological denoising. In: Proceedings of the 14th ACM International Conference on Web Search and Data Mining, pp. 779–787 (2021)
23. Oono, K., Suzuki, T.: Graph neural networks exponentially lose expressive power for node classification. arXiv preprint arXiv:1905.10947 (2019)
24. Rong, Y., Huang, W., Xu, T., Huang, J.: Dropedge: towards deep graph convolutional networks on node classification. arXiv preprint arXiv:1907.10903 (2019)
25. Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., Eliassi-Rad, T.: Collective classification in network data. AI Mag. **29**(3), 93–93 (2008)
26. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)
27. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
28. Wang, H., Van Mieghem, P.: Algebraic connectivity optimization via link addition. In: 3d International ICST Conference on Bio-Inspired Models of Network, Information, and Computing Systems (2010)
29. Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., Weinberger, K.: Simplifying graph convolutional networks. In: International Conference on Machine Learning, pp. 6861–6871. PMLR (2019)
30. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. IEEE Trans. Neural Netw. Learn. Syst. **32**(1), 4–24 (2020)
31. Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.I., Jegelka, S.: Representation learning on graphs with jumping knowledge networks. In: International Conference on Machine Learning, pp. 5453–5462. PMLR (2018)
32. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. arXiv preprint arXiv:1511.07122 (2015)

33. Zeng, H., Zhou, H., Srivastava, A., Kannan, R., Prasanna, V.: Graphsaint: graph sampling based inductive learning method. arXiv preprint arXiv:1907.04931 (2019)
34. Zhang, L., Yan, X., He, J., Li, R., Chu, W.: Drgcn: dynamic evolving initial residual for deep graph convolutional networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, pp. 11254–11261 (2023)
35. Zhang, M., Cui, Z., Neumann, M., Chen, Y.: An end-to-end deep learning architecture for graph classification. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
36. Zhao, L., Jin, W., Akoglu, L., Shah, N.: From stars to subgraphs: uplifting any gnn with local structure awareness. arXiv preprint arXiv:2110.03753 (2021)
37. Zheng, C., et al.: Robust graph representation learning via neural sparsification. In: International Conference on Machine Learning, pp. 11458–11468. PMLR (2020)

A Quantum-inspired Approach to Estimate Optimum-Path Forest Prototypes based on the Traveling Salesman Problem

Maria Angélica Krüger Miranda[1], Felipe Fernandes Fanchini[2], Leandro Aparecido Passos[2], Douglas Rodrigues[2], Kelton Augusto Pontara da Costa[2(✉)], Rafał Sherer[3], and João Paulo Papa[2]

[1] Institute of Computing, Campinas State University - UNICAMP, Campinas, Brazil
[2] São Paulo State University (UNESP), School of Sciences, Bauru, Brazil
kelton.costa@unesp.br
[3] Institute of Computational Intelligence, Czestochowa University of Technology, Czestochowa, Poland

Abstract. Quantum mechanics emerge as a promise for the future of computing, broadening the horizons for solutions concerning complex tasks, e.g., NP-hard problems. Alongside quantum computing, machine learning has become indispensable. This paper explores the potential integration of quantum computing principles into the Optimum-Path Forest (OPF), a graph-based framework comprised of solutions for machine learning, optimization, and image processing. We are particularly interested in the supervised OPF approach, which elects the most representative samples for each class, aka prototypes, as the connected samples from different classes in a minimum spanning tree (MST) computed over the training set. By harnessing quantum parallelism and superposition, this paper introduces a new approach to identifying prototypes employing a quantum-based Traveler Salesman Problem (TSP) algorithm, which provides an alternative to computing MSTs and yields a hybrid version of the OPF classifier. The experiments on established datasets demonstrated the promising potential of this approach while also underscoring the necessity for further research in this field.

Keywords: Quantum Computing · Machine Learning · Quantum Optimization · Optimum-Path Forest.

1 Introduction

The functioning of the human brain has intrigued many scientists since ancient times. The study and replication of cognition for the computational environment begin with modeling the first artificial neuron by Warren McCulloch et al. [16]. From this point on, the field of artificial intelligence and machine learning (ML) gained strength and prominence until becoming fundamental in today's

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 85–98, 2025.
https://doi.org/10.1007/978-3-031-78183-4_6

activities. However, although artificial intelligence has broad applicability, it is constrained to highly complex problems on traditional computers and issues involving the manipulation of large volumes of data, which impose a challenging task even for the supercomputers available today.

Parallel to ML, quantum computing has recently received attention following the introduction of the first quantum computers, which are now a subject of interest for businesses and academics alike, with significant investments in this field. Quantum computers leverage fundamental principles of quantum mechanics, such as superposition and entanglement of states, which are crucial for modeling physical phenomena observed in quantum mechanics [4]. Theoretically, this paradigm could help solve problems that traditional computers cannot tackle in a feasible amount of time, e.g., large number decomposition into prime factors, searching for elements in unordered lists, and optimization problems. Such circumstances have encouraged huge and increasing investments from big companies in developing and advancing this technology.

Recent research indicates that quantum computing may address certain obstacles faced in machine learning [1,8,26], giving rise to a new study area known as Quantum Machine Learning (QML). By integrating concepts from physics, mathematics, and computing, QML broadens the scope of challenges that can be tackled by machine learning. It also has the potential to decrease computational load, enhance classification rates, and significantly improve the efficiency of these algorithms.

While quantum computing holds great promise in theory, its practical application in complex scenarios is hampered by the instability of its atomic units (qubits) and the high resource demand for information representation. Fortunately, researchers have developed hybrid machine learning models that blend quantum and classical algorithms to overcome these limitations. By harnessing the power of quantum concepts, these models can significantly improve the performance of machine learning. Ongoing efforts to adapt conventional ML techniques to the quantum realm and develop hybrid models hold great potential for expanding the field in novel and exciting directions.

This work proposes a hybrid version of the Optimum-Path Forest (OPF), a framework for designing pattern classifiers based on graphs. We are particularly interested in the supervised OPF with a complete graph [19,20], which has been applied to various applications. Some of them include the description of textures in images [18], diagnosis of laryngeal diseases [21], and classification of fingerprints [17], among others [22,23,28].

In short, it implements a novel OPF version that replaces the standard procedure of computing the prototypes, i.e., the nodes representing each class from the training set, by a quantum implementation of the Traveling Salesman Problem (TSP) algorithm. Experiments conducted over a limited[1] amount of samples from the Boat dataset shows that the hybrid approach can obtain similar results

[1] A restricted number of samples is imposed due to the limitation of actual quantum computers.

under a quantum environment. Therefore, the main contributions of this paper are described as follows:

OPF-based algorithms perform pattern classification through a reward-competition process, in which key nodes called prototypes compete among themselves to conquer the remaining samples. What differentiates one OPF from another concerns the amount of labeled data and the problem itself. Regarding supervised learning, the OPF with a complete graph is the most used approach due to its straightforward formulation. The prototypes found by a minimum spanning tree (MST) over the training set are the nearest samples from different classes, further used to start the competition process.

The fundamental idea involves a quantum-inspired TSP algorithm to encounter the OPF prototypes during the training phase, i.e., the most representative samples belonging to distinct classes. Since TSP is NP-hard, quantum computing can solve it by taking advantage of parallelism and processing multiple possibilities simultaneously. Experiments conducted on a limited number of samples[2] from well-known datasets showed that the proposed hybrid approach could achieve similar or superior results in a quantum environment.

The main contributions of this article are described below:

- To introduce a hybrid version of the Optimum-Path Forest classifier that figures prototype computation using quantum computing;
- To implement a novel approach to compute OPF prototypes using Traveling Salesman Problem algorithm; and
- To introduce a hybrid OPF classifier that combines traditional and quantum computing elements.

The remainder of this paper is organized as follows. Section 2 provides the theoretical background concerning the supervised OPF and quantum machine learning, while Section 3 describes the proposed approach. Following, Section 4 describes the datasets and experimental setup. Finally, Sections 5 and 6 state the experimental results and conclusions, respectively.

2 Theoretical Background

2.1 Optimum-Path Forest

Optimum Path Forest is a graph-based framework for machine learning that models dataset samples as the nodes of a graph whose arcs describe the relationship between each pair of instances. The supervised version proposed by Papa et al. [19,20] arranges the training instances as a fully connected graph whose edges are weighted by the distance between pairs of vertices. Further, the model computes an MST by adapting Prim's algorithm, and the nodes connecting samples from distinct classes are marked as prototypes inserted into a set $\mathcal{P}$. Finally, the prototypes compete among themselves in a conquering-like fashion

[2] A restricted number of samples are imposed due to the limitation of real quantum computers.

to offer each remaining vertices an optimum cost computed through the f_{max} function:

$$f_{max}(\langle \boldsymbol{s} \rangle) = \begin{cases} 0 & \text{if } \boldsymbol{s} \in \mathcal{P}, \\ +\infty & \text{otherwise} \end{cases}$$
$$f_{max}(\phi_{\boldsymbol{s}} \cdot \langle \boldsymbol{s}, \boldsymbol{z} \rangle) = \max\{f_{max}(\phi_{\boldsymbol{s}}), d(\boldsymbol{s}, \boldsymbol{z})\}, \tag{1}$$

where $\boldsymbol{s}$ and $\boldsymbol{z}$ are arbitrary nodes, $\phi_{\boldsymbol{s}}$ stands for a path rooted in $\mathcal{P}$ and the terminus at sample $\boldsymbol{s}$, and $\phi_{\boldsymbol{s}} \cdot \langle \boldsymbol{s}, \boldsymbol{z} \rangle$ stands for the concatenation between the path $\phi_{\boldsymbol{s}}$ and the edge $\langle \boldsymbol{s}, \boldsymbol{z} \rangle$. Moreover, $d(\boldsymbol{s}, \boldsymbol{z})$ represents the distance between samples $\boldsymbol{s}$ and $\boldsymbol{z}$. Finally, $f_{max}(\phi_{\boldsymbol{s}})$ denotes the maximum distance among adjacent samples in the path $\phi_{\boldsymbol{s}}$.

The training step is then performed by assigning an optimum cost $C(\boldsymbol{z})$ to each sample $\boldsymbol{z}$ in the training set $\mathcal{D}$:

$$C(\boldsymbol{z}) = \min_{\forall \boldsymbol{s} \in \mathcal{D}} \{\max\{C(\boldsymbol{s}), d(\boldsymbol{s}, \boldsymbol{z})\}\}, \tag{2}$$

where $\boldsymbol{s}$ represents the instance that conquered $\boldsymbol{z}$.

The classification step computes the distances between each test instance and every training node for further computing the minimum cost, similarly to Equation 2. Finally, the test sample is labeled with the same label as the prototype associated with the path that offered the optimum cost. This procedure is repeated until the entire test set is covered.

2.2 Quantum Machine Learning

Quantum Machine Learning regards distinct research areas dependent on the combination of resources it consumes, i.e., classical and quantum data and computing strategy [25]. The most explored area uses classical data and quantum computing, namely the classical-quantum approach, which is usually employed to encompass optimization problems. Such a procedure usually employs quantum annealing, i.e., a method that considers intrinsic quantum properties to solve optimization and probabilistic sampling problems.

Quantum annealing [5] is a process that models a problem as an energy landscape and prepares a ground state composed of n qubits. Further, it employs the qubits' superposition and entanglement properties to simulate different energy states through the natural evolution of quantum physics and find the configuration whose system's energy is the lowest. The technique represents the input data using Hamiltonians, a mathematical tool capable of describing the total energy of a physical system.

Hamiltonians can be computed through the Quadratic Unconstrained Binary Optimization (QUBO) [13], a problem whose solution is given by a binary vector $\boldsymbol{x} \in \Re^n$ that optimizes the following function:

$$\min \boldsymbol{x}^t \boldsymbol{Q} \boldsymbol{x}, \tag{3}$$

where $\boldsymbol{Q}$ is a square symmetric matrix of order n of coefficients.

Most machine learning techniques are based on optimization problems, i.e., such algorithms are designed to minimize an objective (cost) function. In graph-based ML models, it is feasible to express the nodes' connectivity using qubits $|0\rangle$ and $|1\rangle$ through a binary sequence optimized by QUBO to represent the presence or absence of each edge. QUBO is also helpful in solving traditional graph problems like maximum cut, binary linear programming, graph coloring, backpack problems, and traveling salesman problems.

Usually, QUBO problems can be solved using the Quantum Approximate Optimization Algorithm (QAOA) [3], a hybrid variational algorithm developed to cope with combinatorial optimization algorithms. However, a more recent method called Feedback-Based Quantum Optimization (FALQON) [15] shows itself more favorable for the task due to its advantages regarding the guarantee of convergence by avoiding possible crashes in local minima. The algorithm implemented using variational quantum circuits is based on optimal path theory and the Trotter-Suzuki Theorem [12].

3 Quantum-inspired Prototype Computation

This paper discusses the challenge of prototype selection in quantum computing. The conventional method of selecting prototypes in the OPF classifier involves the MST. However, adapting it to the quantum context can be challenging since it requires graph optimization. We are unaware of any paper that presents an efficient approach to computing an MST in the quantum domain. The paper proposes a novel approach to solving TSP using quantum algorithms, leveraging that a TSP visiting all nodes once creates a Hamiltonian cycle that includes all edges of an MST. Hence, one can simulate the TSP graph into an MST by eliminating one edge from each cycle.

The TSP models a graph whose nodes represent cities and whose edges denote the distance between each pair of nodes. Given an origin and a destination, the method's objective is to find the lowest-cost route that visits all cities exactly once and returns to the origin. Regarding computational complexity, it is classified as an NP-hard problem, which means no known algorithm can provide the global best solution in polynomial time.

Let $G = (V, E)$ be a graph where V and E denote the set of vertices and edges, respectively, and each edge $(v_i, v_j) \in E$ has an associated weight w_{ij}, i.e., a distance between its corresponding vertices $v_i, v_j \in V$. The TSP can be described using a binary sequence that indicates whether the edge (v_i, v_j) is present ($e_{ij} = 1$) or not ($e_{ij} = 0$) in the problem solution. One can mathematically formulate this problem through the minization of the following cost function:

$$C(x) = \sum_i \sum_{j>i} w_{ij} e_{ij} + \left(\sum_i \sum_{j>i} e_{ij} - n \right)^2 + \left[\sum_i \left(\sum_{j \neq i} e_{ij} - 2 \right)^2 \right], \tag{4}$$

where the first term concerns the lowest-cost path, the second ensures that the number of edges must equal the number of nodes n, and the third guarantees that each vertex must contain two edges.

The TSP can be adapted to the quantum computing domain by adjusting the cost function presented in Equation 4 in Hamiltonian terms and representing the variable e_{ij} as $\tilde{Z}_{ij}$, described as follows:

$$H_c = \sum_i \sum_{j>i} w_{ij} Z_{ij} + P_1 \left(\sum_i \sum_{j>i} Z_{ij} - n \right)^2 + P_2 \left[\sum_i \left(\sum_{j \neq i} Z_{ij} - 2 \right)^2 \right], \tag{5}$$

where P_1 and P_2 are ad-hoc parameters (weights for each restriction), and Z is an operator that plays the role of the quantum computing[3]. The Hamiltonian H_c describes all graph configurations and those that do not meet the above restrictions will be penalized so that they are not considered possible solutions when searching for the lowest energy state [10].

4 Methodology

This section describes the datasets and experimental setup employed to evaluate the proposed approach.

4.1 Datasets

1. **Boat** [14]: a synthetic dataset that comprises 100 samples distributed into three classes, each represented by two variables. The choice is motivated by the ease of manipulation and visualization.
2. **Breast Tissue** [24]: a small real-world dataset containing 106 breast tissue instances represented by nine features distributed into six irregular pattern classes, i.e., carcinoma, fibro-adenoma, mastopathy, glandular, connective, and adipose.
3. **Heart** [7]: a real dataset containing 303 samples represented by 13 features and distributed across two classes.
4. **Diabetes** [11]: a real dataset containing 768 samples represented by 8 features and distributed over two classes.
5. **Ionosphere** [27]: a real dataset containing 351 samples represented by 35 features and distributed over two classes

[3] Due to the lack of space, we did not detail the formalism concerning quantum operators.

6. **LungCancer** [6]: a real dataset with 59 samples represented by 6 features across two classes.

Processing large matrices poses hardware limitations, limiting the usage of the complete datasets. We randomly selected eight samples from each dataset only, respecting the label proportions. Four were used to compose the training set, and the remaining were part of the test set.

4.2 Experimental Setup

This work employs the OPFython [2], a Python-based library that implements the OPF framework. The supervised OPF addressed in this manuscript models the training set as a complete graph whose adjacency relationship is established by the Euclidean distance between the vertices. Afterward, the graph nodes are normalized to assemble the Hamiltonian described in Equation 5, which is further minimized using FALQON [29]. The prototypes (connected samples in the TSP graph solution with different labels) are then selected, and the standard (classifical) OPF training procedure is performed. Figure 1 depicts the proposed approach.

After the dataset is partitioned into four samples for training and four for testing purposes, the Quantum TSP replaces the standard procedure to generate the prototypes. The method employs the QuTip [9] library to assemble the matrix operations and the Hamiltonian, whose weights $P_1 = 9$ and $P_2 = 7$ (Equation 5) were selected through a grid search in the range $\{2, 3, \ldots, 30\}$. Further, the FALQON algorithm is performed during $10,000$ iterations to optimize the Hamiltonian, i.e., to minimize the system's energy state.

Finally, the prototypes are derived from the closed Hamiltonian cycle obtained in the previous step. This selection process is akin to the standard OPF, whereby nodes connected to samples from distinct classes are designated prototypes. The algorithm then proceeds to the subsequent training phase, which assembles the forest of optimum paths.

The procedure was constructed using Python 3.11 with the support of the OPFython libraries in version 1.0.12, Qiskit 0.43.0, and Qutip 4.7.1. Furthermore, the Pipenv virtual environment manager was used to control the project's dependencies. The experiments were conducted using a Windows 10 system running on an AMD® Ryzen 5 3500U processor with 16 GB of memory and a Radeon® 540X Series GPU.

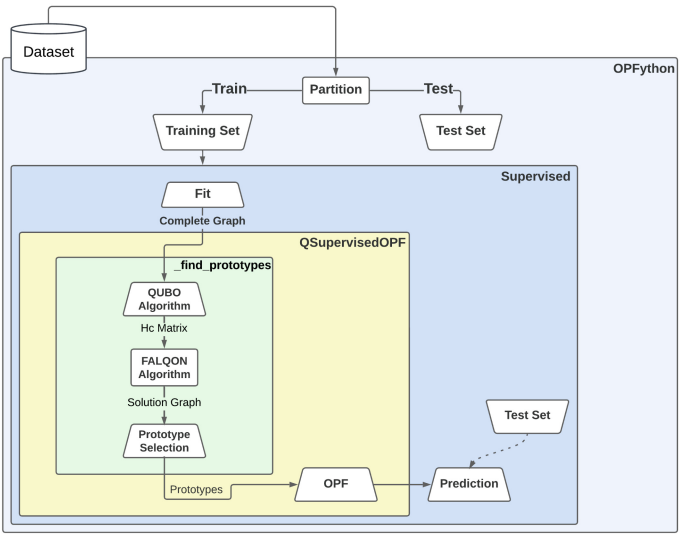


Fig. 1. The pipeline describes the OPF training using the TSP to find prototypes where the light green module runs in a quantum computer.

5 Experiments and Results

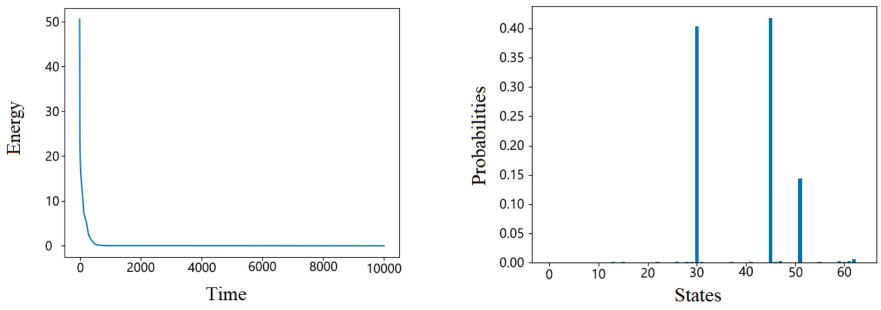
The experiments involved two scenarios: (i) the first evaluates FALQON's convergence to the minimum energy state and its compatibility with the TSP graph solution, and (ii) the second scenario assesses the model's effectiveness.

5.1 Convergence Analysis

The main focus of these experiments is to verify whether the FALQON algorithm converges to the lowest energy state and whether this state adequately represents the solution graph of the TSP problem. Due to the extensive computational complexity demanded by large matrix computations in a quantum context, the experiments were repeated 100 times per dataset using a considerably reduced number of randomly selected balanced instances per execution, i.e., four samples for training and four for testing purposes.

In the first experiment, we considered the Boat dataset under two scenarios: eight and ten samples. Figure 2a depicts the convergence curve, i.e., the system's energy is minimized throughout the iterations. Furthermore, Figure 2b indicates the probability associated with each graph model. Since we have a graph with four nodes, the total number of edges is six for a complete graph. Considering QUBO formulation assigns a binary value to each edge, i.e., it will compose the solution or not, we have 2^6 possible solutions (states). The one with the highest probability is the solution that minimizes the system's energy. It is worth noting

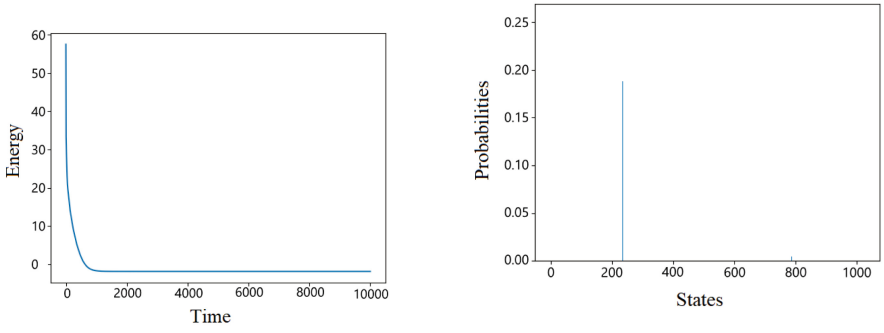
that both the classical and the quantum models achieved accuracy of 100%. While classical OPF took an average training time of 0.5 seconds, its counterpart quantum version required 27 seconds.



(a) (b)

Fig. 2. (a) Convergence analysis and (b) the probability associated to each possible solution (state) for the first scenario.

An additional experiment considered the Boat dataset with ten samples, i.e., five for training and five for testing purposes. Figures 3a and 3e illustrate the convergence curve and the probability associated with each state (now we have 2^10 possible solutions). Both models achieved an accuracy of 50%. Training averaged 0.96 seconds for the classical OPF and 5 hours for the quantum version.



(a) (b)

Fig. 3. (a) Convergence analysis and (b) the probability associated to each possible solution (state) for the second scenario.

The second experiment considers all datasets mentioned earlier to evaluate the efficiency of quantum TSP algorithms. Table 1 presents the averaged accuracy obtained for each dataset, and Table 2 shows the training time for each

OPF version. It is essential to highlight that the number of executions for the quantum algorithms was 5,000, and each model was trained 25 times on each dataset to calculate the averaged accuracy.

Table 1. Average accuracies for each dataset and algorithm

	Traditional OPF	Quantum OPF (TSP))
Boat	**0.85**	0.79
Breast Tissue	0.75	**0.76**
Heart	**0.58**	0.53
Diabetes	**0.63**	0.56
Ionosphere	**0.62**	0.54
LungCancer	**0.83**	0.7
Average	**0.6866**	0.6466

Table 2. Average training time for each dataset

	Traditional OPF	Quantum OPF (TSP)
Boat	**0.01592**	19.55148
Breast Tissue	**0.01826**	25.65221
Heart	**0.01989**	15.63850
Diabetes	**0.02731**	24.67932
Ionosphere	**0.02542**	24.30720
LungCancer	**0.02202**	25.43112
Average	**0.02147**	22.543305

Although the classical version of OPF achieved the best accuracy and training times, the quantum model also presented similar accuracy. However, in terms of execution time, quantum algorithms exhibited longer average times due to the need to simulate the algorithm on a conventional machine. It is important to note that implementing the TSP algorithm requires controlling the collapse of the quantum state after the simulation, which can be challenging and time-consuming. We also highlight the main idea of this manuscript is not (yet) to present a quantum-based approach that can be more efficient than classical OPF but to show its viability instead. We expect these results to be more efficient when more robust quantum computers are available.

5.2 Discussion

One can observe that classical and TSP-based OPF versions achieved similar results in the first set of experiments, and quantum TSP was able to find the

minimum energy solution. Although the classical and quantum models did not reach 100% accuracy in the second scenario, i.e., the Boat dataset with five samples for training, they exhibited similar behavior. Concerning the entire dataset experiment, both versions achieved close effectiveness, with classical OPF being much faster. Again, we have no access to native quantum computers, i.e., all experiments were simulated in a classical computer, affecting TSP-based OPF efficiency.

Extensive research efforts are dedicated to optimizing algorithms and increasing their efficiency. Quantum searches, such as Grover's algorithm, have been a particular area of focus, as they can significantly accelerate search processes in some instances. This method could enhance the performance of MST algorithms. However, perfecting its practical application remains challenging, as current outcomes are less satisfactory than those produced by classical computing. Moreover, quantum computing technology is still in its early stages, which means that even promising theories may need more practical implementation due to technological constraints. Despite these challenges, the FALQON algorithm has consistently achieved optimal results for the TSP in various test scenarios. Its reliability makes it a valuable tool for finding optimal solutions.

Furthermore, the reliable results from these experiments underscore the practical and valuable applications of quantum modules in optimization problems, especially as quantum technology becomes more readily available. This research implies that the quantum method exhibits excellent potential despite the constraints of present simulation technology. As quantum technologies progress, the integration of algorithms such as FALQON holds even greater promise, given the possibility of its full deployment instead of the simplified approach.

Regarding the computation burden, training models using the quantum module take considerably longer than the classical module, mainly when dealing with the five-vertex graph. This is because the module is ran and simulated on a classical computer rather than being implemented through quantum circuits that are parameterized on quantum hardware computers.

6 Conclusions

This study has demonstrated that it is feasible to integrate quantum implementation elements into machine learning models. Although access to quantum computers is still limited, adapting these models to the quantum context can significantly enhance performance and pave the way for promising future applications, thereby advancing research at the intersection of quantum computing and machine learning.

Upon careful analysis, this work successfully achieves its intended purpose by exploring the fundamental principles of quantum computing and optimization and utilizing the OPF classifier, which boasts a multitude of practical applications. The outcomes of this research establish a robust groundwork for further investigation and demonstrate the potential of quantum techniques in augmenting conventional machine learning models. These findings imply that leveraging

quantum methodologies to enhance the OPF classifier can lead to significant progress at the intersection of quantum computing and classical machine learning.

6.1 Challenges

During the preparation of the quantum module, several challenges were faced. The first challenge involved working with large-scale matrices, which presented a significant technical difficulty. The second challenge related to the FALQON algorithm simulation using the Qiskit library required additional efforts to handle this resource effectively.

Moreover, when solving the Traveling Salesman Problem using the QUBO formulation, each edge in the graph is represented as a qubit, meaning that a graph with n edges will need n qubits to be properly represented in a quantum context. However, this approach presents computational challenges, especially when determining the size of matrices used to describe the operators in the Hamiltonian. To represent these operators, n matrices of order 2^n are required for every n number of edges in the graph. For instance, a complete graph model with four samples is formed by six edges, resulting in an operator with dimensions of $6 \times 64 \times 64$ (six quadratic matrices of order 64). As the number of edges grows, so does the memory needed to store these matrices, making it challenging to train models with large datasets.

6.2 Future Works

Based on the study's results and analysis, several promising directions for future research can improve the algorithm's efficiency. Initially, deep research in quantum optimization is essential to enhance the FALQON algorithm so that quantum simulations can be carried out through the Qiskit library and experiments using real quantum hardware.

Additionally, investigating techniques for manipulating large matrices and modeling quantum graphs can expand the training set, enabling the analysis of the quantum solution's advantage in larger datasets compared to classical models.

Acknowledgements. The authors are grateful to the São Paulo Research Foundation (FAPESP) grants 2013/07375-0, 2019/07665-4, 2021/04655-8, 2023/03726-4, 2023/10823-6, 2023/12830-0, and 2023/14427-8, to the Brazilian National Council for Scientific and Technological Development (CNPq) grant 308529/2021-9, to the Petrobras grant 2023/00466-1, and to the Office of Naval Research (ONR) grant N62909-24-1-2012.

References

1. Dang, Y., Jiang, N., Hu, H., Ji, Z., Zhang, W.: Image classification based on quantum k-nearest-neighbor algorithm. Quantum Information Processing **17**(9) (Aug 2018). https://doi.org/10.1007/s11128-018-2004-9

2. de Rosa, G.H., Papa, J.P.: Opfython: A python implementation for optimum-path forest. Software Impacts p. 100113 (2021https://doi.org/10.1016/j.simpa.2021.100113
3. Farhi, E., Goldstone, J., Gutmann, S.: A quantum approximate optimization algorithm (2014https://doi.org/10.48550/ARXIV.1411.4028
4. Feynman, R.P.: Simulating physics with computers. Int. J. Theor. Phys. **21**(6–7), 467–488 (1982). https://doi.org/10.1007/bf02650179
5. Finnila, A.B., Gomez, M.A., Sebenik, C., Stenson, C., Doll, J.D.: Quantum annealing: A new method for minimizing multidimensional functions. Chem. Phys. Lett. **219**(5–6), 343–348 (1994)
6. Hong, Z., Yang, J.: Lung Cancer. UCI Machine Learning Repository (1992)
7. Janosi, Andras, S.W.P.M., Detrano, R.: Heart Disease. UCI Machine Learning Repository (1988)
8. Jeswal, S.K., Chakraverty, S.: Recent developments and applications in quantum neural network: A review. Archives of Computational Methods in Engineering **26**(4), 793–807 (2018). https://doi.org/10.1007/s11831-018-9269-0
9. Johansson, J., Nation, P., Nori, F.: Qutip 2: A python framework for the dynamics of open quantum systems. Comput. Phys. Commun. **184**(4), 1234–1240 (2013) https://doi.org/10.1016/j.cpc.2012.11.019, https://www.sciencedirect.com/science/article/pii/S0010465512003955
10. Jordan, S.: Traveling santa problem (2018), http://quantumalgorithmzoo.org/
11. Kahn, M.: Diabetes. UCI Machine Learning Repository
12. Kato, T.: Trotter's product formula for an arbitrary pair of self-adjoint contraction semigroup. Topics in Func. Anal., Adv. Math. Suppl. Studies **3**, 185–195 (1978)
13. Kochenberger, G., Hao, J.K., Glover, F., Lewis, M., Lü, Z., Wang, H., Wang, Y.: The unconstrained binary quadratic programming problem: a survey. J. Comb. Optim. **28**, 58–81 (2014)
14. Kuncheva, L.: Artificial data sets (2005), https://lucykuncheva.co.uk/activities/
15. Magann, A.B., Rudinger, K.M., Grace, M.D., Sarovar, M.: Feedback-based quantum optimization. Physical Review Letters **129**(25) (Dec 2022https://doi.org/10.1103/physrevlett.129.250502
16. McCulloch, W.S., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. Bull. Math. Biophys. **5**(4), 115–133 (1943). https://doi.org/10.1007/bf02478259
17. Montoya-Zegarra, J.A., Papa, J.P., Leite, N.J., da Silva Torres, R., Falcão, A.X.: Novel Approaches for Exclusive and Continuous Fingerprint Classification, p. 386–397. Springer Berlin Heidelberg (2009https://doi.org/10.1007/978-3-540-92957-4_34
18. Montoya-Zegarra, J.A., Papa, J.P., Leite, N.J., da Silva Torres, R., Falcão, A.: Learning how to extract rotation-invariant and scale-invariant features from texture images. EURASIP Journal on Advances in Signal Processing **2008**(1) (Mar 200https://doi.org/10.1155/2008/691924
19. Papa, J.P., Falcão, A.X., Albuquerque, V.H.C., Tavares, J.M.R.S.: Efficient supervised optimum-path forest classification for large datasets. Pattern Recogn. **45**(1), 512–520 (2012)
20. Papa, J.P., Falcão, A.X., Suzuki, C.T.N.: Supervised pattern classification based on optimum-path forest. Int. J. Imaging Syst. Technol. **19**(2), 120–131 (2009)
21. Papa, J.P., Spadotto, A.A., Falcao, A.X., Pereira, J.C.: Optimum path forest classifier applied to laryngeal pathology detection. In: 2008 15th International Conference on Systems, Signals and Image Processing. IEEE (Jun 2008https://doi.org/10.1109/iwssip.2008.4604414

22. Passos, L.A., Jodas, D.S., Ribeiro, L.C., Akio, M., De Souza, A.N., Papa, J.P.: Handling imbalanced datasets through optimum-path forest. Knowl.-Based Syst. **242**, 108445 (2022)
23. Ribeiro, P.B., Passos, L.A., Da Silva, L.A., da Costa, K.A., Papa, J.P., Romero, R.A.: Unsupervised breast masses classification through optimum-path forest. In: 2015 IEEE 28th International Symposium on Computer-Based Medical Systems. pp. 238–243. IEEE (2015)
24. S, J., Jossinet, J.: Breast Tissue. UCI Machine Learning Repository (2010), https://doi.org/10.24432/C5P31H
25. Schuld, M., Petruccione, F.: Supervised Learning with Quantum Computers. Springer International Publishing (2018). https://doi.org/10.1007/978-3-319-96424-9
26. Schuld, M., Sinayskiy, I., Petruccione, F.: Prediction by linear regression on a quantum computer. Physical Review A **94**(2) (Aug 2016https://doi.org/10.1103/physreva.94.022342
27. Sigillito, V., W.S.H.L., Baker, K.: Ionosphere. UCI Machine Learning Repository (1989)
28. de Souza, R.W.R., De Oliveira, J.V.C., Passos, L.A., Ding, W., Papa, J.P., de Albuquerque, V.H.C.: A novel approach for optimum-path forest classification using fuzzy logic. IEEE Trans. Fuzzy Syst. **28**(12), 3076–3086 (2019)
29. Wakeham, D., Ceroni, J.: Feedback-based quantum optimization (falqon) (05 2021)

Face to Cartoon Incremental Super-Resolution Using Knowledge Distillation

Trinetra Devkatte[1], Shiv Ram Dubey[1(✉)], Satish Kumar Singh[1], and Abdenour Hadid[2]

[1] Computer Vision and Biometrics Lab, Indian Institute of Information Technology, Allahabad, Allahabad, India
{mit2021096,srdubey,sk.singh}@iiita.ac.in
[2] Sorbonne Centre for Artificial Intelligence, Sorbonne University, Abu Dhabi, Abu Dhabi, United Arab Emirates
abdenour.hadid@ieee.org

Abstract. Facial super-resolution/hallucination is an important area of research that seeks to enhance low-resolution facial images for a variety of applications. While Generative Adversarial Networks (GANs) have shown promise in this area, their ability to adapt to new, unseen data remains a challenge. This paper addresses this problem by proposing an incremental super-resolution using GANs with knowledge distillation (ISR-KD) for face to cartoon. Previous research in this area has not investigated incremental learning, which is critical for real-world applications where new data is continually being generated. The proposed ISR-KD aims to develop a novel unified framework for facial super-resolution that can handle different settings, including different types of faces such as cartoon face and various levels of detail. To achieve this, a GAN-based super-resolution network was pre-trained on the CelebA dataset and then incrementally trained on the iCartoonFace dataset, using knowledge distillation to retain performance on the CelebA test set while improving the performance on iCartoonFace test set. Our experiments demonstrate the effectiveness of knowledge distillation in incrementally adding capability to the model for cartoon face super-resolution while retaining the learned knowledge for facial hallucination tasks in GANs. The code is released at https://github.com/TRINETRA-DEVKATTE/ISR-KD-Network.

Keywords: Knowledge Distillation · Incremental Learning · Face to Cartoon Super-Resolution

1 Introduction

Facial super-resolution/hallucination is a crucial field of research that aims to enhance the quality of low-resolution facial images for various applications, including security systems, medical imaging, and entertainment [24]. Generative Adversarial Networks (GANs) [7] have shown promising results in facial super-resolution tasks [31]. However, one of the challenges faced by GANs is with their limited ability to adapt to new and unseen data. This limitation becomes particularly critical in real-world scenarios where new facial data is continuously generated, such as in surveillance systems or video streaming platforms.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 99–114, 2025.
https://doi.org/10.1007/978-3-031-78183-4_7

In this work, we address this challenge by exploring the potential of incremental learning in the context of GAN-based facial super-resolution. Incremental learning enables the model to continually learn and adapt to new data while retaining knowledge acquired from previous training stages [27]. The proposed Incremental Super-Resolution with Knowledge Distillation (ISR-KD) aims to develop a unified framework for facial hallucination that can handle various settings, including different types of faces and different levels of detail. The proposed ISR-KD leverages the benefits of knowledge distillation [10] to retain the performance of the pre-trained GAN-based super-resolution network while incrementally learning from new data.

To evaluate the effectiveness of our approach, we consider a pre-trained GAN-based super-resolution network on the CelebA dataset, which consists of a large collection of celebrity faces, and then incrementally train the network on the iCartoonFace dataset for super-resolution, which contains cartoon-style images. The knowledge distillation loss is utilized for training. The experimental results demonstrate the effectiveness of knowledge distillation in incrementally expanding the model's capability for facial hallucination tasks within the GAN framework. The ability to incrementally adapt to new data and retain previously learned knowledge makes the proposed approach highly suitable for real-world applications where the facial data distribution evolves over time. The major contributions of this paper are as follows:

- This paper proposes an Incremental Super-Resolution technique using Knowledge Distillation (ISR-KD) for exploiting the existing knowledge for super-resolution on new types of images without training from scratch.
- The proposed model is originally developed and trained for face super-resolution and incrementally trained for cartoon face super-resolution.
- The proposed model is able to improve the cartoon face super-resolution performance with negligible performance drop for original face super-resolution.

In the remainder of this paper, Sect. 2 provides a comprehensive review of literature while Sect. 3 presents the proposed ISR-KD framework. Sections 4 and 5 describe the experimental setup and the experimental results, respectively. Finally, concluding remarks are drawn in Sect. 6.

2 Related Work

2.1 Incremental Learning and Knowledge Distillation

Incremental learning has been widely exploited in computer vision for different applications to incrementally add new classes to a trained model [30]. Knowledge distillation is also heavily utilized to transfer the gained knowledge from one model to other model or one type of data to other type of data [8,21]. Welling *et al.* [25] proposed herding selection criterion to choose samples from the previous dataset. The incremental Classifier and Representation Learning (iCaRL) was proposed by Rebuffi *et al.* [20] which first extracts features using new data and then performs classification using nearest mean of exemplars rule, after that it combines the classification and distillation loss to adjust the exemplar. End-to-End Incremental Learning (EEIL) was proposed by Castro *et al.* [2] to perform feature extraction and classification. EEIL uses a joint loss function for

classification and distillation. The problem of class imbalance was addressed by Hou *et al.* [11] by proposing a unified classifier and incremental learning. Bias Correction Layer (BiC) was introduced by Wu *et al.* [27] to handle the last fully connected layer being biased towards new classes.

Chenshen *et al.* [26] proposed memory replay GANs which combats catastrophic forgetting problems by joint retraining and aligning replays. Mengyao *et al.* proposed Lifelong GAN [29] which used knowledge distillation to combat catastrophic forgetting by encouraging the model to produce visually similar results to a pre-trained model. The Learning without Forgetting (LwF) [15] method proposed by Li and Hoiem in 2017 prevents catastrophic forgetting by adding task-specific parameters to the original model when learning a new task. However, LwF requires storing parameters for each learned task and is dependent on the correlation between the tasks. Dhar *et al.* (2019) proposed Learning without Memorizing (LwM) [4] that allows a model to learn new classes incrementally without requiring data from base classes. This is achieved by restricting the divergence between student and teacher models using attention maps generated from the gradient flow information.

Though incremental learning is widely used to add new classes and knowledge distillation is heavily utilized for light-weight models, they are not well explored for super-resolution over new type of images. In this paper, the incremental learning facilitated by knowledge distillation is exploited for face to cartoon incremental super-resolution.

2.2 Face Super-Resolution

In recent years, deep learning models have shown outstanding performance for Face Super-Resolution (FSR) [13]. Hao *et al.* [5] proposed PCA-SRGAN which pays attention to the cumulative discrimination in the orthogonal projection space spanned by a PCA projection matrix of face data to improve the performance of GAN-based models on super-resolving face images. The Edge and Identity Preserving Network (EIPNet) [14] addresses the distortion of facial components by providing edge information and data distributions. A generative and controllable face super-resolution (GCFSR) framework is introduced in [9] that reconstructs high-resolution images while preserving identity information without additional priors. A deep FSR method with iterative collaboration between two recurrent networks is proposed in [19] by leveraging the facial landmarks for image recovery and accurate landmark estimation. A supervised pixel-wise GAN (SPGAN) is investigated in [32] that performs the super-resolution at different scales while considering face identity. The denoising diffusion probabilistic models are combined with image-to-image translation to perform super-resolution via repeated refinement [23].

A pre-prior guided approach is exploited in [18] that extracts facial prior information from high-resolution images and embeds them into low-resolution images to improve face reconstruction performance. Shuang *et al.* [16] uses a multi-scale deep network that incorporates both global and local facial priors to generate high-quality super-resolved face images. First, the feature extraction module extracts multi-scale features of the input image, then the super-resolution module utilizes these features along with the facial parsing prior to generate high-quality super-resolved face images. The 3D facial priors are incorporated into face super-resolution in [12] by exploiting the facial structures and identity information for improved performance.

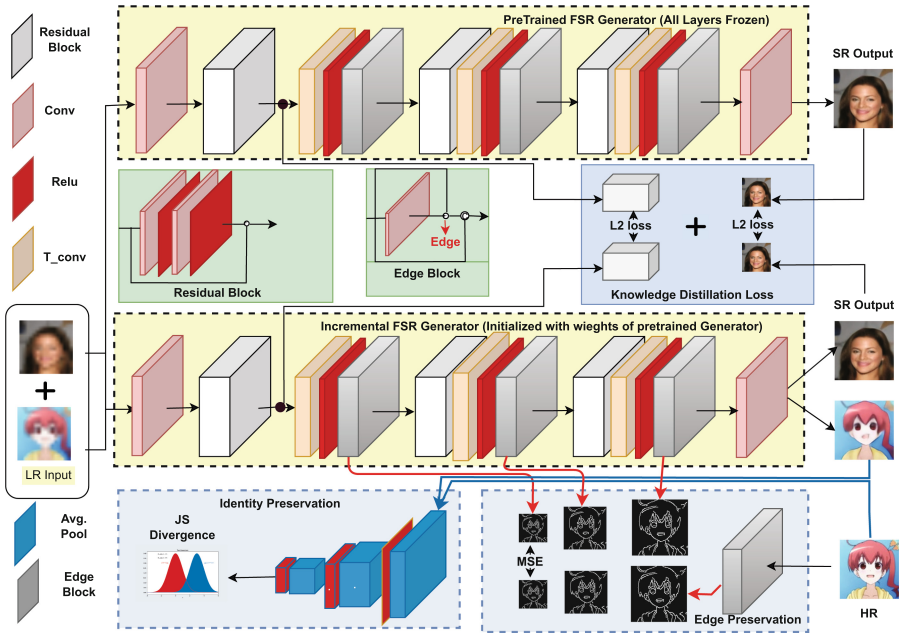


Fig. 1. Proposed face to cartoon incremental super-resolution method using knowledge distillation. Conv, ReLU, and T_conv represent Convolution Layer, ReLU Layer, and Transpose Convolution Layer, respectively. Pre-trained FSR Generator is trained on CelebA Dataset. The incremental FSR Generator is initialized with weights of Pre-trained FSR Generator and trained on combined CelebA and iCartoonFace images using the proposed method.

A SPARNet architecture is proposed in [3] for face super-resolution by leveraging spatial attention mechanisms to capture key face structures effectively. SPARNet achieves promising performance, even for very low-resolution faces. A self-attention learning network (SLNet) is proposed in [28] for three-stage face hallucination. SLNet leverages the interdependence of low and high-level spaces to achieve better reconstruction. A CNN-Transformer Cooperation Network (CTCNet) is investigated in [6] for face hallucination by incorporating a local-global feature cooperation module and a feature refinement module to enhance the local facial details and global facial structure restoration.

From the above, it appears that all existing works do not address the face super-resolution with incremental learning, where the network is trained on one type of faces and extended to other type of faces. This paper proposes the hallucination for face to cartoon incremental learning scenario.

3 Proposed Methodology

3.1 Problem Description

We tackle the problem of incremental super-resolution, where we consider a trained model to perform super-resolution in source domain, i.e., $I^S_{LR} \rightarrow I^S_{SR}$, and incrementally train it to perform super-resolution in target domain, i.e. $I^T_{LR} \rightarrow I^T_{SR}$, while retaining the performance for super-resolution in source domain. In the experiments, the source and target domains are considered as human faces from CelebA dataset and cartoon faces from iCartoonFace dataset, respectively.

Let $I_{HR} \in \mathbb{R}^{h_{HR} \times w_{HR} \times c}$ be the high resolution image data, and $I_{LR} \in \mathbb{R}^{h_{LR} \times w_{LR} \times c}$ be the corresponding low resolution image data, where h_{HR} and w_{HR} are height and width of high resolution images, h_{LR} and w_{LR} are height and width of low resolution images, and c is the number of channels. The image degradation function ϕ can be described as,

$$I_{LR} = \phi(I_{HR}, \theta),$$

where θ represents the choice of kernel for down-sampling the image, and the random noise is added to the image to account for random variations during the down-sampling.

Face super-resolution (FSR) is the inverse process of image degradation for down-sampling a high resolution image. Mathematically, it is expressed as,

$$I_{SR} = \phi^{-1}(I_{LR}, \delta) = G(I_{LR}, \delta),$$

where G is the FSR generator network with parameters δ, and the super-resolved image is represented by I_{SR}.

To update the FSR generator using incremental learning, we use a pre-trained FSR generator G_S trained on the source domain images (I^S_{LR}, I^S_{HR}) as a starting point. Then, we train our Incremental FSR generator G_T on a new dataset which is created using a combination of source domain and target domain images. Using images from a source domain while incrementally training for a target domain allows us to use knowledge distillation. This helps us in combatting catastrophic forgetting problem. Knowledge distillation is achieved by feeding the low resolution images from source domain I^S_{LR} as input to the pre-trained FSR generator G_S as well as the incremental FSR generator G_T, simultaneously and comparing their outputs, i.e., $I^S_{SR,S} = G_S(I^S_{LR})$ and $I^S_{SR,T} = G_T(I^S_{LR})$, using L2 loss. It Incentivizes the incremental FSR generator G_T to maintain its performance on the source domain task. The proposed method is illustrated in Fig. 1.

The incremental learning process combined with knowledge distillation can be formalized as follows,

$$\delta_{optimal} = \underset{\delta_T}{\operatorname{argmin}} \begin{bmatrix} \mathcal{L}_T(I^T_{HR}, G_T(I^T_{LR}, \delta_T)) + \\ \lambda \mathcal{L}_{kd}(G_T(I^S_{LR}, \delta_T), G_S(I^S_{LR}, \delta_S)) \end{bmatrix}, \tag{1}$$

where $\mathcal{L}_T$ is the loss function measuring the difference between the super-resolved images $I^T_{SR,T} = G_T(I^T_{LR}, \delta_T)$ and the high-resolution images I^T_{HR} in the target domain. $\mathcal{L}_{kd}$ is the knowledge distillation loss function, which measures the difference between the outputs of the incremental FSR generator G_T and the pre-trained

FSR generator G_S when fed with low-resolution images I^S_{LR} from the source domain. δ_S represents the parameters of the pre-trained FSR generator G_S which are obtained by training from scratch on source domain. δ_T represents the parameters of the incremental FSR generator G_T that are first initialized as δ_S and then updated to minimize the combined loss.

The goal of the above objective is to update the parameters δ_T of the incremental FSR generator G_T in a way that minimizes the loss functions for target domain images, while also ensuring that the knowledge distillation loss between G_T and G_S is minimized. It should be noted that δ_S is kept frozen while we train the incremental FSR generator G_T in an incremental fashion. The regularization coefficient λ balances the importance of knowledge distillation loss in the overall objective.

3.2 Knowledge Distillation

The proposed method utilizes a pre-trained Facial Super-Resolution (FSR) generator G_S, as depicted in Fig. 1, which has been trained on source domain images from the CelebA dataset. To ensure the stability of G_S during the training process, all layers of G_S are frozen. This pre-trained FSR generator is then employed for knowledge distillation to mitigate the issue of catastrophic forgetting. Another generator, denoted as G_T (Incremental FSR generator), as shown in Fig. 1, is initialised using weights of G_S and incrementally trained on target domain images from the iCartoonFace dataset and a small subset of images from the source domain CelebA dataset. Low resolution images from source domain I^S_{LR} are given as input to G_S and G_T, the generated super-resolved images $I^S_{SR,S}$ and $I^S_{SR,T}$ are then used for knowledge distillation. Specifically, knowledge distillation is performed by computing the L2 loss between $I^S_{SR,S}$ and $I^S_{SR,T}$. Basically, this loss encourages the generator G_T to match the output of generator G_S for the source domain images, thereby preventing the forgetting of source domain knowledge by the network G_T.

Moreover, the outputs from bottleneck layers of G_S and G_T for low-resolution images from source domain I^S_{LR} are also compared using the L2 loss. This comparison ensures that the bottleneck representation of G_T, which is actively being trained on inputs from the target domain, does not deviate significantly from the bottleneck representation of G_S, which was trained on inputs from the source domain. The knowledge distillation loss is expressed as follows,

$$\mathcal{L}_{kd} = \mathcal{L}_R(I^S_{SR,S}, I^S_{SR,T}) + \mathcal{L}_F(h_t, h_s), \tag{2}$$

where $I^S_{SR,S}$ and $I^S_{SR,T}$ represent the outputs of the G_S and G_T, respectively, while h_s and h_t indicate the hidden layer responses from the G_T and G_S, respectively for the source domain images. $\mathcal{L}_R$ represents the response-based loss, computed on the output logits, and $\mathcal{L}_F$ denotes the feature loss, computed from the hidden layer.

3.3 Edge Block

Traditional generative networks can lead to blurry images as high-frequency edge information is often lost during image resizing and re-scaling. Edge information can be used

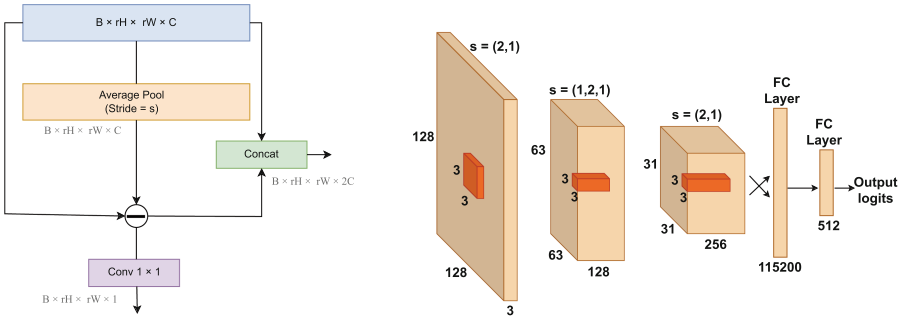


Fig. 2. (*left*) Edge block having an edge extraction layer. Here, B is the batch size, H is the height of tensor, W is the width of tensor, c is the number of channels in tensor, r is the scaling factor and s is the stride variable. (*right*) A schematic diagram of Discriminator architecture. Here, s indicates the stride and the list of numbers adjacent to s indicates the stride of convolution layers grouped with the same number of output channels.

to improve the quality of images generated by face-generative networks by providing additional guidance to the generator during training. The edge block allows the generator to better preserve the fine details of the input images, by capturing the structural differences between two images rather than individual pixels. In addition, incorporating edge information can help to combat the disadvantages of using L2 loss, which is sensitive to changes in individual pixels and causes the blurry generated images.

The edge block, as depicted in Fig. 2(*left*), plays a key role in leveraging edge information within the facial resolution network. This block is designed to be computationally efficient and consists of a low-pass filter constructed using an average pooling layer. The edge block uses a variable kernel size with stride 1 and "SAME" padding to retain the spatial resolution. The kernel size used during the average pool step depends on the size of input image, i.e., kernel size as $(5, 7, 10)$ is used for input tensor of size $(32 \times 32$, 64×64, $128 \times 128)$, respectively. By applying the low-pass filter, a blurred version of the original image is obtained. The difference between the blurred image and the original image yields the edge map, which highlights the edges present in the image. The edge map is then concatenated with the original image and propagated to the subsequent layer. To reduce the number of edge maps to a single-channel representation, a point-wise convolution operation is employed. This single-channel edge map is subsequently compared with the edge map of the high-resolution image, which is obtained using the Canny edge detector [1] by applying adaptive threshold selection [22].

Let E_{HR} be the edge map of the high-resolution image and E_{SR} be the edge map of the generated super-resolved image. The edge loss ($\mathcal{L}_{edge}$) is computed by comparing the generated edge map (E_{SR}) with the ground truth edge map (E_{HR}). Therefore, the edge loss ($\mathcal{L}_{edge}$) can be defined as:

$$\mathcal{L}_{edge} = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} (E_{SR}(i,j) - E_{HR}(i,j))^2, \tag{3}$$

where H and W represent the height and width of the edge maps, respectively. Minimizing the edge loss encourages the generated image to have similar edge structures as the high-resolution ground truth image, leading to enhanced sharpness and preservation of fine details.

3.4 Generator Architecture

The generator network consists of three similar modules with a convolution layer at the beginning and at the end. Each module is made up of a residual block, transpose convolution layer, ReLU activation function and an edge block. The edge block is shown in Fig. 2(*left*). The residual block is made up of two convolution layers, each followed by a ReLU activation function. The transpose convolution performs a $2\times$ up-sampling. Overall, the network performs $8\times$ up-sampling. In Fig. 1, refer to Incremental FSR Generator for detailed visualization. The generator is trained with adversarial loss and other losses described in this paper.

3.5 Discriminator Architecture

The discriminator is a CNN used to distinguish between real and generated images. It is illustrated in Fig. 2(*right*). The network consists of seven convolution layers with leaky ReLU activation in the first six layers and two fully-connected layers. The input to the network is a super-resolution image which is passed through the network to determine if it is a real or a generated image. Initially, the input image passes through the first convolution layer, which has a kernel size of 3×3, a stride of 1, and 128 output channels. The output of this layer is then passed through another convolution layer, which has a kernel size of 3×3, a stride of 2, and 128 output channels. The same process is repeated for two more convolution layers, each with 256 output channels. After these layers, the image is passed through two more convolution layers, with 256 and 512 output channels, respectively. Finally, the image is passed through one last convolution layer, with 512 output channels. The output of this layer is then flattened and passed through two fully-connected layers to determine whether the input image is a real or a generated high-resolution image. The leaky ReLU activation function is used for all convolution layers with a slope of 0.2 for negative activations. The generator and discriminator are trained with an adversarial loss function to classify real and generated images.

3.6 Objective Function

The objective function for Face to Cartoon Incremental Super-Resolution using Knowledge Distillation task is given by loss $\mathcal{L}_{ISR-KD}$ as,

$$\begin{aligned}\mathcal{L}_{ISR-KD} = \lambda_{kd}\mathcal{L}_{kd} + \lambda_{edge}\mathcal{L}_{edge} + \lambda_{ad}\mathcal{L}_{ad} \\ +\lambda_{lce}\mathcal{L}_{lce} + \lambda_{id}\mathcal{L}_{id} + \lambda_{rl}\mathcal{L}_{rl},\end{aligned} \tag{4}$$

where $\mathcal{L}_{kd}$, $\mathcal{L}_{edge}$, $\mathcal{L}_{ad}$, $\mathcal{L}_{lce}$, $\mathcal{L}_{id}$ and $\mathcal{L}_{rl}$ are knowledge distillation loss, edge loss, adversarial loss, luminance-chrominance error, identity loss and reconstruction loss, respectively. λ_{kd} represents the combined effect of L_R and L_F on $\mathcal{L}_{kd}$ as shown in Eq.

(2). λ's are the hyper parameters used as the weights for different loss functions. $\mathcal{L}_{edge}$ loss is defined in Eq. (3), respectively. $\mathcal{L}_{ad}$, $\mathcal{L}_{lce}$, $\mathcal{L}_{id}$ and $\mathcal{L}_{rl}$ losses are described in the rest of this subsection.

Adversarial Loss. The adversarial loss function is defined as,

$$\begin{aligned} L_{ad} &= \mathbb{E}_{x\sim p(I^T_{HR})}[\log D_T(x)] \\ &+\mathbb{E}_{z\sim p(I^T_{LR})}[\log(1 - D_T(G_T(z)))]. \end{aligned} \tag{5}$$

The aim of the Generator is to minimize the second term of equation L_{ad} as it wants to fool the discriminator by predicting the generated samples as real. The aim of the Discriminator is to maximize L_{ad} to make sure that the discriminator can accurately distinguish between samples coming from the probability distribution of high resolution images $p(I^T_{HR})$ and the probability distribution of hallucination images $p(I^T_{SR,T})$.

Luminance-Chrominance Error. Luminance-chrominance error occurs in image super-resolution tasks due to the mismatch between luminance (brightness) and chrominance (color) components.

The difference between two images in the Luminance-chrominance space (i.e., YCbCr) is calculated as,

$$\mathcal{L}_{lce} = \sqrt{\Delta Y^2 + \Delta Cb^2 + \Delta Cr^2}, \tag{6}$$

where ΔY, ΔCb and ΔCr are the difference between super-resolution and high-resolution images in Y, Cb and Cr channels, respectively. Minimizing $\mathcal{L}_{lce}$ improves the Luminance-chrominance preservation in the super-resolution images.

Identity Loss. We use the identity loss as the JS divergence which measures the similarity between two probability distributions. The generated and ground truth images are passed through a pre-trained Inception-V1 network to extract 512-class encoded vectors, denoted as V_{SR} and V_{HR}, respectively. The identity loss is calculated as,

$$\mathcal{L}_{id} = \frac{1}{2}KL(V_{SR}|M) + \frac{1}{2}KL(V_{HR}|M), \tag{7}$$

where KL is the Kullback-Leibler divergence and M is the average distribution of V_{SR} and V_{HR} given as, $M = (V_{SR}+V_{HR})/2$. The identity loss as JS divergence assesses the similarity between high-level features of generated and ground truth images, measuring how closely the generated output aligns with the high-resolution images.

Reconstruction Loss. The reconstruction loss is computed as mean squared error (MSE) between the pixel values of the generated image and the corresponding pixel values of the ground truth image. Mathematically, it can be expressed as,

$$L_{rl} = \frac{1}{N}\sum_{i=1}^{N}(I_{SR_i} - I_{HR_i})^2, \tag{8}$$

where N represents the total number of pixels in the images, and I_{SR_i} and I_{HR_i} denote the pixel values for the i^{th} pixel in the generated and ground truth images, respectively.

Table 1. The Experimental results of the proposed ISR-KD model on different dataset settings. The % change is obtained by comparing with the model on CelebA dataset trained from scratch.

Dataset	ISR-KD Setting	Test Dataset	PSNR	SSIM	FID
CelebA	NA	CelebA	24.2420	0.7097	44.3340
(From Scratch)		iCartoonFace	20.5817	0.5781	113.3130
Cartoon-CelebA-1	0 CelebA and	CelebA	23.5397 (↓2.90%)	0.6817 (↓3.95%)	74.3212 (↓67.64%)
	20,000 Cartoon images	iCartoonFace	20.6933 (↑0.54%)	0.5991 (↑3.63%)	98.8749 (↑12.74%)
Cartoon-CelebA-2	10,000 CelebA and	CelebA	23.9387 (↓1.25%)	0.7007 (↓1.27%)	59.3153 (↓33.79%)
	20,000 Cartoon images	iCartoonFace	20.7170 (↑0.66%)	0.6007 (↑3.91%)	98.0944 (↑13.43%)
Cartoon-CelebA-3	20,000 CelebA and	CelebA	23.9526 (↓1.19%)	0.7005 (↓1.30%)	61.3342 (↓38.35%)
	20,000 Cartoon images	iCartoonFace	20.7170 (↑0.66%)	0.5998 (↑3.75%)	99.3709 (↑12.30%)
Cartoon-CelebA-4	10,000 CelebA and	CelebA	24.0124 (↓0.95%)	0.7047 (↓0.70%)	57.2367 (↓29.10%)
	50,000 Cartoon images	iCartoonFace	20.8310 (↑1.21%)	0.6092 (↑5.38%)	87.6397 (↑22.66%)
Cartoon-CelebA-5	20,000 CelebA and	CelebA	24.1403 (↓0.42%)	0.7096 (↓0.01%)	58.1901 (↓31.25%)
	50,000 Cartoon images	iCartoonFace	20.8387 (↑1.25%)	0.6072 (↑5.03%)	91.8344 (↑ 18.96%)

4 Experimental Settings

Datasets: For experimental analysis, CelebA [17] and iCartoonFace [33] datasets are used in this paper. The CelebA dataset contains 202,599 face images from 10,177 identities. Whereas, the iCartoonFace dataset contains 389,678 cartoon face images from 5,013 identities. The CelebA dataset is used as the source domain and iCartoonFace dataset is used as the target domain. In this paper, we experiment with five different combinations (i.e., Cartoon-CelebA-1, Cartoon-CelebA-2, Cartoon-CelebA-3, Cartoon-CelebA-4 and Cartoon-CelebA-5) by considering varying number of images randomly from CelebA and iCartoonFace datasets as detailed in Table 1.

Experimental Setup: The data augmentation is performed by center cropping to a size of 178×178 pixels, resizing to a size of 128×128 pixels, horizontal flipping with a probability of 0.5, and rotating with 90 and $270°$. The generator and discriminator are trained using the Adam optimizer with learning rate $1e-4$ and Epsilon $1e-8$. $Beta1$ and $Beta2$ for Generator are 0.9 and 0.999, respectively. However, for Discriminator they are 0.5 and 0.9, respectively. The model is trained for 100 epochs. The model is trained and tested on an Nvidia Quadro RTX 6000 GPU using the TensorFlow framework. The values of $\mathcal{L}_R$, $\mathcal{L}_F$, and λ_{edge} hyperparameters are 5, 0.01, and 0.3. However, λ_{ad}, λ_{lce}, λ_{id}, and λ_{rl} are set to 1.

5 Experimental Results and Discussion

5.1 Quantitative Results

Table 1 presents the performance of the proposed ISR-KD network. In case of Cartoon-CelebA-1 dataset combination, where knowledge distillation is not employed as this dataset does not contain any CelebA face images, we observe an increase in the performance on the iCartoonFace test set, but a significant drop in the performance on the CelebA dataset. This decline can be attributed to the catastrophic forgetting problem,

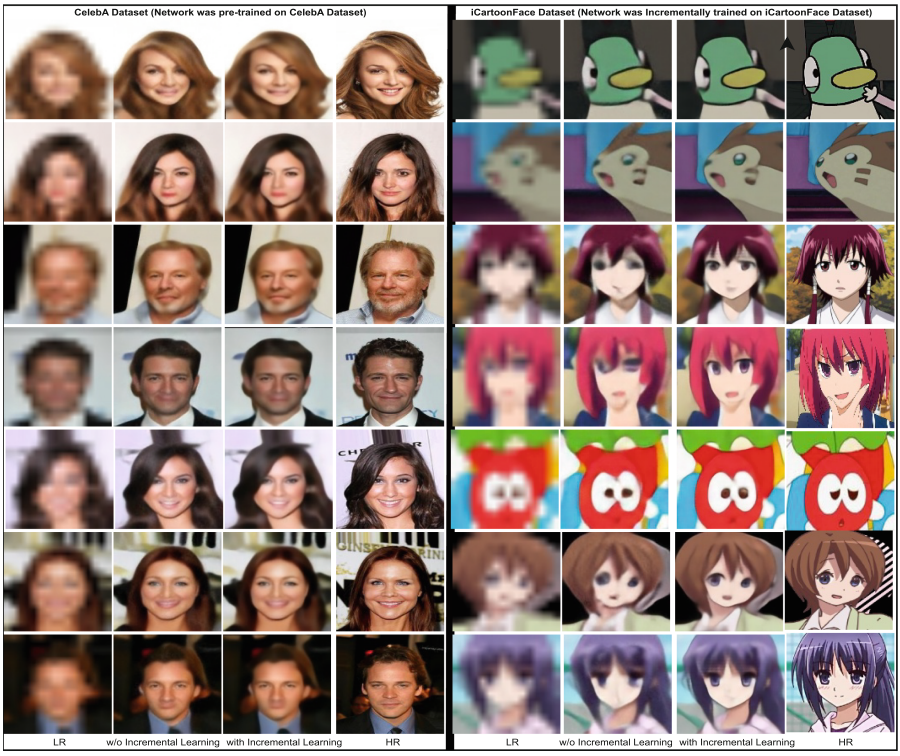


Fig. 3. The generated samples depicting the visual effects of using incremental learning in combination with knowledge distillation for facial super-resolution task. The left half of the image contains the results for CelebA dataset (Source Domain). The right half shows the results after incrementally training on the iCartoonFace dataset (Target Domain).

which occurs when a model forgets previously learned knowledge while training on new data. To mitigate this problem, we introduced incremental hallucination using knowledge distillation by re-feeding some images from the source domain (i.e., CelebA face images) to a pre-trained generator as shown for Cartoon-CelebA-2, Cartoon-CelebA-3, Cartoon-CelebA-4, and Cartoon-CelebA-5 datasets. This additional step introduces a knowledge distillation loss, which helps to overcome catastrophic forgetting. As a result, the performance on the CelebA test set is almost retained, while simultaneously improving the performance on the iCartoonFace test set. An improvement of 5.03% on iCartoonFace dataset and a drop of only 0.01% on CelebA dataset is observed in Table 1 in terms of SSIM on Cartoon-CelebA-5 training dataset setting, which has 20,000 images from CelebA and 50,000 images from iCartoonFace. We note that the number of samples from Cartoon images also plays an important role as the results using Cartoon-CelebA-5 setting are better than the Cartoon-CelebA-3 setting, in spite of having the same number of CelebA samples. Hence, the proposed approach can effectively deal with the catastrophic forgetting issue in the context of super-resolution.

Table 2. Similarity between the samples generated without and with incremental learning using the proposed approach.

Test Results Images	PSNR	SSIM	FID
CelebA Test Set	33.2640	0.9425	18.7007
iCartoon Test Set	28.8210	0.8850	25.3614

5.2 Qualitative Results

Figure 3 illustrates the visual results achieved by combining Incremental training and knowledge distillation. The ISR-KD model was first trained on the entire CelebA training set before undergoing incremental training on a small subset of the iCartoonFace dataset. The left portion of the image displays results for the CelebA dataset. Given that the model was initially trained on the CelebA dataset and subsequently incrementally trained on the iCartoonFace dataset, one might anticipate a substantial decline in the quality of super-resolved (SR) images after incremental training on the iCartoonFace dataset, an occurrence referred to as catastrophic forgetting. However, by employing knowledge distillation, we were able to counteract these effects, and the results clearly indicate that the images with and without incremental learning are nearly indistinguishable. The quality of images super-resolved from the iCartoonFace dataset improved after incremental training. In order to show the degree of degradation on source images and degree of improvement on target images, we compute the average similarity between the generated samples without and with incremental learning on both CelebA and iCartoon test sets in terms of PSNR, SSIM and FID in Table 2. The higher PSNR & SSIM and smaller FID on CelebA test set shows higher similarity between the generated images using without and with incremental learning, which means low degradation. The vice-versa is observed on iCartoon test set which shows more dissimilarity means high improvement in the generated samples after applying the incremental learning.

5.3 Ablation Study on Loss Hyperparameters

Table 3 presents the results of different loss hyperparameter settings on the Cartoon-CelebA-2 dataset. We focus on adjusting $\mathcal{L}_R$ and $\mathcal{L}_F$ values in Eq. (2), while keeping $\lambda edge$ fixed at 0.3 and other hyperparameters set to 1. The $\mathcal{L}_R = 5$ and $\mathcal{L}_F = 0.01$ are used in other experiments as this setting yields the best performance on the iCartoonFace dataset. This choice of hyperparameters shows a good trade-off between maintaining high performance on the CelebA dataset while enhancing the performance on the iCartoonFace dataset.

5.4 Cross-Dataset Analysis

We also perform the cross-dataset analysis by first training the model from scratch on $50,000$ Cartoon images and then applying incremental learning using the proposed approach on a dataset consisting of $20,000$ Cartoon images from source domain and $20,000$ CelebA face images from the target domain. The results reported in Table 4

Table 3. Performance comparison of various $\mathcal{L}_R$ and $\mathcal{L}_F$ loss hyperparameter settings in Eq. (2) on the Cartoon-CelebA-2 dataset with λ_{edge} fixed at 0.3 and all other hyperparameters set to 1.

Hyper-parameters	CelebA Test Dataset			iCartoonFace Test Dataset		
	PSNR	SSIM	FID	PSNR	SSIM	FID
$\mathcal{L}_R$ = 15, $\mathcal{L}_F$ = 0.04	24.0721	0.7086	58.0514	20.6202	0.5920	108.8132
$\mathcal{L}_R$ = 10, $\mathcal{L}_F$ = 0.08	24.1119	0.7082	57.1407	20.7152	0.5962	103.8281
$\mathcal{L}_R$ = 5, $\mathcal{L}_F$ = 0.08	24.0707	0.7070	57.7262	20.7109	0.5982	100.7698
$\mathcal{L}_R$ = 5, $\mathcal{L}_F$ = 0.04	24.0941	0.7069	58.0286	20.7397	0.5991	100.931
$\mathcal{L}_R$ = 5, $\mathcal{L}_F$ = 0.01	23.9387	0.7007	59.3153	20.717	0.6007	98.0944

Table 4. Cross-dataset analysis by first training on Cartoon images from scratch and then incrementally learn on CelebA images using the proposed method.

Train Dataset	iCartoonFace Test Dataset			CelebA Test Dataset		
	PSNR	SSIM	FID	PSNR	SSIM	FID
Cartoon (From Scratch)	20.8648	0.6055	94.0667	23.6665	0.6834	75.2700
Incremental Learning	20.8364	0.6061	90.7447	24.1532	0.7074	56.3191

Table 5. Comparison of proposed method with joint training approach.

Training Type	DataSet	PSNR	SSIM	FID
Incremental	**CelebA**	24.1403	0.7096	58.1901
	iCartoonFace	20.8387	0..6072	91.8344
Joint	**CelebA**	24.2541	0.7197	59.2465
	iCartoonFace	20.7822	0.6049	97.8604

confirm that the proposed ISR-KD improves the performance on CelebA while maintaining the similar performance on iCartoon dataset. This analysis also points out that performing super-resolution on cartoon face images is a difficult problem as compared to normal face images.

5.5 Comparsion with Joint Training Approach

As shown in the Table 5, the incremental learning approach using ISR-KD achieves comparable performance on the source dataset (CelebA) compared to the joint training approach. This is important because it demonstrates that our method can effectively retain knowledge from the original training data. However, on the target dataset (iCartoonFace), the joint training approach shows slightly better performance metrics (PSNR, SSIM). However, it's important to consider the context of our proposed method. ISR-KD is specifically designed for scenarios where the target data is not readily available but becomes accessible over time. In such situations, joint training is not feasible. Our approach offers a solution by allowing for incremental training on the target

Table 6. Results of Extended ISR-KD model when trained on Cartoon-CelebA-2 dataset setting.

Test Dataset	PSNR	SSIM	FID
CelebA	24.19 (0.23 ↓)	0.71 (0.31 ↑)	58.42 (31.76 ↑)
iCartoonFace	20.74 (0.77 ↑)	0.60 (3.94 ↑)	98.02 (13.49 ↑)

data while minimizing performance degradation on the source data. This is particularly valuable for applications where the model needs to continuously adapt to new data streams while maintaining performance on previously learned tasks.

5.6 Performance on Extended Network

In this experiment, the incremental FSR generator network is extended by adding six convolution layers with padding set as SAME to the rear end of the network. The newly added layers are initialized with random weights. The weights of the other layers are initialized from the pre-trained FSR generator (Fig. 1). The extended model is trained on the Cartoon-CelebA-2 dataset. Table 6 shows that increasing the depth of incremental FSR generator leads to better results for source (CelebA) as well as incremental target (iCartoonFace) facial hallucination tasks.

6 Conclusion

In this research paper, we addressed the problem of adapting GANs to new and unseen data in the context of facial hallucination. We combine the incremental learning and knowledge distillation in the proposed ISR-KD framework. The incorporation of knowledge distillation allows the model to retain the performance on previous dataset while enhancing its capability on new dataset. We used the pre-trained GAN-based super-resolution network on the CelebA dataset and incrementally trained it on the combined CelebA and iCartoonFace dataset using our proposed framework. We achieved superior performance on target Cartoon dataset while maintaining the performance on the source CelebA dataset, mitigating the issue of catastrophic forgetting. Future research directions include investigating different knowledge distillation methods, different networks, and evaluating on larger and more diverse datasets.

References

1. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **8**(6), 679–698 (1986)
2. Castro, F.M., Marín-Jiménez, M.J., Guil, N., Schmid, C., Alahari, K.: End-to-end incremental learning. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11216, pp. 241–257. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01258-8_15
3. Chen, C., Gong, D., Wang, H., Li, Z., Wong, K.Y.K.: Learning spatial attention for face super-resolution. IEEE Trans. Image Process. **30**, 1219–1231 (2021)

4. Dhar, P., Singh, R.V., Peng, K.C., Wu, Z., Chellappa, R.: Learning without memorizing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5138–5146 (2019)
5. Dou, H., Chen, C., Hu, X., Xuan, Z., Hu, Z., Peng, S.: PCA-SRGAN: incremental orthogonal projection discrimination for face super-resolution. In: Proceedings of the 28th ACM International Conference on Multimedia, pp. 1891–1899 (2020)
6. Gao, G., Xu, Z., Li, J., Yang, J., Zeng, T., Qi, G.J.: CTCNet: a CNN-transformer cooperation network for face image super-resolution. IEEE Trans. Image Process. **32**, 1978–1991 (2023)
7. Goodfellow, I., et al.: Generative adversarial networks. Commun. ACM **63**(11), 139–144 (2020)
8. Gou, J., Yu, B., Maybank, S.J., Tao, D.: Knowledge distillation: a survey. Int. J. Comput. Vision **129**, 1789–1819 (2021)
9. He, J., Shi, W., Chen, K., Fu, L., Dong, C.: GCFSR: a generative and controllable face super resolution method without facial and GAN priors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1889–1898 (2022)
10. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
11. Hou, S., Pan, X., Loy, C.C., Wang, Z., Lin, D.: Learning a unified classifier incrementally via rebalancing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 831–839 (2019)
12. Hu, X., et al.: Face super-resolution guided by 3D facial priors. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12349, pp. 763–780. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58548-8_44
13. Jiang, J., Wang, C., Liu, X., Ma, J.: Deep learning-based face super-resolution: a survey. ACM Comput. Surv. **55**(1), 1–36 (2021)
14. Kim, J., Li, G., Yun, I., Jung, C., Kim, J.: Edge and identity preserving network for face super-resolution. Neurocomputing **446**, 11–22 (2021)
15. Li, Z., Hoiem, D.: Learning without forgetting. IEEE Trans. Pattern Anal. Mach. Intell. **40**(12), 2935–2947 (2017)
16. Liu, S., Xiong, C., Gao, Z.: Face super-resolution network with incremental enhancement of facial parsing information. In: Proceedings of the 25th International Conference on Pattern Recognition, pp. 7537–7543 (2021)
17. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: Proceedings of the International Conference on Computer Vision, December 2015
18. Lu, T., Wang, Y., Zhang, Y., Jiang, J., Wang, Z., Xiong, Z.: Rethinking prior-guided face super-resolution: a new paradigm with facial component prior. IEEE Trans. Neural Netw. Learn. Syst. **35**, 3938–3952 (2022)
19. Ma, C., Jiang, Z., Rao, Y., Lu, J., Zhou, J.: Deep face super-resolution with iterative collaboration between attentive recovery and landmark estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, June 2020
20. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: iCaRL: incremental classifier and representation learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5533–5542 (2017)
21. Reddy, B.Y., Dubey, S.R., Sanodiya, R.K., Karn, R.R.P.: Context unaware knowledge distillation for image retrieval. In: Proceedings of the International Conference on Computer Vision and Machine Intelligence, pp. 65–77 (2023)
22. Rong, W., Li, Z., Zhang, W., Sun, L.: An improved canny edge detection algorithm. In: Proceedings of the IEEE International Conference on Mechatronics and Automation, pp. 577–582 (2014)
23. Saharia, C., Ho, J., Chan, W., Salimans, T., Fleet, D.J., Norouzi, M.: Image super-resolution via iterative refinement. IEEE Trans. Pattern Anal. Mach. Intell. **45**(4), 4713–4726 (2023)

24. Wang, Z., Chen, J., Hoi, S.C.: Deep learning for image super-resolution: a survey. IEEE Trans. Pattern Anal. Mach. Intell. **43**(10), 3365–3387 (2020)
25. Welling, M.: Herding dynamical weights to learn. In: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 1121–1128 (2009)
26. Wu, C., et al.: Memory replay GANs: learning to generate new categories without forgetting. Adv. Neural Inf. Process. Syst. **31** (2018)
27. Wu, Y., et al.: Large scale incremental learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 374–382 (2019)
28. Zeng, K., Wang, Z., Lu, T., Chen, J., Wang, J., Xiong, Z.: Self-attention learning network for face super-resolution. Neural Netw. **160**, 164–174 (2023)
29. Zhai, M., Chen, L., Tung, F., He, J., Nawhal, M., Mori, G.: Lifelong GAN: continual learning for conditional image generation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 2759–2768 (2019)
30. Zhang, J., et al.: Class-incremental learning via deep model consolidation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 1131–1140 (2020)
31. Zhang, M., Ling, Q.: Supervised pixel-wise GAN for face super-resolution. IEEE Trans. Multimedia **23**, 1938–1950 (2020)
32. Zhang, M., Ling, Q.: Supervised pixel-wise GAN for face super-resolution. IEEE Trans. Multimedia **23**, 1938–1950 (2021)
33. Zheng, Y., et al.: Cartoon face recognition: a benchmark dataset. In: Proceedings of the 28th ACM International Conference on Multimedia, pp. 2264–2272 (2020)

Copula Entropy Based Causal Network Discovery from Non-stationary Time Series

Jing Yang[1,2(✉)] and Xinzhi Rao[1,2]

[1] The Intelligent Interconnected Systems, Hefei University of Technology, Hefei 230009, China
jsjyj0801@163.com, 2022111055@mail.hfut.edu.cn
[2] The School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China

Abstract. The task of deducing the causal network from time series data and identifying relationships among multiple series is increasingly vital across various sectors such as industry, medicine, and finance. Despite numerous algorithms being proposed for this purpose, the majority are predicated on the stationarity assumption. However, in disciplines like climatology and neuroscience, time series often exhibit non-stationarity, characterized by a data distribution that shifts over time. In this paper, we introduce an innovative algorithm designed to discern causal relationships from non-stationary time series. Our approach unfolds in three key steps: Initially, we harness the concept of copula entropy to estimate the conditional transfer entropy, offering a streamlined method for non-parametric conditional independence testing. Subsequently, we introduce the time index, which influences other variables at specific time lags, and by integrating the conditional transfer entropy, we execute the independence tests. This leads us to propose the CE-CDN (Copula Entropy-based Causal Discovery from Non-stationary time series), a two-stage algorithm tailored for learning the causal network and identifying change modules. Finally, through comparative analysis with existing algorithms, our experimental findings indicate that CE-CDN not only excels in managing non-stationary time series but also boasts commendable time efficiency.

Keywords: Causal Discovery · Non-stationarity · Copula Entropy · Time Series

1 Introduction

Time series represent a sequence of data points recorded over time. Analyzing the relationships between these observations at various moments is crucial for enhancing predictive accuracy and informing decision-making processes. However, this analysis is often fraught with challenges, particularly the issue of non-stationarity. In numerous disciplines—ranging from economics [1] to climate research [2] and biological analysis [3]—the time series frequently exhibit non-stationarity. This characteristic indicates a shift in the data distribution over time. As a result, the analysis of causality within non-stationary time series gets extensive attention.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 115–131, 2024.
https://doi.org/10.1007/978-3-031-78183-4_8

Traditional causality methods for time series assume data stability, which can mislead when applied to non-stationary series that change over time. Recent approaches to non-stationary causality are split into two: one [4–6] tracks changes in causal strength, while the other [3] also detects shifts in causal directions. For the latter, it can be further lead to the problem of stationary state detection. We focus on the former as a foundational step for the latter.

In addressing the challenge of non-stationarity, Huang et al. [4] proposed temporal linear causal and nonlinear noise models, yet failed to locate the changing causal module. In a subsequent effort, the same team [5] developed a state-space model that accommodates temporal variations in causal intensity and noise, yet this model is predicated on linearity. Zhang et al. [6] incorporated time or domain indices to delineate the evolving causal modules and to ascertain instantaneous causal relationships. Kang et al. [7] offered a high-order causal entropy for distributed network analysis, though at a high computational cost. Building upon these contributions, we introduce a novel two-stage algorithm, CE-CDN, designed to identify lagged change modules and discern causal relationships that encompass nonlinear. Furthermore, CE-CDN offers a streamlined approach for non-parametric conditional independence testing.

The work of this paper is organized as follows:

(1) We begin by introducing the relationship between mutual information and copula entropy [8]. We then reformulate the conditional transfer entropy as the sum of four distinct copula entropy components. Finally, we estimate the copula entropy using an entropy estimator. This transformation is demonstrated to offer a more effective approach to measuring independence;
(2) At the maximum time delay, we introduce a smoothly varying time-related variable that laggingly affects others to represent the non-stationary mechanism. Subsequently, under the additive noise model and by integrating the conditional transfer entropy with a constraint-based method, we propose the CE-CDN algorithm. This algorithm is designed to learn the relationships among variables and between variables and the time index;
(3) We compare CE-CDN with other causal discovery algorithms for time series, including CCM, HCE, NAVAR, DYNOTEARS, VarLiNGAM and PCMCI(cmiknn and gpdc). Experiments demonstrate that CE-CDN achieves good results on non-linear and non-stationary data, with superior time performance. Furthermore, the algorithm is applied to a real dataset to identify causal connections.

Sections are organized as follows: Sect. 2 introduces related work; Sect. 3 describes preliminary knowledge; Sect. 4 details the CE-CDN; Sect. 5 and 6 analyses the experimental results; Sect. 7 concludes the work of this paper.

2 Related Work

Traditional causal network learning methods for time series fall into three categories: Constraint-based: tsFCI by Entner and Hoyer et al. [9] and PCMCI by Runge et al. [10], which combines PC and MCI. Functional Causal Models: Hyvarinen et al. [11] fused LiNGAM with VAR, and Peter et al. [12] introduced models with independent noise.

Score-based: Pamfil et al. [13] extended NOTEARS to DYNOTEARS, and Sun et al. [14] used convolutional neural networks for time series analysis.

The above algorithms are under the assumption of stationary time series. We now turn to the discussion of non-stationary causal discovery works, which encompass both instantaneous and time-delayed relationships.

For the instantaneous causal discovery, Huang et al. [5] introduced a state-space model where the causal coefficients and noise variance are permitted to vary over time. These parameters are estimated using the SAEM algorithm, albeit under the assumption of linear causality, which may not be universally applicable. Zhang et al. [6] proposed CD-NOD, which utilizes a proxy variable to account for confounding factors, followed by the application of traditional constraint-based methods combined with non-parametric conditional independence tests to learn the causal structure. Fujiwara et al. [15], building on the linear non-Gaussian-acyclic model and the just-in-time framework, introduced JIT-LiNGAM. This model estimates the local linear causal structure from past data upon receiving a new input sample, demonstrating robust nonlinear detection capabilities. However, it has not yet been adapted for time-delay embedding scenarios.

We concentrate on time-delayed causal relationships, where the causality is inherently from the past to the present. To address this, Xing et al. [16] proposed Time-Varying Dynamic Bayesian Networks (TV-DBNs) to model the directed dependence and structural changes in non-stationary entities such as organisms or neural time series. Huang et al. [4] suggested exploiting smooth changes to estimate a time-varying causal model, using temporal information as a common cause and applying Gaussian process regression for estimation. However, this method lacks a clear mechanism for positioning changing modules. Kang et al. [7] introduced the High-Order Causal Entropy (HCE) algorithm, which identifies network structures in a distributed manner. Yet, it struggles with high-dimensional challenges and suffers from high time complexity.

In order to identify lagged change modules and provide a simple method for the non-parameter conditional independence test in causal learning processes, this paper proposes a causal structure learning algorithm CE-CDN that can effectively deal with nonlinear and non-stationary time series.

3 Preliminary Knowledge

3.1 Mutual Information and Transfer Entropy

Mutual information is an important measure in information theory, widely used to quantify the correlation between random variables. The mutual information between two variables, symbolized here as I, can be described by Eq. (1).

$$I\left(X^1, X^2\right) = H\left(X^2\right) - H\left(X^2|X^1\right) = H\left(X^1\right) + H\left(X^2\right) - H(X^1, X^2) \quad (1)$$

The following H represents the differential entropy. When extended to high-dimensional continuous observation variables, the multivariate mutual information of the generalized redundant [17] can be calculated by Eq. (2).

$$I\left(X^1, X^2, \ldots, X^n\right) = H\left(X^1\right) + H\left(X^2\right) + \cdots + H\left(X^n\right) - H(X^1, X^2, \ldots, X^n) \quad (2)$$

Transfer entropy is essentially conditional mutual information, under which the dynamic process is considered simultaneously. Specifically, it measures how the past of X decreases the uncertainty of Y, given the past of Y. When considering the conditional set $\boldsymbol{Z}$, it is called the conditional transfer entropy, as shown in Eq. (3).

$$\begin{aligned} CTE_{X\to Y|\boldsymbol{Z}}(t) &= I(Y_t, X_{t-1}|Y_{t-1}, \boldsymbol{Z}_{t-1}) = H(Y_t|Y_{t-1}, \boldsymbol{Z}_{t-1}) - H(Y_t|X_{t-1}, Y_{t-1}, \boldsymbol{Z}_{t-1}) \\ &= I(Y_t, X_{t-1}, Y_{t-1}, \boldsymbol{Z}_{t-1}) - I(Y_t, Y_{t-1}, \boldsymbol{Z}_{t-1}) - I(X_{t-1}, Y_{t-1}, \boldsymbol{Z}_{t-1}) + I(Y_{t-1}, \boldsymbol{Z}_{t-1}), \end{aligned} \tag{3}$$

where the subscript t-1 denotes the embedded history length of the variables.

3.2 Conditional Transfer Entropy Estimation

As shown in Eq. (3), the calculation of conditional transfer entropy (CTE) can be expressed as the sum of four terms, each representing multivariate mutual information. Subsequently, we will elucidate the relationship between mutual information and copula entropy [8]. Following this, we will demonstrate how the computation of CTE can be reformulated in terms of a sum of copula entropies.

Definition 1 (Copula entropy [8]): let $\boldsymbol{X} \in \mathbf{R}^N$ be a set of random variables, its marginal function $\boldsymbol{u} = [F_1, \ldots, F_{\mathrm{N}}]$ and copula density function is $c(\boldsymbol{u})$, then the copula entropy expression of $\boldsymbol{X}$ is shown in Eq. (4).

$$H_c(\boldsymbol{X}) = -\int c(\boldsymbol{u}) log c(\boldsymbol{u}) d\boldsymbol{u}, \tag{4}$$

where $c(\boldsymbol{u})$ represents the Nth order derivative of copula distribution function $C(\boldsymbol{u})$.

Theorem 1 (Sklar's theorem): If T is a joint distribution, F and G are its marginal distribution, then there is a copula function C for all x and y belong to $\mathbf{R}$, satisfying $T(x, y) = C(F(\mathrm{x}), G(y))$. If F and G are continuous, then C is unique.

Theorem 2: The mutual information of random variables is the opposite number of their joint copula entropy.

Proof of Theorem 2: Take two variables as example, follow Theorem 1, then the joint cumulative distribution function of x and y can be expressed as Eq. (5).

$$P_{XY}(x, y) = C(P_X(x), P_Y(y)), \tag{5}$$

let $u = P_X(x)$, $v = P_Y(y)$, $c(u, v)$ can be expressed as the Eq. (6).

$$c(u, v) = \frac{\partial^2 C(u, v)}{\partial u \partial v} = \frac{\partial^2 P_{XY}(x, y)}{p_X(x)\partial x * p_Y(y)\partial y} = \frac{p_{XY}(x, y)}{p_X(x) * p_Y(y)}, \tag{6}$$

we can obtain the relationship between I and the copula entropy, as shown in Eq. (7).

$$I(X, Y) = \iint p_{XY}(x, y) log \frac{p_{XY}(x, y)}{p_X(x) * p_Y(y)} dxdy = \iint c(u, v) log c(u, v) dudv$$

$$= \iint c(u,v)p_X(x)p_Y(y)logc(u,v)dxdy = -H_c(u,v), \tag{7}$$

similarly, multivariate mutual information shares the same conclusion.

As described in Theorem 2, the conditional transfer entropy in Eq. (3) can finally be composed of four-term copula entropy, as shown in Eq. (8).

$$\begin{aligned} CTE_{X\to Y|\boldsymbol{Z}}(t) &= -H_c(V_t, U_{t-1}, V_{t-1}, \boldsymbol{W}_{t-1}) + H_c(V_t, V_{t-1}, \boldsymbol{W}_{t-1}) \\ &+ H_c(U_{t-1}, V_{t-1}, \boldsymbol{W}_{t-1}) - H_c(V_{t-1}, \boldsymbol{W}_{t-1}), \end{aligned} \tag{8}$$

where $U, V, \boldsymbol{W}$ separately denotes the marginal distribution of $X, Y, \boldsymbol{Z}$. Consequently, the calculation of CTE can ultimately be reduced to the problem of estimating information entropy.

KNN estimation [17] is usually used to calculate the entropy value of continuous random variables, whose calculation formula is shown in Eq. (9).

$$\widehat{H}\left(X^1, X^2, \ldots, X^d\right) = -\psi(k) + \psi(N) + logc_d + \frac{d}{N}\sum_{i=1}^{N} log\varepsilon(i), \tag{9}$$

where the digamma function $\psi(x) = \Gamma'(x)/\Gamma(x)$ denotes the digamma function, d and N are separately mean the dimensions and the number of observation variables, c_d is equal to 1 when adopt the maximum norm, $\varepsilon(i)$ denotes twice the amount of distance from the ith observation of $[X^1, X^2, \ldots, X^d]$ to its kth neighbor.

3.3 Definitions and Assumptions in Non-stationary Time Series Causal Discovery

To obtain causal relationships from observed time series data, several assumptions are necessary, including sufficiency, the Markov condition, and faithfulness. Additionally, building upon the CD-NOD [6], we introduce the concept of changing causal modules. These modules may be associated with changes that could be explained by certain confounders. Subsequently, these confounders could exert influence on other variables at specific time lags, leading to distribution shifts. The following presents definitions and assumptions essential to this paper.

Definition 2 (Time series data): Time-series data is a collection of data recorded in order. For the observed data that include d variables at t time, it can be written as $\boldsymbol{X}_t = \{X_t^1, X_t^2, \ldots, X_t^d\}$. And the $\boldsymbol{X}_{t-\tau}$ means the past of the data (τ denotes the lag time delay).

Definition 3 (Conditional independence): For the set of observed variables $\boldsymbol{X}_t = \{X_t^1, X_t^2, \ldots, X_t^d\}$, and condition set $\boldsymbol{Z} \subseteq X_{t-\tau} \setminus \{X_{t-\tau}^i\}$, if $CMI\left(X_{t-\tau}^i \to X_t^j | \boldsymbol{Z}\right) < \alpha$, then the $X_{t-\tau}^i$ conditional independent to X_t^j, denoted as $X_{t-\tau}^i \perp X_t^j | \boldsymbol{Z}$.

Definition 4 (Causal link): For the set of past observations $\boldsymbol{X}_{t-\tau}$, if $X_{t-\tau}^i \perp X_t^j | \boldsymbol{Z}$ not ture, then there is a causal link from $X_{t-\tau}^i$ to X_t^j, its representation is shown in Fig. 1(a):

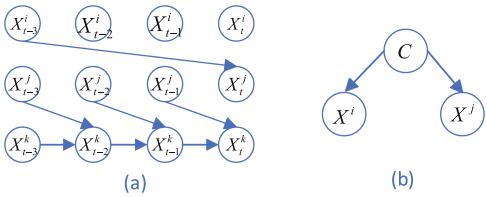


Fig. 1. (a) Causal link. (b) Pseudo confounder.

Assumption 1 (Pseudo causal sufficiency [6]): Assuming the existence of pseudo confounders, a time proxy variable C is introduced, as shown in Fig. 1(b). This variable can be written as a smooth function of time. At each time instance, the values of these confounders are fixed. Therefore, the pseudo causal sufficiency assumption is introduced, considering the causal adequacy of the model in the presence of potential confounding factors.

Assumption 2 (Lagged causal effects [18]): We assume that causality operates in a lagged manner, implying that there is no instantaneous causality at time t. This assumption also ensures that the lagging parent variable is sufficient to satisfy the Markov condition.

4 Algorithm Introduction

In this section, we first introduce and analyze the framework of CE-CDN, which is presented in two parts. The first part concerns the two-stage algorithm for learning the causal network structure. The second part focuses on the independence test method, specifically the conditional transfer entropy calculation process required in the first part. Finally, we analyze the time complexity of these two components.

4.1 Causal Structure Learning of the Non-stationary Time Series

The process of learning the causal network from non-stationary time series data can be divided into two stages.

The first stage involves causal connection discovery, where we calculate the conditional transfer entropy between variables and between variables and time exponents. This involves assessing the conditional mutual information at different time delays, denoted by τ. If the calculated value exceeds a specified threshold, it is added to the corresponding parent node set. As a result, this stage allows us to identify the lagged parent nodes for each variable at time t.

For temporal datasets containing d-dimensional variables and a time index variable, we consider the maximum time delay condition τ_{max}. The condition set comprises the delays τ for each variable. Steps 6 and 7 involve calculating the conditional mutual information between each node and its lagged nodes. If this value exceeds a threshold, the lagged node is added to the parent node set. Consequently, by step 8, we have obtained the lagged parent node set for each variable at time t. Additionally, steps 9

and 10 account for non-stationary factors, specifically detecting causal links between the time index variable C and other variables. This allows us to identify modules that change and affect the nodes at time t in a lagged manner.

After the first stage, we obtain the parent node set $\boldsymbol{Pa}\left(X_t^j\right)$ for variables at time t. However, there may still be incorrect parent nodes present. To ensure that the results converge to the true causal diagram under an infinite sample size, redundancy removal is necessary. Consequently, the second stage involves the false positive removal phase. For the parent set of each variable, the conditional independence test method is the same as in the first stage. The difference lies in the constraint applied to the condition set. In step 15, we test the independence between $X_{t-\tau}^i \subseteq \boldsymbol{Pa}(X_t^j)$ and X_t^j. Step 16 evaluates the calculated value. If the value is less than the threshold, step 17 removes the node from its parent set. Thus, we can eliminate incorrect parent nodes at this stage and obtain the final parent set for the nodes at the last time t. The framework encompassing the two stages is illustrated in Algorithm 1.

Alg.1 Causal discovery from non-stationary time series

Input: Dataset $\boldsymbol{X} = \{X^1, X^2, \ldots, X^d\}$; τ_{max}; α_1, α_2; Time index X^c.
Output: Parent nodes $\boldsymbol{Pa}(\boldsymbol{X}_t)$ of feature nodes at time t.
Initialization: Set of parents $\boldsymbol{Pa}\left(X_t^j\right) = \phi$

1. /* Stage 1: Constraint stage */
2. **for** j = 1 to d **do**
3. **for** $\tau = 1$ to τ_{max} **do**
4. $\boldsymbol{Cond} = \{X_{t-\tau}^1, X_{t-\tau}^2, \ldots, X_{t-\tau}^d, X_{t-\tau}^c\}$
5. **for** i = 1 to d **do**
6. calculate $CMI(X_{t-\tau}^i \to X_t^j | \boldsymbol{Cond} \backslash X_{t-\tau}^i)$
7. **if** $CMI\left(X_{t-\tau}^i \to X_t^j \middle| \boldsymbol{Cond} \backslash X_{t-\tau}^i\right) > \alpha_1$
8. append $X_{t-\tau}^i$ into $\boldsymbol{Pa}\left(X_t^j\right)$
9. calculate $CMI(X_{t-\tau}^c \to X_t^j | \boldsymbol{Cond} \backslash X_{t-\tau}^c)$
10. **if** $CMI\left(X_{t-\tau}^c \to X_t^j \middle| \boldsymbol{Cond} \backslash X_{t-\tau}^c\right) > \alpha_1$
11. append $X_{t-\tau}^c$ into $\boldsymbol{Pa}\left(X_t^j\right)$
12. /* Stage 2: Error parent nodes removing stage */
13. **for** j = 1 to d **do**
14. **for** $X_{t-\tau}^i$ in $\boldsymbol{Pa}\left(X_t^j\right)$ with $i = 1, \ldots d, c$; $\tau = 1, \ldots \tau_{max}$ **do**
15. calculate $CMI\left(X_{t-\tau}^i \to X_t^j \middle| \boldsymbol{Pa}\left(X_t^j\right) \backslash X_{t-\tau}^i\right)$
16. **if** $CMI\left(X_{t-\tau}^i \to X_t^j \middle| \boldsymbol{Pa}\left(X_t^j\right) \backslash X_{t-\tau}^i\right) < \alpha_2$
17. remove $X_{t-\tau}^i$ from $\boldsymbol{Pa}\left(X_t^j\right)$
18. **return** $\boldsymbol{Pa}(\boldsymbol{X}_t)$

4.2 Estimation of the Conditional Transfer Entropy

This subsection outlines the estimation process of conditional transfer entropy, which is a necessary component of the two-stage algorithm described above. The process consists of two main steps, both aimed at preparing for the calculation of copula entropy.

The first step involves obtaining the marginal probability distribution of each variable within the set X, which contains m-dimensional variables. This distribution can be estimated using rank statistics, specifically by sorting the time series data.

In the second step, leveraging the relationship between copula entropy and mutual information, the conditional transfer entropy—essentially a form of conditional mutual information—is converted into the sum of four copula entropies. For the estimation of copula entropy, we employ the k-nearest neighbor entropy estimator proposed by Kraskov et al. As shown in Eq. (10), we set k to 3. d corresponds to the number of variables in H_c at step 23, and N represents the number of samples. c_d is equal to 1 when adopt the maximum norm. For the first term, $\varepsilon(i)$ denotes twice the amount of distance from the ith observation of $[M_t^k, M_{t-1}^j, M_{t-1}^k, M_{t-1}^{z1}, \dots]$ to its kth neighbor.

$$\widehat{H}\left(X^1, X^2, \dots, X^d\right) = -\psi(k) + \psi(N) + logc_d + \frac{d}{N}\sum_{i=1}^{N} log\varepsilon(i) \tag{10}$$

Consequently, the final result can be derived by estimating the sum of these four copula entropies. The framework of the overall process is depicted in Algorithm 2.

Alg.2 Conditional transfer entropy estimation

Input: Temporal data set $\boldsymbol{X} = \{X^1, X^2, \dots, X^m\}$; Samples N.
Output: $CTE(X^j \to X^k | X^{z1}, X^{z2}, X^{z3} \dots)$, mark it as CTE
19. /* Step 1: estimate marginal distribution */
20. **for** i = 1 to m **do**
21. $M_t^i = \frac{1}{N}\sum_{t=1}^{N} I\{X_t^i \le x_t^i\}$ (I: Indicator function)
22. /* Step 2: CTE estimation */
23. $CTE = -H_c\left(M_t^k, M_{t-1}^j, M_{t-1}^k, M_{t-1}^{z1}, \dots\right) + H_c(M_t^k, M_{t-1}^k, M_{t-1}^{z1}, \dots) + H_c\left(M_{t-1}^j, M_{t-1}^k, M_{t-1}^{z1}, \dots\right) - H_c(M_{t-1}^k, M_{t-1}^{z1}, \dots)$
24. calculate each of four terms by entropy estimator
25. **return** CTE

4.3 Time Complexity Analysis

In this section, we focus on discussing the time complexity of the CE-CDN algorithm. We denote the dimension by m, the number of samples by n, and the maximum time delay by k. The first part of the algorithm, which estimates the conditional transfer entropy, has a time complexity that primarily consists of two stages: the rank stage and the k-nearest neighbor estimation stage.

During the rank stage, according to Eq. (8), the maximum dimensions for ranking would be $m + 1$. By applying a rank sorting algorithm to these dimensions, the time complexity is O ($mnlogn$). In the entropy estimation stage, the time complexity of calculating the distance between each pair of samples is O (n^2). To obtain the distance from the i-th observation to its k-th nearest neighbor, a sorting algorithm is applied to a two-dimensional array. The time complexity for this operation under n samples is O (n^2logn). Thus, the total time complexity for this part is O ($mnlogn + n^2logn$).

For the second part of the algorithm, its time complexity mainly includes the constraint stage and the error parent nodes removal stage. According to the framework of Algorithm 2, the total time complexity for this part is O (m^2k). Therefore, combining this with the first part, the overall time complexity of the CE-CDN algorithm is O ($m^3knlogn + m^2kn^2logn$).

5 Experimental Results and Analysis

5.1 Experiment 1

To verify the effectiveness of our algorithm, we compared it with NAVAR [19], PCMCI [10], HCE [7], DYNOTEARS [13], VarLiNGAM [11] and CCM [20]. The synthesized non-stationary time-series dataset was generated by Eq. (10).

$$X_t^i = a_i f_i\left(\boldsymbol{Pa}\left(X_t^i\right)\right) + b_i g(t-\tau) + \varepsilon_t^i, \tag{11}$$

where f_i represents the dependent function, which can have either a linear or nonlinear relationship with its lagged causes, denoted by $Pa(X^i)$. Under the maximum time delay τ_{max}, the lag of causes can range from 1 to τ_{max}. g is a smooth time-varying function with a time lag of τ. The coefficients a_i and b_i modify the function's response. The final term, ε_t^i, represents Gaussian noise.

In this simulation, we defined the time function $g(t) = cos(t/50)$, and randomly generated time series networks based on functional dependencies, including both linear and nonlinear causal relationships. The network size was varied to include 5, 10, 15, and 20 dimensions, with the maximum time delay set to $\tau_{max} = 5$. Furthermore, to test the algorithm's performance with different sample sizes, we applied it to datasets containing 1000, 2000, and 3000 samples. In each dataset, we identified nodes directly affected by the time index as non-stationary, leading to two scenarios: (1) For a given ratio of non-stationary nodes r_{non}, the number of non-stationary nodes across various dimensions d would be r_{non}* d; (2) With a fixed dimension, the proportion of non-stationary nodes was set to 0.1, 0.2, 0.3, 0.4, and 0.5.

Figure 2 and Fig. 3, which correspond to these two cases, illustrate the performance of different algorithms in reconstructing the causal network with sample sizes of 1000, 2000, and 3000. Additionally, due to the high time complexity of the independence test algorithm in PCMCI using cmiknn, we limited our comparison to the 1000-sample case.

5.2 Experiment 2

To further verify the effectiveness of the algorithm, we conducted Experiment 2. In this experiment, the time function was set to $g(t) = t$, and the data was regenerated using

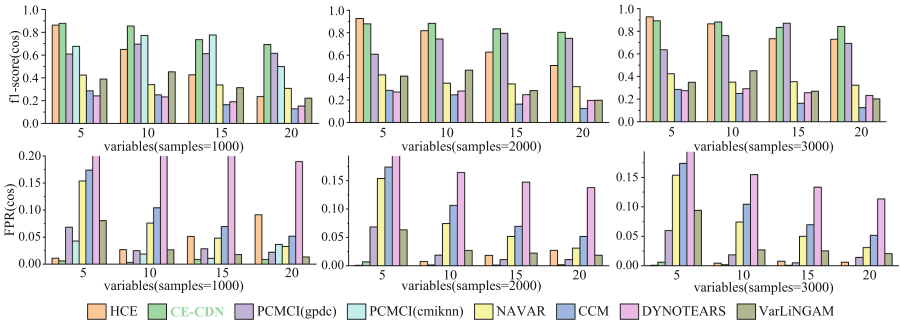


Fig. 2. The F1-score and FPR of the algorithms at the same ratio of non-stationary nodes, with the time function $g(t) = cos(t/50)$.

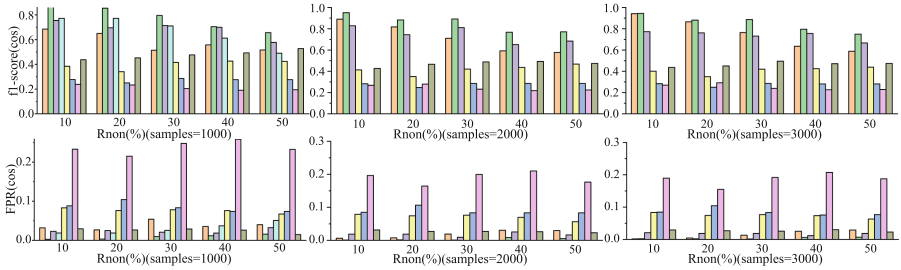


Fig. 3. The F1-score and FPR of the algorithms at different ratios of non-stationary nodes, with the time function $g(t) = cos(t/50)$.

the same experimental setup as described in Experiment 1.

$$X_t^i = a_i f_i\left(\boldsymbol{Pa}\left(X_t^i\right)\right) + b_i g(t-\tau) + \varepsilon_t^i \tag{12}$$

The experimental setup is the same as in Experiment 1. The performance of different algorithms is depicted in Fig. 4 and 5.

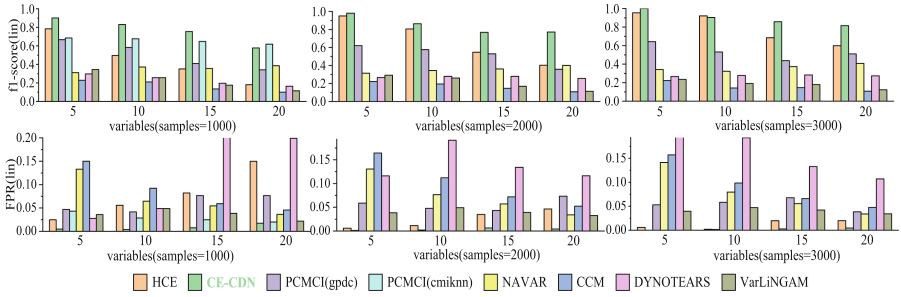


Fig. 4. The F1-score and FPR of algorithms at the same r_{non}. $(g(t) = t)$

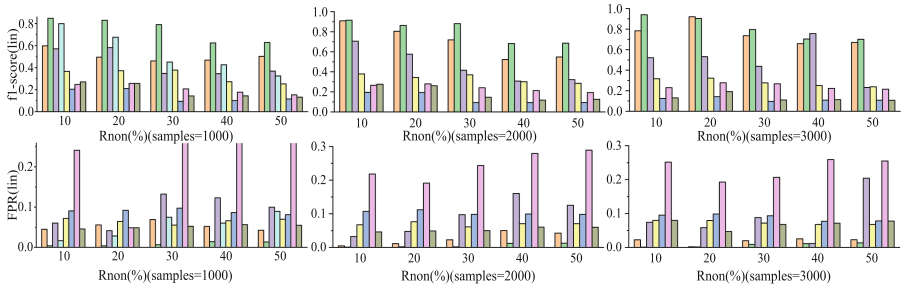


Fig. 5. The F1-score and FPR of algorithms at different r_{non}. $(g(t) = t)$

Furthermore, we conducted a non-stationarity test on the variables directly influenced by the time index. The time series and ADF test results are presented in Fig. 6. By comparing the ADF statistic with the critical values, it can be observed that if the ADF statistic is greater than the critical values and the p-value is greater than the level of significance, then we accept the null hypothesis that the time series data has a unit root, meaning the time series is non-stationary.

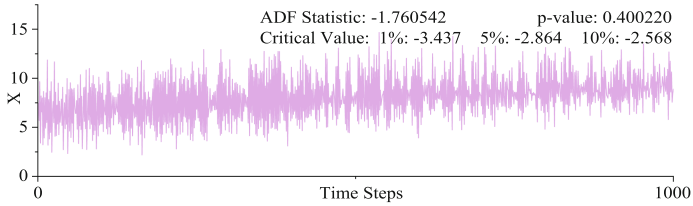


Fig. 6. Non-stationary time series.

5.3 Experiment 3

To compare the performance of algorithms at different maximum time delays, we also conducted Experiment 3. The performance metrics for models with 5, 10, 15, and 20 variables are reported in Table 1.

5.4 Experimental Analysis

The performance of each algorithm at the same non-stationary rate is shown in Fig. 2 and 4, which respectively represent different time functions.

In Fig. 2, it can be observed that our algorithm's False Positive Rate (FPR) is essentially lower than other algorithms, regardless of whether the sample size is 1000, 2000, or 3000. In terms of the F1 score, PCMCI outperforms our algorithm at the 15-variable level for both 1000 and 3000 samples, and HCE performs better than our algorithm at the 5-variable level for 2000 and 3000 samples. Apart from these instances, our algorithm generally outperforms other algorithms. In Fig. 4, it can be observed that PCMCI takes

Table 1. Performance of different algorithms for 5, 10, 15, 20 variables with lag 2 and 4.

		Lag2(Fun1)			Lag4(Fun1)			Lag2(Fun2)			Lag4(Fun2)		
Variables	Algs	TPR	FPR	F1	TPR	FPR	F1	TPR	FPR	F1	TPR	FPR	F1
5	CE-CDN	**0.88**	**0.03**	0.81	0.80	**0.01**	0.83	0.88	**0.01**	**0.93**	0.72	**0.01**	0.78
	HCE	0.82	0.04	0.81	**0.90**	0.02	**0.86**	0.88	0.03	0.87	0.87	0.03	**0.81**
	PCMCI(gpdc)	0.87	0.05	**0.84**	0.87	0.11	0.54	0.87	0.07	0.77	0.87	0.06	0.66
	CCM	0.50	0.09	0.50	0.50	0.13	0.33	0.50	0.09	0.50	0.50	0.13	0.33
	NAVAR	0.70	0.04	0.71	0.57	0.09	0.43	0.70	0.04	0.71	0.62	0.1	0.45
	Var-LiNGAM	0.52	0.19	0.41	0.50	0.06	0.44	0.55	0.15	0.46	0.49	0.06	0.46
	DYNOTEARS	**0.88**	0.21	0.48	0.88	0.83	0.31	**0.89**	0.34	0.51	**0.88**	0.30	0.36
10	CE-CDN	0.87	**0.02**	**0.85**	0.87	**0.01**	**0.88**	0.84	**0.003**	**0.90**	0.80	**0.002**	**0.87**
	HCE	0.88	**0.02**	0.84	0.86	0.02	0.76	0.87	0.04	0.77	0.85	0.05	0.58
	PCMCI(gpdc)	0.88	0.05	0.75	0.88	0.08	0.48	0.88	0.03	0.78	**0.88**	0.08	0.47
	CCM	0.50	0.04	0.50	0.50	0.25	0.33	0.50	0.04	0.50	0.50	0.07	0.33
	NAVAR	0.65	**0.02**	0.67	0.65	0.05	0.48	0.62	0.02	0.68	0.61	0.04	0.48
	Var-LiNGAM	0.52	0.09	0.42	0.48	0.06	0.36	0.50	0.08	0.43	0.52	0.04	0.41
	DYNOTEARS	**0.90**	0.32	0.38	**0.89**	0.29	0.24	**0.93**	0.30	0.38	**0.88**	0.24	0.25
15	CE-CDN	0.87	**0.01**	**0.84**	0.83	**0.02**	**0.83**	0.81	**0.007**	**0.84**	0.81	**0.007**	**0.80**
	HCE	0.88	0.03	0.76	0.80	0.05	0.49	0.87	0.03	0.74	0.82	0.07	0.41
	PCMCI(gpdc)	**0.93**	0.04	0.71	**0.92**	0.07	0.45	0.89	0.04	0.69	0.89	0.08	0.39
	CCM	0.50	0.04	0.47	0.50	0.05	0.32	0.50	0.03	0.50	0.50	0.04	0.33
	NAVAR	0.61	**0.01**	0.66	0.60	0.03	0.45	0.59	0.01	0.65	0.57	0.03	0.46
	Var-LiNGAM	0.51	0.06	0.43	0.51	**0.02**	0.45	0.50	0.05	0.43	0.48	0.03	0.43
	DYNOTEARS	0.91	0.27	0.33	0.90	0.25	0.20	**0.90**	0.28	0.29	**0.90**	0.25	0.19
20	CE-CDN	0.86	**0.01**	**0.88**	0.84	**0.01**	**0.80**	0.82	**0.01**	**0.81**	0.86	**0.01**	**0.75**
	HCE	0.87	0.05	0.64	0.77	0.09	0.32	0.83	0.04	0.63	0.80	0.11	0.27
	PCMCI(gpdc)	0.92	0.06	0.61	0.92	0.07	0.39	0.89	0.04	0.62	**0.92**	0.08	0.34
	CCM	0.50	0.04	0.43	0.50	0.04	0.29	0.50	0.03	0.49	0.50	0.04	0.33
	NAVAR	0.62	**0.01**	0.68	0.60	0.02	0.46	0.59	**0.01**	0.66	0.61	0.02	0.49
	Var-LiNGAM	0.51	0.04	0.43	0.50	0.02	0.45	0.51	0.03	0.45	0.50	**0.01**	0.53
	DYNOTEARS	**0.95**	0.23	0.30	**0.97**	0.20	0.20	**0.90**	0.24	0.27	0.88	0.88	0.17

a leading position at the 20-variable level for 1000 samples, and HCE is slightly higher than our algorithm at the 10-variable level for 3000 samples. The FPR shows the same performance as in Fig. 2.

The performance of each algorithm at different non-stationary rates is depicted in Fig. 3 and 5, which respectively represent different time functions.

In Fig. 3, it can be observed that our algorithm generally outperforms other algorithms in terms of the F1 score and FPR. In Fig. 5, with a sample size of 3000, the F1 scores of HCE and PCMCI are superior to our algorithm at the non-stationary rates of 0.2 and 0.4, respectively. In all other cases, our algorithm takes precedence.

In terms of performance at maximum lag delays of 2 and 4, as can be observed from Table 1, DYNOTEARS exhibits excellent performance in True Positive Rate (TPR). However, in terms of overall performance measured by the F1 score, our algorithm generally outperforms other algorithms.

Through the above analysis, it is evident that our algorithm generally performs better than other algorithms. For CCM, which is based on state-space reconstruction theory, there are several considerations. It is effective for nonlinear causal problems, particularly in the analysis of small-scale and short-term time series. CCM has high recognition sensitivity for weakly coupled systems but may produce errors under strong coupling conditions. As it requires analysis on an established state space, it is susceptible to the

influence of noise points. Besides, since the algorithm is often used for the analysis of bivariate systems, its performance in multivariate analysis systems may be poor.

NAVAR extends the vector autoregressive model to perform Granger causal analysis and introduces neural networks to learn the contribution of variables to a specific observed variable in the past. It uses the contribution value as a score to represent causal relationships, offering outstanding advantages in capturing nonlinear relationships but has limitations in dealing with higher-order interactions.

Table 2. Running time (second) of different algorithms at various samples.

		g(t)=t				g(t)=cos(t/50)			
Samples	Algs	N=5	N=10	N=15	N=20	N=5	N=10	N=15	N=20
1000	CE-CDN	**24**	**62**	**140**	**475**	**18**	**64**	**137**	**260**
	HCE	131	601	2046	5721	66	437	1622	4459
	PCMCI(cmiknn)	5439	15157	36400	655323	3419	11501	40114	149989
	PCMCI(gpdc)	1684	3882	10098	17560	1753	3451	8716	18237
	NAVAR	1165	1343	1563	1879	1092	1287	1535	1810
	CCM	51	162	349	576	40	152	370	601
2000	CE-CDN	83	234	509	971	66	249	517	957
	HCE	253	945	3378	8633	108	887	3018	7157
	PCMCI(gpdc)	12198	27957	58786	120686	13938	28605	52586	139523
	NAVAR	2285	2639	3238	3839	2177	2581	3016	4036
	CCM	**51**	**162**	**349**	**576**	**40**	**152**	**370**	**601**
3000	CE-CDN	198	599	1181	2203	139	534	1135	2176
	HCE	433	1308	4596	12177	193	1366	4402	9865
	PCMCI(gpdc)	29138	83087	-	-	25245	89483	-	-
	NAVAR	3440	4204	4899	5787	3218	4075	4800	5461
	CCM	**75**	**234**	**432**	**746**	**52**	**198**	**411**	**749**

PCMCI combines the PC algorithm for the selection of the conditional set and further tests in the MCI stage to control the false positive rate. However, the algorithm operates under the assumption of stationarity. Therefore, when considering non-stationary time exponents, there may be some impact on performance.

What's more, we also provide a comparison of the running time of non-parametric method algorithms, as shown in Table 2, representing the mean running time across sample sizes of 1000, 2000, and 3000. From the sample size of 1000, it is observed that CE-CDN's time performance is superior to other algorithms, and it demonstrates better performance in processing sample data compared to the non-parametric methods PCMCI (cmiknn/gpdc). In the samples of 2000 and 3000, CCM shows a lower running time. However, it exhibits poor performance in terms of F1 score and FPR.

In our approach, we propose using copula entropy to estimate conditional transfer entropy (CTE) and perform conditional independence tests through CTE. In the process of estimating conditional transfer entropy, first, we transform it into the sum of four multivariate mutual information terms. Then, based on the relationship between mutual information and copula entropy, we further transform it into the sum of four copula entropies. Finally, by estimating each copula entropy using the KNN method, we obtain an estimate of the conditional transfer entropy. As estimating copula entropy necessitates obtaining the marginal probability distributions of variables, time series values are

converted to ordinal ranks, which greatly enhances the algorithm's tolerance for non-stationarity and non-linear outliers. Additionally, compared to the direct estimation of mutual information for each multivariate set, this approach reduces both computational effort and time complexity. Hence, CE-CDN shows excellent performance in processing sample data within a reasonable time frame.

In summary, the experimental results demonstrate that the CE-CDN algorithm can effectively address the issue of constructing causal networks from non-linear and non-stationary time series.

6 Real Data Experiment

Climate-related time-series data were collected from NOAA Physical Sciences Laboratory. In the 1948 to 2023 seasonal reanalysis dataset of the NCEP / NCAR model [10], the sea level pressure in the Western Pacific (WPAC) was selected, defined by the regional coordinates of approximately 5° southern latitude to 5° northern latitude and 130° to 150° east longitude. Additionally, we collected air temperature data from the Central Pacific (CPAC), Eastern Pacific (EPAC), and Tropical Atlantic (ATL), with specific coordinates as follows:

- Central Pacific (CPAC): 5° southern latitude to 7.5° northern latitude, 150° to 120° west longitude
- Eastern Pacific (EPAC): 5° southern latitude to 7.5° northern latitude, 103° to 80° west longitude
- Tropical Atlantic (ATL): 5° northern latitude to 20° northern latitude, 55° to 30° west longitude.
- The time series of the above is shown in Fig. 7. A point on the timeline represents the running average of a specific year's data.

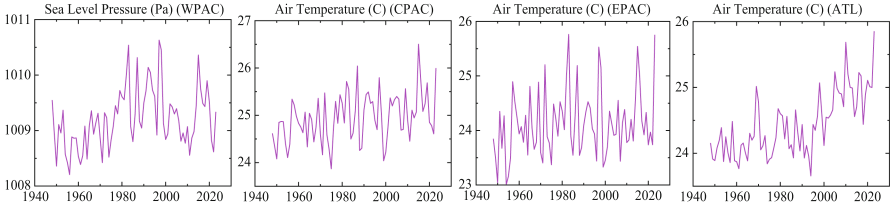


Fig. 7. Timeseries of climate data and Discovered causal relationships.

Additionally, the discovered causal links, with a maximum time delay set to 2, are illustrated in Fig. 8. Here we compare our result with PCMCI and DYNOTEARS, since other methods have discovered fewer causal links.

The blue-shaded area corresponds to the time index. As shown in the causal diagram, PCMCI and DYNOTEARS have not identified the relationship between the time index and the variables. In our discovered results, EPAC and ATL laggly affects the WPAC [21], and WPAC laggly affects the ATL [22]. Additionally, CPAC is influenced by EPAC

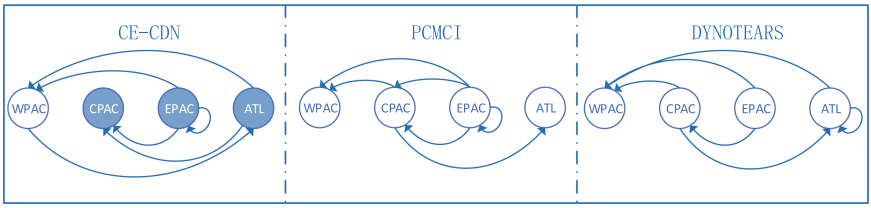


Fig. 8. Causal links returned by CE-CDN, PCMCI and DYNOTEARS (Color figure online)

[23] and ATL [24]. When the temperature of the Tropical Atlantic rises, it leads to an increase in sea-level air pressure. Concurrently, the temperature in the Western Pacific is relatively low, forming a downdraft. The low pressure formed in the Western Pacific causes high-pressure air from the Tropical Atlantic to flow over the Western Pacific. Therefore, changes in the temperature of the Tropical Atlantic may lead to changes in sea level pressure in the Western Pacific.

Furthermore, when the Eastern Pacific experiences reduced temperatures due to the influence of the Peruvian cold current, the high air pressure sinks to form low pressure, and high pressure forms at sea level. At the same time, the temperature in the Western Pacific Ocean is rising due to adverse currents, causing the air to expand and rise, forming a lower pressure area with high pressure at high altitudes.

Due to the influence of the air pressure gradient force, the high-pressure airflow in the Western Pacific flows towards the low-pressure area in the Eastern Pacific. Simultaneously, the high sea-level airflow in the Eastern Pacific flows towards the low-pressure area at sea level in the Western Pacific, forming the Walker circulation. Thus, temperature changes in the Eastern Pacific will impact the sea level pressure in the Western Pacific, as mentioned in the analysis of sea level changes by Zuo et al. [25].

7 Conclusion

In this paper, we propose an algorithm for learning causal networks from non-stationary time series. It mainly includes two parts, firstly, estimating the conditional transfer entropy using copula entropy, which provides an efficient calculation method. After that, we introduce a time index with a lagged effect on other variables to represent changing causal modules. Under the assumption of pseudo causal sufficiency, we learn the causal relationships using a constrained approach with non-parametric conditional independence tests. The experimental results show that the CE-CDN algorithm can effectively address the issue of reconstructing causal networks from non-stationary data. Additionally, it also performs better than the majority of baselines in terms of time performance. In the future, we will further study the causal relationships within non-stationary time series, especially the challenges posed by instantaneous causal links and the issue of changing causal directions. Following work will conduct intensive research.

References

1. Papana, A., Kyrtsou, C., Kugiumtzis, D., Diks, C.: Detecting causality in non-stationary time series using partial symbolic transfer entropy: evidence in financial data. Comput. Econ. **47**, 341–365 (2016)
2. Mudelsee, M.: Trend analysis of climate time series: a review of methods. Earth Sci. Rev. **190**, 310–322 (2019)
3. Cai, R., Huang, L., Chen, W., Qiao, J., Hao, Z.: Learning dynamic causal mechanisms from non-stationary data. Appl. Intell. **53**, 5437–5448 (2023)
4. Huang, B., Zhang, K., Schölkopf, B.: Identification of Time-Dependent Causal Model: a gaussian process treatment. In: Proceedings of the 24th International Conference on Artificial Intelligence, pp. 3561–3568. AAAI (2015)
5. Huang, B., Zhang, K., Gong, M., Glymour, C.: Causal discovery and forecasting in non-stationary environments with state-space models. In: International Conference on Machine Learning, pp. 2901–2910. PMLR (2019)
6. Zhang, K., Huang, B., Zhang, J., Glymour, C., Schölkopf, B.: Causal discovery from nonstationary/heterogeneous data: skeleton estimation and orientation determination. In: International Joint Conference on Artificial Intelligence, pp. 1347–1353. IJCAI, US (2017)
7. Kang, M., Chen, D., Meng, N., Yan, G., Yu, W.: Identifying unique causal network from nonstationary time series. arXiv preprint arXiv:2211.10085 (2022)
8. Ma, J., Sun, Z.: Mutual information is copula entropy. Tsinghua Sci. Technol. **16**(1), 51–54 (2011)
9. Entner, D., Hoyer, P.O.: On causal discovery from time series data using FCI. In: Proceedings of the 5th European Workshop on Probabilistic Graphical Models, pp. 121–128. PGM (2010)
10. Runge, J., Nowack, P., Kretschmer, M., Flaxman, S., Sejdinovic, D.: Detecting and quantifying causal associations in large nonlinear time series datasets. Sci. Adv. **5**(11), eaau4996 (2019)
11. Hyvärinen, A., Zhang, K., Shimizu, S., Hoyer, P.O.: Estimation of a structural vector autoregression model using non-gaussianity. J. Mach. Learn. Res. **11**(5), 1709–1731 (2010)
12. Peters, J., Janzing, D., Schölkopf, B.: Causal inference on time series using restricted structural equation models. In: Proceedings of the 26th International Conference on Neural Information Processing Systems, pp. 154–162. NIPS, Curran Associates Inc., Red Hook (2013)
13. Pamfil, R., et al.: DYNOTEARS: structure learning from time-series data. In: International Conference on Artificial Intelligence and Statistics, pp. 1595–1605. PMLR (2020)
14. Sun, X., Schulte, O., Liu, G., Poupart, P.: NTS-NOTEARS: learning nonparametric DBNs with prior knowledge. arXiv preprint arXiv:2109.04286 (2021)
15. Fujiwara, D., Koyama, K., Kiritoshi, K., Okawachi, T., Izumitani, T., Shimizu, S.: Causal discovery for non-stationary non-linear time series data using just-in-time modeling. In: Conference on Causal Learning and Reasoning, pp. 880–894. PMLR (2023)
16. Song, L., Kolar, M., Xing, E.: Time-varying dynamic Bayesian networks. In: Proceedings of the 22nd International Conference on Neural Information Processing Systems, pp. 1732–1740. Curran Associates Inc., Red Hook (2009)
17. Kraskov, A., Stögbauer, H., Grassberger, P.: Estimating mutual information. Phys. Rev. E **69**(6), 066138 (2004)
18. Runge, J.: Causal network reconstruction from time series: from theoretical assumptions to practical estimation. Chaos Interdisc. J. Nonl. Sci. **28**(7) (2018)
19. Bussmann, B., Nys, J., Latré, S.: Neural additive vector autoregression models for causal discovery in time series. In: Soares, C., Torgo, L. (eds.) DS 2021. LNCS, vol. 12986, pp. 446–460. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-88942-5_35
20. Sugihara, G., et al.: Detecting causality in complex ecosystems. Science **338**(6106), 496–500 (2012)

21. Huang, L., He, J., Lu, C.: Review and prospect of research on the western Pacific subtropical high. J. Arid Meteorol. **30**(2), 255–260 (2012)
22. Wang, B., Jiao, Z., Cao, J.: A study on the relationship between the sea surface temperature and the ridge of the subtropical high in the western Pacific. J. Yunnan Univ. (Nat. Sci. Edn.) **27**(4), 332–336 (2005)
23. Tan, R., Rong, G.: Equatorial Pacific ST warming and transmission characteristics of El Nino process. J. Appl. Meteorol. Sci. **3**(2), 165–172 (1992)
24. Zheng, J., Liu, Q.: Interactions between the tropical Pacific Ocean and the tropical Atlantic SST master modes. Oceanologia Et Limnologia Sinica **6**(2), 799–806 (2010)
25. Zuo, J., Yu, Y., Chen, Z.: The analysis of sealevel variation factor along China coast. Adv. Earth Sci. **9**(5), 48–53 (1994)

DSparsE: Dynamic Sparse Embedding for Knowledge Graph Completion

Chuhong Yang, Bin Li(✉), and Nan Wu

Beijing Institute of Technology, Beijing, China
{3120230733,binli,wunan}@bit.edu.cn

Abstract. Addressing the incompleteness problem in knowledge graph remains a significant challenge. Current knowledge graph completion methods have their limitations. For example, ComDensE is prone to overfitting and suffers from the degradation with the increase of network depth while InteractE has the limitations in feature interaction and interpretability. To this end, we propose a new method called dynamic sparse embedding (DSparsE) for knowledge graph completion. The proposed model embeds the input entity-relation pairs by a shallow encoder composed of a dynamic layer and a relation-aware layer. Subsequently, the concatenated output of the dynamic layer and relation-aware layer is passed through a projection layer and a deep decoder with residual connection structure. This model ensures the network robustness and maintains the capability of feature extraction. Furthermore, the conventional dense layers are replaced by randomly initialized sparse connection layers in the proposed method, which can mitigate the model overfitting. Finally, comprehensive experiments are conducted on the datasets of FB15k-237, WN18RR and YAGO3-10. It was demonstrated that the proposed method achieves the state-of-the-art performance in terms of Hits@1 compared to the existing baseline approaches. An ablation study is performed to examine the effects of the dynamic layer and relation-aware layer, where the combined model achieves the best performance.

Keywords: Knowledge Graph · Graph Completion · Link Prediction · Sparse Embedding

1 Introduction

Knowledge Graph (KG) is a directed heterogeneous graph that represents concepts, entities, and their relationships in a structured form using knowledge triples. Knowledge triples are typically represented as (s, r, o), where s, r and o denote the subject entity, the relation, and the object entity, respectively. KGs have a wide range of applications in various fields, including natural language processing, information retrieval, recommendation systems, and semantic web

This work was supported in part by the National Natural Science Foundation of China under Grant 62371045. The corresponding author is Bin Li.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 132–146, 2025.
https://doi.org/10.1007/978-3-031-78183-4_9

technologies. They are used to represent and organize knowledge in a structured and machine-readable format, which can be used to power intelligent applications and services.

Some well-known KGs, including Wikidata [26] and DBpedia [1], contain billions of knowledge triples, but they are often incomplete, which poses a significant challenge in the field of knowledge graph research. To address this issue, knowledge graph completion has emerged as an important task, which aims to predict missing knowledge triples. Typically, link prediction that focuses on predicting the missing entity in a knowledge triple is adopted for knowledge graph completion. Graph embedding, which uses low-dimensional, dense, and continuous vectors to represent nodes and relationships in knowledge graphs, is the basis of most link prediction methods. Existing link prediction models can be categorized into tensor decomposition models [13], translational models, and deep learning models [18]. Recently, pre-trained language models, such as Large Language Model, have also been introduced to solve KG incompletion problem [7,31].

In this paper, we propose a deep learning model, called DSparsE, for KG completion, where a new model that includes a relation-aware layer and a dynamic layer to extract features and residual connections used for decoding is proposed for link prediction. The previous multilayer perceptron (MLP) model ComDensE [12] retains both shared fully connected layers and relation-aware fully connected layers, and concatenates their results by a projection layer to achieve feature fusion. The relation-aware layer can be seen as a MLP with dynamic weights that changes with the input data. However, this dynamic processing is not comprehensive because the weights of the shared layers are still fixed, which limits the network's expressive power. MoE [9,20] and CondConv [29] were proposed in 2017 and 2019, respectively. The former divides the fully connected layer into several expert layers and uses a separate network to generate the combination weights of these expert layers. It takes the expert blocks with the top k weights for feature fusion. The latter uses dynamic convolution kernels based on input data for convolution operations. These dynamic methods give the network greater flexibility and have been shown to have good application potential. Thus DSparsE introduces a dynamic structure similar to MoE into the encoding end, and takes the results of all expert blocks for weighted fusion.

Compared to fully connected networks, convolutional layers introduce position-related sparse connections, which suppress overfitting effectively, save computing resources, and capture feature correlations between adjacent pixels efficiently. However, in link prediction for knowledge graph completion, the input of neural network is a one-dimensional embedding vector, which does not naturally have correlation information like pixels in images. Most of the aforementioned convolution-based models [5,11,16,24] attempt to enhance the interaction between entity and relation embedding vectors in different dimensions. These methods achieved good results on many datasets, but they still suffer from insufficient feature interaction and interpretability. Therefore, this paper uses sparse

layers with adjustable sparsity to replace all dense layers. Sparse layers can be seen as an upgrade to convolutional layers, while at the same time alleviating the overfitting issues faced by dense layers through unstructured pruning [6].

In addition, the research of ComDensE [12] shows that the effect of a single wide network layer is even better than a deep network, which has the degradation problem. This paper introduced residual connections [8] to solve the degradation problem when deepening network models. In summary, the contributions of this paper are listed as follows:

- We propose a novel link prediction model for knowledge graph completion, which introduces a *shallow but wide* dynamic layer and a relation-aware layer to the encoding end and a *deep but thin* residual structure to the decoding end. This enables neural networks to perform better information fusion and has the potential to deepen the network layers.
- By replacing all the fully connected layer with sparse layers, our model not only mitigates overfitting risks effectively, but also preserves its capability of feature interactions. Moreover, at comparable interaction levels, fixed sparse structures demonstrate enhanced predictability compared to other methods like dropout or downscale of output dimensions.
- A serial of tests and ablation studies were conducted on FB15k-237, WN18RR, and YAGO3-10 demonstrate that our proposed model achieves state-of-the-art performance in terms of Hits@1. Furthermore, by applying t-SNE dimensionality reduction to the output of the gating layer within the dynamic layer, it was discovered that the gating structure distributes weights to expert blocks based on the semantic information of entity-relation pairs.

2 Background and Related Works

A knowledge graph is a collection of triples (facts) that represent relationships between entities, denoted as $\mathcal{G} = \{(s, r, o)\} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, where $s \in \mathcal{E}$ and $o \in \mathcal{E}$ are the triple subject and object, respectively, and $r \in \mathcal{R}$ is the relationship between them. Link prediction for KG completion can be viewed as a point-wise learning to rank problem, where the goal is to learn a scoring function that maps an input triple (s, r, o) to a score $\psi(\cdot)$: $\mathcal{E} \times \mathcal{R} \times \mathcal{E} \mapsto \mathbf{R}$.

Related works on link prediction for knowledge graph completion are summarised as follows:

- **Tensor decomposition models**
  Tensor decomposition models treat the link prediction as a task of tensor decomposition. It encodes the knowledge graph as a three-dimensional tensor, which is incomplete due to the incompleteness of knowledge graph. This tensor is decomposed into a combination of low-dimensional vectors, from which the embeddings of entities and relations can be obtained. The model learns the relationships between vectors by setting a loss function and predicts the existence and correlation of underlying facts in knowledge graph. Typical tensor decomposition models include DistMult [28], ComplEx [23],

TuckER [2], etc. Although these models are mostly lightweight and easy to train, they are sensitive to sparse data and have limited modeling capabilities.

– **Translational models**
Translational models are based on the assumption that the relationship between entities can be represented by the translation of an entity vector. A typical translational model is TransE [4]. This model learns the embeddings of entities and relations by minimizing the energy function, and predicts the existence of underlying facts in knowledge graph. Translational models are simple and easy to train, but they are not suitable for modeling symmetric relations and complex relations. To address these issues, TransH [27], TransR [14], and TransD [10] models are proposed to enhance the modeling capability by dynamically mapping entities and relations and suppress the homogenization tendency of embedding vectors. Moreover, some improved methods based on TransE introduce additional computation overhead.
– **Deep learning models**
Deep learning models for link prediction can be categorized as models based on simple MLPs, models based on convolutional neural networks (CNNs), models based on graph neural networks (GNNs), and models based on recurrent neural networks (RNNs) or transformers. In link prediction, these networks usually take the entity and relation embeddings as input, and obtain a vector after encoding and decoding the input data through several neural network linear and nonlinear layers. Networks based on simple MLPs, such as ComDensE [12], add a relation-aware component, which generates different weight matrices for different relations that appear in the training set based on the common layer. Models based on CNNs include ConvE [5], ConvKB [16], ConvR [11], and InteractE [24]. These methods convert embedding vectors into two-dimensional feature maps in different ways and apply filters for convolution. More specifically, ConvR uses the relation embedding vector as the convolution kernel, while InteractE enhances the interaction between features by reshaping them into a checkerboard pattern. Models based on GNNs include R-GCN [19] and CompGCN [25]. These methods use graph convolutional networks to grab the neighborhood information of entities and relations and aggregate them into the entity embedding vector. Those methods naturally take advantage of the graph structure and achieve good results on some datasets. However, the parallelization challenge caused by the heterogeneous graph structure limits the performance of these methods. Some methods use fine-tuning pre-trained language models for link prediction, such as KG-BERT [30] and Rhelphormer [3]. Although these models achieve good performance, they suffer from high complexity and require external information beyond knowledge graph.

3 DSparsE for Link Prediction

This paper proposes a novel neural network model called DSparsE for link prediction, whose architecture is shown in Fig. 1. The proposed DSparsE model consists of two parts: encoder and decoder. The encoding part includes a dynamic

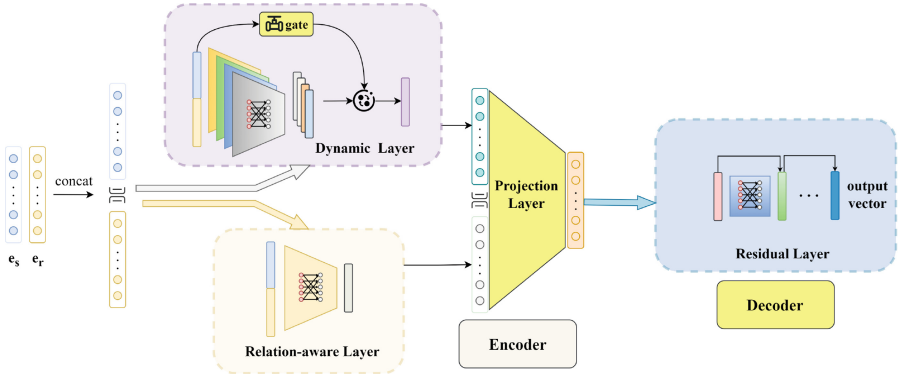


Fig. 1. The architecture of DSparsE. The encoding end is composed of a dynamic layer and a relation-aware layer. The decoding end is composed of several residual MLP layers. **Note that all the dense layers are replaced by sparse layers with certain sparsity degrees.**

layer, a relation-aware layer and a projection layer. The decoding part is a residual layer. Note that the dense MLPs are replaced by sparse MLP layers in DSparsE, where a sparsity degree is used to measure the sparsity of weight matrix. The sparsity degree is a hyperparameter that can be adjusted according to dataset.

In DSparsE, the robustness of the network is enhanced through dynamic module augmentation, which leverages randomly initialized sparse unstructured pruning via a weight matrix combined with joint learning from stacked expert blocks. This structure maximizes the network's expressive power and allows the network to deepen to improve its performance. Since deepening the network may lead to a decrease in accuracy, the residual connections in DSparsE can alleviate this effect.

In Fig. 1, DSparsE takes the d-dimensional head node embedding $\boldsymbol{e}_s$ and the d-dimensional relation embedding $\boldsymbol{e}_r$ as inputs. These two embeddings are concatenated to form a $2d$-dimensional vector, which is further passed through a dynamic layer and a relation-aware layer in parallel. The output features of these two layers are concatenated and further passed through a projection layer. A detailed introduction of the aforementioned layers in DSparsE is as follows.

3.1 Dynamic Layer

The dynamic layer consists of multiple sparse MLP layers and a gate layer. This network structure enhances the robustness of the model and improves the prediction performance. The dynamic layer takes the concatenation of input vectors $[\boldsymbol{e}_s; \boldsymbol{e}_r]$ and produces k different output vectors $\boldsymbol{e}_{out_1}, \boldsymbol{e}_{out_2}, ..., \boldsymbol{e}_{out_k}$ through k parallel MLP layers. The output vector of the dynamic layer is obtained by taking a weighted combination of these output vectors $\boldsymbol{e}_{out_1}, \boldsymbol{e}_{out_2}, ..., \boldsymbol{e}_{out_k}$. The

combination weights are determined by a gate layer, which includes a dense fully connected layer and a softmax layer controlled by a temperature parameter t. The output of the gate layer can be denoted by

$$\boldsymbol{g} = \text{softmax}(\Omega_{gate}([\boldsymbol{e}_s; \boldsymbol{e}_r]/t)). \tag{1}$$

where $\Omega_{gate}(\cdot)$ is a affine function. Thus the output of dynamic layer can be written as

$$\boldsymbol{e}_{out,D} = \sum\nolimits_{i=1}^{k} g_i \times \boldsymbol{e}_{out_i}, \tag{2}$$

where $\boldsymbol{g} \triangleq [g_1, g_2, \cdots, g_k]$.

3.2 Relation-Aware Layer

To achieve more accurate feature extraction, we introduce a sparse relation-aware layer that changes dynamically with the input relation [10,12]. This can be viewed as part of the network dynamic nature (See [12] for more details). The output of the sparse relation-aware layer is given by

$$\boldsymbol{e}_{out,R} = f(\Omega_R^{\alpha}([\boldsymbol{e}_s; \boldsymbol{e}_r])). \tag{3}$$

where $\Omega_R^{\alpha}(\cdot)$ is a sparse affine function with sparsity degree $\alpha \in (0, 1)$, and f denotes the activation function.

3.3 Projection Layer

The projection layer in DSparsE is a sparse MLP layer. With the input being the vector concatenation of $\boldsymbol{e}_{out,D}$ and $\boldsymbol{e}_{out,R}$, the output of the projection layer is a d-dimensional vector and can be given by

$$\boldsymbol{e}_{out,P} = f(\Omega_P^{\alpha}([\boldsymbol{e}_{out,D}; \boldsymbol{e}_{out,R}])), \tag{4}$$

where $\Omega_P^{\alpha}(\cdot)$ is a sparse affine function with sparsity degree $\alpha \in (0, 1)$.

3.4 Residual Layer

A residual block consists of a sparse MLP layer, a batchnorm layer, an activation layer (such as ReLU), a dropout layer and a residual connection. The input and output of the residual block has the same dimension. The decoder of DSparsE is a stack of multiple residual blocks, where the output of the i-th residual block is formulated as

$$\boldsymbol{e}_{Res_i} = f(\text{BN}(\Omega_{Res_i}^{\alpha}(\boldsymbol{e}_{Res_{i-1}})) + \boldsymbol{e}_{Res_{i-1}}), \tag{5}$$

where $\text{BN}(\cdot)$ denotes the batch normalization operation and $\boldsymbol{e}_{Res_i}$ the output vector of the i-th residual block, and $\Omega_{Res_i}^{\alpha}(\cdot)$ is a sparse affine function with sparsity degree α. Note that $\boldsymbol{e}_{Res_0}$ is set to the output of the projection layer $\boldsymbol{e}_{out,P}$. As introduced in [8,15], residual connections ease the training of deep networks and prevent the degradation of deep networks.

3.5 Sparse Structure of MLP

It is known that a dense layer may have a large number of useless parameters, which lead to poor model generalization and increase training difficulty [6]. Although a convolutional layer is a sparse and parameter-sharing linear layer and the number of parameters is much lower than that of a dense (fully connected) layer, it leads to insufficient information exchange and difficulty in extracting features. Moreover, convolving the feature embeddings of nodes and relations does not have a physical interpretation. To tackle this issue, we can introduce sparse MLP layer in DSparE. In the training stage, the elements of weight matrix are initialized randomly with zeros in certain probability, which leads to a sparse MLP layer. Given the parameters of a dense MLP layer $\boldsymbol{W}$ and a sparsity degree α, the parameters of a sparse MLP layer $\boldsymbol{W}^{\alpha}$ can be formulated as

$$W_{i,j}^{\alpha} = \begin{cases} 0 & \text{with probability } \alpha, \\ W_{i,j} & \text{with probability } 1-\alpha. \end{cases} \tag{6}$$

A sparse MLP layer can be viewed from two directions. On one hand, it can be viewed as the result of a convolutional layer with enhancing interaction and removing weight sharing. On the other hand, it can be viewed as the result of pruning a dense layer.

Finally, we calculate the scores and loss function. The score is obtained by taking the dot product of the output of the residual layer $\boldsymbol{e}_{decode}$ and the object entity embedding vector $\boldsymbol{e}_o$ and further applying the sigmoid function, which is formulated as

$$\psi = \sigma(\boldsymbol{e}_{decode} \cdot \boldsymbol{e}_o), \tag{7}$$

where '$\cdot$' denotes the dot product and σ denotes the sigmoid function. For the loss function, we adopt the binary cross entropy loss function

$$\mathcal{L} = -\frac{1}{N} \sum\nolimits_i y_i \log \psi(s, r, o_i) + (1 - y_i) \log(1 - \psi(s, r, o_i)), \tag{8}$$

where N is the number of entities, y_i is the label of the i-th entity, the label $y_i = 1$ of the entity o_i if $(s, r, o_i) \in \mathcal{G}$, and $y_i = 0$ otherwise.

4 Experiments and Analysis

4.1 Datasets and Evaluation Settings

In our experiments, we use 1-N training strategy introduced by [5] to train DSparsE and evaluate the performance of DSparsE on three typical datasets: FB15k-237 [22], WN18RR [5] and YAGO3-10 [21]. Our evaluation of link prediction is conducted in the filtered setting, where we calculate scores for all other potential triples in the test set that are not present in the training, validation, or test set. To generate these potential triples, we corrupt the subjects for object prediction. We use mean reciprocal rank (MRR) and Hits at N (Hits@N) metrics to evaluate the performance of our model on these datasets. To ensure robust evaluation, we train and evaluate our models five times and average the performance results.

Table 1. A comparison of prediction performance on different datasets. The best result is in **bold**, and the second best result is underlined.

Model	FB15k-237			WN18RR			YAGO3-10		
	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR
TransE [4]	0.199	0.471	0.290	0.422	0.512	0.465	–	–	–
TransD [10]	0.148	0.461	0.253	–	0.508	–	–	–	–
DistMult [28]	0.155	0.419	0.241	0.390	0.490	0.430	0.240	0.540	0.340
CompGCN [25]	0.264	0.535	0.355	**0.443**	**0.546**	**0.494**	–	–	–
R-GCN [19]	0.151	–	0.249	–	–	–	–	–	–
ConvE [5]	0.237	0.501	0.325	0.400	0.520	0.430	0.350	0.620	0.440
ConvKB [16]	–	0.517	**0.396**	–	0.525	0.248	–	–	–
TuckER [2]	0.266	0.544	0.358	**0.443**	0.526	0.470	–	–	–
ComplEx [23]	0.158	0.428	0.247	0.410	0.510	0.440	0.260	0.550	0.360
RESCAL [17]	0.269	0.548	0.364	0.417	0.487	0.441	–	–	–
RotatE [17]	0.241	0.533	0.338	0.417	0.552	0.462	0.402	0.670	0.495
KG-BERT [30]	–	0.420	–	–	0.524	–	–	–	–
ComDensE [12]	0.265	0.536	0.356	0.440	0.538	0.473	–	–	–
InteractE [24]	0.263	0.535	0.354	0.430	0.528	0.463	0.462	0.687	0.541
DSparsE (proposed)	**0.272**	**0.551**	0.361	**0.443**	0.539	0.474	**0.464**	**0.690**	**0.544**

4.2 Prediction Performance

Table 1 shows the performance of the proposed method compared to existing methods. It can be seen that DSparsE reached the state-of-the-art performance on FB15k-237, WN18RR and YAGO3-10 in terms of Hits@1. On FB15k-237, it achieves a improvement of 2.6% and 3.4% in Hits@1 compared to ComDensE and InteractE, respectively. On WN18RR, the improvement is not significant compared to CompGCN and TuckER, but it still outperformed those models based on translation and deep learning. On YAGO3-10, DSparsE achieves the state-of-the-art performance on all the metrics, which highlights the effectiveness of the proposed model. Furthermore, DSparsE performs better than those models based on feature convolution. For instance, on FB15k-237, it achieves a improvement of 14.8% in Hits@1 compared to ConvE and 6.6% in Hits@10 compared to ConvKB. KG-BERT, a link prediction model based on BERT pre-trained language model, performs average on small knowledge graph like FB15k-237 and WN18RR, and its accuracy is much lower than DSparsE. It is observed that DSparsE outperforms KG-BERT with a 24% and 3% improvement in Hits@10 on FB15k-237 and WN18RR, respectively.

4.3 Ablation Studies and Further Experiments

The Effect of Sparsity Degree. To further explore the effect of sparsity degree, we applied sparse structure to replace the dense layers in DSparsE, ComDensE, and InteractE. The performance comparison is shown in Fig. 2. It can be observed that the accuracies of both ComDensE and DSparsE models first

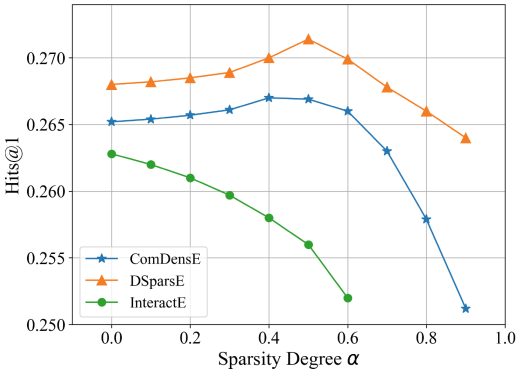


Fig. 2. Hits@1 of InteractE, ComDensE, and DSparsE under different sparsity degrees on FB15k-237.

increase and then decrease with the increase of sparsity, and the highest accuracies of both models appear at a sparsity of around 0.5. This indicates that low sparsity in the network can lead to overfitting, limiting its potential, while increasing sparsity properly can mitigate this issue. However, excessively high sparsity reduces the number of effective parameters and disrupts neuron connections, diminishing the network's expressive power and impairing training due to decreased neuronal interaction. Note that DSparsE is less adversely affected by the increase of sparsity compared to ComDensE, owing to its marginally greater parameter count and more complex structure.

On the other hand, the performance of the InteractE model demonstrates a consistent decrease with the increase of sparsity. This trend is due to the model architecture of InteractE, where the final feature decoding layer is only an MLP layer. The experiment results indicate that introducing increased sparsity over the sparse interactions already captured by the earlier convolutional layers adversely affects the model's predictive performance.

Furthermore, the results demonstrate that enhancing a network's effectiveness can be achieved by introducing random sparsity. However, two questions arise:

1. **Can we achieve a similar performance by reducing the scale of the linear layer?**
2. **Can we achieve a similar performance by increasing the dropout probability?**

To address the first question, we do experiments by reducing the output dimension in the linear layers. Specifically, for a linear layer with output dimension d, we downscale the output dimension to $\hat{d} = \alpha d$. For the second question, we do experiments by increasing the dropout rate to $\hat{p} = p + \alpha(1 - p)$, where p is the original dropout rate. The results are shown in Fig. 3. It indicates that decreasing the number of neurons significantly degrades the performance, whereas increasing dropout rate drastically deteriorated the performance. This is

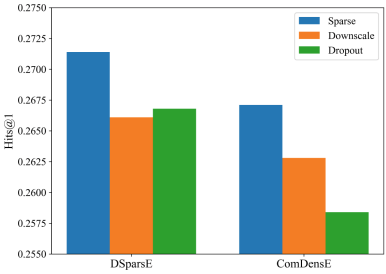


Fig. 3. The effects of downscale and dropout. *Sparse* represents the proposed sparse structure, *Downscale* means cutting off part of the output dimension of the network, and *Dropout* means adding extra dropout based on the original dropout layer. The experiment is conducted on FB15k-237.

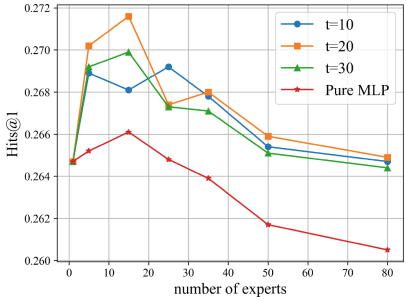


Fig. 4. Hits@1 of DSparsE under different numbers of experts and temperatures on FB15k-237. t denotes the temperature and *Pure MLP* denotes an MLP layer which has the same number of parameters as the dynamic layer.

due to the fact that reducing the neuron number confines the output to a smaller subspace, limiting expressive freedom. On the other hand, since each training iteration changes the dropout mask, an excessively high dropout introduces more uncertainty, thus diminishing network stability.

The Effect of Experts. Figure 4 illustrates the performance of DSparsE in Hits@1 scores under different expert and temperature settings on FB15k-237. The experiment results indicate that the prediction performance first increases and then decreases with the rising number of expert blocks. The increase of performance be explained from two aspects. On one hand, in contrast to a non-partitioned fully connected structure (i.e., a very wide fully connected layer), the expert blocks in the dynamic layer represent a form of regular sparse connections. These sparse connections are further integrated through a decision layer, namely a gating layer, forming a hypernetwork structure, which brings robustness to the entire network. On the other hand, the expert blocks in the dynamic layer can be viewed as sub-modules in an ensemble learning framework. This ensemble learning architecture can effectively suppress the propagation of errors, reducing the variance in prediction results.

However, when the number of expert blocks becomes large, the performance deteriorates. This is due to an increase in network parameters introduces additional training complexity, diminishing the network's generalization performance. Moreover, the gating network is fundamentally a multi-classifier. An excessive number of categories increases the decision-making complexity of the network.

Table 2. The ablation study on dynamic layer and relation-aware layer on FB15k-237. *D*, *R* and *Res* denote the dynamic layer, relation-aware layer, and residual layer, respectively.

	Hits@1	Hits@10	MRR
D + R + Res (Proposed)	**0.272**	**0.551**	**0.361**
D + Res	0.254 (−0.018)	0.526	0.345
R + Res	0.266 (−0.006)	0.538	0.355
Res($depth = 1$)	0.237 (−0.035)	0.499	0.325
Res($depth = 3$)	0.236 (−0.036)	0.509	0.325
Res($depth = 5$)	0.238 (−0.034)	0.511	0.325
Res($depth = 10$)	0.235 (−0.037)	0.504	0.324

Another key factor is the temperature of the dynamic layer. High temperature values lead to weight homogenization. Conversely, low temperature values can render many experts ineffective in learning, thus degrade the performance.

The Effects of Dynamic Layer and Relation-Aware Layer. Our ablation studies demonstrate that both the dynamic and relation-aware layers are essential for achieving the optimal performance, as shown in Table 2. The Dynamic layer compensates for the relation-aware layer's lack of interconnectedness, facilitating the integration of diverse relational knowledge. The expert layer's gating output is determined by head-relation pairs, fostering a more entity-aware weighting system and enabling the connection of different knowledge types. The interaction between these two layers yields enhanced performance, highlighting their synergistic effect. Furthermore, if both the dynamic layer and relation-aware layer are removed and only the decoder with the residual connection is left, it leads to a significant performance degradation. This decline in performance cannot be mitigated by increasing the number of layers in the decoder. The result indicates that both the encoder and the decoder are indispensable for link prediction in DSparsE.

Furthermore, we disclose some insights of the gate layer. Each entity-relation pair passed through the gating layer yields an output vector. Using the t-SNE dimensionality reduction technique, these high-dimensional vectors can be visualized in Fig. 5 and Fig. 6. Each point in these figures represents a unique entity-relation pair, distinguished by different colors corresponding to different relationships. The visualization results reveal the following observations:

- The entity-relation pairs with the same relationship type tends to cluster together, which indicates the proximity within the space of the gated layer outputs.
- The spatial distribution of clusters is significantly influenced by the nature of the relationships. For instance, relationships denoting inverse meanings (e.g., nominee_inv and nominee) or semantic opposites (e.g., place of birth vs. place

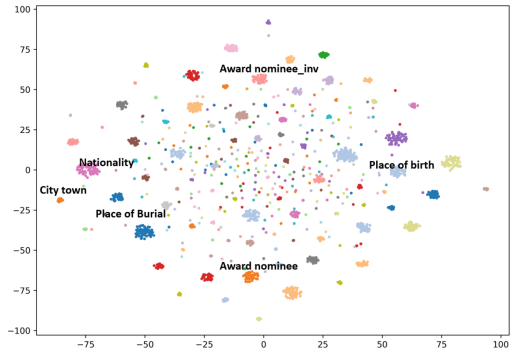


Fig. 5. The output of gated layer for each entity-relation pair. Each point represents an entity-relation pair in latent space after t-SNE reduction. The color of a point represents the relation type.

Table 3. Hits@1 of DSparsE under different residual layer depths on FB15k-237. Note that the *Wide linear layer* means a wide and shallow network with the same number of parameters replaces the deep structure with residual connections in decoder.

Depth of layers	With residual connection	Without residual connection	Wide linear layer
$depth = 1$	0.2682	0.2671 (−0.0011)	–
$depth = 2$	0.2691	0.2522 (−0.0169)	0.2633 (−0.0058)
$depth = 3$	**0.2716**	0.2276 (−0.0440)	0.2550 (−0.0166)
$depth = 4$	0.2681	0.1908 (−0.0773)	0.2511 (−0.017)
$depth = 100$	0.2639	0.02338 (−0.2401)	0.2490 (−0.0149)

of burial) exhibit divergence in the reduced dimensional space. Conversely, relationships with similar semantics (e.g., nationality and city town) are close in the latent space. This demonstrates that DSparsE can capture various associations between entities and relations.

– Alterations in the head entity of a relation pair result in minor shifts within the vector output. For a fixed relation, the relative positions of entity within its corresponding cluster does not display a discernible pattern. This is due to the relatively lower frequency of triples involving individual nodes compared to those associated with a particular relation type, posing challenges in accurately modeling semantic information [4]. However, certain examples, such as Mariah Carley and Dmitri Shostakovich (notable in the music domain) demonstrate proximity within clusters pertaining to specific relations.

The Effect of Residual Blocks. With the increasing scale of dataset, a deeper decoding layer is expected to ensure that the network's performance. However, simply increasing MLP layers leads to a rapid degradation in performance. Employing residual connections maintains the expressive potential of

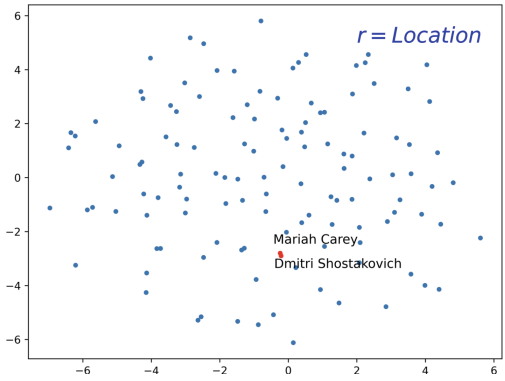


Fig. 6. The distribution of different entities in the same relation cluster (**e.g., a relation named *Location***). The points that close to each other are semantic similar in latent space.

the network effectively. The presence of residual connections in DSparsE ensures the expressive capability when deepening the network. Table 3 shows the performance of DSparsE in Hits@1 under different numbers of residual blocks on FB15k-237. It can be observed that the accuracy decreases rapidly as the number of layers increases if the residual blocks are replaced with fully connected layers. In our model, the residual connections reduce the effect of increasing the number of layers. Furthermore, if a wide and shallow network with the same number of parameters is used to replace the deep structure with residual connections, there is still a performance degradation. These experiments demonstrate that a *shallow but wide* encoder for feature extraction and a *deep but thin* decoder for feature decoding can effectively enhance the performance of link prediction.

5 Conclusion

This paper proposed a new model called DSparsE for knowledge graph completion. By introducing wide dynamic layer and relation-aware layer as an encoder and a deep residual connection layer as a decoder, the model representation power was enhanced effectively. This model employs sparse MLP layers and residual structures to alleviate overfitting, which reduces the difficulty of training deep networks and improves prediction performance. The experiment results demonstrate that the proposed DSparsE achieves the best performance in terms of Hits@1 on the FB15k-237, WN18RR, and YAGO3-10 datasets. Moreover, it was discovered that the hypernetwork structure formed by gated layers can effectively capture the semantic features and semantic associations of entity-relation pairs, with the results of latent space dimensionality reduction exhibiting interesting clustering and intra-cluster deviation phenomena. Ablation studies have further proven that the *shallow-to-deep* network structure of DSparsE improves the link prediction performance.

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: a nucleus for a web of open data. In: Aberer, K., et al. (eds.) ASWC/ISWC -2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-76298-0_52
2. Balažević, I., Allen, C., Hospedales, T.M.: Tucker: tensor factorization for knowledge graph completion. arXiv preprint arXiv:1901.09590 (2019)
3. Bi, Z., Cheng, S., Zhang, N., Liang, X., Xiong, F., Chen, H.: Relphormer: relational graph transformer for knowledge graph representation. arXiv preprint arXiv:2205.10852 (2022)
4. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. Adv. Neural Inf. Process. Syst. **26** (2013)
5. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2D knowledge graph embeddings. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
6. Dettmers, T., Zettlemoyer, L.: Sparse networks from scratch: faster training without losing performance. arXiv preprint arXiv:1907.04840 (2019)
7. He, J., Jia, L., Wang, L., Li, X., Xu, X.: MoCoSa: momentum contrast for knowledge graph completion with structure-augmented pre-trained language models. arXiv preprint arXiv:2308.08204 (2023)
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
9. Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E.: Adaptive mixtures of local experts. Neural Comput. **3**(1), 79–87 (1991)
10. Ji, G., He, S., Xu, L., Liu, K., Zhao, J.: Knowledge graph embedding via dynamic mapping matrix. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (volume 1: Long Papers), pp. 687–696 (2015)
11. Jiang, X., Wang, Q., Wang, B.: Adaptive convolution for multi-relational learning. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 978–987 (2019)
12. Kim, M., Baek, S.: ComDensE: combined dense embedding of relation-aware and common features for knowledge graph completion. In: 2022 26th International Conference on Pattern Recognition (ICPR), pp. 1989–1995. IEEE (2022)
13. Kolda, T.G., Bader, B.W.: Tensor decompositions and applications. SIAM Rev. **51**(3), 455–500 (2009)
14. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 29 (2015)
15. Ma, X., Qin, C., You, H., Ran, H., Fu, Y.: Rethinking network design and local geometry in point cloud: a simple residual MLP framework. arXiv preprint arXiv:2202.07123 (2022)
16. Nguyen, D.Q., Nguyen, T.D., Nguyen, D.Q., Phung, D.: A novel embedding model for knowledge base completion based on convolutional neural network. arXiv preprint arXiv:1712.02121 (2017)

17. Nickel, M., Tresp, V., Kriegel, H.P., et al.: A three-way model for collective learning on multi-relational data. In: ICML, vol. 11, pp. 3104482–3104584 (2011)
18. Rossi, A., Barbosa, D., Firmani, D., Matinata, A., Merialdo, P.: Knowledge graph embedding for link prediction: a comparative analysis. ACM Trans. Knowl. Discov. Data (TKDD) **15**(2), 1–49 (2021)
19. Schlichtkrull, M., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: Gangemi, A., et al. (eds.) ESWC 2018. LNCS, vol. 10843, pp. 593–607. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93417-4_38
20. Shazeer, N., et al.: Outrageously large neural networks: the sparsely-gated mixture-of-experts layer. arXiv preprint arXiv:1701.06538 (2017)
21. Suchanek, F.M., Kasneci, G., Weikum, G.: YAGO: a core of semantic knowledge. In: Proceedings of the 16th International Conference on World Wide Web, pp. 697–706 (2007)
22. Toutanova, K., Chen, D.: Observed versus latent features for knowledge base and text inference. In: Proceedings of the 3rd Workshop on Continuous Vector Space Models and Their Compositionality, pp. 57–66 (2015)
23. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: International Conference on Machine Learning, pp. 2071–2080. PMLR (2016)
24. Vashishth, S., Sanyal, S., Nitin, V., Agrawal, N., Talukdar, P.: InteractE: improving convolution-based knowledge graph embeddings by increasing feature interactions. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 3009–3016 (2020)
25. Vashishth, S., Sanyal, S., Nitin, V., Talukdar, P.: Composition-based multi-relational graph convolutional networks. arXiv preprint arXiv:1911.03082 (2019)
26. Vrandecic, D., Krtoetzsch, M.: WikiData: a free collaborative knowledgebase. Commun. ACM **57**(10), 78–85 (2014)
27. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 28 (2014)
28. Yang, B., Yih, W.t., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. arXiv preprint arXiv:1412.6575 (2014)
29. Yang, B., Bender, G., Le, Q.V., Ngiam, J.: CondConv: conditionally parameterized convolutions for efficient inference. Adv. Neural Inf. Process. Syst. **32** (2019)
30. Yao, L., Mao, C., Luo, Y.: KG-BERT: BERT for knowledge graph completion. arXiv preprint arXiv:1909.03193 (2019)
31. Zhu, Z., Zhang, Z., Xhonneux, L.P., Tang, J.: Neural bellman-ford networks: a general graph neural network framework for link prediction. Adv. Neural. Inf. Process. Syst. **34**, 29476–29490 (2021)

Interpreting Convolutional Neural Network Decision via Pixel-Wise Interaction Hierarchy Graph

Keyang Cheng(✉), Hao Zhou, and Hao Wan

School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang, China
kycheng@ujs.edu.cn, {zhouhao,haowan}@stmail.ujs.edu.cn

Abstract. Although deep neural networks exhibiting superior performance across numerous tasks, their application in high-risk domains is limited by a lack of interpretability and trustworthiness. In this paper, an interaction value calculation method is firstly proposed, which faithfully represents the interaction utility of each variable in the feature map. Secondly, an interpretable method for top-down construction of interaction hierarchy graph based on interaction utility is proposed to understand the visualized knowledge represented by filters and to elucidate the decision-making process of the network. Extensive experiments were carried out on publicly available datasets and models that had been pre-trained. Experimental results indicate that each node in the graph consistently corresponds to the same part of an object across various images belonging to the same category. The faithfulness evaluation shows that the filters involved in the graph nodes have a major role in the network. Furthermore, the quantitative evaluation shows that our method improves over the others by an average of 0.18%, 1.19%, and 2.18% on the EBPG, mIoU, and Bbox metrics, respectively.

Keywords: Explainable AI · Interpretability · Computer Vision

1 Introduction

For the past decade, convolutional neural networks (CNNs) have led the field of computer vision research, demonstrating outstanding performance across various tasks including image classification [13], object detection [6,28], and other related domains [9,12,17]. However, CNNs often lack interpretability and trustworthiness due to their large number of free parameters, especially in high-risk areas such as medical diagnostics [8], financial risk forecast [26] and autonomous driving [16] where they often exhibit unexpected behavior and cause irreparable damage. Therefore, it is vital to open the black box of CNNs and improve their trustworthiness and transparency. We believe that a CNN trained for various scenarios can encode rich information about the object through its multilayer

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 147–161, 2025.
https://doi.org/10.1007/978-3-031-78183-4_10

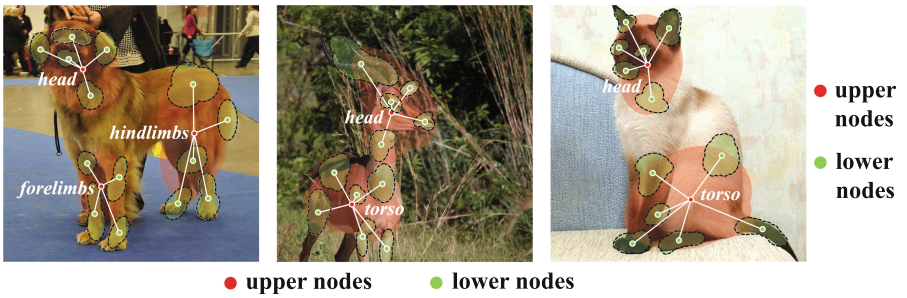


Fig. 1. The interaction hierarchy graph is constructed based on the interaction values of the feature maps, and one or more nodes in the graph represent the information encoded by a particular convolutional filter. The visual knowledge encoded by the CNN convolutional filter is understood by the hierarchical graph and the decision process is clarified by the edges of the graph.

convolution. We provide a perspective that enables us to interpret the information encoded within the convolutional layers and to analyze the CNN decision process, which also presents a number of challenges, such as (i) What kind of visual knowledge do CNN's convolutional filters encode? (ii) Which filters are primarily involved in the recognition results of the model? (iii) What are the relationships between the filters of the different convolutional layers?

Interpretable algorithms are able to make deep learning models understandable to humans. Researchers are increasingly interested in developing methods to visualize and interpret deep neural networks. However, there is a trade-off between performance and interpretability. The pursuit of interpretability can result in degradation of performance, which is undesirable in environments with some serious consequences, such as incorrect classifications or flawed decision-making processes. This paper focuses on post-hoc interpretability, that is, independent methods for interpreting an already trained and fixed target model, without compromising its performance. In this paper, the interaction value of the feature map is calculated by introducing multi-order interactions, which reflect the interaction utility of a variable in the feature map. In addition, we propose hierarchical graphs to understand the information encoded by the filters. As illustrated in Fig. 1, the hierarchical graph is constructed by extracting graph nodes in the feature interaction graph of the specified convolutional layer and computing the between-layer relationships of the nodes in a top-down manner. The contribution of this paper is as follows.

1 The interaction value calculation method is proposed that can reflect the interaction utility of variables, which can signify the interaction of each variable of the feature map in the CNN and provide a basis for interpretability.
2 The interpretable method for constructing hierarchical graphs using top-down of interaction values is proposed to understand the encoding information of filters and clarify the decision paths of CNN.

2 Related Work

Interpretability has several different directions. One branch of interpretability is post-hoc explanation methods, which aim to explain why a trained model made a specific decision in response to a given input. Post-hoc explanation methods can be divided into perturbation-based methods [3,19,25], which perturb the input and measure the degree of influence on the model outcome, and gradient-based methods [20,22,24], which calculate the back propagation of the modified gradient function.

Saliency Maps. Numerous efforts [2,20,29,34] have been made to investigate saliency map methods that reflect the pixels that have the greatest impact on model predictions. The saliency map approach in CNNs is the most intuitive way to explore patterns hidden within neurons. The up-convolutional network [5] transforms the feature map of the convolutional layer into an image. Gradient-based visualization [2,20] evaluates input images that maximize the activation score of neural units by gradients in back propagation. However, the saliency map only knows which areas the model focuses on and does not explain its decision process.

Decision Paths. [30] propose a distillation guided routing method to explore critical data routing paths in neural networks, which can faithfully compress the model into a compact model. In other words, the trustworthiness of the model compression is improved with interpretability. Explanatory Graph [32] generates graphs from CNN where each node represents a partial pattern corresponding to an activation peak in the relevant layer, and each edge connects two nodes in adjacent layers to reflect the interrelationship. [27] combines decision trees and neural networks, allowing neural networks to have decision tree-like interpretability and to output intermediate decisions for prediction. [33] roughly represents the path of a CNN prediction by a decision tree with semantics, where each node in the tree represents a decision pattern of the CNN. But these approaches only superficially decompose the decision paths of the network and do not explain the rationale for such decisions and the knowledge hidden in the decisions.

3 Method

In this section, we design the interaction values to reflect the utility of each variable in the input image or feature map of the model. Graph nodes are extracted from the peaks of the interaction values of the feature graph, and the edges of the graph connect two nodes of the neighbouring layer. The nodes represent the visual knowledge of a particular convolutional filter, and the edges represent the interrelationships of this visual knowledge and the decision paths of the model. The construction process of the interaction hierarchy graph is shown in Fig. 2.

Specifically, firstly, the feature graph of the deep neural network is extracted and its corresponding interaction values are calculated according to Eq. 4. Then, the peaks of each interaction feature graph are found using Eq. 5, and the graph nodes are extracted from the interaction peaks according to 6. Finally, the edges between the upper nodes and the lower nodes are created according to 7. This clarifies (i) which object parts activate which filters in the model, (ii) which filters play an important role in the detection of the model, (iii) how the filters of the different convolutional layers are related to each other, and (iv) How much do these filters contribute to the prediction score. Theoretically, we could explain CNN under different tasks. However, in this study, we focus on the CNN used for object classification to simplify the story.

Preliminaries: Before analyzing the CNN decision process, we first introduce the multi-order interactions [31] between input variables. Given a pre-trained DNN model v and an input image $X = \{1, 2, , ..., n\}$ with n variables, $v(X)$ denotes the network output of X. CNNs are usually inference through interactions between variables rather than individual variables. For example, the interaction between pixels i, j, k can form the edge of an object. If this interaction increases the output of the network, the pattern is considered to have a positive effect on the network. Conversely, if this pattern decreases the output of the network, this pattern has a negative effect on the network.

The multi-order interaction $I^{(m)}(i, j)$ between two input variables $i, j \in X$ is measured for interactions of different complexity. The m-order is considered to represent the contextual complexity. Overall, the multi-order interaction $I^{(m)}(i, j)$ measures the average utility of the interaction between variables i, j under all possible scenarios consisting of m variables. The multi-order interaction $I^{(m)}(i, j)$ can be shown as:

$$I^{(m)}(i, j) = \mathbb{E}_{S \subseteq X \setminus \{i,j\}, |S|=m}[\Delta v(i, j, S)] \tag{1}$$

where $S \subseteq X$ is any combination of m variables, $v(S)$ is the output score when keeping the variables in S constant but replacing the variables in $X \setminus S$ with baseline value [1], and $\Delta v(i, j, S) = v(S \cup \{i, j\}) - v(S \cup \{i\}) - v(S \cup \{j\}) + v(S)$.

3.1 Interaction Value

As shown in Eq. 1, the multi-order interaction is computed only for the input, and we expand it to each layer of the convolutional neural network. $f_{l\sim}(\cdot)$ represents the latter part of the model after the l-th convolutional layer. In other words, the model is divided into two parts starting from the l-th convolutional layer, with $f_{l\sim}(\cdot)$ being the part after this separation. As with $f_{l\sim}(\cdot)$, let $f_{\sim l}(\cdot) : \mathbb{R}^{H \times W \times 3} \mapsto \mathbb{R}^{H_l \times W_l \times D_l}$ be any function that converts the input into a feature map (*i.e.*, the first l layers of the CNN). So the feature map of the l-th convolution layer is $f_{\sim l}(X) = A_l \in \mathbb{R}^{H_l \times W_l \times D_l}$. As before, the two variables i, j belong to A_l. Thus the multi-order interaction of the l-th layer feature map can be denoted as:

$$I_l^{(m)}(i, j) = \mathbb{E}_{S \subseteq A_l \setminus \{i,j\}, |S|=m}[\Delta f_{l\sim}(i, j, S)] \tag{2}$$

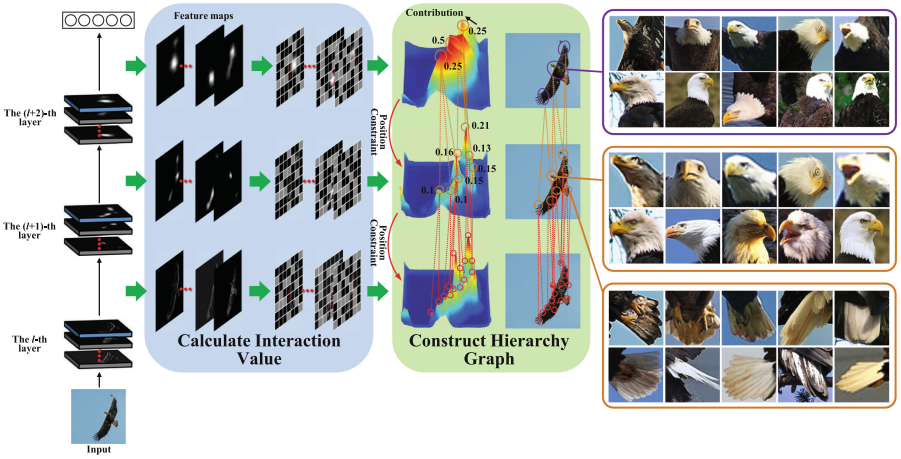


Fig. 2. The overall structure of the proposed method. The feature interaction map is generated by calculating the corresponding interaction values based on the feature maps of the CNN, and graph nodes are extracted from the feature interaction graph. One or more of these nodes correspond to the visual knowledge encoded by the specified convolutional filter. The graph structure is constructed in a top-down manner based on the nodes in each layer, and each edge in the graph connects two nodes in the adjacent layer, which reflects the relationship between the upper and lower layers of the nodes. The creation of edges is determined by the shortest Euclidean distance between the lower node and the upper node. The interaction hierarchy graph reveals the filters that play an important role in the CNN and clarifies their decision paths.

where

$$\begin{aligned}\Delta f_{l\sim}(i,j,S) =& f_{l\sim}(S\cup\{i,j\}) - f_{l\sim}(S\cup\{i\}) \\ & - f_{l\sim}(S\cup\{j\}) + f_{l\sim}(S)\end{aligned} \tag{3}$$

According to the multi-order interactions at the pixel level, we find that variable i is involved in computing interactions of $|A_l|-1$ complexities, which jointly reflect the interaction utility of the variable. Thus, interaction value is proposed to represent the complex interaction utility of a variable in a set of features. For example, a variable i in the feature map A_l which has $|A_l|-1$ types of multi-order interactions in the contextual features. The interaction value of the variable i in the l-layer feature map is defined by the following equation.

$$V(i) = \frac{1}{(|A_l|-1)^2} \sum_{j\subseteq A_l\setminus\{i\}} \sum_{m=0}^{|A_l|-2} I_l^{(m)}(i,j) \tag{4}$$

The interaction value incorporates all orders of multi-order interaction between variable i and any other variable, which represents the interaction utility of variable i in the convolutional layer.

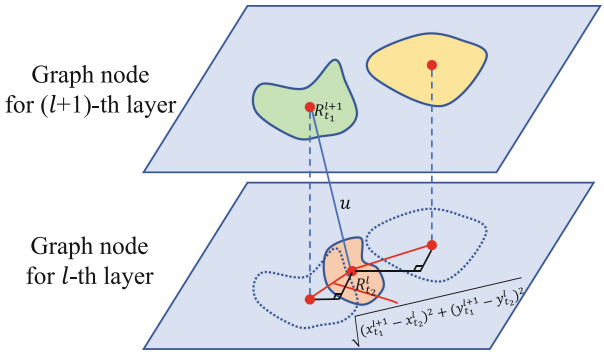


Fig. 3. Mapping the upper level nodes into the lower level space and inferring the attribution of the lower level nodes by the minimum Euclidean distance.

3.2 Interaction Hierarchy Graph

The interaction value of each variable in the feature map A_l is calculated by Eq. 4 to obtain the feature interaction map $V_l = V(i \subseteq A_l) \in \mathbb{R}^{H_l \times W_l \times D_l}$. Each variable in the feature interaction map V_l corresponds to the interaction value of the feature map A_l.

To construct the hierarchical graph, positional constraints on the upper and lower nodes are incorporated, mapping the lower nodes into the upper layer and selecting edges based on the Euclidean distance between the lower and upper nodes. The mask containing the node information $M_l \in \mathbb{R}^{H_l \times W_l \times D_l}$ is generated via Eq. 5, which leaves the first λ of the variables in the feature interaction graph unchanged and sets the others to 0.

$$M_l = M_\lambda(\mathcal{S}(V_l), V_l) \tag{5}$$

where $\lambda \in [0, 1]$. Let $\mathcal{S}(\cdot)$ be a sorting function such that $\mathcal{S}_r(V_l)$ is the r^{th} largest element in V_l. We define the mask function $M_\lambda(\cdot)$ such that $M_\lambda(\mathcal{S}(V_l), V_l)$ replaces all the variables in the feature interaction map with the mask distribution. The M_l embodies variables that have excellent interaction utility in the feature interaction map.

For an interaction feature map, the set of continuous variables on the two-dimensional space is considered to be a graph node. $M_l^k \in \mathbb{R}^{H_l \times W_l}$ is the feature interaction map of the k-th filter in the l-th convolutional layer. There is not necessarily one node in M_l^k, there may be more or none. Therefore we extract the e graph nodes of the l-th convolutional layer by the Graphing operation.

$$\left\{R_t^{k,l}\right\}_{t=1}^{e} = \text{Graphing}\left(M_l\right), k \leq D_l \tag{6}$$

where $R_t^{k,l} = \{(x, y), c, u\}$ denotes a node in the graph that corresponds to the node of the feature interaction graph of the k-th filter in the l-th convolutional layer, specifically the t-th graph node in the l-th convolutional layer. (x, y) is the

Algorithm 1. Construct the Interaction Hierarchy Graph

Input: feature map A_l of the l-th conv-layer, node sets G_{l+1} in the upper conv-layer.
Output: node sets G_l.

Calculate the interaction values for each variable of the feature map to obtain the feature interaction map.
for i in A_l **do**
  $V_l(i) = \frac{1}{(|A_l|-1)^2} \sum_{j \subseteq A_l \setminus \{i\}} \sum_{m=0}^{|A_l|-2} I_l^{(m)}(i,j)$
end for
Mask containing the graph node information is derived from the feature interaction graph.
$M_l = M_\lambda(S(V_l), V_l)$
Convert mask to graph node.
$\left\{R_t^{k,l}\right\}_{t=1}^{e} = \text{Graphing}\,(M_l)$ where $R_t^{k,l} = \{(x,y), c, u\}$
for t_1 in $[1,2,...,e_{l+1}]$, t_2 in $[1,2,...,e_l]$ **do**
  if $\min \sqrt{(x_{t_1}^{l+1} - x_{t_2}^{l})^2 + (y_{t_1}^{l+1} - y_{t_2}^{l})^2}$ and l is not top layer **then**
    $u_{t_2}^{l} := t_1$
  else
    $u_{t_2}^{l} := -1$
  end if
end for
return $\left\{R_t^{k,l}\right\}_{t=1}^{e} \in G_l$

position of the node, c is the information of the node and u is the edge of the node to some node in the previous convolution layer. The creation of the edge u is illustrated in Fig. 3. Map the upper nodes in the lower space, calculate the Euclidean distance between nodes in the lower layer and all nodes in the upper layer, and create an edge between the lower node and the upper node with the smallest distance. The specific calculation is shown in Eq. 7.

$$u_{t_2}^{l} := \begin{cases} t_1\Big|_{1 \le t_1, t_2 \le e_l, e_{l+1}} \min \|\mathcal{P}_{t_1}^{l+1}, \mathcal{P}_{t_2}^{l}\|_2 & , l \text{ is top layer} \\ -1 & , l-1 \text{ is top layer} \end{cases} \tag{7}$$

where e_l is the number of nodes in the l-th layer of the graph and $\mathcal{P}_{t_1}^{l} = (x_{t_1}^{l}, y_{t_1}^{l})$ is the position of the t_1-th node in the l-th convolutional layer. The construction of the hierarchical graph is shown in Algorithm 1.

The information encoded by some nodes in the lower layers of the hierarchical graph together constitute the information encoded by a node in the upper layers.

These graph nodes encode the same information in the same class of images.

3.3 Filter Contribution

Above we have constructed interaction hierarchy graphs to clarify the decision making process of CNN and to understand the visual knowledge associated with some of the filters through graph nodes. Next, we explore the contribution of

Fig. 4. Visual knowledge of filters representation. Some critical filters in the same class of images represent the same visual knowledge, but in other classes of images, these filters encode irrelevant parts.

filters in CNN. Since one or more nodes in the hierarchical graph correspond to a filter, we rename the filter's sort number. Let the set of filters involved in the graph be $K = \{k_1, k_2, ...\}$. The score of the filter in the CNN is calculated as shown in Eq. 8.

$$S_l^{k_i} = f_{\sim l}(A_l) - f_{\sim l}(\mathbb{M}^{k_i}(A_l)) \tag{8}$$

where $k_i \in K$, and $\mathbb{M}^{k_i}(A_l)$ represents the value of the k_i-th channel $A_l^{k_i}$ of the feature map replaced by the baseline value.

From the scores of the filters obtained from the above equation, the contribution of each filter is calculated by the softmax function with the following formula.

$$C_l^{k_i} = \frac{S^{k_i}}{\sum_{k \in K} S_l^k} \tag{9}$$

where $C_l^{k_i}$ represents the value of the contribution of the K_i-th filter in the l-th convolution layer. We replace the channel of the feature map with the baseline value, which corresponds to the filter for which the contribution will be sought. The difference between the pre-replacement score and the post-replacement score is the filter score and the contribution is derived from the filter score. In summary, we constrain the flow through the forward propagation to obtain the filter contribution.

In this part, we only address the filters involved in the hierarchical diagram and consider that the rest of the filters play a minimal role in the prediction of the model's results.

Table 1. Quantitative evaluation results of the state-of-the-art approach compared with our method on ResNet-50.

Metric	EBPG	mIoU	Bbox
Grad CAM [20]	60.08	32.16	60.25
Grad CAM++ [2]	47.78	30.16	58.66
Extremal Perturbation [7]	63.24	26.29	52.34
RISE [18]	32.86	27.40	55.55
Score CAM [29]	35.56	31.0	60.02
Integrated Gradient [24]	40.62	15.41	34.79
FullGrad [23]	39.55	20.20	44.94
Ours method	**65.07**	**32.8**	**61.49**

4 Experiments

In this section, the experimental setup is first introduced in terms of data sets and models. Secondly, the representation of the images of the filters in the network in the same and different classes is visualised. Then, convolution filters are perturbed to demonstrate the faithfulness of the interaction hierarchy graph. Finally, we compare a range of saliency methods to demonstrate the superiority of our approach in localization evaluation.

4.1 Experimental Settings

Datasets: The following datasets were utilized in our experiments: (1) The ILSVRC2012 [4] is a portion of the extensive, manually-labeled ImageNet, which is organized based on the WordNet hierarchy. Each hierarchy node is associated with hundreds of images. This dataset contains 1000 leaf categories, with the entire hierarchy comprising 1860 nodes. (2) PASCAL VOC 2007 [11] is a dataset focused on object detection, including 4952 images across 20 categories. (3) CIFAR-10 [14] is a subset of the Tiny Images dataset, composed of 60,000 32×32 color images distributed among 10 categories.

Types of CNNs: To demonstrate the broad applicability of the proposed method, we considered three types of CNNs to prove our approach, i.e. the VGG-16 [21], the ResNet-50 [10], the ResNet-101 [10] and the AlexNet [15].

4.2 Visual Knowledge of Filters

In this experiment, we tested whether the critical filters in the hierarchical graph represent the same target part in different images of the same class. We constructed the interaction hierarchy graph on a VGG-16 network to extract filters with outstanding roles based on node information. Specifically, as shown in Fig. 4, we visualized some of the filters in layers 10 and 13 of the VGG-16 network. For example, in the third row, the 299th filter in layer 13 represents the head and

Table 2. Quantitative evaluation results of the state-of-the-art approach compared with our method on VGG16.

Metric	EBPG	mIoU	Bbox
Grad CAM [20]	55.44	26.52	51.70
Grad CAM++ [2]	46.29	28.10	55.59
Extremal Perturbation [7]	**61.19**	25.44	51.20
RISE [18]	33.44	27.11	54.59
Score CAM [29]	46.42	27.71	54.98
Integrated Gradient [24]	36.87	14.11	33.97
FullGrad [23]	38.72	26.61	54.17
Ours method	59.72	**28.29**	**57.12**

body parts of different Siamese cat images, while the 435th and 503rd filters in layer 10 represent the nose and ear areas of the Siamese cat, respectively. In the left part, the results of the visualization of the filters for different images of the same category are indicated separately. It is easy to see that these filters represent the same visual knowledge in the same category of images. In the right part, these filters represent unrelated visual knowledge on other categories of images, and these filters are not present in the hierarchical graph of the image, *i.e.* these filters play a minor role in the prediction of the image. Experimental results show that our approach is able to understand the visual information encoded by the filters in the network from within the convolutional layer, which is consistent across images or datasets.

4.3 Quantitative Evaluation

This part measures the accuracy of the significance location evaluation by objective quantification. The saliency map of our method is derived from the feature interaction map corresponding to the top 10% of the filter contribution ranking. A number of metrics were quantified in the experiment, including Energy-Based Pointing Game (EBPG), mean Intersection over Union (mIoU) and Bounding box (Bbox).

Inspired by **Energy-Based Pointing Game (EBPG)**, we binarised the sample images using the bounding boxes of the prediction category. We specify the ground-truth mask as G and the prediction mask as P. The metric then indicates how many attention regions are within the predicted bounding box, represented as $Proportion = (\|G \odot P\|_1 / \|P\|_1) \times 100$.

mean Intersection over Union (mIoU) is a recognised metric in image segmentation that is used to anlayze the localization ability and significance of attributes captured in the saliency map. $mIoU = \frac{1}{k} \sum_{i=1}^{k} (P \cap G)/(P \cup G) \times 100$, where k is the number of classes. This metric evaluates global localization.

Bounding box (Bbox) is used to quantify the extent to which the imputation method identifies and localizes the region of interest. Assuming that the bounding box contains n pixels, we evaluate the number of top k pixels in the prediction mask P. $Bbox = (k/n) \times 100$, where $k \leq n$.

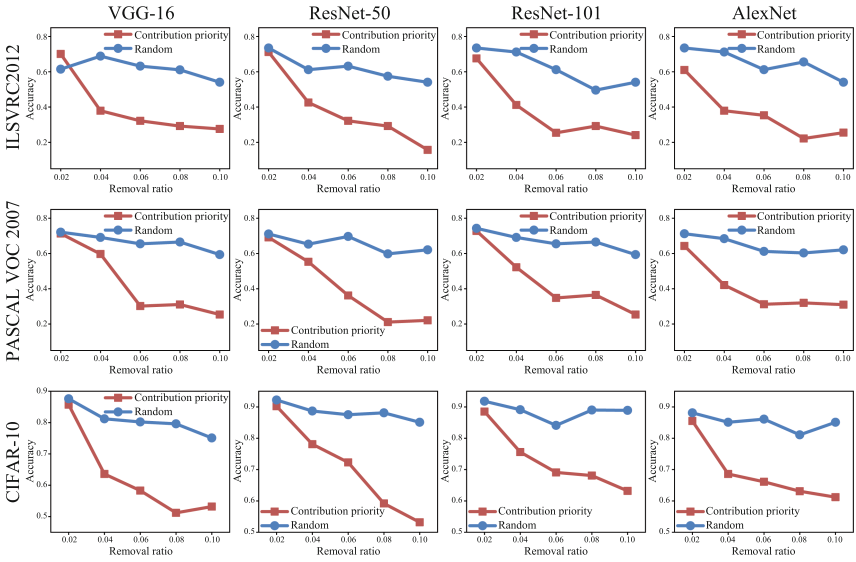


Fig. 5. Filter perturbations for training different CNNs on various image datasets, with the red line being contribution-first driven perturbations and the blue line being random perturbations. (Color figure online)

We compared our methods with the most advanced XAI methods, such as Grad CAM [20], Grad CAM++ [2], Extremal Perturbation [7], RISE [18], Score CAM [29], Integrated Gradient [24] and FullGrad [23]. Tables 1 and 2 show the results of comparing our method with other methods on VGG-16 and ResNet-50 trained on the PASCAL VOC 2007 dataset. For each metric, the best is shown in bold and the second best is underlined. All values are reported as percentages. Our method achieves excellent results for different metrics and models.

4.4 Faithfulness Evaluation

The reliability of the filters corresponding to the nodes in the interaction hierarchy graph was verified. It is misleading to rely solely on the visualization of the model's convolutional filters for visual evaluation. Therefore, perturbations of the convolutional kernel were introduced to ensure the visual interpretability of the hierarchical graph. We selected the top 10% of filters in the model based on the previously calculated filter contributions considered. The filters were selected from the filter contribution ranking. In Sect. 3.3, the filter contribution calculation method within the convolutional layer is presented, and here we extend it to the convolutional part of the whole model, i.e. the contribution of a particular convolutional filter in the whole network. The score of each filter is calculated according to Eq. 8, and then the global contribution of each filter is calculated by putting all the scores into Eq. 9. Briefly, K in Sect. 3.3 refers to the set of filters involved in the hierarchical graph within a convolutional layer, which is

Table 3. Recognition evaluation results of the state-of-the-art approach compared with our method on ResNet-50.

Metric	AD(%)	AI(%)
Grad CAM [20]	35.80	36.58
Grad CAM++ [2]	41.77	32.15
Extremal Perturbation [7]	39.38	34.27
RISE [18]	39.77	37.08
Score CAM [29]	35.36	37.08
Integrated Gradient [24]	66.12	24.24
FullGrad [23]	65.99	25.36
Ours method	**31.70**	**39.36**

replaced here with the set of filters for the whole network. As shown in Fig. 5, it corresponds to a different dataset and model for each row and column, respectively. The red line records the trend in accuracy for the gradual removal of the top 10% filters, which shows a sharp decline. The blue line records the trend in accuracy for the gradual random removal of the 2% to 10% filters, which shows an erratic slow increase or decrease.

Moreover, the Hadamard product between the input sample and attention mechanism primarily serves to obstruct the input sample, allowing for an assessment of accuracy changes in the target class during the measurement of model confidence in the classification task. To mitigate the influence of the convolution kernel in the region of highest correlation, the partial correlation convolution kernel is set to zero. Comparison methods like LIME, RISE, and Score CAM are used to perturb the model input. Therefore, faithfulness is measured using Average Drop and Average Increase. **Average Drop (AD)** measures the maximum positive deviation between the prediction made using the input image and the prediction obtained with the saliency map. The AD is defined as follows: $AD = \frac{1}{N}\sum_{i=1}^{N}(\frac{1}{Y_i^c}\max(0, Y_i^c - O_i^c)) \times 100$, where Y_i^c represents the predicted score for class c on image i, and O_i^c is the predicted score for class c when using the explanation map region as input. Here, N is the total number of images in the dataset. The Average Drop indicates how the saliency map affects the scores, with a lower AD value reflecting a higher credibility of the model. Conversely, **Average Increase (AI)** describes the situation where the saliency map results in a higher score. It is given by: $AI = \sum_{i=1}^{N}(\frac{1}{N}\text{Sign}(Y_i^c < O_i^c)) \times 100$. A higher AI score signifies greater confidence in the model's interpretability. Table 3 summarizes the experimental results on the ILSVRC2012 dataset, where lower values for average drop and higher values for average increase are desirable. The algorithm proposed in this paper achieves an AD rate of 31.7% and an AI rate of 39.36%, with the AD rate being 3.66% superior and the AI rate being 2.28% better compared to other methods.

The experimental results indicate that the filters selected from the interaction hierarchy graph are critical to the prediction of the input image and the model.

5 Conclusion

In this paper, interpretable interaction hierarchy graph methods are proposed to improve the interpretability of deep neural networks. Interaction value based on multi-order interactions is proposed, which can represent the interaction utility of variables within the network. A top-down hierarchical graph construction method based on interaction values is proposed to understand the visual encoding of convolutional filters and to clarify the decision-making process of the network. A method for calculating the contribution of convolutional filters is proposed to differentiate the significance of filters in the network. Experimental results show that interaction hierarchy graph significantly enhances visual interpretation and validates the interpretability of the model.

Acknowledgements. This work was supported by National Natural Science Foundation of China [62372215] and Jiangsu Science and Technology Project [BE2022781].

References

1. Ancona, M., Oztireli, C., Gross, M.: Explaining deep neural networks with a polynomial time algorithm for Shapley value approximation. In: International Conference on Machine Learning, pp. 272–281. PMLR (2019)
2. Chattopadhay, A., Sarkar, A., Howlader, P., Balasubramanian, V.N.: Grad-CAM++: generalized gradient-based visual explanations for deep convolutional networks. In: 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 839–847. IEEE (2018)
3. Deng, H., Ren, Q., Zhang, H., Zhang, Q.: Discovering and explaining the representation bottleneck of DNNs. In: International Conference on Learning Representations (2022). https://openreview.net/forum?id=iRCUlgmdfHJ
4. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255. IEEE (2009)
5. Dosovitskiy, A., Brox, T.: Inverting visual representations with convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4829–4837 (2016)
6. Fan, Q., Zhuo, W., Tang, C.K., Tai, Y.W.: Few-shot object detection with attention-RPN and multi-relation detector. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4013–4022 (2020)
7. Fong, R., Patrick, M., Vedaldi, A.: Understanding deep networks via extremal perturbations and smooth masks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 2950–2958 (2019)
8. Hadji Misheva, B., Hirsa, A., Osterrieder, J., Kulkarni, O., Fung Lin, S.: Explainable AI in credit risk management. Credit Risk Manag. **57**, 203–216 (2021)
9. Hase, P., Bansal, M.: Evaluating explainable AI: which algorithmic explanations help users predict model behavior? In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 5540–5552 (2020). https://doi.org/10.18653/v1/2020.acl-main.491. https://aclanthology.org/2020.acl-main.491
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016

11. Hoiem, D., Divvala, S.K., Hays, J.H.: PASCAL VOC 2008 challenge. World Lit. Today **24** (2009)
12. Hu, Y., et al.: Efficient semantic segmentation by altering resolutions for compressed videos. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 22627–22637 (2023)
13. Kim, Y.J., Bae, J.P., Chung, J.W., Park, D.K., Kim, K.G., Kim, Y.J.: New polyp image classification technique using transfer learning of network-in-network structure in endoscopic images. Sci. Rep. **11**(1), 1–8 (2021)
14. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. Commun. ACM **60**(6), 84–90 (2017)
16. Loquercio, A., Segu, M., Scaramuzza, D.: A general framework for uncertainty estimation in deep learning. IEEE Robot. Autom. Lett. **5**(2), 3153–3160 (2020)
17. Mohankumar, A.K., Nema, P., Narasimhan, S., Khapra, M.M., Srinivasan, B.V., Ravindran, B.: Towards transparent and explainable attention models. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 4206–4216 (2020). https://doi.org/10.18653/v1/2020.acl-main.387. https://aclanthology.org/2020.acl-main.387
18. Petsiuk, V., Das, A., Saenko, K.: RISE: randomized input sampling for explanation of black-box models. In: British Machine Vision Conference (BMVC) (2018). http://bmvc2018.org/contents/papers/1064.pdf
19. Ribeiro, M.T., Singh, S., Guestrin, C.: "Why should i trust you?" explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1135–1144 (2016)
20. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-CAM: visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 618–626 (2017)
21. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations (2015). http://arxiv.org/abs/1409.1556
22. Smilkov, D., Thorat, N., Kim, B., Viégas, F., Wattenberg, M.: SmoothGrad: removing noise by adding noise. arXiv preprint arXiv:1706.03825 (2017)
23. Srinivas, S., Fleuret, F.: Full-gradient representation for neural network visualization. Adv. Neural Inf. Process. Syst. **32** (2019)
24. Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: International Conference on Machine Learning, pp. 3319–3328. PMLR (2017)
25. Tonekaboni, S., Joshi, S., Campbell, K., Duvenaud, D.K., Goldenberg, A.: What went wrong and when? Instance-wise feature importance for time-series black-box models. Adv. Neural. Inf. Process. Syst. **33**, 799–809 (2020)
26. Torrent, N.L., Visani, G., Bagli, E.: PSD2 explainable AI model for credit scoring. arXiv preprint arXiv:2011.10367 (2020)
27. Wan, A., et al.: NBDT: Neural-backed decision tree. In: International Conference on Learning Representations (2021). https://openreview.net/forum?id=mCLVeEpplNE
28. Wang, C.Y., Bochkovskiy, A., Liao, H.Y.M.: YOLOv7: trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7464–7475 (2023)

29. Wang, H., et al.: Score-CAM: score-weighted visual explanations for convolutional neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp. 24–25 (2020)
30. Wang, Y., Su, H., Zhang, B., Hu, X.: Interpret neural networks by identifying critical data routing paths. In: proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 8906–8914 (2018)
31. Zhang, H., Li, S., Ma, Y., Li, M., Xie, Y., Zhang, Q.: Interpreting and boosting dropout from a game-theoretic view. In: International Conference on Learning Representations (2020)
32. Zhang, Q., Wang, X., Cao, R., Wu, Y.N., Shi, F., Zhu, S.C.: Extraction of an explanatory graph to interpret a CNN. IEEE Trans. Pattern Anal. Mach. Intell. **43**(11), 3863–3877 (2020)
33. Zhang, Q., Yang, Y., Ma, H., Wu, Y.N.: Interpreting CNNs via decision trees. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6261–6270 (2019)
34. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2921–2929 (2016)

Denoising Optimization-Based Counterfactual Explanations for Time Series Classification

Omar Bahri(✉), Peiyu Li, Pouya Hosseinzadeh, Soukaina Filali Boubrahimi, and Shah Muhammad Hamdi

Department of Computer Science, Utah State University, Logan, UT, USA
{omar.bahri,pouya.hosseinzadeh,peiyu.li,
soukaina.boubrahimi,s.hamdi}@usu.edu

Abstract. Counterfactual explanations are valuable tools in Explainable AI (XAI) for understanding complex machine learning models. While successful in tabular and image data, existing optimization-based counterfactual methods struggle with time series due to generated explanations containing unrealistic high-frequency oscillations, which significantly hinders their plausibility. This paper proposes a novel approach to address this issue. Our method leverages the Discrete Fourier Transform (DFT) to eliminate oscillations from counterfactual explanations generated by any optimization-based technique. This allows for the utilization of established counterfactual explanation algorithms from other domains while ensuring the resulting explanations are temporally coherent and interpretable for time series data. We demonstrate the effectiveness of our approach on time series classification tasks using datasets from the University of California Riverside (UCR) archive. Experiments show significant improvement in the plausibility of counterfactual explanations generated by three state-of-the-art methods. This paves the way for a more efficient application of existing counterfactual explanation techniques to the time series domain.

Keywords: Counterfactual Explanations · XAI · Time Series Classification · Discrete Fourier Transform.

1 Introduction

Counterfactual explanation models have gained significant traction in the realm of Explainable Artificial Intelligence (XAI), aiding in elucidating the decision-making processes of complex machine learning models and offering actionable insights for model improvement. They offer stakeholders valuable explanations,

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 162–179, 2025.
https://doi.org/10.1007/978-3-031-78183-4_11

shedding light on why a specific decision was made and how slight changes in input variables could lead to different outcomes. For example, rather than simply rejecting a mortgage application, a bank could provide a rationale through a counterfactual explanation, saying something along the lines of: "*Had your yearly salary been $25,000 higher, your mortgage would have been approved*". Counterfactual explanations, generated post hoc after training the black-box classifier, involve constructing synthetic data points that lead to different model outputs, known as counterfactual instances.

While prominently utilized in tabular and image data domains, counterfactual models have encountered challenges in achieving similar success within the domain of time series. Factors such as the absence of curated datasets, the inherently non-intuitive characteristics of time series data, and its heightened complexity collectively contribute to this limitation [11,20]. In the existing literature, methods generally adhere to two main approaches. Some aim to produce a counterfactual explanation from the original input by incorporating features from the available training data [1–3,8,15–17]. Others optimize a loss function to find the smallest perturbation required to alter the model output while preserving the fidelity of the original data [9,19,23,24]. These optimization-based methods have originally been introduced for tabular and image data domains, where they have proven to be effective, providing interpretable explanations by perturbing individual features or pixels within reasonable bounds. In recent works, the same methods have been applied to time series [3,10,14,16]. However, as we will show in the experiments, the continuous and sequential nature of time series data complicates the perturbation process, resulting in unrealistic counterfactual instances characterized by high-frequency oscillations throughout their length. These noisy explanations fail to capture the temporal coherence of the original data, diminishing their interpretability and utility for stakeholders.

In this work, we propose an approach to resolve this issue. Our solution can be easily introduced to any optimization-based counterfactual explanation algorithm, without significant overhead. It eliminates the noise from the counterfactual instance by applying a low-pass filter in the frequency domain and/or clipping the amplitudes of each frequency band to match time series in the original dataset. Our objective is to further the progress of time series counterfactual explanation techniques by facilitating a more efficient utilization of the well-established literature in the tabular and image domains. Through experiments on datasets from the University of California Riverside (UCR) [7] time series classification archive, we show that our approach highly improves the plausibility of the counterfactual explanations generated by three state-of-the-art methods. The rest of this paper is organized as follows. In Sect. 2, we introduce the counterfactual explanation problem and the main optimization-based approaches. In Sect. 3, we describe our proposed approach. In Sect. 4, we discuss the details of the experiments and implementation. In Sect. 5, we present the results and discuss the findings. Finally, we conclude with a summary in Sect. 6.

2 Preliminaries

2.1 Problem Formalism

Consider a dataset with N time series instances $\boldsymbol{\mathcal{D}} = \{T_1, T_2, ..., T_N\}$, such that each $T_i = (t_1, t_2, ..., t_L) \in \mathbb{R}^L$, where L is the length of the time series (i.e. number of time steps) is mapped to a class from the mutually exclusive set $\boldsymbol{\mathcal{C}} = \{C_1, C_2, ..., C_M\}$. Given a black-box classification model with prediction function $f : \mathbb{R}^L \rightarrow \boldsymbol{\mathcal{C}}$; $T \mapsto C$ trained on the dataset and an instance $T \in \boldsymbol{\mathcal{D}}$ with class prediction $f(T) = C_m$, a counterfactual explanation $T' = T + \delta$ is generated by introducing a perturbation $\delta \in \mathbb{R}^L$ to T to achieve $f(T') = f(T+\delta) = C_{m'}$ where $C_{m'} \neq C_m$ is the target counterfactual class. Several criteria have been used throughout the literature to assess the quality of a counterfactual explanation. We focus on the two most important ones:

- **Proximity:** The counterfactual explanation has to be close to the original time series instance. The L_1-, L_2-, and L_{inf}-norms have all been used as proximity measures [3,8,10,19,23]. Formally, $||\delta||_{p\in\{1,2,\infty\}}$ has to be minimal.
- **Plausibility:** The counterfactual explanation has to be realistic. Thus, it must lie within the data manifold of the training data. The use of autoencoders and outlier detection methods such as the local outlier factor [6], isolation forests [18], and one-class support vector machine [21] have occurred in the literature [3,8,23]

Optimization-based counterfactual explanation methods aim to find the best perturbation δ by minimizing a loss function that incorporates desired counterfactual criteria.

2.2 Wachter's Method

Wachter et al. [24] proposed one of the first and most famous optimization-based approaches for generating counterfactual explanations. The loss function defined by the authors includes two terms:

$$\lambda(f(T') - C_{m'}) + d(T, T') \tag{1}$$

The first term represents the quadratic distance between the black-box prediction and the target class probability. It encourages the generation of a time series instance belonging to the target class. The second term represents the distance between the original data instance and the counterfactual explanation. It ensures that the explanation remains as close as possible to the original data. λ is a parameter that controls how confident the model should be in predicting T' as belonging to $C_{m'}$ relative to the distance between T and T'. The authors suggest using high values of λ and the following distance function:

$$d(T, T') = \sum_{i}^{L} \frac{|t_i - t'_i|}{MAD_i} \tag{2}$$

where the nominator represents the Manhattan distance (L_1-norm) between the two time series and the denominator represents the median absolute deviation (MAD) at each time step.

$$MAD_i = median_{j \in N}(|T_{j,i} - median_{k \in N}(T_{k,i})|) \tag{3}$$

Wachter et al. [24] used the Adam optimizer to find the best perturbation (assuming that the black-box model is differentiable) by randomly initializing T' and minimizing the loss function. This process is repeated multiple times and the explanation with the minimum loss value is selected.

2.3 Contrastive Explanation Method

Dhurandhar et al. [9] introduced the Contrastive Explanation Method (CEM), a method for finding pertinent positives, the minimal set of features required for a certain classification target, and pertinent negatives, the set of features that have to be modified to achieve the classification target. The second problem is equivalent to finding counterfactual explanations [11,23]. CEM extends Wachter's loss function by replacing the distance function with an elastic net regularizer function [27] and adding a third term that encourages the generated explanation to be within the data manifold of the original dataset. It generates counterfactual explanations by minimizing the following loss function:

$$\lambda(f(T') - C_{m'}) + d(T, T') + \gamma||\delta - AE(T + \delta)||_2^2 \tag{4}$$

The first term is similar to Wachter's method's. The second term is the elastic net regularizer [27] defined as:

$$d(T, T') = \beta||\delta||_1 + ||\delta||_2^2 \tag{5}$$

The third term represents the L_2 reconstruction loss of $T' = T + \delta$ using an autoencoder model AE trained on $\boldsymbol{\mathcal{D}}$. The loss function has three hyperparemeters: $\lambda, \beta, \gamma \geq 0$. The best counterfactual explanation is found by optimizing the loss function using the FISTA algorithm [4]. CEM was originally proposed for tabular and image data. In addition, Labaien et al. [14] have shown that it is also suitable for explaining time series data.

2.4 Counterfactual Explanations Guided by Prototypes

Van Looveren and Klaise [23] introduced Counterfactual Explanations Guided by Prototypes (CEGP), a method that extends CEM by adding a new term to the loss function that encourages the generated counterfactual instance to fall within the data manifold of the target class $C_{m'}$:

$$\lambda(f(T') - C_{m'}) + d(T, T') + \gamma||\delta - AE(T + \delta)||_2^2 + \theta||ENC(T') - proto_{m'}||_2^2 \tag{6}$$

ENC represents the encoder part of AE and $proto_{m'}$ is the average encoding of the K (a hyperparameter) nearest neighbors of $ENC(T)$ from class $C_{m'}$. Therefore, the fourth term represents the Euclidean distance (L_2-norm) between the prototype of the target class and the encoding of the counterfactual explanation.

In case the AE is not available, CEGP uses k-d trees [5] to compute the target class prototype $proto_{m'}$ by building a k-d tree using all $C_{m'}$ instances and selecting the nearest element of the tree to T. The loss function becomes:

$$\lambda(f(T') - C_{m'}) + d(T, T') + \theta||T' - proto_{m'}||_2^2 \tag{7}$$

such as the last term is now computed in the original data space.

2.5 TimeX

Filali Boubrahimi et al. [10], a model-agnostic counterfactual explanation algorithm for time series that works similarly to CEGP when the AE is not available. Instead of using k-d trees to compute class prototypes, TimeX uses dynamic barycenter averaging. In addition, TimeX generates a saliency map to find the most significant contiguous segment in the time series and limits the gradient updates to it. This results in less sparse perturbations.

2.6 Discrete Fourier Transform

The Discrete Fourier Transform (DFT) is a fundamental tool in signal processing and time series analysis. It provides a way to decompose a signal into its constituent frequencies, revealing the underlying periodic components that make up the signal. The DFT transforms time series $T = (t_1, t_2, ..., t_L)$ into a sequence of complex numbers called Fourier coefficients $DFT(T) = (X_1, X_2, ..., X_L)$, which represent the signal in the frequency domain. Each Fourier coefficient is a complex number with a real and imaginary part $X_k = (real_k, imag_k)$ computed as $X_k = \sum_{i=1}^{L} t_i e^{-j2\pi \frac{k}{L+1} i}$ such as $j = \sqrt{-1}$ is the imaginary unit.

The initial Fourier coefficients are associated with lower frequency bands or the gradually evolving segments within a signal. On the other hand, the later coefficients correspond to higher frequency bands or the fast-changing segments. Typically, the primary Fourier coefficients are employed for signal characterization, facilitating low-pass filtering and signal smoothing.

The Inverse Discrete Fourier Transform (IDFT) is the reverse process of the DFT. It takes a sequence of complex numbers representing a signal in the frequency domain and reconstructs the original signal in the time domain. Given a sequence of Fourier coefficients $X = (X_1, X_2, ..., X_L)$, where each coefficient is a complex number with real and imaginary parts, the IDFT computes the corresponding time series $T = (t_1, t_2, ..., t_L)$ as $t_i = \frac{1}{L} \sum_{k=1}^{L} X_k e^{j2\pi \frac{ik}{L}}$.

3 Proposed Approach

3.1 Motivation

The optimization-based counterfactual explanation approaches described in the previous section have proven their success for tabular and image data. However, when used to explain time series models, the counterfactual instances tend to look unrealistic due to the nature of the perturbation strategy. By optimizing the loss function and modifying the original time series based on the gradients, the methods end up producing noisy explanations with high-frequency oscillations throughout their length. While this might be mitigated for tabular data by limiting the range of possible values for each feature to fit the original data, it is not possible to implement for time series due to their continuous and sequential nature, which makes it challenging to constrain the perturbations within reasonable bounds without disrupting the temporal coherence of the data. The use of autoencoders and class prototypes to guide the perturbations and ensure the respect of the original data manifold is another solution that works well for tabular and image data. However, as we show in Sect. 5.1, such methods still generate unrealistic time series explanations. In this work, we propose a solution that leverages the DFT to eliminate the oscillations problem for all optimization-based counterfactual explanation methods. We to contribute to the advancement of time series counterfactual explanation techniques by enabling a more effective application of the already established literature in the tabular and image domains.

3.2 Low-Pass Filtering

The first approach to eliminate the oscillations is to decompose the time series into its frequency components using DFT, apply a low-pass filter to remove frequency bands higher than those commonly found in the original dataset, and reconstruct the time series using the IDFT. This is a common approach used in signal smoothing; the only difference is that the threshold is determined based on the original data. In this section, we describe its introduction to the counterfactual optimization process.

Let $\boldsymbol{\mathcal{D}}_{m'} = \{\forall T_i | T_i \in C_{m'}\}$ be the set of original data instances from class $C_{m'}$. To generate a counterfactual explanation with target class $C_{m'}$, the first step is to determine the threshold value th above which frequencies will be filtered from $\boldsymbol{\mathcal{D}}_{m'}$. We describe this process in Algorithm 1. We define p as the percentile of frequencies to discard from $\boldsymbol{\mathcal{D}}_{m'}$. p is the only user-defined parameter. It controls how conservative the filtering will be, i.e. lower values of p result in the inclusion of frequency bands that are rarely present in $\boldsymbol{\mathcal{D}}_{m'}$. Algorithm 1 starts by computing the distribution of power values across different frequency components $\overline{PSD}$ as the average of all Power Spectrum Density (PSD) arrays of each $T \in \boldsymbol{\mathcal{D}}_{m'}$. At the same time, the algorithm keeps track of the frequency bands returned by `DFT(T)` in $freqs$ (lines 1–14). Next, the algorithm sorts the power values in $\overline{PSD}$ in ascending order to determine the value pow_p required for a frequency band to be considered significant (lines 15–17). Then,

Algorithm 1. `GET_THRESHOLD()`

Inputs: Dataset $\boldsymbol{\mathcal{D}}$, target class $C_{m'}$, percentile of frequencies to discard p.
Output: Low-pass filter threshold th.

Compute average PSD:
$PSDs \leftarrow$ empty list
$freqs \leftarrow$ empty set
for T in $\boldsymbol{\mathcal{D}}_{m'}$
    if $T \in C_{m'}$
        $PSD \leftarrow$ empty list
        for k, z in `DFT(T)` ▷ k is the frequency and z is its Fourier coefficient
            PSD.`append`($|z|^2$)
            $freqs$.`add`(k)
        $PSDs$.`append`(PSD)
$\overline{PSD} \leftarrow$ `mean`($PSDs$)
Sort $\overline{PSD}$ in ascending order and get power value at p:
$\overline{PSD} \leftarrow$ `sort`($\overline{PSD}$)
$pow_p \leftarrow \overline{PSD}[\frac{p}{100} \times$ `length`($\overline{PSD}$)$]$
Get significant frequency components:
$significant\_freqs \leftarrow$ empty list
for k in $freqs$
    if $\overline{PSD}[k] \geq pow_p$
        $significant\_freqs$.`append`(k)
Return highest frequency component:
$th \leftarrow$ `max`($significant\_freqs$)
return th

Algorithm 2. `FILTER()`

Inputs: Time series T, low-pass filter threshold th.
Output: Filtered time series.

Compute `DFT`(T) and select frequency bands lower than th with their corresponding Fourier coefficients:
$dft \leftarrow$ `DFT`(T)
$coeffs \leftarrow$ empty list
for k, z in `DFT(T)`
    if $k \leq th$
        $coeffs$.`append`((k, z))
Reconstruct the filtered time series using IDFT:
$T \leftarrow$ `IDFT`($coeffs$)
return T

the significant frequencies are selected depending on their values in $\overline{PSD}$ (lines 18–24). Finally, it returns th as the highest frequency component within the significant frequencies (lines 25–27).

Once th is computed, integrating the low-pass filtering procedure into the counterfactual search is a straightforward process: it is applied to the potential counterfactual time series $T + \delta$ (with δ being the gradient) at each optimization

step as shown in Algorithm 5. The low-pass filtering procedure is described in Algorithm 2. First, the time series is transformed to the frequency domain using DFT (line 2). Then, coefficients corresponding to frequency bands higher than th are discarded (lines 3–6), and the remaining ones are used to reconstruct the filtered time series using IDFT (lines 7–9).

3.3 Frequency Clipping

The previous approach can efficiently eliminate the oscillations. However, completely discarding some frequency bands results in information loss correlated with their power. Therefore, we propose another solution that preserves all frequency components within the signal. The idea is to limit the amplitude of each frequency component to values within their ranges in the original data. In Algorithm 3, we compute the ranges of real and imaginary components of the Fourier coefficients present in $\boldsymbol{\mathcal{D}}_{m'}$. Similar to p in the low-pass filtering approach, we define p_{low} and p_{up} as user-defined parameters that reduce the ranges of values to the most common ones in $\boldsymbol{\mathcal{D}}_{m'}$.

Algorithm 3 starts by computing the DFT of each time series $\boldsymbol{\mathcal{D}}_{m'}$ and storing the real and imaginary components of the resulting Fourier coefficients in arrays dedicated to each frequency band; these arrays represent the distributions of the amplitudes in the original data $\boldsymbol{\mathcal{D}}_{m'}$. At the same time, the algorithm keeps

Algorithm 3. `GET_FREQ_RANGES()`

Inputs: Dataset $\boldsymbol{\mathcal{D}}$, target class $C_{m'}$, lower and upper percentiles p_{low} and p_{up}.
Output: Lists of lower and upper bounds of real and imaginary ranges for each frequency r_{low}, r_{up}, i_{low}, and i_{up}.

1: Get all real and imaginary components of the Fourier coefficients:
2: $reals$, $imags$ ←arrays of size $L =$ `length(T)` ▷ each array element is an empty list
3: $freqs$ ← empty set
4: **for** T in $\boldsymbol{\mathcal{D}}_{m'}$
5: **if** $T \in C_{m'}$
6: PSD ← empty list
7: **for** k, z in `DFT(T)` ▷ k is the frequency and z is its Fourier coefficient
8: $real$, $imag$ ← $z[0], z[1]$
9: $reals[k]$.`append`$(real)$
10: $imags[k]$.`append`$(imag)$
11: $freqs$.`add`(k)
12: Sort each list in $reals$ and $imags$ in ascending order and return lower and upper bounds of ranges:
13: $r_{low}, r_{up}, i_{low}, i_{up}$ ← arrays of size $L =$ `length(T)` ▷ each array element is an empty list
14: **for** k in $freqs$
15: $real[k] \leftarrow$ `sort`$(real[k])$
16: $imag[k] \leftarrow$ `sort`$(imag[k])$
17: $r_{low}[k], r_{up}[k] \leftarrow real[k][\frac{p_{low}}{100} \times L], real[k][\frac{p_{up}}{100} \times L]$
18: $i_{low}[k], i_{up}[k] \leftarrow imag[k][\frac{p_{low}}{100} \times L], imag[k][\frac{p_{up}}{100} \times L]$
19: **return** $r_{low}, r_{up}, i_{low}, i_{up}$

track of the frequency bands returned by `DFT(T)` in $freqs$ (lines 1–14). Then, it sorts the distributions of amplitudes at each frequency band and selects the corresponding lower and upper boundaries based on p_{low} and p_{up}.

Similarly to the low-pass filtering approach, the frequency clipping is performed to the potential counterfactual explanation $T + \delta$ at each optimization step as shown in Algorithm 5. The clipping process is described in Algorithm 4. First, the time series is transformed to the frequency domain using DFT (line 3).

Algorithm 4. `CLIP()`

Inputs: Time series T, lists of lower and upper bounds of real and imaginary ranges for each frequency r_{low}, r_{up}, i_{low}, and i_{up}.
Output: Frequency clipped time series.

1: Compute `DFT`(T) and clip the real and imaginary components of the coefficients:
2: $coeffs \leftarrow$ empty list
3: $dft \leftarrow$ `DFT`(T)
4: **for** k, z in `DFT(T)`
5: $real$, $imag \leftarrow z[0], z[1]$
6: $real$, $imag \leftarrow$ `min`$(real, r_{low}[k])$, `min`$(imag, i_{low}[k])$
7: $real$, $imag \leftarrow$ `max`$(real, r_{up}[k])$, `min`$(imag, i_{up}[k])$
8: $z \leftarrow real + j.imag$ ▷ $j = \sqrt{-1}$
9: $coeffs$.`append`$((k, z))$
10: Reconstruct the clipped time series using IDFT:
11: $T \leftarrow$ `IDFT`$(coeffs)$
12: **return** T

Algorithm 5. `EXPLAIN()`

Inputs: Time series to explain T, black-box classifier prediction function f, target class $C_{m'}$, boolean values indicating whether the low-pass filter and/or clipping should be applied is_filter and is_clip.
Output: Counterfactual explanation T'.

1: Initialize counterfactual:
2: $T' \leftarrow T$
3: **for** number of iterations or until convergence
4: Minimize loss function:
5: $\delta \leftarrow$ `argmin` LOSS()
6: Get potential counterfactual:
7: $T' = T' + \delta$
8: **if** is_filter
9: $T' \leftarrow$ `FILTER`(T')
10: **if** is_clip
11: $T' \leftarrow$ `CLIP`(T')
12: Update counterfactual if conditions are met
13: **if** $f(T' + \delta) == C_{m'}$ & $d(T' + \delta, T) < d(T', T)$ ▷ $d()$ is a distance function
14: $T' = T' + \delta$
15: **return** T'

Then, each real and imaginary components of each Fourier coefficient are clipped into the ranges defined by r_{low}, r_{up}, i_{low}, and i_{up} (lines 5–7). Next, the clipped coefficients are reconstructed (line 9). Finally, the IDFT is used to reconstruct the final time series (line 12).

3.4 Time Complexity

Wachter's method is the least computationally demanding of the counterfactual algorithms introduced in Sect. 2. We show that our approach does not increase the complexity of its counterfactual search. Let n be the number of parameters in the black-box classification model. At each search iteration, the model prediction $f(T)$ is evaluated with a complexity of approximately $\mathcal{O}(nL)$, and its gradients are computed with the same complexity $\mathcal{O}(nL)$. The low-pass filtering and frequency clipping in Algorithm 2 and Algorithm 4 have a time complexity of $\mathcal{O}(L^2)$ corresponding to the DFT and IDFT steps. Since $n >> L$ for typical black-box models, introducing our approach to the counterfactual search does not add to its complexity. Furthermore, using Fast Fourier Transform (FFT) instead of DFT reduces the complexity of our approach to $\mathcal{O}(L\ logL)$.

4 Experimental Setup

To evaluate the addition of the low-pass filtering and clipping procedures to optimization-based counterfactual explanation methods, we experiment with the three methods described in Sect. 2, namely Wachter's method, CEM, and CEGP. We generate counterfactual explanations using all three methods as described in Algorithm 5: first with $is\_filter = is\_clip = False$ (denoted as: Orig.), then with $is\_filter = True$ and $is\_clip = False$ (denoted as: F.), with $is\_filter = False$ and $is\_clip = True$ (denoted as: C.), and finally with $is\_filter = is\_clip = True$ (denoted as: F.C.). For CEGP, we generate explanations with the help of an AE as described in Eq. 6 (CEGP-AE) and using k-d tree prototypes as described in Eq. 7 (CEGP-KD).

4.1 Black-Box Classification Models

We conduct the experiments using Residual Network (ResNet) [25] as the black-box classification model. ResNet is a classic deep learning architecture that achieved good results on the UCR time series classification archive in a recent benchmark [12]. We keep the default parameter configuration and train the model for 1500 epochs as suggested in the work by Fawaz et al. [12].

4.2 Datasets

We evaluate the counterfactual explanations generated from the UCR archive [7] of univariate time series datasets. First, we train ResNet models on the predefined training sets of all 117 fixed-length datasets from UCR (we discard the

11 variable-length ones to avoid preprocessing issues). Then, we test the models' classification performances on their respective test sets and select the datasets with f1-scores of 100% from the UCR archive to ensure that the counterfactual explanations are robust, i.e. that the label did not flip due to uncertainty on the classification model side. The seven selected datasets are described in Table 1.

Table 1. Datasets Descriptions

Dataset	Train Size	Test Size	Classes	Length	Data Type
BME	30	150	3	128	Simulated
Coffee	28	28	2	286	Spectro
GunPoint	150	150	2	150	Motion
Meat	60	60	3	448	Spectro
Plane	105	105	7	144	Sensor
Trace	100	100	4	275	Sensor
ShapeletSim	20	180	2	500	Simulated

4.3 Implementation Details

We used the implementations of Wachter's method, CEM, and CEGP provided in the Alibi library [13]. For TimeX, we used the public code provided by the authors [10]. Our additions to the code are made public in our project website[1]. We used the following parameter values for all datasets: $p = 5$, $p_{low} = 25$, and $p_{up} = 75$. Further tuning should result in better results.

As to the AE models used for CEM and CEGP, we used the architecture of Encoder originally proposed by Serrà et al. [22] for the encoder part. Encoder is a hybrid deep convolutional neural network inspired by FCN [25]. It consists of three convolutional blocks, each followed by instance normalization, PReLU activation, dropout, and max-pooling layers. The third convolutional block includes an attention mechanism. The only difference is that we replace the instance normalization with batch normalization layers. Then, we build a decoder model that mirrors the encoder structure using transposed convolutional layers and upsampling. Additionally, it ensures the output shape matches the input shape by adjusting the convolutional layer parameters. We train the models for 3000 epochs (with early stopping after 200 epochs) using the code and parameter configuration from [12] and tune the learning rate and dropout rate for each dataset. In addition, we select the optimization algorithm and loss function that performs best for each dataset. The code for training the AEs and the parameters used for each dataset are also available on the project website[2].

[1] https://sites.google.com/view/CFOPT/home
[2] https://sites.google.com/view/CFOTP/home

5 Experimental Results

In this section, we evaluate the counterfactual explanations generated using all methods. Due to space restrictions, we only show the average ranks in Table 2 and Table 3. The full results can be found on the project website. Similarly, we include figures for other datasets on the website.

5.1 Plausibility

To assess the improvements brought by the filtering and clipping approaches to each of the three methods, we compute the outlier factors (ratio of explanations detected as outliers compared to the original dataset) using isolation forests (IF) [18] and the one class support vector machine (OC-SVM) [21] method (on the raw time and the matrix profile [26] (OC-SVM MP) representations of the time series). Table 2 shows the average ranks of each approach for each evaluation measure. While the use of the filtering approach did not significantly improve the plausibility, the clipping produced better explanations in all cases.

Table 2. Outlier factor average ranks

Avg. Rank	IF				OC-SVM				OC-SVM MP			
	Orig.	F.	C.	F.C.	Orig.	F.	C.	F.C.	Orig.	F.	C.	F.C.
Wachter	2.67	3.17	1.67	**1.50**	3.14	2.86	**1.29**	**1.29**	3.14	3.14	**1.29**	1.43
CEM	3.14	2.57	1.14	**1.00**	3.14	2.86	**1.00**	1.14	2.86	3.00	1.14	**1.00**
CEGP-KD	2.86	3.29	**1.29**	1.43	3.29	3.43	**1.00**	**1.00**	2.71	3.14	**1.29**	**1.29**
CEGP-AE	2.86	2.71	**1.00**	1.14	2.86	2.71	**1.00**	**1.00**	3.29	2.71	**1.00**	**1.00**
TimeX	2.83	2.33	**1.67**	**1.67**	2.17	2.17	**1.33**	**1.33**	2.20	2.20	**1.40**	**1.40**

5.2 Proximity

To evaluate the explanations in terms of proximity, we show the average ranks of the L_1, L_2, and L_{inf} norms in Table 3. Optimization-based approaches typically do best when it comes to these measures, at the expense of plausibility. The results in Table 3 and on the website show that, overall, our approach does not significantly increase the distance between the explanation and the explained time series. In fact, some improvements might be observed, particularly when using an AE to guide the perturbations.

Table 3. Proximity average ranks

Avg. Rank	$\mathbf{L}_1$				$\mathbf{L}_2$				$\mathbf{L}_{inf}$			
	Orig.	F.	C.	F.C.	Orig.	F.	C.	F.C.	Orig.	F.	C.	F.C.
Wachter	**1.17**	1.83	3.50	3.50	**1.57**	2.00	3.00	3.43	**1.83**	**1.83**	3.00	3.33
CEM	2.43	3.14	**2.14**	2.29	**2.14**	2.57	2.71	2.57	2.57	2.71	2.43	**2.29**
CEGP-KD	**1.86**	2.29	2.57	3.29	**2.00**	2.14	2.86	3.00	**1.86**	2.29	2.86	3.00
CEGP-AE	2.43	**2.29**	2.57	2.71	2.43	**2.29**	2.57	2.71	2.57	2.86	**1.86**	2.71
TimeX	**1.00**	2.00	3.33	3.67	**1.17**	1.83	3.33	3.67	**1.00**	2.00	3.50	3.50

5.3 Visual Plausibility

We evaluate the counterfactuals by visually comparing them to the original time series. We select the Coffee dataset which contains spectrographs of two coffee bean types: Arabica and Robusta. Each plot in Fig. 1, 2 and 3 contains all time series from the corresponding class. The original spectrographs in Fig. 1. show the existence of a main characteristic feature that differentiates Arabica beans from Robusta beans, located in the rectangular boxes. For the Arabica class, the three peaks are aligned along a descending diagonal line, whereas for the Robusta class, the three peaks form a reversed V shape with the highest one in the middle.

Figure 2 and Fig. 3 display the counterfactual explanations generated using the state-of-the-art methods before and after introducing our clipping approach. Without clipping, all explanations were affected by the high-frequency noise issue, which correlates with the high outlier factors. The use of the clipping step eliminates the oscillations, resulting in time series similar to the original ones. Moreover, the clipping produced the characteristic segments of both Arabica and Robusta instances, whereas the original methods did not.

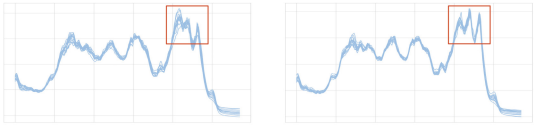


Fig. 1. Original Coffee time series.

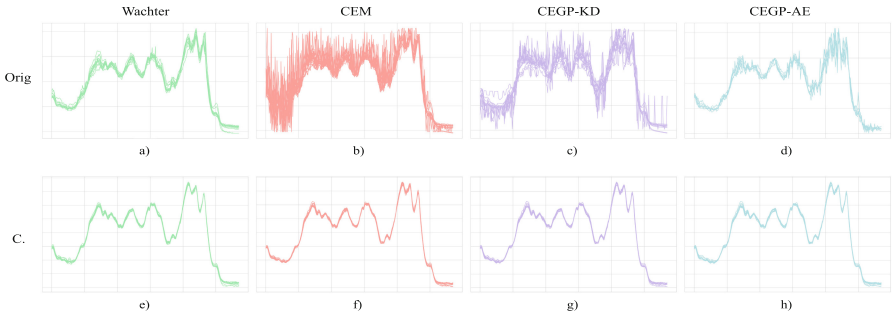


Fig. 2. Coffee (Arabica) counterfactual explanations generated with Wachter's method (a and e), CEM (b and f), CEGP-KD (c and g), and CEGP-AE (d and h).

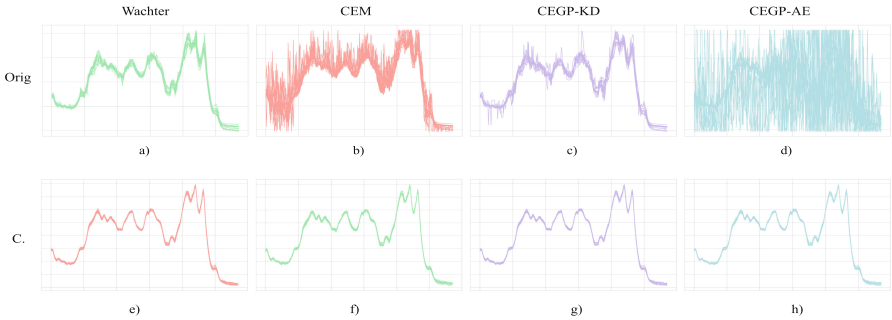


Fig. 3. Coffee (Robusta) counterfactual explanations generated with Wachter's method (a and e), CEM (b and f), CEGP-KD (c and g), and CEGP-AE (d and h).

5.4 PCA

We use Principal Component Analysis (PCA) to visualize the distributions of the counterfactual explanation samples by projecting them onto the 2-D space defined by the first two principal components of the training set. In Fig. 4, each class is represented by a combination of color and shape, with the unfilled shapes representing original training data samples and the filled shapes representing the counterfactual data. Comparing the plots in the first row to those in the second row clearly demonstrates that the explanations generated using our clipping approach respect the class distributions of the target class, whereas the ones produced by the original methods still belong to their original class data manifold.

5.5 Sensitivity Analysis

The experiments we presented above were performed with $p = 5.0$, $p_{low} = 25$, and $p_{up} = 75$. In this section, we explore how changing these values affects the resulting explanations on the Coffee dataset. We define q such that $p_{low} = 100 - q$

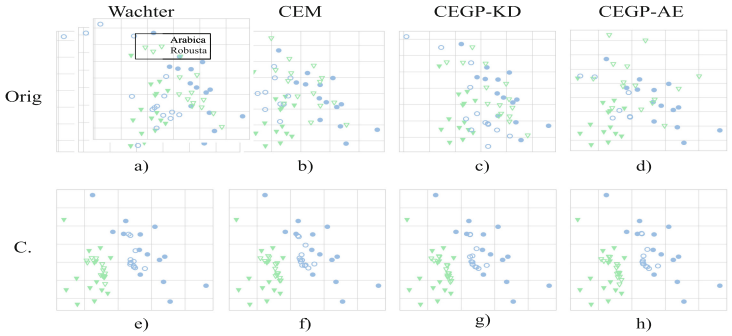


Fig. 4. First two PCA components of Coffee explanations generated with Wachter's method (a and e), CEM (b and f), CEGP-KD (c and g), and CEGP-AE (d and h).

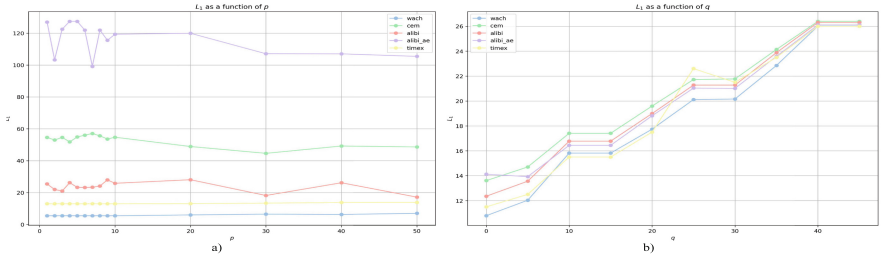


Fig. 5. L_1 sensitivity to p (a) and to q (b).

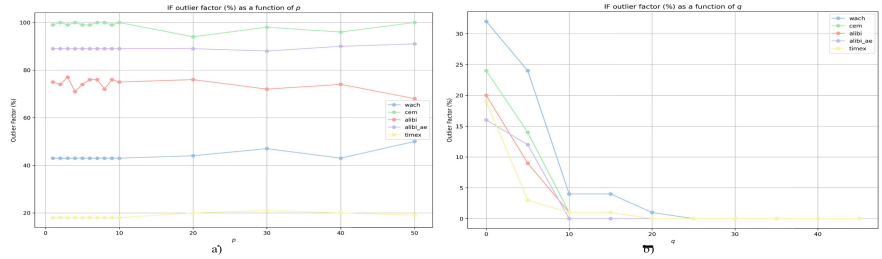


Fig. 6. IF sensitivity to p (a) and to q (b).

and $p_{up} = q$. First, we set $q = 25$ and generate counterfactual explanations with different values of p. Figure 5 shows the L_1 proximity and IF outlier factor results as a function of p for all algorithms with the addition of the smoothing step (F.). Except for a slight decrease in L_1 values for the CEM counterfactuals as p increases, there is no significant change when varying this parameter. Then, we set $p = 5.0$ and experiment with different values of q. Figure 6 shows the L_1 proximity and IF outlier factor results. On one hand, we can see a clear increase

in L_1 values as p increases. On the other hand, the outlier factor decreases as p increases. Therefore, choosing an appropriate value for p is a crucial design decision that depends on which criterion is more important for the given scenario.

6 Conclusion

In conclusion, our work presents a significant step forward in addressing the challenges associated with applying optimization-based counterfactual explanation methods to time series data. By introducing a novel approach that leverages DFT to mitigate unrealistic high-frequency oscillations in generated explanations, we have significantly enhanced the plausibility of generated time series counterfactuals. Our experiments on datasets from the UCR archive underscore the effectiveness of our approach in improving the quality of explanations generated by state-of-the-art methods. Through this contribution, we aim to bridge the gap between counterfactual explanation techniques developed for tabular and image data domains and their application to time series.

Acknowledgments. This project has been supported in part by funding from CISE and GEO Directorates under NSF awards #2204363, #2240022, #2301397, and #2305781.

References

1. Ates, E., Aksar, B., Leung, V.J., Coskun, A.K.: Counterfactual explanations for multivariate time series. In: International Conference on Applied Artificial Intelligence (ICAPAI), pp. 1–8 (2021)
2. Bahri, O., Boubrahimi, S.F., Hamdi, S.M.: Shapelet-based counterfactual explanations for multivariate time series (2022). arXiv:2208.10462
3. Bahri, O., Li, P., Filali Boubrahimi, S., Hamdi, S.M.: Temporal rule-based counterfactual explanations for multivariate time series. In: 2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA), Nassau, Bahamas, pp. 1244–1249 (2022)
4. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imag. Sci. **2**(1), 183–202 (2009)
5. Bentley, J.L.: Multidimensional binary search trees used for associative searching. Commun. ACM **18**, 509–517 (1975)
6. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. ACM SIGMOD Rec. **29**(2), 93–104 (2000)
7. Dau, H.A., et al.: The UCR Time Series Archive (2018)
8. Delaney, E., Greene, D., Keane, M.T.: Instance-based counterfactual explanations for time series classification. In: Sánchez-Ruiz, A.A., Floyd, M.W. (eds.) ICCBR 2021. LNCS (LNAI), vol. 12877, pp. 32–47. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86957-1_3
9. Dhurandhar, A., et al.: Explanations based on the missing: towards contrastive explanations with pertinent negatives. In: Advances in Neural Information Processing Systems, pp. 592–603 (2018)

10. Filali Boubrahimi, S., Hamdi, S.M.: On the mining of time series data counterfactual explanations using barycenters. In: International Conference on Information and Knowledge Management. Proceedings, pp. 3943–3947. Association for Computing Machinery, New York (2022)
11. Guidotti, R.: Counterfactual explanations and how to find them: literature review and benchmarking. Data Min. Knowl. Disc. **38**, 1–55 (2022)
12. Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., Muller, P.A.: Deep learning for time series classification: a review. Data Min. Knowl. Disc. **33**(4), 917–963 (2019)
13. Klaise, J., Looveren, A.V., Vacanti, G., Coca, A.: Alibi explain: algorithms for explaining machine learning models. J. Mach. Learn. Res. **22**(181), 1–7 (2021)
14. Labaien, J., Zugasti, E., De Carlos, X.: Contrastive explanations for a deep learning model on time-series data. In: Song, M., Song, I.-Y., Kotsis, G., Tjoa, A.M., Khalil, I. (eds.) DaWaK 2020. LNCS, vol. 12393, pp. 235–244. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59065-9_19
15. Li, P., Bahri, O., Boubrahimi, S.F., Hamdi, S.M.: SG-CF: shapelet-guided counterfactual explanation for time series classification, pp. 1564–1569. Institute of Electrical and Electronics Engineers Inc. (2022)
16. Li, P., Bahri, O., Boubrahimi, S.F., Hamdi, S.M.: CELS: counterfactual explanations for time series data via learned saliency maps. In: Proceedings - 2023 IEEE International Conference on Big Data, BigData 2023, pp. 718–727 (2023)
17. Li, P., Boubrahimi, S.F., Hamdi, S.M.: Motif-guided time series counterfactual explanations. In: Rousseau, J.J., Kapralos, B. (eds.) ICPR 2022. LNCS, vol. 13645, pp. 203–215. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-37731-0_16
18. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation forest. In: Proceedings - IEEE International Conference on Data Mining, ICDM, pp. 413–422 (2008)
19. Mothilal, R.K., Sharma, A., Tan, C.: Explaining machine learning classifiers through diverse counterfactual explanations. In: Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency, Barcelona, Spain (2020)
20. Rojat, T., Puget, R., Filliat, D., Ser, J.D., Gelin, R., Díaz-Rodríguez, N.: Explainable artificial intelligence (XAI) on timeseries data: a survey (2021)
21. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural Comput. **13**(7), 1443–1471 (2001)
22. Serrà, J., Pascual, S., Karatzoglou, A.: Towards a universal neural network encoder for time series. In: International Conference of the Catalan Association for Artificial Intelligence, vol. 308, pp. 120–129 (2018)
23. Van Looveren, A., Klaise, J.: Interpretable counterfactual explanations guided by prototypes. In: Oliver, N., Pérez-Cruz, F., Kramer, S., Read, J., Lozano, J.A. (eds.) ECML PKDD 2021. LNCS (LNAI), vol. 12976, pp. 650–665. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86520-7_40
24. Wachter, S., Mittelstadt, B., Russell, C.: Counterfactual explanations without opening the black box: automated decisions and the GDPR. Harvard J. Law Technol. **31**(2) (2018)
25. Wang, Z., Yan, W., Oates, T.: Time series classification from scratch with deep neural networks: a strong baseline. In: Proceedings of the International Joint Conference on Neural Networks, pp. 1578–1585 (2017pg

26. Yeh, C.C.M., et al.: Matrix profile i: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets, pp. 1317–1322 (2017)
27. Zou, H., Hastie, T.: Regularization and variable selection via the elastic net. J. R. Statist. Soc. B **67**, 301–320 (2005)

Improving Adaptive Runoff Forecasts in Data-Scarce Watersheds Through Personalized Federated Learning

Zaipeng Xie[1,2(✉)], Xiangqin Zhang[2], Yunfei Wang[2], Xuanyao Jie[2], Wenhao Fang[2], and Yanping Cai[1,2]

[1] Key Laboratory of Water Big Data Technology of Ministry of Water Resources, Hohai University, Nanjing, China
yanpingcai@hhu.edu.cn

[2] College of Computer Science and Software Engineering, Hohai University, Nanjing, China
{zaipengxie,zhangxiangqin,yunfeiwang,xuanyaojie,wenhaofang}@hhu.edu.cn

Abstract. Runoff forecasting plays a crucial role in water resource management and flood mitigation, but it often faces significant challenges due to data deficiency and decentralized datasets. Inadequate hydrological data in many watersheds hinders the development of accurate prediction models, while organizational barriers and concerns about data privacy lead to information silos, preventing effective collaborative modeling. To address these issues, we propose FedHydroDSW, a novel personalized federated learning framework for runoff forecasting in data-scarce watersheds. Our approach enables decentralized model training across multiple organizations without requiring direct data sharing, thus preserving data privacy. FedHydroDSW incorporates unique model similarity metrics and parameter adaptation methods, utilizing insights from data-rich areas to refine forecasts in data-scarce regions. Experiments on the CAMELS dataset show that FedHydroDSW significantly outperforms standalone and conventional federated learning methods, with an average 30.1% increase in NSE, and reductions of 28% in RAE and 26.6% in RMSE across multiple data-scarce basins. By enabling secure cross-silo learning, our FedHydroDSW strategy paves the way toward more equitable runoff forecasting globally. The customized training and pattern recognition techniques represent major progress in deploying federated intelligence for impactful hydrological predictions. Our code is publicly available at https://github.com/xyjie37/FedHydroDSW.

Keywords: Runoff forecasting · Federated learning · Data-scarce watersheds · Personalized modeling · Hydrological information silos · Privacy-preserving collaboration

1 Introduction

Accurate runoff forecasting [19,28] is critical for effective water resource management and flood mitigation, as it enables informed decision-making and proactive

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 180–198, 2025.
https://doi.org/10.1007/978-3-031-78183-4_12

measures to minimize the socioeconomic and environmental impacts of hydrological extremes. Traditional physical-driven runoff prediction methods [21] typically rely on mathematical models describing rainfall-runoff relations within specific catchment areas, drainage basins, or watersheds. Recently, data-driven-based runoff prediction methods [13], which are based on statistical relationships between input and output data, have gained significant attention. Particularly, neural network-based methods [8–10] have emerged as powerful tools for runoff forecasting. These approaches possess remarkable fitting capabilities, enabling them to effectively capture complex relationships between input and output variables without explicit knowledge of specific hydrological processes.

Neural network-based models [18] are increasingly employed for hydrological predictions, with their effectiveness heavily reliant on both the quality and volume of available data. Studies [4,11] have indicated that models gain in accuracy when trained on data from multiple basins compared to those relying on single-basin datasets. However, challenges arise due to the diverse nature of hydrological data [3] and the commonality of data silos [7], which are created by security and privacy concerns that limit data sharing. These barriers significantly impede the progression of sophisticated neural network models for runoff predictions, necessitating extensive data sharing [24]. Furthermore, many global watersheds, particularly those in less developed regions, are characterized by an acute lack of hydrological data, stemming from inadequate monitoring infrastructure and limited data collection resources [22]. This issue is compounded by information silos, which isolate the scant data within specific entities, hindering shared access due to data privacy, regulatory restrictions, or a lack of collaboration incentives. Addressing the scarcity of data is paramount for refining models but also for facilitating fair and sustainable management of water resources, especially given the uncertainties associated with global climate change.

Federated learning has recently emerged as a promising approach [2] for enabling collaborative model development, allowing multiple entities, such as watershed management organizations across different states, to collaboratively develop models without the need to share sensitive or proprietary data directly. This approach effectively enables the aggregation of hydrological models from varied sources while safeguarding privacy and security. Despite FL's success in diverse domains [20,27], its application in hydrological runoff prediction has been limited, raising concerns about the capability of FL-derived neural network models to generalize effectively across different hydrological scenarios.

This study introduces FedHydroDSW, a self-adaptive personalized federated learning framework designed to address the challenges associated with data-scarce watersheds and hydrological information silos. The distinction between data-rich and data-scarce watersheds is: data-rich basins have extensive years of hydro-meteorological datasets, while data-scarce basins have limited years of data. This categorization allows us to leverage insights from data-rich areas to refine forecasts in data-scarce regions, enhancing accuracy and reliability. Our approach enables decentralized model training across organizations without direct data sharing, preserving privacy. FedHydroDSW incorporates unique

model similarity metrics and parameter adaptation methods. We evaluate FedHydroDSW using the CAMELS hydrological dataset and demonstrate significant improvements in prediction accuracy compared to local modeling and conventional federated learning approaches, particularly for watersheds with limited data availability. The main contributions of this paper are:

- We introduce FedHydroDSW, a novel personalized federated learning framework for runoff forecasting in data-scarce watersheds. This method enables decentralized model training across multiple organizations without the need for direct data sharing, preserving data privacy.
- FedHydroDSW incorporates unique model similarity metrics and parameter adaptation methods, utilizing insights from data-rich areas to refine forecasts in data-scarce regions. This addresses the challenges of data scarcity and information silos in hydrological forecasting.
- Experiments on the CAMELS dataset show that FedHydroDSW significantly outperforms standalone and conventional federated learning methods, with an average 30.1% increase in NSE, and reductions of 28% in RAE and 26.6% in RMSE across multiple data-scarce basins. This validation not only proves the effectiveness of the proposed method but also underscores its potential as a versatile tool in hydrological modeling and water resource management.

2 Related Work

Recent advances in runoff prediction leverage neural networks for their feature learning and nonlinear mapping capabilities. Notable developments include Backpropagation (BP) neural networks [25], Recurrent Neural Networks (RNN) [28], Long Short-Term Memory (LSTM) [11], and Gated Recurrent Units (GRU) [8,26]. These models have significantly enhanced runoff forecasting accuracy by utilizing complex hydrological data. Applications, such as the GRU-based model [26], highlight the potential of neural networks to improve long-term runoff predictions. Ding et al. [6] introduced an interpretable spatio-temporal attention-based LSTM method that shows promise in flood forecasting. Moishin et al. [17] combined Convolutional Neural Networks (CNNs) with LSTM to predict floods using daily rainfall data. However, the effectiveness of these neural network models heavily relies on the availability of large, high-quality hydrological datasets. Variations in data size and quality can result in overfitting, underfitting, and compromised generalization capabilities.

Access to diverse data sources significantly improves runoff prediction models by providing a broad spectrum of environmental, climatic, and geographical information. This variety leads to higher model accuracy, improved generalization across different regions, and better adaptability to evolving conditions. Kratzert et al. [11] demonstrated the advantages of integrating multi-basin data by training an LSTM neural network for runoff prediction, showing superior performance through data fusion across various basins. Similarly, Han et al. [9] developed a rainfall-runoff model using data from multiple sub-watersheds,

proving its efficiency over traditional physical-based models through centralized training. Nonetheless, the issue of data silos remains largely unaddressed, restricting data sharing and consequently limiting the models' ability to generalize. This limitation affects prediction accuracy and the development of the models. Moreover, while data isolation in distributed data systems offers opportunities for collaborative learning, it also presents significant challenges, such as data privacy and integration issues. Overcoming these information silos is imperative to unlock the full potential of neural networks in hydrological forecasting.

The persistence of information silos makes federated learning (FL) an essential solution. Rasha et al. [2] propose that FL provides a decentralized framework that enables collaborative learning while maintaining data privacy. This approach is particularly beneficial in fields where data diversity and continuous model improvement are crucial. For instance, Zhang et al. [27] utilized FL for traffic flow prediction without data sharing, and Savi et al. [20] applied it to enhance short-term energy consumption predictions. Chen et al. [5] introduced FedLGAN, combining FL, GANs, and LSTM for detecting and fixing anomalies in hydrological data. These examples underscore federated learning's ability to mitigate the limitations caused by data isolation, enhancing the evolution and adaptability of neural network methods in data-sensitive domains. However, applying federated learning in hydrological prediction remains challenging.

Hydrological data scarcity, especially in less developed areas, arises from infrastructural, financial, geographical, and logistical challenges [12,14]. Limited resources, geographical barriers, and poor coordination between agencies, along with restrictive data-sharing policies, hinder the establishment and maintenance of monitoring systems. Environmental changes and natural disasters further exacerbate data collection efforts, worsened by technological disparities and a shortage of skilled professionals. Researchers have proposed various methods to improve runoff prediction in data-deficient basins [11,14,16,30]. For instance, Kratzert et al. [11] and Ma et al. [14] suggested leveraging regional data to develop and fine-tune LSTM models for specific data-scarce basins. However, these methods often overlook the issues of information silos and data sharing, assuming that models for data-rich basins can be directly adapted to data-scarce environments, which may not always hold true due to unique basin characteristics. Thus, effectively predicting runoff in data-scarce basins remains a significant challenge, and traditional federated learning approaches have yet to provide a comprehensive solution to these complex issues.

3 Methodology

3.1 Problem Formulation

We address the challenge of runoff prediction in a scenario where hydrological data is decentralized, leading to uneven data distribution across different geographic regions. In this setting, certain watersheds suffer from data scarcity due to technological constraints, hindering traditional centralized modeling approaches. These traditional methods also face additional challenges such

as privacy, regulatory, and organizational barriers, complicating the aggregation and collaborative training of models with data from various regions. To counter these issues, we propose a federated training framework. This innovative approach involves a network of nodes, each representing different basins. By integrating data from these diverse nodes without actual data transfer, the framework facilitates a rich learning environment that benefits from the unique characteristics and data available in each basin. Specifically, we categorize basins into two groups: data-rich basins $\mathcal{R} = \{\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_r\}$ and data-scarce basins $\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_s\}$. In our framework, r and s indicate the numbers of data-rich and data-scarce basins, respectively, with $r \gg s$. Data-rich basins are characterized by extensive hydro-meteorological datasets Φ_i^{Train}, unlike their data-scarce counterparts, which possess limited datasets Ψ_j^{Train}. Our objective is to enhance the accuracy of runoff predictions across all basins by minimizing the local model prediction loss. This loss is calculated using the following formulas for each basin type, aiming to reduce the discrepancies between predicted and observed runoff values:

$$\begin{cases} \mathcal{L}(w_i) = \dfrac{1}{|\Phi_i^{\text{Train}}|} \displaystyle\sum_{\phi_i, \chi_i \in \Phi_i^{\text{Train}}} |f(w_i; \phi_i) - \chi_i|^2, \\ \mathcal{L}(u_j) = \dfrac{1}{|\Psi_j^{\text{Train}}|} \displaystyle\sum_{\psi_j, \zeta_j \in \Psi_j^{\text{Train}}} |f(u_j; \psi_j) - \zeta_j|^2, \end{cases} \tag{1}$$

Here, $|\Phi_i^{\text{Train}}|$ and $|\Psi_j^{\text{Train}}|$ denote the sizes of the training sets for the data-rich basins $\mathcal{R}_i$ and data-scarce basins $\mathcal{S}_j$, respectively. The loss functions $\mathcal{L}(w_i)$ and $\mathcal{L}(u_j)$ quantify the prediction accuracy for each basin type, where w_i and u_j represent the model parameters. Input features and observed runoff values are denoted by (ϕ_i, χ_i) for data-rich basins and (ψ_j, ζ_j) for data-scarce basins. The model's predictions are given by $f(w_i; \phi_i)$ and $f(u_j; \psi_j)$ for the respective basin types. By leveraging federated learning, we aim to find optimal, personalized models for each basin that utilize knowledge from data-rich environments $\mathcal{R}$ to improve predictions in data-scarce ones $\mathcal{S}$. This approach seeks to mitigate the challenges posed by data scarcity and technological constraints while preserving data privacy by avoiding direct data sharing between basins.

3.2 FedHydroDSW: Federated Runoff Forecast for Data-Scarce Watershed

We introduce FedHydroDSW, a novel federated learning approach for intricate temporal hydrological data analysis in runoff prediction. This method promotes privacy-preserving model training across varied hydrological conditions by enabling indirect data sharing. FedHydroDSW excels in assimilating insights across regions, particularly enhancing the capabilities of data-scarce areas through the knowledge derived from data-abundant counterparts. It features a dynamic model adjustment mechanism during training to refine predictive performance by harnessing inter-basin similarities, thus improving accuracy

and mitigating convergence challenges. An illustrative diagram of the FedHydroDSW methodology is provided in Fig. 1.

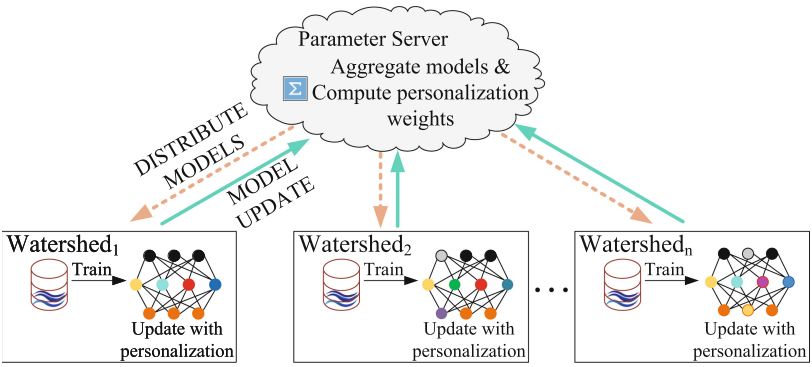


Fig. 1. Overview of the FedHydroDSW algorithm for collaborative runoff prediction across multiple watersheds. The Parameter Server coordinates model exchange, aggregation, and personalization, while each watershed performs local training and model updating in an iterative process.

It is worth noting that we use a uniform model structure across basins for simplicity, scalability, and efficient aggregation. While heterogeneous models tailored to each node's data and computational capacity could enhance performance, they pose challenges in aggregation, fairness, and consistency. Addressing these issues may require techniques like federated distillation and meta-learning. Our current work focuses on uniform models, but we plan to explore heterogeneous structures in future research.

Runoff Prediction Neural Model: We optimize our model for accurate runoff predictions in data-scarce watersheds by balancing complexity with simplicity to avoid overfitting and underfitting. Our approach utilizes a local BiLSTM model [23], known for managing complex temporal data, with integrated dropout techniques for effective regularization, enhancing learning without overfitting, and maintaining long-term dependencies. The model setup includes a learning rate of 0.001, a batch size of 256 samples, time steps of 30, and five features per sample. It comprises an input layer, two LSTM layers with a 20% dropout rate to prevent overfitting, and a fully connected output layer producing one prediction value per batch. The use of the Tanh activation function adds nonlinearity, enabling the model to capture complex data patterns, culminating in reliable runoff predictions.

Self-adaptive Personalized Local Training: Our approach innovatively combines joint training methodologies across both data-rich and data-scarce

hydrological basins. By exploiting the inherent similarities in hydrological dynamics across diverse basins, our method enables data-scarce basins to leverage insights from their data-rich counterparts. This symbiotic relationship not only augments the predictive accuracy of runoff models in data-scarce regions but also enhances the models' out-of-distribution prediction capabilities. Conversely, data-rich basins benefit from this methodology by refining their models through the diversified training data pool. Distinct from traditional techniques, which primarily focus on training with data-rich basins before making adjustments for data-scarce ones, our method emphasizes the mutual adaptability between the two during the federated training phase.

Building on this foundation, we propose a personalized federated learning approach, inspired by [29], tailored to the unique challenges of hydrological prediction across basins with varying data availability. Unlike conventional federated learning, which often fails to exploit similarities between clients fully, our method ensures that basins are exposed only to models from other basins with relevant similarities during each update round. This selective sharing mechanism mitigates the risk of model convergence towards suboptimal solutions. Given the constraints of federated learning, direct computation of data similarity between basins is not feasible. We assess similarity through the adaptability of models across different basins, allowing for a tailored and efficient update process for each basin, particularly focusing on those with scarce data. To ensure a comprehensive evaluation of model performance, we also monitor Mean Absolute Error (MAE) throughout the training, in addition to validation loss. We calculate these metrics at the end of each local training round for both the training and validation sets. This multi-metric approach provides a more robust assessment of model adaptability and performance, especially in the early stages of training.

The procedure begins with basins uploading their model parameters (updated in round t) to the parameter server (PS). For data-rich basins, these parameters are denoted as w_{set}^t. For data-scarce basins, the parameters are denoted as u_{set}^t. The PS then redistributes these parameters among the computing nodes for the next training round, spearheading the global training initiative. For a data-scarce basin $\mathcal{S}_j$, this means receiving a bespoke set of models ($w_{\text{set}}^t \cup u_{\text{set}}^t$) tailored for personalized updates. The update process in basin $\mathcal{S}_j$ unfolds in three steps:

Step 1: $\mathcal{S}_j$ evaluates the received models' adaptability to the specific runoff prediction task at hand. Models that demonstrate lower adaptability compared to the current model of $\mathcal{S}_j$ are excluded from further consideration. Due to the inherent challenges in directly comparing complex models, we shift our focus towards assessing the performance of these models on the local dataset. This dataset is divided into training and validation sets, denoted as Ψ_j^{Train} and Ψ_j^{Val}, respectively. The adaptability is quantified by calculating both the validation loss and the MAE metric:

$$\mathcal{L}(v_k) = \frac{1}{|\Psi_j^{\text{Val}}|} \sum_{\psi_j, \zeta_j \in \Psi_j^{\text{Val}}} |f(v_k; \psi_j) - \zeta_j|^2, \tag{2}$$

$$\mathrm{MAE}(v_k) = \frac{1}{|\Psi_j^{\mathrm{Val}}|} \sum_{\psi_j, \zeta_j \in \Psi_j^{\mathrm{Val}}} |f(v_k; \psi_j) - \zeta_j|, \tag{3}$$

where v_k represents parameters from any basin being evaluated on $\mathcal{S}_j$, $f(v_k; \psi_j)$ is the predicted runoff, and ζ_j is the observed runoff.

To address potential instability in early training stages and balance the emphasis between immediate performance and long-term improvement, we use a composite loss $\mathcal{L}_C$ based on the validation loss ($\mathcal{L}$) and the MAE metric (MAE):

$$\mathcal{L}_C(v_k) = \gamma \cdot \mathcal{L}(v_k) + (1 - \gamma) \cdot \mathrm{MAE}(v_k) \tag{4}$$

where the dynamic weighting factor γ is adjusted linearly based on the training progress:

$$\gamma = 0.5 + 0.5 \cdot \frac{\text{current\_round}}{\text{total\_rounds}} \tag{5}$$

This ensures equal weighting between MAE and validation loss at the start of training ($\gamma = 0.5$) and gradually increases emphasis on validation loss, reaching $\gamma = 1.0$ at the final round. This approach allows the model to benefit from both metrics throughout the training while simplifying the hyperparameter tuning. We consider a model to be adapting well if its $\mathcal{L}_C(v_k)$ shows a consistent decrease over multiple evaluation periods. This dynamic weighting approach allows us to balance between multiple model performance indicators early in training while gradually shifting focus toward generalization capability as training advances. Models with lower $\mathcal{L}_C$ are considered more adaptable and are retained for further consideration in the subsequent steps.

Step 2: $\mathcal{S}_j$ assigns weights to models reflecting their adaptability for enhancing runoff prediction accuracy. The process entails comparing the composite losses of received models with the local model's previous round composite loss, $\mathcal{L}_C(u_j^{t-1})$. This comparison aims to determine each model's relative improvement or decline in performance, thereby assessing their contribution to the personalized update for this round. Utilizing the local model from the previous round, u_j^{t-1}, as a benchmark facilitates a clear evaluation of each external model's impact without overshadowing the local model's ongoing contribution. The adaptability weight for each received model is calculated using the formula:

$$a_k = \frac{\mathcal{L}_C(u_j^{t-1}) - \mathcal{L}_C(v_k^t)}{\|v_k^t - u_j^{t-1}\|} \tag{6}$$

Here, $\|\cdot\|$ denotes the $L2$ norm. A positive a_k value signals that a model performs better on the local validation set than $\mathcal{S}_j$'s current model, indicating its potential utility for the prediction task. We consider a model to be adapting well if its $\mathcal{L}_C$ shows a consistent decrease over multiple rounds. Conversely, a negative a_k suggests incompatibility, prompting its exclusion by setting a_k to zero. To ensure equity among models, weights $a_k > 0$ are normalized, yielding final weights:

$$a_k^* = \max(a_k, 0) / \sum_{n=1}^{k} \max(a_k, 0) \tag{7}$$

Subsequently, the local model's weight is derived from comparing its performance between rounds $t-1$ and t through:

$$a_j = \frac{\mathcal{L}_C(u_j^{t-1}) - \mathcal{L}_C(u_j^t)}{|u_j^t - u_j^{t-1}|} \tag{8}$$

This approach fine-tunes the model's accuracy and provides a criterion for ceasing training early based on successive improvements in model performance.

Step 3: The focus shifts to constructing and applying the weighted model for updates on the local training set, following the groundwork laid in **Step 1** and **2**. Here, $\mathcal{S}_j$ integrates the normalized weights, acquired for both the received models and its own model u_j^t, represented collectively as $a = \{a_1^*, a_2^*, \ldots, a_k^*, a_j\}$. The composite model for local updates, denoted as $\widehat{u}_j^t$, is synthesized by

$$\widehat{u}_j^t = a_1^* v_1^t + a_2^* v_2^t + \ldots + a_k^* v_k^t + a_j u_j^t, \tag{9}$$

where each term represents a component model, weighted appropriately according to its relevance and performance as determined in the preceding steps.

Upon determining the composite weights, $\mathcal{S}_j$ proceeds to update its model on the local training dataset, Ψ_j^{Train}, to generate new model parameters for the ensuing round, denoted u_j^{t+1}. This approach deviates from standard federated learning practices by basing updates on the aggregated insights from $\widehat{u}_j^t$, instead of merely updating from u_j^t. This strategy ensures that the update leverages broader knowledge, incorporating beneficial features and learned parameters from other models while maintaining a focus on local data relevance.

Subsequently, $\mathcal{S}_j$ uploads the updated parameters u_j^{t+1} to the PS, completing the cycle. This iterative process, encapsulating **Steps 1** through **3**, continues until the predetermined number of global communication rounds. The sequential operations executed by basin $\mathcal{S}_j$ are methodically outlined in pseudocode format to facilitate understanding and replication, as exemplified in Algorithm 1. Note that we employ the Adam algorithm for optimization and parameter updates are computed by modifying the corrected moments.

To sum up, our FedHydroDSW framework addresses several critical challenges in hydrological modeling for data-scarce watersheds. The personalized federated learning approach enables knowledge transfer from data-rich to data-scarce basins without direct data sharing, preserving privacy while mitigating data scarcity. Our dynamic weighting strategy adjusts γ dynamically during training to address early-stage model instability. Initially balancing between MAE and validation loss, it gradually emphasizes generalization as training progresses, preventing overfitting to limited local data in scarce basins. The composite loss $\mathcal{L}_C$ combines MAE for overall model optimization with validation loss for point-wise accuracy. Finally, our adaptive model aggregation method ensures that each basin's model remains tailored to its unique hydrological characteristics while benefiting from collective learning, addressing the challenge of heterogeneity across watersheds. Together, these components synergistically work to improve prediction accuracy and model adaptability in data-scarce watersheds.

Algorithm 1: The training of FedHydroDSW on participating agents

Input : Set of models $v_{\text{set}}^t = \{v_1^t, v_2^t, \ldots, v_k^t\}$, number of local training rounds E, learning rate α, total rounds T
Output: Updated model u_j^{t+1}

for *iteration* t *from* 1 **to** E **do**
  Get $\gamma = 0.5 + 0.5 \cdot (t/T)$;
  Get composite losses $\mathcal{L}_{C,\text{set}} = \{\mathcal{L}_C(v_1^t), \mathcal{L}_C(v_2^t), \ldots, \mathcal{L}_C(v_k^t)\}$ using Ψ_j^{Val};
  Compute and normalize the weights to get $\{a_1^*, a_2^*, ..., a_k^*, a_j\}$ using $\mathcal{L}_C(v_k)$;
  Construct the weighted model $\widehat{u}_j^t$ for local updates: $\widehat{u}_j^t = a_1^* v_1^t + a_2^* v_2^t + \ldots + a_k^* v_k^t + a_j u_j^t$;
  Update the local model based on the training set Ψ_j^{Train} and $\widehat{u}_j^t$ using the ADAM optimization: $u_j^{t+1} = \widehat{u}_j^t - \alpha \cdot \nabla \mathcal{L}_C(\widehat{u}_j^t)$;
return *Updated model* u_j^{t+1}

4 Empirical Analysis

4.1 Dataset and Preprocessing

Our study employs the CAMELS dataset [1], encompassing over 35 years of daily meteorological data and observed discharge records from 1980, distributed across 18 hydrological units (HUCs) as categorized by the U.S. Geological Survey. Preprocessing of the dataset is crucial for ensuring analytical accuracy and involves a rigorous selection of vital climatological variables. This phase includes the selection of key external data features (precipitation, short wave radiation, maximum and minimum temperatures, atmospheric pressure), normalization ($x' = (x - x_{mean})/x_{std}$) to achieve uniformity, determination of an appropriate time step length, and reshaping of data into a three-dimensional format tailored for the BiLSTM model. This preparation facilitates a multivariate single-step prediction on a daily scale, formulated as $\hat{y}^{t+1} = f(y^{t-T+1}, \ldots, y^t; x^{t-T+1}, \ldots, x^t)$, where $f(\cdot)$ denotes the model's mapping function from input data to output predictions, and T signifies the chosen time step length. The processed dataset for each basin, whether data-rich or data-scarce, consists of input features and corresponding observed discharge values.

4.2 Design of the Experimental Study

Our experimental setup evaluates the FedHydroDSW framework's runoff forecasting capabilities, emphasizing its performance in data-scarce environments compared to traditional methods.

We examine three coastal U.S. Hydrologic Unit Codes (HUCs) across New England, California, and the Pacific Northwest. Each region includes a consistent configuration of seven data-rich watersheds and two data-scarce watersheds (North River at Shattuckville, MA, and Swift River near Roxbury, Maine in New England; Pescadero Creek near Pescadero, CA, and Redwood Creek at

Orick, CA in California; and Minam River at Minam, OR, and St. Joe River at Calder, ID in the Pacific Northwest). This selection enables a comprehensive assessment across varied hydrological conditions. Data for the data-rich watersheds span the last 20 years, while the data-scarce watersheds are limited to the most recent two years, creating a controlled environment to test the framework's adaptability. The datasets are divided into training, validation, and testing segments, reflecting the data availability within each basin, to thoroughly examine FedHydroDSW's predictive accuracy. During the training process, we implement the dynamic weighting factor γ as in Eq. (5) that balances the importance of validation loss and MAE in our composite loss function $\mathcal{L}_C$. This γ is adjusted linearly from 0.5 to 1.0 over 50 global communication rounds, gradually shifting emphasis from equal weighting to greater focus on validation loss.

The final model evaluation utilizes the Nash-Sutcliffe Efficiency Coefficient (NSE), Root Mean Square Error (RMSE), and Relative Absolute Error (RAE) as metrics to gauge the model's accuracy and reliability. NSE evaluates the model accuracy with a range of $(-\infty, 1]$, where 1 signifies perfect model fit and values above 0.5 are deemed acceptable. RAE assesses error relative to a benchmark, with scores ranging from $[0, +\infty)$; below 1 suggests better-than-average performance, while above 1 indicates poorer performance. RMSE measures prediction error dispersion, highlighting outliers.

$$\begin{cases} NSE = 1 - \sum_{i=1}^{n} \left(\hat{y}_i - y_i\right)^2 / \sum_{i=1}^{n} \left(y_i - \bar{y}\right)^2 \\ RAE = \sum_{i=1}^{n} |y_i - \hat{y}_i| / \sum_{i=1}^{n} |y_i - \bar{y}| \\ RMSE = \sqrt{\sum_{i=1}^{n} \left(\hat{y}_i - y_i\right)^2 / |n|} \end{cases} \tag{10}$$

Here n is the sample size, $\hat{y}_i$ is the predicted discharge, y_i is the observed discharge, and $\bar{y}$ is the mean observed discharge.

Our FedHydroDSW framework adopts a specialized Local-BiLSTM architecture for each basin by applying a consistent dropout rate of 0.1 for regularization across different time steps. We tailor the model's complexity based on the data's richness: employing batch sizes of 128 for well-documented basins and 64 for those less studied. Each basin undergoes three training epochs before model updates are shared globally, over 50 communication rounds, adhering to a training-validation division at a ratio of 70 to 30. The performance of our FedHydroDSW method is benchmarked against four leading approaches:

- **Local-BiLSTM** [23]: Only local historical data is employed for training runoff prediction models, acting as the baseline for local model evaluations.
- **Multi-FineTuned** [11]: A multi-source fine-tuning strategy, involving initial pre-training on datasets from both data-rich and data-scarce basins, with subsequent fine-tuning on data-scarce basins.
- **Transfer-A** [14]: Only input and output layer weights are updated after initial pretraining with data-rich sources, preserving all other LSTM and recurrent weights. This maintains fundamental behaviors while tailoring input-output dynamics to the target dataset.

– **Transfer-B** [14]: Only the weights related to input, output, and LSTM hidden state transformations are updated, while input-to-LSTM layer weights remain unchanged. This method retains foundational knowledge while adapting to the new dataset's characteristics.
– **FedAvg** [15]: Each node independently trains a model with local data; the central server then averages updates to form a global runoff prediction model.

4.3 Experimental Result and Discussion

Experiment I: We streamline the comparison of runoff prediction models across selected HUCs to gauge their prediction accuracy. This experiment aims to identify the most effective model under different regional climates and scenarios of data availability. In data-scarce regions, we fine-tune models using the only available data from 2006 to 2007 and then evaluate their performance over six months in 2008. Figure 2 provides a compelling visual representation of the experimental results, specifically illustrating the comparative analysis across

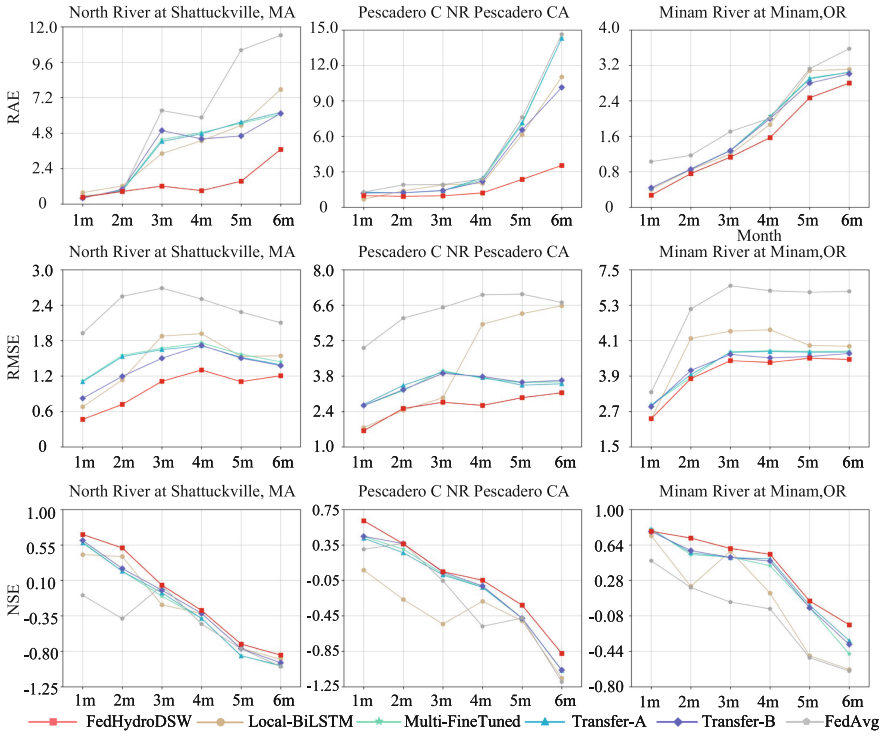


Fig. 2. RAE, RMSE, and NSE Performance Comparison of Hydrological Forecast Models Across North River at Shattuckville, MA; Pescadero Creek near Pescadero, CA; and Minam River at Minam, OR, Over a Six-Month Period in 2008.

the North River at Shattuckville, MA; Pescadero C NR Pescadero CA; and Minam River at Minam, OR. This analysis showcases the performance differentials between the FedHydroDSW framework and established methodologies under the lens of data scarcity.

The experiment showcases that FedHydroDSW consistently achieves lower RAE and RMSE values and higher NSE scores compared to the benchmarks set by conventional methods. In data-scarce regions where conventional models typically struggle due to insufficient training data, FedHydroDSW substantially reduces prediction errors and aligns more closely with observed runoff trends. These improvements demonstrate FedHydroDSW's ability to effectively leverage federated learning, enabling the model to benefit from the collective data insights gathered from various watersheds without directly sharing sensitive data. By allowing data-scarce basins to fine-tune models based on insights derived from data-rich counterparts, the framework ensures that each local model is optimally adjusted to the specific hydrological conditions and data distributions of its corresponding watershed.

Experiment II: We test the FedHydroDSW method's ability for mid to long-term runoff forecasting over a year in three data-scarce basins, evaluating the model's reflection of runoff trends and response to seasonal changes against actual data. Figure 3 shows the FedHydroDSW model's runoff predictions from October 1, 2008, to September 30, 2009, for three US watersheds: North River at Shattuckville, MA; Pescadero C NR Pescadero CA; and Minam River at Minam, OR. These data-scarce basins, representing diverse climates—humid continental, Mediterranean, and maritime, respectively—highlight FedHydroDSW's long-term forecasting ability across different hydrological conditions.

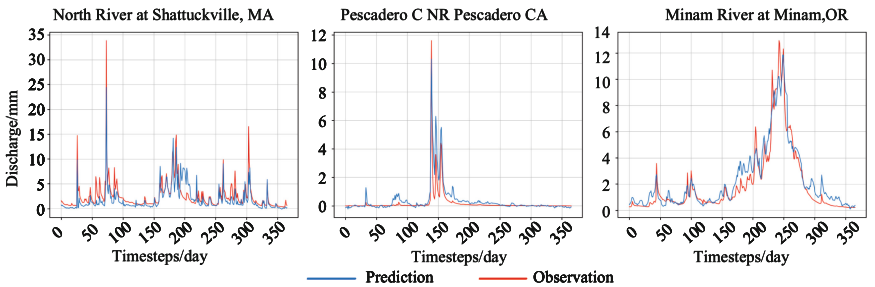


Fig. 3. Analysis of observed versus predicted discharges in three distinct watersheds during October 1, 2008 - September 30, 2009, utilizing FedHydroDSW.

The precision of the FedHydroDSW predictions is notably apparent in the North River at Shattuckville, MA, where the model's forecasts align closely with actual observations, effectively capturing peak flow events indicative of the

seasonal variability. Similarly, in Pescadero C NR Pescadero CA, the framework exhibits adaptability to the unique conditions of central California, accurately reflecting discharge fluctuations despite environmental differences from the North River. The inferred performance in the Minam River at Minam, OR, although not detailed quantitatively, suggests FedHydroDSW's consistent predictive accuracy extends to the diverse hydrological landscapes of the Pacific Northwest. Collectively, these results underscore the FedHydroDSW model's potential as a federated solution for runoff forecasting across different geographic regions.

Experiment III: We evaluate the generalization of our models on unknown data-scarce watersheds with the available data from 2006–2007, conducting five rounds of fine-tuning to assess adaptability and performance. After fine-tuning, we assess the average model effectiveness and robustness for 2008–2010 across various settings. Table 1 displays the evaluation of generalization capabilities for several runoff prediction models across three untrained, distinct watersheds: Swift River near Roxbury, Maine (ID:01055000), Redwood Creek at Orick, California (ID:11482500), and St. Joe River at Calder, Idaho (ID:12414500). The models' performance is assessed to determine their accuracy and reliability in predicting hydrological behavior in environments beyond their training regions.

Table 1. Evaluation of Generalization Performance for 2008–2010 by using the performance metrics for various hydrological models within the Swift River near Roxbury, Maine (ID:01055000); Redwood Creek at Orick, California (ID:11482500); and the St. Joe River at Calder, Idaho (ID:12414500).

Models	ID:01055000			ID:11482500			ID:12414500		
	RAE	RMSE	NSE	RAE	RMSE	NSE	RAE	RMSE	NSE
Local-BiLSTM	0.56	1.22	0.51	0.62	1.79	0.48	0.66	1.85	0.41
FedAvg	0.79	1.71	0.42	1.08	4.57	0.42	1.25	2.49	0.58
Multi-FineTuned	0.52	1.10	0.58	0.56	1.61	0.52	0.54	1.58	0.68
Transfer-A	0.51	1.08	0.58	0.57	1.57	0.53	0.54	1.54	0.69
Transfer-B	0.51	1.10	0.58	0.59	1.60	0.50	0.52	1.57	0.68
FedHydroDSW	**0.48**	**1.01**	**0.66**	**0.38**	**1.10**	**0.68**	**0.45**	**1.36**	**0.73**

In this table, the FedHydroDSW model consistently demonstrates superior performance across all three watersheds. Specifically, the FedHydroDSW achieves the highest NSE scores and the lowest RAE and RMSE scores, indicating its ability to predict runoff accurately. For the Swift River watershed in New England, FedHydroDSW shows notable improvements over the other models, achieving an NSE score of 0.66, which surpasses the next-best model's performance by significant margins. This trend continues across the other watersheds, with FedHydroDSW recording the highest NSE scores of 0.68 and 0.73 for the

Redwood Creek and ST JOE RIVER watersheds, respectively. The RAE and RMSE metrics underscore FedHydroDSW's robustness, demonstrating its ability to maintain lower error rates than competing models, effectively capturing both general trends and subtle nuances in hydrological data.

4.4 Ablation Study

We evaluate the contribution of each component of the FedHydroDSW framework through a series of ablation experiments. These experiments assess the impact of personalized federated learning, model similarity metrics, parameter adaptation methods, and the dynamic weighting factor (γ) on the model's performance. The results of these experiments collected on Pescadero C NR Pescadero CA watershed and are summarized in Table 2.

Table 2. Ablation Study Results on Pescadero C NR Pescadero CA from 2008–2010 by using the performance metrics

Model Configuration	NSE	RMSE	RAE
FedAvg (Baseline)	0.47	1.43	1.37
Without Model Similarity Metrics	0.55	1.29	1.25
Without Parameter Adaptation Methods	0.58	1.25	1.21
Without Dynamic Weighting Factor (γ)	0.59	1.26	1.20
FedHydroDSW (Full Model)	0.63	1.23	1.18

To establish a baseline, we used a traditional federated learning approach (FedAvg) without personalization, achieving an NSE of 0.47, RMSE of 1.43, and RAE of 1.37. Removing model similarity metrics while retaining parameter adaptation and the dynamic weighting factor improved the NSE to 0.55, RMSE to 1.29, and RAE to 1.25. Eliminating parameter adaptation and using static parameters across all basins resulted in an NSE of 0.58, RMSE of 1.25, and RAE of 1.21. Using a fixed weighting factor ($\gamma = 0.5$) yielded an NSE of 0.59, RMSE of 1.26, and RAE of 1.20.

We also conducted a series of ablation experiments focusing on four key hyperparameters: learning rate, local batch size, SGD momentum, and the starting value of γ. These experiments aimed to assess the impact of each hyperparameter on model performance, measured by the RAE, RMSE, and NSE. The results are summarized in Table 3. The results indicate that a learning rate of 0.01 yields the best performance, achieving an NSE of 0.63, RMSE of 1.23, and RAE of 1.18. Similarly, a local batch size of 128 and 256 both resulted in an NSE of 0.63, RMSE of 1.23–1.25, and RAE of 1.18. Higher learning rates resulted in poorer performance, highlighting the need for balance in these parameters. A momentum of 0.5 produced the best results with an NSE of 0.63, while a very high momentum (0.99) caused instability. For γ, a value of 0.5 was optimal, with

an NSE of 0.63 and other values showing decreased performance, underscoring the need for a balanced initial setting.

These experiments underscore the importance of each component in enhancing FedHydroDSW's predictive accuracy. Personalized federated learning adapts to local data while leveraging global knowledge, and model similarity metrics and parameter adaptation ensure the aggregated model fits each watershed's unique characteristics. The dynamic weighting factor prevents overfitting by balancing immediate performance with long-term generalization. The ablation studies also reveal that optimal settings for learning rate, batch size, SGD momentum, and γ are critical for improving performance, providing insights for effectively configuring the framework for runoff forecasting in data-scarce watersheds.

Table 3. Performance metrics for different hyperparameters in the ablation study of FedHydroDSW. The table shows the impact of varying learning rates, batch sizes, SGD momentum, and starting values of γ on RAE, RMSE, and NSE metrics.

	Learning Rate				Local Batch Size			
Metric	0.001	0.01	0.1	0.5	32	64	128	256
RAE	1.21	1.18	1.09	1.25	1.23	1.20	1.18	1.18
RMSE	1.51	1.23	1.23	1.93	1.28	1.24	1.23	1.25
NSE	0.41	0.63	0.61	0.03	0.58	0.62	0.63	0.63
	SGD Momentum				Starting Value of γ			
Metric	0.3	0.5	0.7	0.9	0.1	0.3	0.5	0.7
RAE	1.22	1.18	1.18	1.14	1.18	1.19	1.18	1.18
RMSE	1.29	1.23	1.27	1.39	1.24	1.23	1.23	1.26
NSE	0.58	0.63	0.55	0.50	0.58	0.6	0.63	0.61

5 Discussion

The FedHydroDSW framework significantly advances runoff forecasting, especially for data-scarce watersheds. This framework uniquely integrates model similarity metrics, parameter adaptation methods, and a dynamic weighting factor to address challenges related to time series data scarcity and hydrological forecasting heterogeneity. One of the key improvements is the validation loss with MAE evaluation, which ensures a balance between training performance and generalization, leading to robust and reliable runoff predictions. Additionally, the framework enhances predictive accuracy by integrating adaptable models from data-rich basins into data-scarce ones, as validated by our experiments. This approach helps mitigate some of the performance disparities and aims to enhance the reliability of the forecasts.

Using a uniform model architecture across participating nodes facilitates efficient aggregation and dissemination of model updates, providing a common

baseline for performance comparison. While we acknowledge that nodes with limited data may benefit from reduced model complexity, the decision to use uniform models ensures scalable deployment and seamless integration of updates. However, heterogeneous models tailored to each node's data and computational capacity could further enhance performance but pose challenges in aggregation, fairness, and consistency. Techniques such as federated distillation and meta-learning may address these issues in future research. Our future work will further expand the analysis to diverse watersheds with varying climates and data availability scenarios. By utilizing publicly available hydrological datasets, we aim to conduct a comprehensive global assessment of FedHydroDSW's performance and generalizability. This will help address the current focus on a limited set of coastal US watersheds and ensure broader applicability and robustness.

6 Conclusion

This study introduces FedHydroDSW, a federated learning framework for collaborative, privacy-aware training of runoff forecast models across organizations, addressing data silos in hydrology. FedHydroDSW enhances prediction accuracy by integrating personalized federated learning, model similarity metrics, parameter adaptation methods, and a dynamic weighting factor, ensuring tailored and balanced performance. Empirical evaluations confirm FedHydroDSW's superior performance over conventional models, especially in data-scarce settings. FedHydroDSW facilitates real-time environmental monitoring and promotes sustainable practices through efficient data use and privacy-centric learning. Future efforts will explore using heterogeneous models in federated learning to better accommodate varying data complexities across nodes, ensuring robust performance in diverse watersheds.

References

1. Addor, N., Newman, A.J., Mizukami, N., Clark, M.P.: The CAMELS data set: catchment attributes and meteorology for large-sample studies. Hydrol. Earth Syst. Sci. **21**(10), 5293–5313 (2017)
2. Al-Huthaifi, R., Li, T., Huang, W., Gu, J., Li, C.: Federated learning in smart cities: privacy and security survey. Inf. Sci. **632**, 833–857 (2023)
3. Balti, H., Abbes, A.B., Mellouli, N., Farah, I.R., Sang, Y., Lamolle, M.: Multi-dimensional architecture using a massive and heterogeneous data: application to drought monitoring. Futur. Gener. Comput. Syst. **136**, 1–14 (2022)
4. Chen, C., Hui, Q., Xie, W., Wan, S., Zhou, Y., Pei, Q.: Convolutional neural networks for forecasting flood process in internet-of-things enabled smart city. Comput. Netw. **186**, 107744 (2021)
5. Chen, Z., Ni, X., Li, H., Kong, X.: FedLGAN: a method for anomaly detection and repair of hydrological telemetry data based on federated learning. PeerJ Comput. Sci. **9**, e1664 (2023)
6. Ding, Y., Zhu, Y., Feng, J., Zhang, P., Cheng, Z.: Interpretable spatio-temporal attention LSTM model for flood forecasting. Neurocomputing **403**, 348–359 (2020)

7. Farooq, M.S., Tehseen, R., Qureshi, J.N., et al.: FFM: flood forecasting model using federated learning. IEEE Access **11**, 24472–24483 (2023)
8. Gao, S., et al.: Short-term runoff prediction with GRU and LSTM networks without requiring time step optimization during sample generation. J. Hydrol. **589**, 125188 (2020)
9. Han, H., Morrison, R.R.: Data-driven approaches for runoff prediction using distributed data. Stoch. Env. Res. Risk Assess. **36**, 1–19 (2021)
10. Kan, G., et al.: A new hybrid data-driven model for event-based rainfall-runoff simulation. Neural Comput. Appl. **28**(9), 2519–2534 (2017)
11. Kratzert, F., Klotz, D., Brenner, C., Schulz, K., Herrnegger, M.: Rainfall-runoff modelling using long short-term memory (LSTM) networks. Hydrol. Earth Syst. Sci. **22**(11), 6005–6022 (2018)
12. Kratzert, F., Klotz, D., Herrnegger, M., Sampson, A.K., Hochreiter, S., Nearing, G.S.: Toward improved predictions in ungauged basins: exploiting the power of machine learning. Water Resour. Res. **55**(12), 11344–11354 (2019)
13. Kratzert, F., Klotz, D., Shalev, G., Klambauer, G., et al.: Benchmarking a catchment-aware long short-term memory network (LSTM) for large-scale hydrological modeling. Hydrol. Earth Syst. Sci. **2019**, 1–32 (2019)
14. Ma, K., Feng, D., Lawson, K., Tsai, W.P., et al.: Transferring hydrologic data across continents–leveraging data-rich regions to improve hydrologic prediction in data-sparse regions. Water Resour. Res. **57**(5), e2020WR028600 (2021)
15. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics, pp. 1273–1282 (2017)
16. Meresa, H.: Modelling of river flow in ungauged catchment using remote sensing data: application of the empirical (SCS-CN), artificial neural network (ANN) and hydrological model (HEC-HMS). Model. Earth Syst. Environ. **5**(1), 257–273 (2019)
17. Moishin, M., et al.: Designing deep-based learning flood forecast model with ConvLSTM hybrid algorithm. IEEE Access **9**, 50982–50993 (2021)
18. Nguyen, G., et al.: Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey. Artif. Intell. Rev. **52**(1), 77–124 (2019)
19. Reddy, B.S.N., Pramada, S., Roshni, T.: Monthly surface runoff prediction using artificial intelligence: a study from a tropical climate river basin. J. Earth Syst. Sci. **130**(1), 1–15 (2021)
20. Savi, M., Olivadese, F.: Short-term energy consumption forecasting at the edge: a federated learning approach. IEEE Access **9**, 95949–95969 (2021)
21. Sood, A., Smakhtin, V.: Global hydrological models: a review. Hydrol. Sci. J. **60**(4), 549–565 (2015)
22. UNESCO, et al.: The United Nations world water development report 2021: Valuing water. United Nations (2021)
23. Van Houdt, G., Mosquera, C., Nápoles, G.: A review on the long short-term memory model. Artif. Intell. Rev. **53**, 5929–5955 (2020)
24. Verma, D., White, G., de Mel, G.: Federated AI for the enterprise: a web services based implementation. In: 2019 IEEE International Conference on Web Services (ICWS), pp. 20–27 (2019)
25. Wang, J., et al.: Application of BP neural network algorithm in traditional hydrological model for flood forecasting. Water **9**(1), 48 (2017)
26. Xiang, Z., Demir, I.: Distributed long-term hourly streamflow predictions using deep learning-a case study for state of Iowa. Environ. Model. Softw. **131**, 104761 (2020)

27. Zhang, C., Zhang, S., James, J., Yu, S.: FastGNN: a topological information protected federated learning approach for traffic speed forecasting. IEEE Trans. Industr. Inf. **17**(12), 8464–8474 (2021)
28. Zhang, J., Chen, X., Khan, A., et al.: Daily runoff forecasting by deep recursive neural network. J. Hydrol. **596**, 126067 (2021)
29. Zhang, M., Sapra, K., Fidler, S., Yeung, S., et al.: Personalized federated learning with first order model optimization. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, 3–7 May 2021 (2021)
30. Zhu, Y., Feng, J., Yan, L., Guo, T., Li, X.: Flood prediction using rainfall-flow pattern in data-sparse watersheds. IEEE Access **8**, 39713–39724 (2020)

Stagger-Cache MITM: A Privacy-Preserving Hierarchical Model Aggregation Framework

Anupam Gupta(✉), Pabitra Mitra, and Sudip Misra

Indian Institute of Technology, Kharagpur, Kharagpur, India
anupamguptacal@kgpian.iitkgp.ac.in

Abstract. In the era of widespread intelligent frameworks and models, we are often surrounded by systems that house multiple models for varied task-specific predictions. Given the general expanse of large intelligent frameworks, we explore the use case of a large-scale setup with locally differentiated machine learning models organized in a multi-tier hierarchy. We specifically aim to understand how to support continuous tier-specific inference queries in a hierarchical multi-model setting while optimizing for network costs, storage space and respecting local privacy constraints. This paper explores two intuitive frameworks in this regard and performs a thorough comparative analysis between the frameworks outlining numeric support points for the approaches. The paper also introduces a novel dynamic protocol, termed Meet-in-the-Middle (MITM), that combines the benefits of the previously introduced protocols using a novel meet-in-the-middle, staggered model-caching approach. Thorough testing and analysis of MITM on a distributed agricultural disease-prediction dataset displays the superiority of the novel protocol over the previously introduced frameworks, demonstrating a reduction of up to 80% in real-time communication cost, 80% in memory utilization and 50% in inference latency while maintaining comparable metrics of accuracy at even higher levels of the multi-tier hierarchical framework.

Keywords: Multi-Layer Inferencing · Machine Learning · Optimized Model Caching · Federated Learning · IIoT

1 Introduction

Machine learning models are widely deployed across various systems for data-driven decision-making. The easy-to-train and adaptive nature of most machine learning algorithms, allow us to choose from multiple available models and apply them for specific tasks by fine-tuning on datasets, loss functions, and model parameters [12]. The deployed models often follow a hierarchy arising from the natural organization of the data generation process. For example, traffic prediction can be performed using models deployed at street, borough, or city granularity built using street traffic data [7].

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 199–216, 2025.
https://doi.org/10.1007/978-3-031-78183-4_13

Inference queries may be serviced at various tiers of the hierarchy. Hierarchy-sensitive inference may be performed either directly using models stored at nodes in all the levels of the hierarchy or by aggregating the predictions from finer granularity models [1]. The choice determines the trade-off between freshness, privacy, accuracy, inference latency, and communication cost. As the expanse of local models increases in large infrastructure setups, inferencing of multi-model setups is going to become increasingly important [17].

We study the large-scale inference problem at multiple granularities from a 'model caching' point of view. A staggered meet-in-the-middle caching at various levels of granularity is found to be an effective approach in comparison to complete caching and zero caching scenarios. In this protocol the nodes at a few of the levels, cache models and thus infer with almost no latency but some computation, while others do not perform computationally elaborate inference and simply aggregate the predictions from lower-level nodes by communicating over the network. The cached layers are selected based on certain global optimizing criteria in a staggered manner over time. The protocol provides a low inference latency federated learning framework that can be applied over sensor, edge, fog, and cloud devices. We demonstrate the efficacy of the approach using a case study on crop disease forecasting at the granularity of fields, cities, sub-districts, districts, and states.

2 Related Work

The problem of data and model aggregation has been studied in IoT, sensor networks, and federated learning domains. A majority of the existing literature in data aggregation covers aggregation approaches across flat networks and only a few address hierarchical networks [2], especially in an IoT setting. Aggregation, in an intelligence setting, has been discussed elaborately as well, in the context of ensemble methods in machine learning [15,20] and gradient boosting frameworks [5]. However, similar to data aggregation studies, ensemble learning techniques have generally been applied to flat networks and not multi-tier hierarchies, especially in inference gathering. No identifiable work, in data aggregation, addresses the edge intelligence use case, specifically - optimizing the duplex flow of information that must occur to perform inferences, if intelligent models are present at edge nodes and inference is requested at a higher tier node in the organizational hierarchy.

In recent years, hierarchical systems have found a natural fit in IoT (Internet of Things) frameworks [8], where they are being studied extensively to reduce the global transmission costs in the field of federated learning - a model aggregation and consensus protocol where intelligent models are only present at the edge of the network [3,13] - a setting that is very similar to ours. In this context, Model caching, as a technique, is also being recently studied for efficient cloud-edge computation offloading for machine learning tasks [6,18,19]. These studies use optimization techniques to handle the trade-off between inference latency, edge computation, and communication cost.

In this paper, we experiment with caching - at tiers - as a method to perform compute inference at all applicable tiers while only needing a fraction of the network and communication cost. Caching, for ML inference, in hierarchical frameworks, has not been explored yet. Caching, as a software system performance-enhancing concept, has been largely explored. However, our work focuses on using the same idea of caching in a hierarchical inference setting. We outline the ideology, protocols, and implementations to both create the cache from scratch as well as intelligently and regularly refresh it and outline the protocols for cache hits and misses in this setting. In addition, our novel MITM protocol is significantly different from all works in this area since it can support multiple levels of edge model heterogeneity - i.e. varying number of leaf nodes, varying number of models - with large or no correlation in each node and a varying number of labeled datapoints per such model. To the best of our knowledge, this is the only aggregation work that recognizes the presence of multiple, potentially differing task models in individual nodes and still successfully correlates all such task-specific inference values processed at any requested tier of interest in the hierarchy.

3 Contributions of This Paper

In this paper, we make the following significant contributions:

- We discuss the problem statement of a hierarchical tier setup where data points lie only at the last layer of setup but inference is important across multiple tiers.
- The paper outlines and discusses in detail two different frameworks for swift, multi-tier inferencing and outlines the drawbacks and benefits of both approaches.
- The paper introduces a novel Meet-In-the-Middle framework that dynamically establishes tiers where intermittent models can be cached for swift inferencing. We provide detailed algorithms for tier selection, cache creation, query completion on cached and non-cached tiers, a dynamic algorithm for identifying cache staleness across tiers, and protocols for cache-refresh in our tiered architectural setup.
- We conduct detailed experiments to tackle an agricultural pest-forecasting use case that is the perfect example of systems that would benefit from swift, multi-tier prediction - specifically for locally applicable, quick forecasting.
- We provide thorough experimental results to outline the benefits of MITM over traditional multi-tier inferencing frameworks and demonstrate how intermittent caching helps us implement swift multi-tier inferencing while enforcing tangible savings in precision loss, inference latency, and communication cost across the board.

4 Problem Setup

We consider a setup where a logical hierarchy exists for the data organization as well as queries. In the case of spatial data, the hierarchies may be defined

in terms of geographical granularity units and queries allow a choice of the granularity unit. We also assume that compute nodes responsible for answering queries corresponding to a particular granularity exist in the network, however, data storage and model training happen only in the leaf level of the hierarchy tree. This ensures locality and privacy preservation. We describe our setting and problem statement, in more detail, through mathematical notations, in the subsequent sections.

4.1 Mathematical Notation

Let us assume there are $T = \{T_1, \ldots, T_l\}$ tiers of interest in our hierarchical framework. Let us assume that each tier - $\{T_i\}$ has a set of nodes of computation denoted by C_i. In Figure 1, we have 13 compute nodes in the bottom-most tier (T_1), 4 in the tier above that (T_2), 4 above that (T_3) and so on.

Let $C_{i,j}$ denote the j-th compute node in tier T_i. Each compute node can house multiple models within it - one specific to each task. We use $\theta(C_{i,j})$ to denote the set of independent models in the compute node $C_{i,j}$. Consequently, we use $\theta_k(C_{i,j})$ to denote the k-th model in $\theta(C_{i,j})$ where $k < |\theta(C_{i,j})|$. We assume that at time $t = 0$ all physical models exist at the lowest level of the hierarchy T_1, i.e.,

$$|\theta(C_{i,j})| = 0 \quad \ldots \quad \forall i \in [2, |T|], \forall j \in [0, |C_i|] \tag{1}$$

A query string, represented by $(x_{input}, \tilde{i})$, consists of an input vector x_{input} to be serviced at tier $T_{\tilde{i}}$ i.e. we wish to obtain the inference on x_{input} as seen from tier T_i. Syntactically, we aim to compute and obtain the result R (Table 1):

$$R(x_{input}, \tilde{i}) = \{\theta_k(C_{\tilde{i},j})(x_{input}) \; | \forall k \in C_{\tilde{i},j}|\} \tag{2}$$

4.2 Performance Criteria

To assess the efficacy of model aggregation frameworks, we outline our metrics of importance. We revisit them in the experiment section.

- *Accuracy* - Measured in terms of the error in regression prediction of the machine learning models at the required tier.
- *Inference Latency* - Real-time forecast requires a reduced inference latency.
- *Resource Conservation*
  1. *Network Communication* - Given the resource-constrained conditions of most intelligent field devices, it is important to conserve the amount of network communication overhead that each device incurs during continuous inferencing.
  2. *Memory and Computation* - Available Memory and Compute Costs are also valuable metrics for resource-constrained devices.

Table 1. Table of Notations

Notation	Description
T_i	Tier of Hierarchy
C_i	Set of Nodes of Computation in Tier T_i
$C_{i,j}$	j^{th} node in C_i
$\theta(C_{i,j})$	Set of models in the compute node $C_{i,j}$
$\theta_k(C_{i,j})$	k-th model in $\theta(C_{i,j})$
$R(x_{input}, \tilde{i})$	Inference result of x_{input} at tier T_i
SD	Subdistrict Tier in our agricultural forecasting use-case study
$\Omega(\eta)$	For any node η in tier T_i, $\Omega(\eta)$ are the nodes in T_{i-1} that are children of η in the hierarchy
$\Gamma(N)$	An inverse index of Task D to Predicted Values for Task D from all nodes in set N
$\phi(L)$	A Function to return cached models at L if L contains a model cache else false

– *Data Privacy* - In this paper, we recognize that data at the edge is often local and sensitive. Hence, we undertake architectural decisions at each level of each protocol to reduce the security risks of over-network communication [14]. We achieve this by adopting measures to conserve data locality (no data leaves local aggregation nodes) and enable knowledge transfer, when needed, through appropriate model sharing only.

5 Multi-tier Inference Frameworks

In the next section, we present two intuitive multi-tier inference framework ideologies and detail their implementation, benefits, and drawbacks.

5.1 Framework A: Bottom-Up

An intuitive approach to tier-specific inference is the simple aggregation of the model output from the lowest tier nodes. This framework builds prediction values at each tier from the bottom up. The input conditions are passed down to the bottommost tier that houses the task models and a singular value for each task and each node is then built up by performing weighted aggregation till the tier of interest.

Inspired by studies of aggregation in hierarchical networks [4,9], we present a weighted averaging approach for this aggregation (Figure 1). On reception of (x_{input}, $\tilde{i}$) at the bottom-most layer, we generate the set:

$$R_{bottom}(x_{input}) = \{[\theta_k(C_{1,0})(x_{input})), \theta_{k+1}(C_{1,0})(x_{input}))...], [\theta_k(C_{1,1})(x_{input})), \theta_{k+1}(C_{1,1})(x_{input})), ...]\} \quad (3)$$

and recursively build up to $R_{\tilde{i}}$ that can be returned to the user.

Specifically, we generate a *set* where each element is a *list* corresponding to the results of all task models in one computation node.

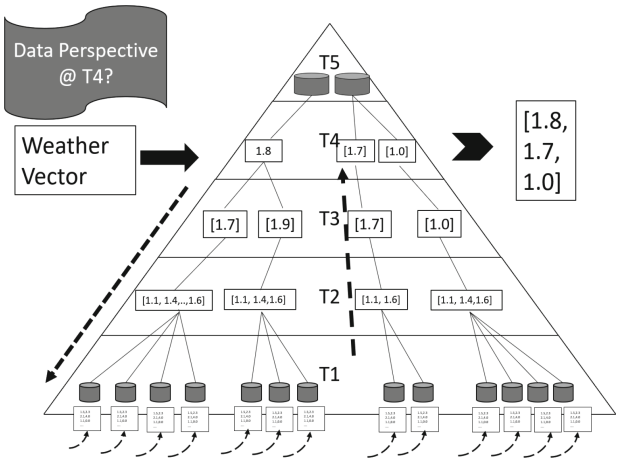


Fig. 1. Tier-specific inference using Framework A (Bottom-Up). Any input data point has to traverse till the end of the computation framework and results are recursively aggregated up till the tier of interest

5.2 Computation Methodology

We outline the steps **for inferencing** below:

- **Step 1:** Compute R_{bottom} using it's definition in Eq. 3.
- **Step 2:** Moving upwards from T_1, For every node η in C_2, compute $\Omega(\eta) =$ [Nodes in C_1 that hierarchically lie under η].
- **Step 3:** For every η, also compute $\Gamma(\Omega(\eta)) =$ an inverse index $\{$"$TaskD$" $\mapsto$ "[Values from R_{bottom} for all models for Task D in $\Omega(\eta)$]"$\}$
- **Step 4:** For every D in $\Gamma(\Omega(\eta))$, we now replace the list of values in $\Gamma(\Omega(\eta))[D]$ with a single weighted average value:

$$M_{1,\eta}^{D} = \sum_{\alpha=0}^{\alpha=|\Gamma(\Omega(\eta))[D]|} w * \Gamma(\Omega(\eta))[D][\alpha] \tag{4}$$

  where w is the total number of training data points on the edge node that generated the value $\Gamma(\Omega(\eta))[D][\alpha]$ i.e. a weighted average for task D.
- **Step 5:** All M values across all tasks in η and across all nodes in C_2 are then aggregated. This, now, becomes our task prediction vector for T_2 and syntactically our $R_{bottom+1}$ that we have built up from R_{bottom}.

We repeat this process recursively, building up from R_{bottom} until we have computed $R_{\tilde{i}}$, which is the $T_{\tilde{i}}$ tier-specific task prediction vector.

5.3 Framework B: Tier-Caching

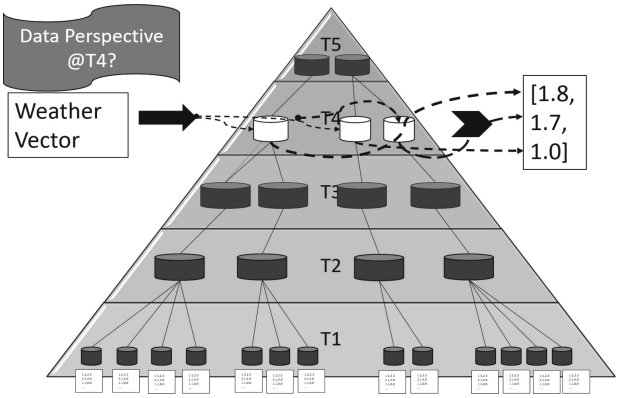


Fig. 2. Tier-specific inference using Framework B (Tier-Caching). Each node in a gray tier contains cached locally-applicable task-specific models.

In contrast to Framework A's ex-situ inference and upward result traversal approach, another intuitive approach to a multi-tier inference framework is to perform inference in situ at the tier of interest itself. Although unutilized in Framework A, we now use our underlying assumption that each tier contains physical fog nodes as necessary. In a multi-tier setup where physical models and training data reside only on the bottom-most tier, we need to cache one model per applicable task at every framework tier to support in-situ tier inference.

However, *How do we generate models to cache at tiers that originally never held any models?* Intuitively, to generate cache models at higher tiers of the framework, we need to build and generate the models from the bottom-up, one tier at a time.

Through the years, several model aggregation protocols have been explored, some of these include bootstrap aggregating, random forest algorithms, and adaptive boosting in ensemble learning [9,16], however, their direct adaptation to our use-case is not feasible as we want to generate models, not evaluation datapoints, that we can cache. Given our constraint of resource-hungry devices, it is not possible to train and store multiple models for one task at each fog node. Instead, we propose choosing a **single model** to cache per task, from all relevant models in the previous tier. We do this by comparing the applicability of the models, as measured by evaluation performance against an unbiased, separated, and mixed global dataset, that is local to the tier where we're generating the cache. This is the setting for complete caching.

A detailed description of the framework is outlined below. Similar to framework A, we outline building the cache for tier T_2 from the models in tier T_1.

5.4 Computation Methodology

During *Preprocessing* :

- **Step 1:** For every node η in C_2, compute $\Omega(\eta)$ = [Nodes in C_1 that hierarchically lie under η].
- **Step 2:** For any node $Z \in C_1$ in $\Omega(\eta)$, we extract and isolate a subset of task-specific training data points that we will use for analysis (hereafter referred to as *evaluation points*). In our experiment, we isolate **2 data points per task per node** before training.
- **Step 3:** For the nodes in $\Omega(\eta)$, we build 2 inverse indices at every computation node - (I_1, I_2). I_1 mapping "$TaskD$" $\mapsto$ "[recorded evaluation points for D]" and I_2 mapping "$TaskD$" $\mapsto$ "[Field models for D]".
- **Step 4:** For every task D in I_1, we compute $I_2[D]$, and evaluate each model in $I_2[D]$ over the dataset $I_1[D]$.
- **Step 5:** An MSE loss is computed against the evaluation dataset for each model. We cache the model with the lowest cumulative MSE, among all models, per D.
- **Step 6:** On performing this iteratively over all nodes in tier T_2, and every task under such node, we compute a single model, per task, per node, that we can create at T_2 using models from tier T_1.

Having computed our model cache set at T_2 we, iteratively, build the model caches until our topmost tier, creating a cache for every tier (Table 2).

At *inference*: - We simply return $\theta_{\bar{i}}[x_{input}]$

Table 2. Benefits and drawbacks of Frameworks A and B

Pros	*Cons*
Framework A: Bottom-Up	
Accurate Inference Predictions	Higher Inference Latency
Reduced memory usage - No caching	Risk for data exposure over network
Updated model learning affect result values quickly	
Framework B: Tier-Caching	
Minimal inference Latency	Higher Initial Setup Duration
Easier to compute at inference	Slower model updates in top tiers
Static Results, Superior Model Filtering	Higher Memory Requirement

6 Framework of Meet-In-The-Middle (MITM) Staggered Caching

Our analysis of previous Frameworks A and B outlines the low-latency benefits of in-situ and high accuracy benefits of ex-situ inferences. We now introduce a meet-in-the-middle framework, that combines benefits from both our initial frameworks and mitigates their drawbacks. The approach is illustrated in Fig. 3.

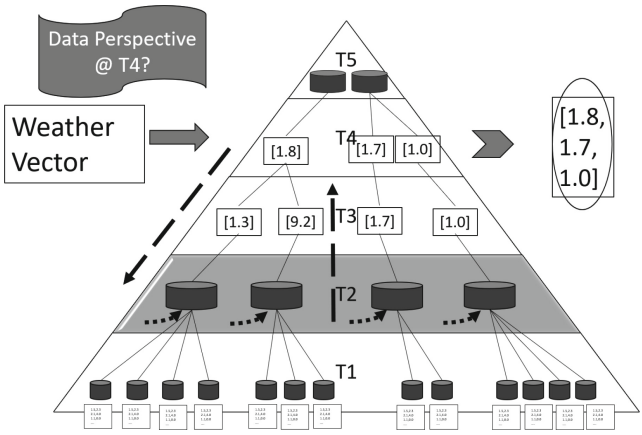


Fig. 3. Tier-specific inference using our MITM Framework (Staggered-Caching). Dark Gray represents cached tiers. Input data traversal is at maximum till this tier.

In a multi-tier hierarchical framework, we introduce the concept of Staggered Caching, which is the idea of caching intermittent and often unevenly distributed layers in our hierarchical framework. Let $\phi(L)$ denote a function that returns cached models at tier L if tier L is cached, else false.

6.1 Computation Methodology

- **Cache Hit** - Return $\phi(\tilde{i})[x_{input}]$.
- **Cache Miss** - If $\phi(\tilde{i})$ is false, find the closest tier below - $\tilde{j} = argmax_i f(i) := \{\phi(i) \neq False; \forall i < \tilde{(i)}\}$. We infer the values at tier $T_{\tilde{j}}$ by computing $R_{\tilde{j}} = \phi(\tilde{j})[x_{input}]$ first. We can then build $R_{\tilde{j}}$ up to $R_{\tilde{j}+1}$ and finally to $R_{\tilde{i}}$ using the same weighted averaging aggregation logic we defined for Framework A (Eq. 3).

Since the number of cached tiers is generally much lesser than the total number of tiers, the framework's evaluation performance is heavily dependent on the placement of the cached tiers. To maximize variance in tier placement and hence optimize framework performance, we propose a novel algorithm for optimal tier placement, contingent on an input of constraint weights, that are generally

use-case dependent. We outline and explain these points of consideration (input constraints) of tier selection below:

- **Distance between Cached tier and Non-Cached tier** - p_{comm} in Algorithm 1 - To reduce the communication overhead incurred for non-cache tier inference, we need to prioritize reducing the overall distance between non-cached and cached tiers.
- **Proximity to bottom models** - p_{local} in Algorithm 1 - Repeated Aggregation causes loss in precision.To preserve local applicability to ground reality, it is better to cache lower and perform aggregated inference in higher tiers.
- **Concurrent Cached Tiers** - p_{conc} in Algorithm 1- This constraint covers a negative penalty - i.e. We want to avoid concurrent cached tiers since higher variance (Staggering) in cache-tier placement will lead to a better-amortized performance.

Algorithm 1 MITM Tier Selection. Given an input of tier penalties(Section 6.1) among N total tiers, an optimal set of K tiers for model caches is selected.

Require: N, K ▷ *total tiers, caches needed*

Require: $p_{comm}, p_{conc}, p_{local}$ ▷ *Constraint Penalties*
Ensure: $N, K \geq 1 | N \geq K$
  $minWt, minTiers \leftarrow 0, []$
  function COMPUTE(upperWt,lowerWt,tiers)
    currentWt = upperWt + lowerWt
    if $len(tiers) > K$ **then**
      return
    if $currentWt > minWt$ **then**
      return
    else if $len(tiers) == K$ **then**
      if $currentWt < minWt$ **then**:
        return currentWt, tiers
      return
    $maxtier \leftarrow -1$
    if $len(tiers) > 0$ **then**
      $maxtier \leftarrow tiers[len[tiers) - 1]]$
    for $i = maxtier + 1, \ldots, N$ **do**
      $tiersAbove \leftarrow (i - (maxtier + 1)) * p_{comm}$
      $lowerWt \leftarrow lowerWt + tiersAbove + p_{local}$
      if maxtier != -1 and maxtier - i = -1 **then**
        $lowerWt \leftarrow lowerWt + p_{conc}$
      $upperWt \leftarrow (N - (i + 1)) * p_{comm}$
      tiers.append(i)
      val, res = COMPUTE(upperWt, lowerWt, tiers)
      if res exists **then**
        if $val < minWt$ **then**
          $minWt \leftarrow val$
          $minTiers \leftarrow res$
      tiers.pop()
    return minWt, minTiers ▷ *minimum weight, tiers for cache*

Based on the weights placed on each of the above constraints, Algorithm 1, recursively computes a global weight for each possible combination and outputs a set of tiers that optimally fits our needs. We also provide a novel request frequency, cache staleness, and time-based cache refresh protocol in our Supplementary Appendix, to build on top of our algorithm.

6.2 Benefits and Drawbacks

The MITM Model Caching Frameworks combines multiple benefits from Frameworks A and B. Some of these include:

1. **Low Inference Latency** (in comparison to Framework A): We match the latency performance of Framework A in the case of a cache hit, and outperform A in the case of a cache miss since we *atmost* traverse to the last cached tier.
2. **Low Memory Requirements** (in comparison to Framework B): Fewer cached tiers mean a lesser memory requirement for MITM in comparison to Framework B.
3. **Mitigation of Security Concerns** - We further mitigate the concern related to over-network data transmission by conserving data locality and only propagating up from the last cached tier instead of from the bottom-most tier.
4. **Static Updates** (with support for dynamic cache updates) - As cached models remain generally static, we can plan how often we want to update our caches.

7 Experimental Results

In this section, we present thorough experimental results for our three defined frameworks against a popularly used, public, Government of India agricultural dataset.

7.1 Use Case: Crop Disease Forecasting

We explore multi-tier inferencing frameworks in the context of an agricultural disease forecasting task that spans multiple geographical locations and is stretched vertically over 5 tiers. The final task of the dataset is to gather and present granularity-specific forecasting results. We study the following tier configuration for our use case:

1. Fields (Tier T_1 - Bottom-most Tier) - Holds physical field models.
2. City (Tier T_2)
3. Subdistrict (SD) (Tier T_3),
4. District (Tier T_4), and
5. State (Tier T_5 - Topmost Tier)

Given the one-to-many relationship between compute nodes of higher tiers and the lower ones, we visualize a pyramid hierarchy for our use case. Deep learning regression models, one per disease per node (so far referred to as tasks in theory), are trained on local data at the lowest tier to predict the extent of crop disease from agro-meteorological features.

Datasets. We use the "Cotton" plant dataset generated by the Indian Council of Agricultural Research (ICAR)'s Crop Pest Disease Support System project [10] for our experiments. The dataset contains 15,912 data points, aggregated from 1982 to 2009, recorded across 6 cities in India. The dataset contains metrics per city, across multiple years. Each data point is an 11-element vector depicting: {*Year, Week, Max Temperature, Minimum Temperature, Relative Humidity (2 metrics), Rainfall, Wind Speed, Sunshine Duration, Evaporation*} *(X) and* {*corresponding Pest Value*}*(Y)*.. Each city per year records different metrics for multiple pests. These pests may be individual to that local city or shared between neighboring cities, however, no such restriction in data persists. We create our test set by aggregating, removing, and grouping a minor subset of random data points for every disease across every city. We use a **90-10% training-test split** for this purpose.

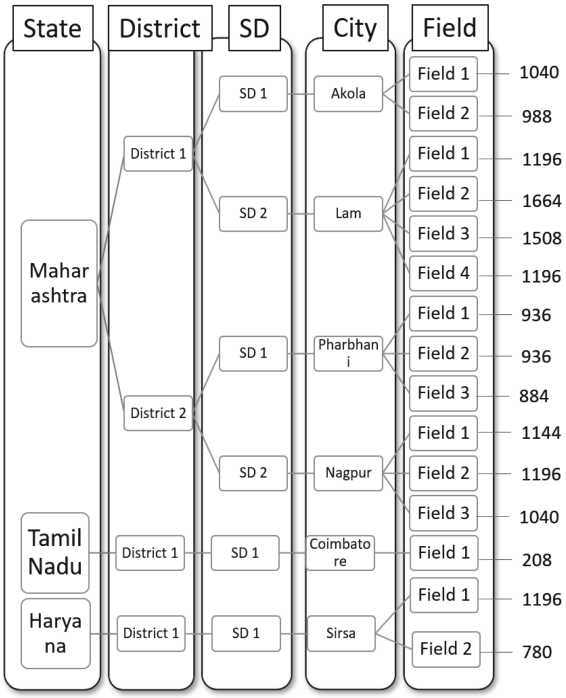


Fig. 4. Hierarchical Dataset Distribution and total data points per field.

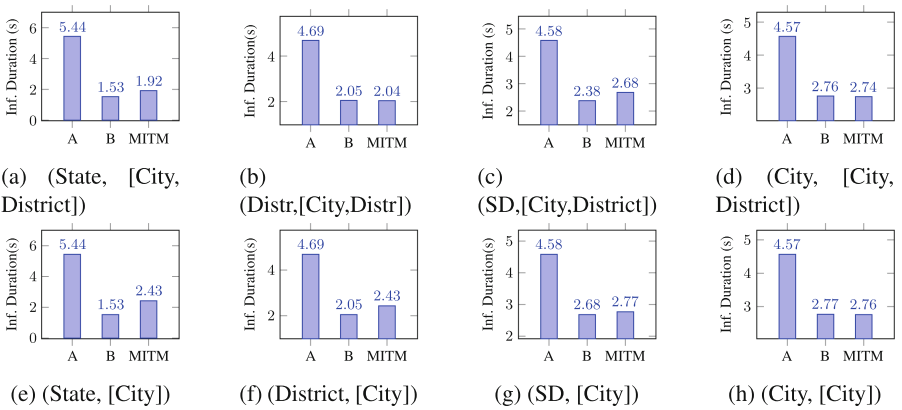
Preprocessing. As part of our experiments, we first perform adequate dataset cleaning, specifically zeroing empty attributes and ensuring all attributes are metric constraint sanitized. We then divide the aggregated data into the hierarchy defined in Fig. 4. We use this hierarchy in experiments, specifically, to introduce multiple levels of heterogeneity that we can test around - differing numbers of tier-to-sub-tier relationships, differing diseases, units for diseases, and the number of data points for the diseases in each field. We make sure to distribute city-specific data points into fields under that city only and only partition data, in year-long batches, so each model gets to observe and learn time-of-year and pest-level correlations properly. This allows the possibility of the same disease to show in numerous fields but with varying data points, local to their geography, which is a trend we are likely to find in real life as well.

Experiment Setup. Each field model contains one regression model per disease that it contains data for. Each model is a 2-layer Neural Network, implemented using Keras, with 32 fully connected neurons and 1 output neuron. Each model is trained to optimize for the Mean Squared Error loss function and we rely on the Adam optimizer for faster convergence. Each model is trained over 50 global iterations using a 32-datapoint batch size and a learning rate of 0.001 for all experiments. All experiments are run on Google Colaboratory's Nvidia Tesla K80 GPU Back-end with randomly initialized starting parameters for each field model. We aim to answer-"Given an input weather vector, and a tier, what are the different pests, and their inferred disease values, from the granularity of the specified tier".

Test Notation. We utilize the (tier of request, [cached tiers]) notation for MITM experiment results. We use the notation Π_{tier} to signify that T_{tier} is cached in this setting.

7.2 Results: Inference Latency

We present a detailed analysis of inference duration across all three frameworks in Fig. 5. We test across multiple settings and find that Framework B outperforms all other frameworks, owing to its in-situ inference. In contrast, Framework A can take almost 2x as much inference time, in some cases. **On the other hand, the MITM Staggered Caching Framework performs much closer to B all around, specifically in the (State, [City]) setting, which is a cache miss, despite which, MITM records almost 50% lesser latency than framework A**. During Cache hits, MITM performs, expectedly, at par with Framework B, while for Cache misses, MITM performs slightly worse than B but much better than A throughout, thus establishing MITM's superiority over Framework A in inference latency.



(a) (State, [City, District]) (b) (Distr,[City,Distr]) (c) (SD,[City,District]) (d) (City, [City, District])

(e) (State, [City]) (f) (District, [City]) (g) (SD, [City]) (h) (City, [City])

Fig. 5. Inference Duration metrics across Framework A, B and MITM in (tier of interest, [cached tiers]) pairings. MITM performs at par with complete caching in most cases.

7.3 Results: Accuracy

Through our experiments, we define accuracy as how close the output inferences at the tier of interest ($\tilde{i}$) are, to the disease metrics recorded at the physical model level. To compare the differences across the frameworks, we use the mean squared error (MSE) measure (the lower, the better). Experimental Results in Table 3 show that Framework A records the smallest MSE values while B shows maximum deviation from ground truth. This is expected since Framework A is similar to weighted averaging in ensemble learning, which performs well for value aggregation [11]. In contrast, B performs sub-optimally since the local applicability of data is lost when we perform inference at a higher tier using models that were optimal at lower tiers. Our model selection algorithm reduces this by using a global evaluation dataset to compute the best globally applicable model. **Owing to this, we note that MITM performs almost as well as A, and much better than B across most scenarios, as seen in the settings of (State, [City]) and (State, [SD]) queries (theoretical worst case)**.

7.4 Results: Memory Used

We report the amount of constant memory required for caching across all frameworks. Since framework A performs no caching, we compare caching memory requirements across frameworks B and MITM only. In addition to reduced memory footprint in comparison to B, we also notice that as the tier of caching moves higher, our memory for cache storage reduces, since the number of nodes of computation decreases as one moves up the pyramid resembling hierarchy(Table 4).

Table 3. Accuracy(MSE, lower the better) across Frameworks A, B and MITM. MITM performs almost at par with A across worst-case settings (Π_{City},$\tilde{i}$=State)

		$\tilde{i}$ = State		
	Π_{City}	Π_{SD}	$\Pi_{District}$	Π_{State}
A	1041.84	1041.84	1041.84	1041.84
B	1726.9	1726.9	1726.9	1726.9
MITM	1084.63	1097.0	1246.65	1683.87
		$\tilde{i}$=District		
	Π_{City}	Π_{SD}	$\Pi_{District}$	Π_{State}
A	1033.74	1052.34	1075.39	x
B	1133.24	1206.64	1175.3	x
MITM	1147.66	1182.6	1176.1	x
		$\tilde{i}$=SD		
	Π_{City}	Π_{SD}	$\Pi_{District}$	Π_{State}
A	1058.75	1036.21	x	x
B	1180.20	1146.84	x	x
MITM	1190.68	1142.54	x	x

Table 4. Memory used for Caching across Frameworks B and MITM (in KB). MITM records a significant drop in usage across all settings.

	Π_{City}	$\Pi_{SubDistrict}$	$\Pi_{District}$	Π_{State}
B	604.36	604.36	604.36	604.36
MITM	117.92	117.92	141.77	104.48
-(%)	80.48	80.48	76.54	82.71

7.5 Results: Data Transmitted over Network

We also measure the data transmitted over the network *during inference* in Table 5. Since Framework B caches models at every tier, we measure B's metric as 0. In contrast, in A, communication overhead increases proportionally to the interest tier. However, with MITM, a reduction in data communicated over the network is witnessed, in comparison to A, even as we cache lower tiers and our request tier is much higher. **This is specifically seen in the $\tilde{i}$ = *State* setting, where MITM sees a reduction of almost 50% per query (cached at City), against A. The metric reduces further as our cache tier moves higher, upto 80% for District caches.**

7.6 Experiments Discussion

We thoroughly assess the performance of all three frameworks across multiple caching scenarios and associated metrics. **Intelligent MITM tiering pro-**

Table 5. Network Transmission Cost Comparison (in KB). 'x' marks non-existent combinations. MITM outperforms Framework A in all cases.

	$\tilde{i}$=*City*	$\tilde{i}$=*SD*	$\tilde{i}$=*District*	$\tilde{i}$=*State*
A	0.968	1.472	1.976	2.384
B	0	0	0	0
		Π_{City}		
	$\tilde{i}$=*City*	$\tilde{i}$=*SD*	$\tilde{i}$=*District*	$\tilde{i}$=*State*
MITM	0	0.504	1.008	1.416
		Π_{SD}		
	$\tilde{i}$=*City*	$\tilde{i}$=*SD*	$\tilde{i}$=*District*	$\tilde{i}$=*State*
MITM	x	0	0.504	0.912
		$\Pi_{District}$		
	$\tilde{i}$=*City*	$\tilde{i}$=*SD*	$\tilde{i}$=*District*	$\tilde{i}$=*State*
MITM	x	x	0	0.448

vides a significant advantage over no-caching and complete caching frameworks. Although we present individual learnings in each section, a holistic perspective paints an impressive picture. An example of this can be seen in the [State, [City]] tier, where we see almost similar accuracy as Framework A in 50% of the inference time, half the data transmitted over the network per query, and using 80% less memory than B. Suitable, smart placement of intermittent caching provides the benefits of both fast and accurate inference as well as low memory and data traversal costs.

8 Conclusion

We address the problem of hierarchical tier-specific inference in a multi-tier multi-model setup with edge models. Two frameworks in this regard are studied and assessed, and a staggered meet-in-the-middle model caching approach is proposed. Empirical studies on a crop disease prediction task show that the proposed MITM method significantly outperforms purely bottom-up and tiered caching protocols. The proposed approach may easily be extended to other continuous learning hierarchical inference tasks like traffic data analysis and large IIoT settings.

References

1. Al-Atat, G., Fresa, A., Behera, A.P., Moothedath, V.N., Gross, J., Champati, J.P.: The case for hierarchical deep learning inference at the network edge. In: Proceedings of the 1st International Workshop on Networked AI Systems. NetAISys '23, ACM (2023). https://doi.org/10.1145/3597062.3597278
2. Chen, Y., Liestman, A., Liu, J.: Energy-efficient data aggregation hierarchy for wireless sensor networks. In: Second International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QSHINE'05) (2005). https://doi.org/10.1109/QSHINE.2005.21
3. Cui, Y., Cao, K., Zhou, J., Wei, T.: Optimizing training efficiency and cost of hierarchical federated learning in heterogeneous mobile-edge cloud computing. Trans. Comp.-Aided Des. Integ. Cir. Sys. **42**(5), 1518–1531 (sep 2022). https://doi.org/10.1109/TCAD.2022.3205551, https://doi.org/10.1109/TCAD.2022.3205551
4. Deligiannakis, A., Kotidis, Y., Roussopoulos, N.: Hierarchical in-network data aggregation with quality guarantees. In: Bertino, E., Christodoulakis, S., Plexousakis, D., Christophides, V., Koubarakis, M., Böhm, K., Ferrari, E. (eds.) Advances in Database Technology - EDBT 2004, pp. 658–675. Springer, Berlin Heidelberg, Berlin, Heidelberg (2004)
5. Einziger, G., Goldstein, M., Sa'ar, Y., Segall, I.: Verifying robustness of gradient boosted models. Proceedings of the AAAI Conference on Artificial Intelligence **33**(01), 2446–2453 (2019). https://doi.org/10.1609/aaai.v33i01.33012446
6. Guo, T., Walls, R.J., Ogden, S.S.: Edgeserve: Efficient deep learning model caching at the edge. In: Proceedings of the 4th ACM/IEEE Symposium on Edge Computing. p. 313–315. SEC '19, ACM (2019). https://doi.org/10.1145/3318216.3363370, https://doi.org/10.1145/3318216.3363370
7. He, Q., Dong, Z., Chen, F., Deng, S., Liang, W., Yang, Y.: Pyramid: Enabling hierarchical neural networks with edge computing. In: Proceedings of the ACM Web Conference 2022. p. 1860–1870. WWW '22, ACM, New York, NY, USA (2022). https://doi.org/10.1145/3485447.3511990, https://doi.org/10.1145/3485447.3511990
8. Hou, W., Wen, H., Zhang, N., Lei, W., Lin, H., Han, Z., Liu, Q.: Adaptive training and aggregation for federated learning in multi-tier computing networks. IEEE Transactions on Mobile Computing pp. 1–13 (2023). https://doi.org/10.1109/TMC.2023.3289940
9. Huang, F., Xie, G., Xiao, R.: Research on ensemble learning. In: 2009 International Conference on Artificial Intelligence and Computational Intelligence. vol. 3, pp. 249–252 (2009). https://doi.org/10.1109/AICI.2009.235
10. ICAR: Crop pest decision support system. http://www.icar-crida.res.in:8080/naip/index.jsp (2023), http://www.icar-crida.res.in:8080/naip/index.jsp, accessed: 2023-08-16
11. Jesus, P., Baquero, C., Almeida, P.S.: A survey of distributed data aggregation algorithms. IEEE Communications Surveys Tutorials **17**(1), 381–404 (2015). https://doi.org/10.1109/COMST.2014.2354398
12. Murshed, M.G.S., Murphy, C., Hou, D., Khan, N., Ananthanarayanan, G., Hussain, F.: Machine learning at the network edge: A survey. ACM Comput. Surv. **54**(8) (2021). https://doi.org/10.1145/3469029, https://doi.org/10.1145/3469029
13. Qu, Z., Duan, R., Chen, L., Xu, J., Lu, Z., Liu, Y.: Context-aware online client selection for hierarchical federated learning. IEEE Trans. Parallel Distrib. Syst. **33**(12), 4353–4367 (2022). https://doi.org/10.1109/TPDS.2022.3186960

14. Sarraute, C., Buffet, O., Hoffmann, J.: Pomdps make better hackers: Accounting for uncertainty in penetration testing. Proceedings of the AAAI Conference on Artificial Intelligence **26**(1), 1816–1824 (2021). https://doi.org/10.1609/aaai.v26i1.8363
15. Sheng, V.S., Zhang, J.: Machine learning with crowdsourcing: A brief summary of the past research and future directions. Proceedings of the AAAI Conference on Artificial Intelligence **33**(01), 9837–9843 (2019). https://doi.org/10.1609/aaai.v33i01.33019837
16. Wan, S., Yang, H.: Comparison among methods of ensemble learning. In: 2013 International Symposium on Biometrics and Security Technologies. pp. 286–290 (2013). https://doi.org/10.1109/ISBAST.2013.50
17. Wu, Q., Chen, X., Ouyang, T., Zhou, Z., Zhang, X., Yang, S., Zhang, J.: Hiflash: Communication-efficient hierarchical federated learning with adaptive staleness control and heterogeneity-aware client-edge association. IEEE Trans. Parallel Distrib. Syst. **34**(5), 1560–1579 (2023). https://doi.org/10.1109/TPDS.2023.3238049
18. Yao, M., Chen, L., Zhang, J., Huang, J., Wu, J.: Loading cost-aware model caching and request routing for cooperative edge inference. In: ICC 2022 - IEEE International Conference on Communications. pp. 2327–2332 (2022). https://doi.org/10.1109/ICC45855.2022.9838823
19. Zhang, J., Shen, Y., Wang, Y., Zhang, X., Wang, J.: Dual-timescale resource allocation for collaborative service caching and computation offloading in iot systems. IEEE Trans. Industr. Inf. **19**(2), 1735–1746 (2023). https://doi.org/10.1109/TII.2022.3186039
20. Zhou, K., Yang, Y., Qiao, Y., Xiang, T.: Domain adaptive ensemble learning. IEEE Trans. Image Process. **30**, 8008–8018 (2021). https://doi.org/10.1109/TIP.2021.3112012

ViT^2 - Pre-training Vision Transformers for Visual Times Series Forecasting

Vasileios Kochliaridis(✉), Ioannis Pierros, Georgios Romanos, and Ioannis Vlahavas

School of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece
vkochlia@csd.auth.gr

Abstract. Computer Vision has witnessed remarkable advancements through the utilization of large Transformer architectures, such as Vision Transformer (ViT). These models achieve impressive performance and generalization capability when trained on large datasets and can be fine-tuned on custom image datasets through transfer learning techniques. On the other hand, time series forecasting models have struggled to achieve a similar level of generalization across diverse datasets. This paper presents ViT^2, a framework composed of four modules, that addresses probabilistic price forecasting and generalization for cryptocurrency markets. The first module injects noise into the time series data to increase sample availability. The second module transforms the time series data into visual data, using Gramian Angular Fields. The third module converts the ViT architecture into a probabilistic forecasting model. Finally, the fourth module employs Transfer Learning and fine-tuning techniques to enhance its performance on low-resource datasets. Our findings reveal that ViT^2 outperforms State-Of-The-Art time series forecasting models across the majority of the datasets evaluated, highlighting the potential of Computer Vision models in the probabilistic time series forecasting domain. The code and models are publicly available at: https://github.com/kochlisGit/VIT2.

Keywords: Gramian Angular Fields · Vision Transformers · Probabilistic Time series Forecasting

1 Introduction

Inspired by the Transformer scaling achievement and performance in Natural Language Processing (NLP) [10], multiple works in the Computer Vision (CV) domain combined Convolutional Neural Networks (CNNs) with Self-Attention architectures [2], while others replaced them entirely [11]. An important large Transformer model that has demonstrated exceptional capabilities in generalizing patterns and handling complex image recognition tasks is the Vision Transformer (ViT) [4].

ViT has shown remarkable generalization results on large-scale datasets, like ImageNet [3]. Moreover, it can be fine-tuned for more domain-specific datasets

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 217–231, 2025.
https://doi.org/10.1007/978-3-031-78183-4_14

with less volume of data, through Transfer Learning. This process involves taking a pre-trained model, which has already learned a rich set of features from the large image dataset and adapting it to a new, but related task [13], reducing the required data availability, as well as the required training time and computational resources.

While Transfer Learning is highly successful in the CV and NLP domains, it is less straightforward and effective in time series forecasting. This can be partly because multi-dimensional time series can be highly diverse even in similar domains. Such diversity is particularly present in cryptocurrency markets, as demonstrated in Figure 1, making it challenging to find a global approach or model that can generalize well within the time series domain [15] in the same way that models like ViT do with images.

Fig. 1. Candlesticks for DAI-USD symbol pair in 3 different exchanges. It can be noticed that this pair presents different trend in Exmo exchange. Additionally, the high-low price ranges, represented by the vertical lines, are quite different in all 3 exchanges.

Several State-Of-The-Art Autoencoders and Transformer architectures have been proposed for probabilistic time series forecasting, including Temporal Fusion Transformers (TFT) [6], N-BEATS [8] and NHITS [2], which stand out due to their impressive performance. However, analysis of these models in several tasks, such as M3 and M4 competitions [9], in financial tasks [5], as well as our work, reveal that such models achieve poor generalization performance, especially in tasks with low-resource data and gain small advantage by employing Transfer Learning. Because there are several symbols (cryptocurrency tokens) from various cryptocurrency exchanges in our work, these limitations would normally require models to be uniquely trained for each symbol of each exchange, requiring a lot of training time and computational resources to achieve overall satisfying performance.

To tackle these limitations, several time series to image encoding techniques have been proposed [1,9,12,16], which enable the utilization of CNN-based models and large transformers that achieve high performance in image classification tasks for time series forecasting tasks. Although these models demonstrate promising results, there has been limited research on pre-trained transformers specifically designed for probabilistic time series forecasting. Most existing

research focuses on time series classification, while lacking the exploration of Transfer Learning [14].

In this paper, we propose ViT^2, a framework designed to overcome the limitations of Transfer Learning and generalization challenges in the probabilistic time series forecasting domain by implementing four modules. The first module collects a large-scale dataset and injects Gaussian Noise to generate synthetic time series samples. The second module applies Gramian Angular Fields (GAF) transformations to time series features, converting them into images which are then stacked along the depth axis, forming multi-channel images. The third module modifies the ViT architecture to be compatible with probabilistic time series forecasting tasks and trains it on the generated dataset. The final module applies Transfer Learning and fine-tunes the model on a smaller-scale dataset, in order to evaluate it in terms of performance and generalization capability.

Additionally, we present comprehensive comparison between the impact of Transfer Learning techniques of our framework against several leading probabilistic forecasting models, such as the TFT, N-BEATS, and N-HiTS. Our findings demonstrate that ViT^2 achieves very satisfying results compared to previous State-Of-The-Art approaches, highlighting the robustness and effectiveness of CV-based models in the probabilistic time series forecasting domain. Our contribution can be summarized as follows:

- Integrate GAF with ViT for converting time series data into images, leveraging ViT's strengths in image classification for probabilistic time series forecasting.
- Implement an end-to-end framework, which benefits from Transfer Learning techniques, in comparison to leading probabilistic time series forecasting models, such as TFT, N-Beats, NHits.

The remainder of this paper is organized into the following sections. Section 2 presents all the research prior to this work. Section 3 reviews all the necessary literature. Section 4 covers thoroughly the four modules that compose our framework. Section 5 presents the datasets, the metrics and the experimental setup and discusses the experimental results. Finally, Section 6 concludes this work and proposes future extensions.

2 Related Work

The Temporal Fusion Transformer (TFT) is a significant advancement in time series forecasting, integrating the strengths of Long-Short-Term-Memory Networks (LSTMs) and attention mechanisms to address complex forecasting tasks [6]. TFT employs a multi-head attention mechanism and Gated Residual Networks to selectively focus on relevant inputs and make accurate forecasts. Although TFT demonstrates robust performance across several popular time series forecasting tasks, it requires very long training time and heavy computational resources and the effectiveness of Transfer Learning in TFT is still unexplored.

N-BEATS [8] is also a popular neural network architecture, designed to address the computational intensity and required resources of transformer-based models in time series forecasting. N-BEATS utilizes a stack of residual blocks, allowing for a deep understanding of the time series components, such as trend and seasonality, through its backward and forward residual links. Later, N-HiTS was built upon the N-BEATS framework to introduce hierarchical temporal aggregation, further enhancing the model's ability to capture multiple time scales and complex patterns from the given inputs [2].

Qi-Qiao et al. [5] explored applied Transfer Learning in LSTM Networks for forecasting financial time series in their study . The authors pre-trained LSTMs on two similar datasets before fine-tuning them on target-specific datasets, thereby demonstrating the effectiveness of Transfer Learning in enhancing LSTM model performance. However, in their research, they did not explore the potential benefits of incorporating additional dataset sources and measure their impact on LSTM's forecasting capability. Furthermore, they completely disregarded alternative architectures, such as CNN models, which have also proven very efficient in computer vision tasks through Transfer Learning.

Barra et al. [1]diverged from traditional time series forecasting methodologies by adopting an ensemble of CNN models, such as ResNet and AlexNet for predicting stock prices within the *S&P500* market. They further enhanced their model by transforming the time series data into images through the Gramian Angular Field (GAF) technique, a method which has proven quite effective in time series classification tasks. While the GAF imaging technique boosted the performance of the ensemble model, their models consist solely of Residual blocks, as well as Convolutional and Pooling layers, ignoring completely more advanced architectures, such as Transformers. Additionally, despite their model demonstrating satisfying generalization capabilities, the potential benefits of applying transfer learning techniques to these models was not explored, according to the authors.

Finally, in a recent study by Semenoglou et al [9], the authors proposed a method of encoding time series data into visual formats for analysis, specifically by transforming the data into 2D grayscale plots and then utilizing CNN-based models like ResNet [9]. This technique outperformed the performance of leading time series forecasting models, including N-BEATS and NHits. However, we noticed that the proposed conversion process to 2D plots inadvertently includes non-essential information, such as black pixels surrounding the actual time series data, represented in white color. Moreover, the study overlooks alternative methods of converting time series to images, such as GAF, which might have offered improved results by minimizing the inclusion of irrelevant data.

To conclude this short literature review, combining time series to image encoding methods with Computer-Vision-based models have yielded very promising results in the domain of time series forecasting. However, in comparison to previous works, our approach suggests pre-training the Computer Vision models in large-scale datasets, before applying them on target-specific datasets. Furthermore, we utilize the Vision Transformer (ViT), due to its exceptional

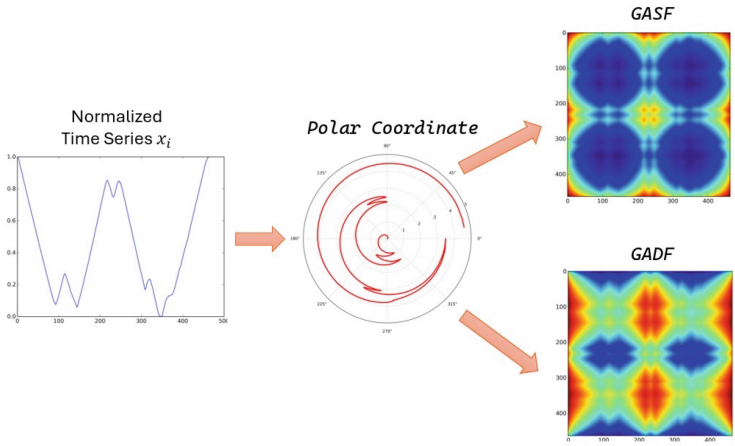


Fig. 2. The encoding process of GAF. The input time series becomes a 2D image.

performance and generalization capability in image recognition tasks and adapt it into a probabilistic forecasting model. Lastly, we apply noise into the inputs, aiming to train ViT with further samples, and thus enhance the robustness and accuracy of its forecasts.

3 Background

In this section, we provide all the necessary literature that was utilized in our work. This consists of the GAF, an imaging technique which has proven quite efficient in time series recognition tasks, as well as the original ViT architecture, which is a State-Of-The-Art deep learning model designed for Computer-Vision-related tasks, such as image classification.

3.1 Gramian Angular Fields

GAF is a time series to image transformation technique that allows deep learning models designed for visual data to recognise and capture complex patterns from time series data. [12]. This transformation is achieved by first normalizing the time series in ranges $[-1.0, 1.0]$ or $[0.0, 1.0]$ and then encoding them into matrices based on angular information, capturing the temporal correlation between different points in the series. The process involves computing the trigonometric functions (cosine and sine) of the cumulative sum of time series data, which are then mapped into a polar coordinate system. The resulting GAF can be either a Gramian Angular Summation Field (GASF) or a Gramian Angular Difference Field (GADF), depending on whether the summation or difference of angles is used, as presented in Figure 2.

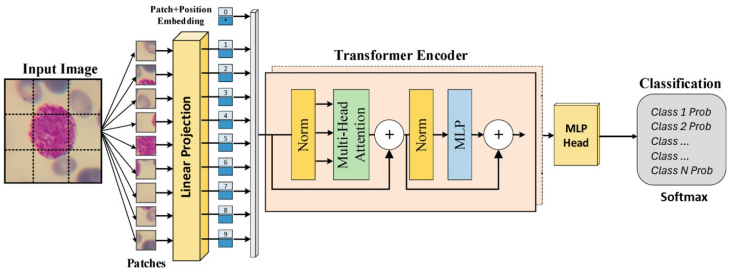


Fig. 3. The original Vit Architecture designed for image classification tasks.

3.2 Vision Transformer

Unlike conventional CNNs that process images through localized filters, ViT divides an image into a sequence of fixed-size patches and then processes each patch sequentially to capture both local and global image features as presented in Figure 3. This approach enables ViT to learn complex patterns, such as different parts of the image that interact together, in a way that deep CNNs can struggle with due to their inherently local nature of processing [4]. ViT has demonstrated remarkable performance on various image recognition benchmarks, often outperforming state-of-the-art CNNs, especially when pre-trained on large-scale image datasets and then fine-tuned for specific tasks via Transfer Learning techniques.

4 Methodology

In this section, we describe the methodology of our work, which is composed in four modules, as described in Figure 4. The first module applies data augmentation via Gaussian noise injection into candlesticks data, which are described in Section 4, further increasing data availability. The second module applies the GAF method to transform every time series features into images, which are then stacked together along the depth axis to form multi-channel images. The third module modifies the original ViT architecture to be compatible with the multi-channel input images and converts it into a probabilistic forecasting model. Then, the modified architecture is trained using the constructed images. The final module applies Transfer Learning and fine-tunes the trained model on a low-resource datasets.

4.1 Module 1 - Data Preprocessing

The ViT has showcased satisfying performance and generalization capability on large-scale datasets. Consequently, the first module employs a data augmentation strategy to enhance the data availability, aiming to further improve the framework's performance. Because the generated GAF images are heavily dependent on the input time series, applying data augmentations on the constructed images

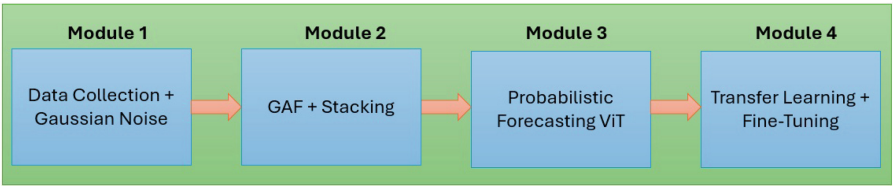


Fig. 4. The four modules of constructing probabilistic forecasting ViT model.s

could potentially distort the market trends and price signals. Hence, the data augmentations are applied directly on the time series prior to GAF Transformations.

To generate new time series data, we inject noise to the original input data, defined as $D \sim \mathcal{N}(0, \sigma^2)$, which is normally distributed with mean $\mu = 0$ and standard deviation $\sigma = e$, with σ being a small constant value ($\sigma << 1.0$) defined by the user. Subsequently, each new sample is generated as $x'_i = x_i + d_i$, as demonstrated in Figure 5.

Fig. 5. Comparison of the original close prices of BTCUSDT (c_t) with a noisy BTCUSDT ($c'_t = c_t + d_{i,t}$), $D \sim \mathcal{N}(0, 0.1)$.

In the final step of our data preparation process, we focus on extracting the target values for our model, which correspond to the closing price of each cryptocurrency token for the subsequent time step, denoted as y_{t+1}. Because the targets are absolute prices, it can be challenging to directly predict them. To address this, we employ the Log Returns transformation on the target values,

which is described in Equation 1. This is a common transformation in financial time series forecasting, because it standardizes the scale of price changes, and helps stabilize their variance.

$$y_{t+1} = ln(\frac{c_{t+1}}{c_t}) \tag{1}$$

4.2 Module 2 - Visualizing Time Series

The second module begins by normalizing the data using MinMax technique, as described in Equation 2, allowing the GAF transformation to be applied. Then, it segments each input time series x_i into timeframes $f_{i,t}$, each of size S, with S being a user-defined setting. Finally, it utilizes the Gramian Angular Summation Field (GASF) to convert the timeframes into 2D matrices, defined as $m_{i,t} \in^{S \times S}$. Finally, each image is stacked together, forming a multi-channel image defined as $I_{i,t} \in^{S \times S \times N}$, with N being the number of features. The entire pipeline of this module can be visualized in Figure 6

$$x'_i = \frac{x_i - min(x_i)}{max(x_i) - min(x_i)} \tag{2}$$

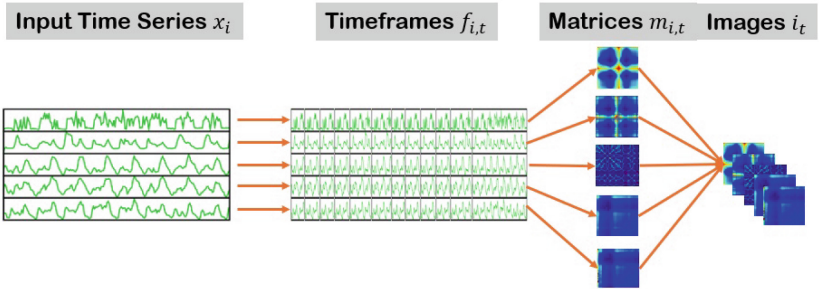


Fig. 6. Pipeline of multivariate time series to multi-channel images. Each image has N channels.

4.3 Module 3 - Probabilistic Forecasting ViT

The original ViT-base architecture features an input layer of shape $(224, 224, 3)$, followed by the Transformer-Encoder model, which is paired with an MLP head. Its output is composed of a final linear layer whose units match the number of classes to be recognized, followed by a softmax activation function that outputs the class probabilities. In contrast, the input layer is adjusted to (S, S, N) dimensions, in order to be compatible with our input images. Moreover, the final linear layer and softmax activation are replaced with singular linear unit, which predicts the target value y_{t+1}.

Finally, the Multi-Quantile loss function is utilized, which is also used in TFT architecture [6]. Unlike traditional regression loss functions, such as Mean Squared Error (MSE) and Mean Absolute Error (MAE), which primarily estimates the mean of the target variable, quantile regression aims to predict specific quantiles simultaneously, providing a more comprehensive view of the possible outcomes. The Multi-Quantile loss function requires the model to predict K outputs, as presented in Figure 7, each one for a single quantile $q_j \in (0, 1.0)$. The quantile loss for a single quantile is described in Equation 3.

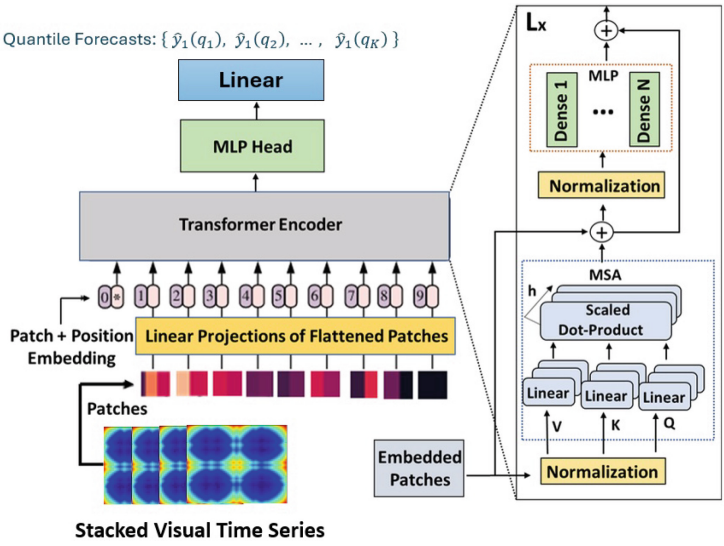


Fig. 7. Probabilistic Forecasting ViT architecture

$$L_q(y, \hat{y}) = \begin{cases} q \cdot (y - \hat{y}) & \text{if } y \geq \hat{y}, \\ (1 - q) \cdot (\hat{y} - y) & \text{if } y < \hat{y}. \end{cases} \tag{3}$$

4.4 Module 4 - Transfer Learning & Fine-Tuning

To further enhance the performance of our framework, we initially train our model on the large-scale dataset, which contains a large collection of time series datasets. This provides a significantly larger volume of data for the model to train on, improving its learning capacity and ability to generalize domain-specific time series.

Subsequently, we transfer the weights of the pre-trained model to a new model, which we then fine-tune it on the target dataset, which is the small-scale dataset. Despite the reduced size and the fewer time series samples in this dataset, the model's forecasting accuracy remains robust due to the foundational knowledge acquired during the initial training phase of the pre-trained

model. This strategy effectively reduces the data requirements for training on new, domain-specific datasets, making it a highly efficient method for adapting our model to varied contexts.

5 Experiments and Discussion

In this section, we analyze the historical market datasets that are used to conduct ViT^2 evaluation. A range of experimental procedures are additionally described to assess the performance of our framework. This section also provides details about the hyper-parameter tuning and concludes by presenting the experimental results.

5.1 Datasets

In this study, we gathered data from 151 cryptocurrency datasets, which are publicly available through the *CryptoDataDownload* platform[1]. These datasets include 14 unique exchanges and feature 46 unique symbols. Each dataset includes daily candlestick records, composed of Open-High-Low-Close-Volume (OHLCV) data points: **'Open'** represents the price at which trading began on a given day, **'High'** denotes the highest price reached during the day, **'Low'** reflects the day's lowest price, **'Close'** indicates the final trading price for the day, and **'Volume'** accounts for the total traded volume of transactions. The total count of candlesticks in each dataset is between 1000 and 2000 samples.

In addition to candlesticks, we also calculate popular technical indicators, i.e. pattern-based mathematical formulas computed by the candlesticks and are used by traders who follow technical analysis, including *EMA, VWAP, RSI, CCI* and *ADX* [7]. Finally, we include the day of week for each sample, represented by a number from 0 to 6.

5.2 Experimental Setup

We train ViT^2 in our dataset along with several other state of the art approaches, including TFT, NBEATS, NHits and ForCNN-ResNet, which are presented in Section 2. Each model is tasked to predict the logarithmic returns of the next day's close price from a given timeframe, as described in Section 4.

To measure the performance of the forecasting models, we concatenate the collected data into two distinct sets: a large-scale dataset and a small-scale dataset, with the small-scale dataset being composed of 10% of the total data. Initially, models are trained and evaluated using the large-scale dataset. Subsequently, by employing Transfer Learning, each model is fine-tuned and further evaluated on the smaller dataset. Additionally, we repeat the same process by excluding samples generated through noise injection (**NNI**), as detailed in Section 4, in order to underscore the impact of sample generation. Lastly, each

[1] https://www.cryptodatadownload.com/.

model is trained directly on the small-scale datasets without the use of Transfer Learning (**NTL**), allowing us to assess their performance in domains with low-resource data.

To ensure the integrity of our evaluation and avoid biases and overfitting issues, we divided the datasets into train, validation and test sets, as demonstrated in Figure 8. We evaluate each model's performance on the validation and test sets, using the Mean Squared Error (MSE) and the Mean Absolute Error (MAE), which are also used in [2,6,8] to measure the performance of N-BEATS and NHits.

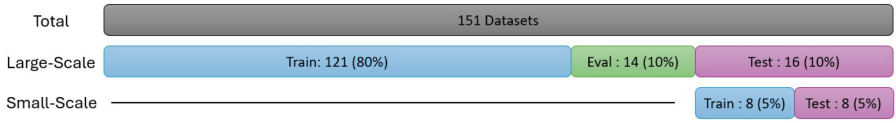


Fig. 8. Train-Test Splits of large-scale and small-scale datasets

5.3 Hyper-parameter Tuning

The hyperparameter space of ViT^2 is extensive, so we tune only the most sensitive hyperparameters. The primary variable of ViT^2 is the timeframe size, which is selected to be $S = 28$, after experimenting with various sizes varying from 16 to 48, in order to balance computational efficiency with model performance. The selected optimization algorithm is Adam, known for its effectiveness in Deep Learning models, with the learning rate set at 0.001 after a thorough experimentation within the [0.001, 0.01] interval, ensuring the model's learning process was neither too slow nor prone to converging to a local minimum point. Moreover, we include 7 quantiles of [0.02, 0.1, 0.25, 0.5, 0.75, 0.9, 0.98], a strategy which was employed in the work of the previous forecasting models. Each model is trained for 1000 epochs in both datasets, with early stopping patience of 200 epochs to address any overfitting issues. Finally, we summarize the hyperparameter selection in Table 1.

To enhance the fairness of our comparison, we manually fine-tuned the most sensitive hyperparameters for the baseline models as well. Specifically, for the TFT, we adjusted the hidden state size to 128 units and employed 2 LSTM layers of 64 units each. Our experimentation identified 4 attention heads and a dropout rate of 10% as the optimal configuration. In the case of N-Beats, we utilized 2 blocks with layer widths of 32 and 512 respectively, alongside a dropout rate of 10%. For N-HITS, we implemented 3 blocks and optimized the architecture by incorporating batch normalization, which improved its performance. With ForCNN-ResNet, we kept the original ResNet-50 architecture as detailed in the original paper. Across all models, we set the learning rate to 0.001 and the weight decay to 0.001.

Table 1. ViT^2 Hyperparameters

Parameter	Value
Timeframe Size (S)	28
Learning Rate (lr)	0.001
Quantiles (Q)	$[0.02, 0.1, 0.25, 0.5, 0.75, 0.9, 0.98]$
Epochs (e)	1000
Early Stopping Patience (p)	200
Gaussian Noise (σ)	0.05

5.4 Results

Table 2. Performance Comparison in the Large-Scale dataset

Exchange	Symbol	Model (MSE/MAE)				
-	-	Vit^2	TFT	N-BEATS	NHits	ForCNN-ResNet
Binance	**FILUSDT**	0.0037/0.0400	0.0072/0.0605	**0.0014/0.0256**	0.0024/0.0354	0.0051/0.0501
Bitbay	**ETHUSD**	0.0113/0.0506	0.0023/0.0304	**0.0028/0.0290**	0.0023/0.0304	0.02139/0.0808
Bittrex	**BTCUSDT**	**0.0010/0.0225**	0.0130/0.0850	0.0037/0.0434	0.0046/0.0493	0.0043/0.0512
CEX	**BNBUSD**	**0.0024/0.0330**	0.0068/0.0611	0.0041/0.04473	0.0038/0.0437	0.0038/0.0422
Exmo	**DOGEUSD**	0.0061/0.0487	0.0034/0.0475	**0.0006/0.0176**	0.0014/0.0272	0.0067/0.0572
Exmo	**TRXUSD**	**0.0025/0.0331**	0.0065/0.0586	0.0029/0.0347	0.0026/0.0379	0.0031/0.3981
Poloniex	**ADAUSDT**	**0.0017/0.0295**	0.0102/0.0752	0.0021/0.0315	0.0030/0.0397	0.0053/0.0621
Poloniex	**DAIUSDT**	**0.0001/0.0018**	0.0023/0.0303	0.0016/0.0223	0.0017/0.0249	0.0012/0.0136

In our initial experiment, we evaluated each model's performance on the test set after training on the large-scale dataset, which included noise injection. The outcomes are detailed in Table 2. Notably, our method surpassed the previous state-of-the-art approaches in both MSE and MAE metrics across 5 out of 8 datasets (we report on the same 8 datasets as in the small-scale test set). The results highlight the superior generalization ability of ViT^2 compared to traditional models designed specifically for time series forecasting, as well as the

Table 3. Performance Comparison in the Small-Scale dataset via Transfer Learning & without Noise Injection (**NNI**)

Exchange	Symbol	Model (MSE/MAE)				
-	-	Vit^2-**NNI**	TFT-NNI	N-BEATS-NNI	NHits-NNI	ForCNN-ResNet-NNI
Binance	**FILUSDT**	0.0037/0.0399	0.0067/0.0588	**0.0014/0.0253**	0.0028/0.0332	0.0039/0.0412
Bitbay	**ETHUSD**	0.0114/0.0505	0.0022/0.0288	**0.0017/0.0224**	0.0018/0.0243	0.0171/0.0582
Bittrex	**BTCUSDT**	**0.0010/0.0225**	0.0138/0.0871	0.0033/0.0401	0.0031/0.0396	0.0037/0.0398
CEX	**BNBUSD**	**0.0024/0.0330**	0.0048/0.0500	0.0032/0.0389	0.0035/0.0409	0.0102/0.0411
Exmo	**DOGEUSD**	0.0061//0.0386	0.0047/0.0585	**0.0003/0.0323**	0.0034/0.0416	0.0116//0.0423
Exmo	**TRXUSD**	**0.0025/0.0331**	0.0062/0.0574	0.0021/0.0320	0.0025/0.0353	0.0038/0.0626
Poloniex	**ADAUSDT**	**0.0017/0.0295**	0.0106/0.0762	0.0027/0.0352	0.0034/0.0416	0.0025/0.0228
Poloniex	**DAIUSDT**	**0.0001/0.0015**	0.0024/0.0360	0.0001/0.0084	0.0022/0.0273	0.0025/0.0201

Table 4. Performance Comparison in the Small-Scale dataset via Transfer Learning & Noise Injection

Exchange	Symbol	Model (MSE/MAE)				
-	-	Vit^2	TFT	N-BEATS	NHits	ForCNN-ResNet
Binance	**FILUSDT**	0.0032/0.0383	0.0062/0.0575	**0.0011/0.0233**	0.0029/0.0365	0.0035/0.0399
Bitbay	**ETHUSD**	0.0111/0.0495	0.0021/0.0277	0.0017/0.0217	**0.0016/0.0209**	0.0282/0.0713
Bittrex	**BTCUSDT**	0.0009/0.0188	0.0114/0.0864	0.0030/0.0383	**0.0009/0.0153**	0.0109/0.0677
CEX	**BNBUSD**	**0.0015/0.0283**	0.0075/0.0614	0.0031/0.0364	0.0021/0.0335	0.0112/0.0430
Exmo	**DOGEUSD**	0.0076//0.0427	0.0038/0.0434	**0.0002/0.0207**	0.0014/0.0259	0.0110//0.0377
Exmo	**TRXUSD**	**0.0031/0.0376**	0.0051/0.0521	0.0021/0.0320	0.0031/0.0389	0.0051/0.0542
Poloniex	**ADAUSDT**	**0.0012/0.0244**	0.0097/0.0673	0.0026/0.0342	0.0021/0.0298	0.0069/0.0631
Poloniex	**DAIUSDT**	**0.0001/0.0011**	0.0022/0.0341	0.0001/0.0046	0.0008/0.0220	0.0028/0.0488

Table 5. Performance Comparison in the Small-Scale dataset, without Transfer Learning (NTL)

Exchange	Symbol	Model (MSE/MAE)				
-	-	Vit^2-NTL	TFT-NTL	N-BEATS-NTL	NHits-NTL	ForCNN-ResNet-NTL
Binance	**FILUSDT**	0.0070/0.0427	0.0092/0.6644	0.0064/0.0435	**0.0045/0.0422**	0.0192/0.0877
Bitbay	**ETHUSD**	0.0143/0.0556	0.0022/0.0291	**0.0012/0.0209**	0.0020/0.0274	0.0188/0.0626
Bittrex	**BTCUSDT**	**0.0022/0.0233**	0.0138/0.0888	0.0076/0.0642	0.0096/0.0711	0.0202/0.0799
CEX	**BNBUSD**	0.0069/0.0497	0.009/0.0757	**0.0005/0.0512**	0.0005/0.0554	0.0198/0.0682
Exmo	**DOGEUSD**	**0.0096//0.0427**	0.3437/0.5755	0.0212/0.1386	0.0104/0.0781	0.0148//0.0583
Exmo	**TRXUSD**	0.0051/0.0376	0.0151/0.0953	**0.0029/0.0402**	0.0051/0.0523	0.0199/0.0743
Poloniex	**ADAUSDT**	0.0048/0.0301	0.0176/0.0429	0.0101/0.0721	**0.0033/0.0439**	0.0212/0.0652
Poloniex	**DAIUSDT**	0.0003/0.0098	0.0013/0.0079	**0.0002/0.0025**	0.0027/0.0243	0.0004/0.0212

ForCNN's approach, especially in *Poloniex-DAIUSDT* dataset, which achieved the same MSE as N-BEATS, but with the lowest MAE score. The accuracy of its forecasts in that particular dataset is presented in Figure 9.

In the subsequent experiments, we focus on training and evaluating the models using the small-scale dataset. Before this training phase, we apply transfer learning techniques to the models that were previously trained on the large-scale dataset. This involves fine-tuning each model specifically for the small-scale dataset. We repeat this process twice: once without implementing noise injection, thereby maintaining the original number of training samples from the large-scale dataset, and once with noise injection as described in Section 4. The results of these experiments are presented in Table 3 for the experiment without the noise injection and Table 4 for the experiment with noise injection. It can be observed that ViT^2 demonstrated a small advantage over the baseline models in most datasets. Furthermore, it can be noticed that all models improved slightly with the introduction of noise injection, marking it as a fast yet effective form of data augmentation.

In the last experiment, we train every model directly in the small-scale dataset, without utilizing Transfer Learning. As shown in Table 5, N-BEATS and NHits demonstrate the strongest overall performance, while the CV-based models underperform. This indicates CV-based models, especially ViT^2, which utilizes transformers, struggle without prior pre-training. The impact of utiliz-

ing Transfer Learning for the CV-based models can be noticed by the increased errors in several datasets. Notably, while ViT^2 showed the most promising results on the Exmo-DOGEUSDT dataset with an MSE of 0.0096 without the use of Transfer Learning, its error is dropped to 0.0076 when utilizing the pre-trained weights.

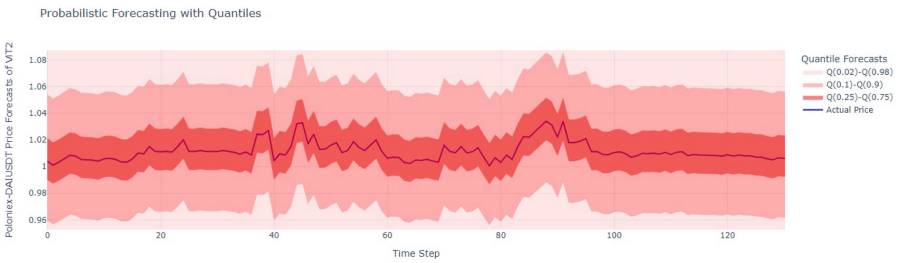


Fig. 9. Quantile Forecasts for Poloniex-DAIUSDT prices.

6 Conclusion & Future Work

In this paper, we introduce ViT^2, a novel framework aimed at addressing the challenges of generalization performance and Transfer Learning limitations in the domain of time series probabilistic forecasting, specifically focusing on cryptocurrency price estimation. Our approach leverages Gramian Angular Fields, in order to transform time series data into images effectively, so that Computer-Vision-based models can be employed. We base our framework on the ViT architecture and modify it into a probabilistic forecasting model by integrating the Multi-Quantile loss function.

Furthermore, we increase the performance of our framework,by enriching our dataset by incorporating noise injection, which is a simple yet effective technique to increase sample availability. Subsequently, we pretrain ViT on a large volume of data and then fine-tune it on the target-specific dataset, which has fewer samples. Through the utilization of these techniques, we surpass both previous State-Of-The-Art methodologies and popular time series forecasting models in most of the experiments, including TFT, N-BEATS and NHits.

In the future we aim to expand ViT^2 into other time-series-related forecasting tasks, as well as experiment with more recent Computer-Vision-based architectures, which have the potential to further contribute to the field of time series forecasting.

References

1. Barra, S., Carta, S.M., Corriga, A., Podda, A.S., Recupero, D.R.: Deep learning and time series-to-image encoding for financial forecasting. IEEE/CAA Journal of Automatica Sinica **7**(3), 683–692 (2020)
2. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European conference on computer vision. pp. 213–229. Springer (2020)
3. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
4. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
5. He, Q.Q., Pang, P.C.I., Si, Y.W.: Transfer learning for financial time series forecasting. In: PRICAI 2019: Trends in Artificial Intelligence: 16th Pacific Rim International Conference on Artificial Intelligence, Cuvu, Yanuca Island, Fiji, August 26–30, 2019, Proceedings, Part II 16. pp. 24–36. Springer (2019)
6. Lim, B., Arık, S.Ö., Loeff, N., Pfister, T.: Temporal fusion transformers for interpretable multi-horizon time series forecasting. Int. J. Forecast. **37**(4), 1748–1764 (2021)
7. Lin, T.C.: The new investor. UCLA L. Rev. **60**, 678 (2012)
8. Oreshkin, B.N., Carpov, D., Chapados, N., Bengio, Y.: N-beats: Neural basis expansion analysis for interpretable time series forecasting. In: International Conference on Learning Representations (2019)
9. Semenoglou, A.A., Spiliotis, E., Assimakopoulos, V.: Image-based time series forecasting: A deep convolutional neural network approach. Neural Netw. **157**, 39–53 (2023)
10. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
11. Wang, H., Zhu, Y., Green, B., Adam, H., Yuille, A., Chen, L.C.: Axial-deeplab: Stand-alone axial-attention for panoptic segmentation. In: European conference on computer vision. pp. 108–126. Springer (2020)
12. Wang, Z., Oates, T.: Imaging time-series to improve classification and imputation. In: Proceedings of the 24th International Conference on Artificial Intelligence. pp. 3939–3945 (2015)
13. Weiss, K., Khoshgoftaar, T.M., Wang, D.: A survey of transfer learning. Journal of Big data **3**, 1–40 (2016)
14. Wen, Q., Zhou, T., Zhang, C., Chen, W., Ma, Z., Yan, J., Sun, L.: Transformers in time series: A survey. arXiv preprint arXiv:2202.07125 (2022)
15. Wen, Q., Zhou, T., Zhang, C., Chen, W., Ma, Z., Yan, J., Sun, L.: Transformers in time series: A survey. In: Elkind, E. (ed.) Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23. pp. 6778–6786. International Joint Conferences on Artificial Intelligence Organization (8 2023). https://doi.org/10.24963/ijcai.2023/759, survey Track
16. Zheng, Z.: A novel air quality prediction method based on gaf and dense net. In: 2022 International Conference on Machine Learning and Intelligent Systems Engineering (MLISE). pp. 105–110. IEEE (2022)

waLLMartCache: A Distributed, Multi-tenant and Enhanced Semantic Caching System for LLMs

Soumik Dasgupta, Anurag Wagh, Lalitdutt Parsai, Binay Gupta, Geet Vudata, Shally Sangal, Sohom Majumdar, Hema Rajesh, Kunal Banerjee(✉), and Anirban Chatterjee

Walmart Global Tech, Bangalore, India
{soumik.dasgupta, anurag.wagh, lalitdutt.parsai, binay.gupta, geet.vudata, shally.sangal, sohom.majumdar, hema.rajesh, kunal.banerjee1, anirban.chatterjee}@walmart.com

Abstract. In recent times, Large Language Models (LLMs) have become an important tool for various business use cases. The power of these LLMs can be leveraged in improving developer productivity, as a knowledge base, for creating applications such as question answering and information retrieval systems. Unfortunately, these benefits come at the price of high usage costs and latency. The managed LLM services charge based on tokens (words) processed which become very significant with scale. Even self-hosted open source LLMs turn out to be expensive because hosting requires expensive GPUs and scaling to many requests would need significant horizontal scaling of these resources. In this context, an LLM focused caching system can significantly reduce usage costs as well as latency. This problem is addressed by GPTCache. The current work termed waLLMartCache advances GPTCache by incorporating the following features: (i) we introduce the support for a new database Redis in GPTCache (our pull request is already merged with GPTCache main branch) – this is used as L2 cache in our designed system, (ii) presently, GPTCache is implemented to be run on a single node which we enhance to span across multiple nodes to handle industry-scale requests and consequently, we also designed a distributed eviction manager, (iii) we further create partitions for individual tenants (clients) so that these can be hosted together while maintaining semantic separations, (iv) we present a *decision engine* that decides whether to cache an LLM response based on our business use-cases, and (v) we showcase that loading FAQs (which can be set to be stored persistently in the memory) while booting the LLM cache can be a simple yet effective strategy to boost cache hits significantly. Although this system is in-house to our company, we believe that the methodology shared in this paper is generic enough to be adopted by any organization.

Keywords: LLM · GPTCache · Redis · Distributed system · Eviction manager

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 232–248, 2025.
https://doi.org/10.1007/978-3-031-78183-4_15

1 Introduction

Large Language Model (LLM) is considered as one of the biggest disruptors in technology nowadays. Although language models have persisted for many years [7], LLMs have become immensely popular since the release of GPT-3 [9] by OpenAI that later caught more attention with the release of GPT-3.5 [32] and GPT-4 [21]. These LLMs have been found to be good at various tasks, such as answering questions, summarizing texts, synthesizing new content, etc. while newer applications involving the LLMs are being invented almost every day. However, accessing the GPTs can be quite costly [20]; moreover, accessing them using APIs typically involves non-uniform latencies – the situation is further exacerbated if too many requests come from a user and consequently, the service is temporarily revoked for that user; in such cases, the service is resumed only after some cool-down period. Such untoward events hamper developer productivity and customer experience significantly which, in turn, may lead to revenue and/or reputation loss for businesses that are dependent on the GPTs.

To alleviate these problems, GPTCache was introduced in [5]. GPTCache, similar to the concept of the standard caches, stores responses from previous queries sent to the LLMs and whenever a *"similar"* query is posed again, the response is returned from the cache instead of invoking the LLM again. Thus, GPTCache not only saves the cost incurred by availing the services from the GPTs but also reduces the response speed by up to 10x [5]. Moreover, returning the response from the cache is unaffected by network fluctuations – thereby making the system more stable; inadvertent lockout periods from the GPT services are also reduced if not avoided altogether.

Notably, GPTCache, in contrast to what its name may suggest, is applicable to any LLM such as, Gemini [3], Claude [4], Llama2 [28], Falcon [2], Mistral [16] and Dolly [12] just to name a few. Although GPTCache's benefits are undoubtedly great, adopting it off-the-shelf for industry use-cases posed many challenges, especially when it came to scaling the cache to a distributed system to handle thousands of requests per minute. Consequently, we had to make upgrades to the GPTCache which resulted into its updated version that we termed as *waLLMartCache*. Our system waLLMartCache advances GPTCache by incorporating the following features:

1. We introduce the support for a new database Redis [27] in GPTCache (our pull request is already merged with GPTCache main branch[1]) – this is used as L2 cache in our designed system.
2. Currently, GPTCache is implemented to be run on a single node which we enhance to span across multiple nodes to handle industry-scale requests and consequently, we also designed a distributed eviction manager.
3. We further create partitions for individual tenants (clients) so that these can be hosted together while maintaining semantic separations.
4. We present a *decision engine* that decides whether to cache an LLM response based on retail business use-cases.

[1] https://github.com/zilliztech/GPTCache/releases/tag/0.1.35

5. We showcase that loading FAQs (which can be set to be stored persistently in the memory) while booting the LLM cache can be a simple yet effective strategy to boost cache hits significantly.

It is important to note that although waLLMartCache is in-house to our company, we believe that our methodology is generic enough to be adopted by any organization.

The paper is organized as follows. Section 2 covers the related literature. Section 3 describes GPTCache in brief. Section 4 explains our enhancements made on top of GPTCache along with some empirical evidence of waLLMartCache's efficacy. Section 5 concludes the paper with a summary of the work done and some potential future directions.

2 Related Work

Cache memory is a well-known concept in computer architecture whereby certain information (instructions and/or data) is stored in a designated memory that has a lower latency than the primary memory [15]. Semantic caching [11] stores data in cache based on its meaning in contrast to traditional caching techniques which typically exploit the concept of *locality of reference* [15] and does not usually look into the meaning of the data. The idea of semantic caching is utilized in [17] to convert cache misses to cache hits while accessing web sources through user queries. In another work [19], semantic caching is used in the context of database queries where a differentially private cache of earlier responses is employed to answer the current workload while maintaining a privacy budget.

GPTCache [5] presents an open-source semantic cache that is specifically geared towards providing faster LLM responses in addition to lowering the costs of LLM usage. We will cover GPTCache, which is the foundation for the current work, in detail in Section 3. GPTCache has already been integrated into applications such as, AI Tutor [6], social and e-commerce recommender [31], and power-aware model serving framework [25]. Note that LLM inference times vary based on the lengths of the generated responses, and since LLM serving systems generally serve requests on a First-Come-First-Serve (FCFS) basis, they often suffer from head-of-line blocking issues. The authors of [24] plan to explore GPTCache in conjunction with a speculative shortest-job-first scheduler to tackle this problem. The same FCFS related problem is also addressed by the authors of [10] who build a Language-Model-as-a-Service (LMaaS) based on efficient batching of the LLM queries which, in turn, relies on predicting the lengths of the generated responses. In the work described in [33], the authors propose a distillation-based method to fine-tune the existing embeddings for better caching prediction. A similar distillation based strategy is adopted in [1] and compared against GPTCache – however, it may be noted that this method specifically focuses on text-to-image generation prompts. In another related work [26], how to generate semantically similar prompts has been explored – these prompts should help in determining validity of cache hits and misses, and thus contribute to evaluating the efficacy of the semantic caches.

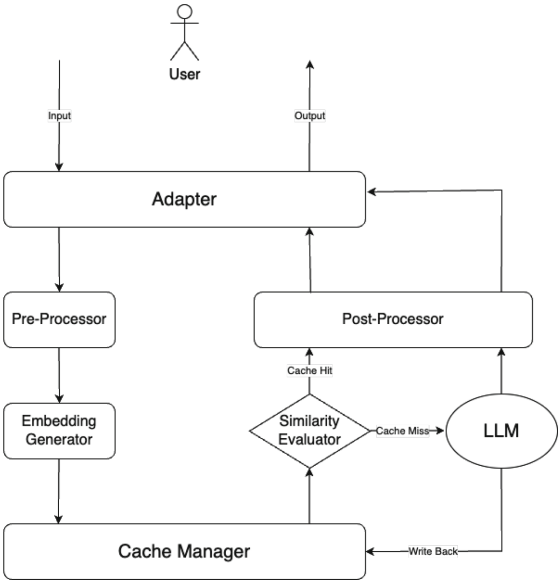


Fig. 1. The overall architecture of GPTCache. Note that in the original paper [5], the Post-Processor module is applied only in case of cache hits but we apply it for both cache hits and misses to adhere to the best industry practices.

3 GPTCache in a Nutshell

In this section, we briefly describe GPTCache whose details can be found at [5]. Conceptually, GPTCache is similar to other caches – whenever a new query is fired, we first check if that query has already been posed to the LLM or not and if a match is found, then the response is returned immediately from the cache; otherwise, we follow the traditional route of sending the query to the LLM and returning its response to the user – however, additionally, the query and the response are stored in the cache for future references. Below are the modules that are present in GPTCache as shown in Fig. 1. While explaining the GPTCache, we also underline some salient features where our system design deviates from the original one even when the aspects of distributed system, multi-tenancy, etc. are not considered.

3.1 Adapter

This module serves as the interface for GPTCache that is accountable for converting LLM queries into cache protocols and converting cache results into LLM responses. Basically, the user should experience identical input/output behavior from the system irrespective of whether caching is enabled or not; the adapter is charged with ensuring a smooth and uniform user experience.

3.2 Pre-processor

This module removes some of the redundant information that may be present in the prompt, e.g., user identifiers. To understand an interesting example where pre-processing may be required, let us consider the case where a long chat is going on between the user and the LLM – however, every LLM typically has a restriction on the number of tokens that it can process in the input prompt, and hence upon nearing the maximum token limit, the pre-processing module may compress the initial conversations or drop them altogether based on the assumption that the recent chats influence the response considerably more than the older ones. Note that dropping early chats may at times incur some performance degradation compared to compression but on the other hand, it may have significant economic impact – in industry, one typically works with the individual clients to employ the best possible pre-processing strategy that suits their performance needs while reducing their costs.

3.3 Embedding Generator

This module translates user queries into embedding vectors which are later used for similarity based retrievals.

3.4 Cache Manager

This module lies at the core of GPTCache and primarily performs three subtasks.

- **L1 Cache Storage:** When a user query arrives, it is first converted into an embedding vector and stored in a vector database such as, Milvus [29], Pinecone [22], Qdrant [23] or Weaviate [30]. For each user query, along with an embedding vector, a unique scalar id is also generated that is used subsequently in the L2 cache.
- **L2 Cache Storage:** This storage stores the unique scalar ids generated by the L1 cache along with the corresponding LLM responses. Any database (including simple structural databases) that can store key-value pairs efficiently can be used as the L2 cache.
- **Eviction Management:** This sub-module clears the cache by following a pre-determined policy, for example, the Least Recently Used (LRU) policy or the First-In-First-Out (FIFO) policy, to maintain the cache capacity.

It may be worth noting that the original paper [5] uses the terms *vector storage* and *cache storage* whereas, we use *L1 cache storage* and *L2 cache storage*, respectively. We deviate from the original terminology because of two reasons: (i) we found the terms *vector storage* and *cache storage* confusing because both are part of GPTCache, and (ii) L1 and L2 storages are commonly used terms in the context of caching and also imply the order in which these storages are accessed (similar to standard caches).

3.5 Similarity Evaluator

This module finds the most similar query posed earlier based on the embedding vectors generated. Typically, Euclidean distance or cosine similarity is used to find the most similar query; however, if the distance is found to be greater than some threshold d, then it is considered to be a *cache miss* and the LLM is actually invoked.

3.6 Post-processor

This module massages the LLM response before it is outputted to the user. In an industry context, we apply *content filtering* to filter the texts that may be toxic or derogatory [13] which may lead to reputation loss. Furthermore, we also scan for *Personal Identifiable Information (PII)* data and redact these from the outputted responses to adhere to laws and regulations, wherever applicable. It may be noted that while in the original GPTCache, *post-processing* is applied only in the case of cache hits, we apply it both during cache hits and misses to ensure that content filtering and redacting PII data are applied (among other similar processes) so that AI is always used responsibly for both internal and external uses.

4 waLLMartCache: An Enhanced Cache for LLMs

In this section, we delve into the details of the significant enhancements which have been done to the GPTCache to adapt it for industry requirements.

4.1 Incorporating Redis as a Database

We had experimented with multiple databases before finalizing on Redis [27] as the L2 cache storage. The experimental results with Redis are captured in Table 1. The dataset used for these experiments is obtained from our in-house *Generative AI playground* where any internal associate can pose a query to an LLM and get its response. Although this *"playground"* is good for testing LLMs and not meant for productionization (which may involve use-case specific fine-tuning), we chose this dataset because it covered a large spectrum of the queries (being fired by associates from all the teams) that are of importance to the retail industry, and thus can act as a suitable proxy for our initial generic design choices. The queries present in the dataset typically range from 500 to 1000 tokens. The LLM responses, however, can be very large, and hence we bucketize these responses (referred to as *Content* in Table 1). As shown in Table 1, we also consider the case where contents from all the different buckets are present together and we refer to it as *All*. We check how much scale can Redis handle by deploying 500 and 1000 concurrent users at once. We report the total number of requests, Requests Per Second (RPS) and the average, the median and the P90 (90th percentile) response times for each combination of bucket size and number

Table 1. Experimental results with Redis as an L2 cache storage

Content Size (#tokens)	#Concurrent Users	#Requests	#RPS	Average Response Time (ms)	Median Response Time (ms)	P90 Response Time (ms)
<2K	500	135758	156	119	45	110
	1000	170530	258	614	190	1100
2K to 5K	500	83035	155	141	55	140
	1000	180694	239	1051	180	3800
5K to 10K	500	81536	158	152	91	230
	1000	122708	217	1523	240	4800
>10K	500	91331	154	227	140	400
	1000	103840	180	2463	380	7400
All buckets	500	135365	150	173	48	120
	1000	190668	230	952	200	2100

of concurrent users. Due to our internal non-competition policy, we refrain from mentioning the other databases that we explored. Nevertheless, it may be noted that *the closest competition scaled to only 30% of the RPS that was registered for Redis for the same configuration*. Lastly, our code for supporting Redis in GPTCache is already merged with the open source repository and available for everyone to use.

4.2 Designing a Distributed Cache for LLMs

This enhancement lies at the core of our designed system. We elaborate it with the help of Fig. 2. In case of an industry-scale application, several hundreds of requests may be submitted every second by hundreds of customers and associates. The queries coming from the various users first undergo a load balancer that distributes these queries uniformly across various nodes, each running an instance of our waLLMartCache. After passing through the Adapter, a query comes to the Distributed Cache Manager that has a local L1 Cache Storage Manager and a local L2 Cache Storage Manager. There is no local storage residing in these individual nodes. Once the semantic embedding is obtained for the query, we search for its most similar query in the central L1 Cache Storage. If no similar query is found, then the LLM is invoked and the response is returned to the user along with saving a copy in the L2 Cache Storage. In contrast, if a similar query is found, then the L1 Cache Storage sends the corresponding scalar id to the L1 Cache Storage Manager in the originating waLLMartCache node, which shares the same to the L2 Cache Storage Manager that fetches the saved LLM response from the central L2 Cache Storage that is finally returned to the user. When any of the storages nears its capacity limit, we evict the entries that are deemed to be *erasable* based on the chosen eviction policy, e.g., LRU. Note that these deletions are carried out appropriately so that the two cache stor-

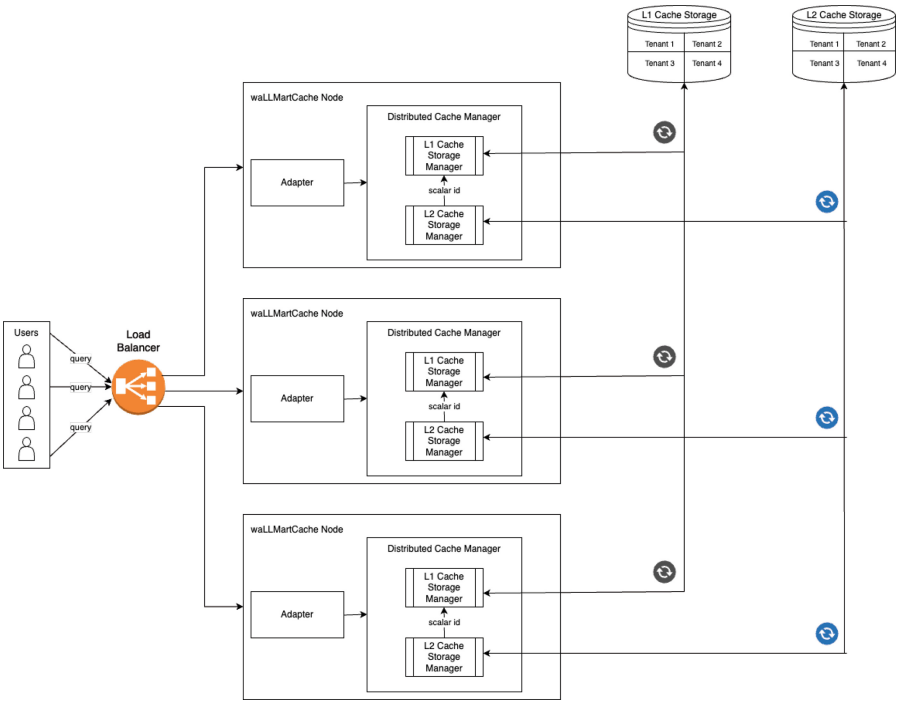


Fig. 2. Design of our distributed cache for LLMs. Note that we do not show each module from Fig. 1 here to maintain clarity and instead focus on the Distributed Cache Manager module specifically.

ages are in sync (albeit the synchronicity may not be achieved instantaneously but within a very small time lag in the order of milli-seconds), i.e., the vector embeddings and the LLM responses that are linked through the unique scalar ids are removed hand-in-hand.

4.3 Integrating Multi-tenancy

One aspect that was not addressed by GPTCache at all was handling multiple tenants (i.e., clients) at the same time. Fundamentally, it is expected that the system should not return a response that originated from a query posed by a different client although it may be similar to the current query – returning such responses may reveal sensitive information and thus violate our privacy policies. To maintain the required semantic separations, we attach a metadata to the query that contains the tenant's id, and we maintain different instances of the L1 cache vector database corresponding to each tenant; upon receiving a new query, we search for similar previous queries only in that vector database that belongs to the current tenant id. Conceptually, one may think that each database is partitioned into buckets based on tenants' ids as shown in Fig. 2.

4.4 Improved Decision Engine for Caching

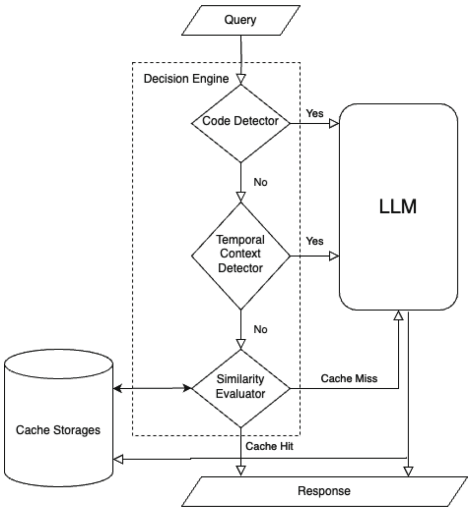


Fig. 3. Design of our decision engine which subsumes the Similarity Evaluator module of GPTCache.

Our *Decision Engine* module subsumes the *Similarity Evaluator* module from GPTCache. As illustrated in Fig. 3, the Decision Engine has three sub-modules that we elaborate upon in this subsection.

– **Code Detector:** This sub-module is used to detect whether the query being sent to the LLM contains code snippets or not. A user typically sends a code snippet to an LLM either to understand what functionality is achieved by the code or to seek the LLM's help in order to optimize it. However, during our experiments, we found that finding vector embedding based semantic similarity between two queries having code snippets is a notoriously difficult task that often leads to false positives, i.e., it results in erroneous cache hits. Two codes may differ by a single operator (say + is replaced by −, or *max* by *min*) and thus produce different outputs – their vector embeddings however will be very similar thereby leading to fallacious cache hits. While such mistakes may happen to code-free texts as well, the number of false positives in such cases is distinctively low. Specifically, in our experiments, we found that true cache hits occurred 90% of the times for code-free queries whereas, the percentage of true cache hits for queries containing codes is only 9%; this number drops further to meagre 3% when the number of tokens for the code containing queries goes beyond 400. Therefore, we decided to include this sub-module and in case it finds that the query includes code, then we send it directly to the LLM and we do not cache such queries and corresponding responses.

- **Temporal Context Detector:** To understand the necessity of this sub-module, suppose we ask the LLM the question: *"What was yesterday's average temperature?"* Obviously, even if the *exact same question* is asked the next day, returning the cached answer would be wrong. Therefore, we do not cache those queries whose responses may differ with time. It is interesting to note that sometimes the responses may differ with time even if there is no temporal construct present in the query, for example, *"What is the price of item X?"* Businesses often employ *dynamic pricing* [8] because of which the price of item X may differ throughout the day, and hence caching the prices may not be advisable. Therefore, this sub-module is deployed with the relevant business intelligence baked in based on a specific client's needs.
- **Similarity Evaluator:** This sub-module is identical to that of GPTCache and already covered in Section 3.5.

It may be noted that the first two sub-modules in the Decision Engine have the positive side-effect of keeping the cache storages restrained.

4.5 Pre-loading (non-volatile) FAQs into the Cache

Table 2. Experimental results for finding the efficacy of pre-loaded FAQs

#Concurrent Users	#Requests	#RPS	Average Response Time (ms)	Median Response Time (ms)	P90 Response Time (ms)	Cache Hit w/o FAQ	Cache Hit w/ FAQ
1000	293304	329	45	35	70	80%	90%

LLMs are being integrated into chat bots. Chat bots are popularly used for handling queries related to *Customer Care* issues among others. Fortunately, many of the use-cases such as, Customer Care, often have to answer the same questions repeatedly. Such common questions and their responses can be collated into Frequently Asked Questions (FAQs) and can be pre-loaded into the cache to boost up its hits – we found this simple idea to be very effective, in fact, it improved the rate of cache hit by 10% as shown in Table 2. For the experiment whose results are reported in this table, we had turned the FAQs non-volatile, i.e., these were never evicted from the cache storages; however, making the pre-loaded FAQs non-volatile is not a mandatory requirement and these may be evicted in a similar fashion as other queries following an eviction policy. In our experience, we found that making the pre-loaded FAQs non-volatile yields superior performance.

Finally, we would like to mention that similar to making Redis available in the open source GPTCache repository, we have plans to open source our other contributions as well – we are currently waiting to receive the necessary approvals from the higher authorities.

4.6 Ablation Study

Table 3. Effects of Redis, distributed cache, decision engine and pre-loading FAQs on semantic caching

Method	Correct Hit			Incorrect Hit			Correct Miss			Incorrect Miss			Acc(%)	
	Reg	Cde	Tmp	Reg	Cde	Tmp	Reg	Cde	Tmp	Reg	Cde	Tmp	Reg	All
Oracle	500	500	0	0	0	0	600	600	1100	0	0	0	100	100
GPTCache	401	422	0	59	435	77	550	222	545	90	21	478	86.4	64.8
WMC(1N)	401	422	0	59	435	77	550	222	545	90	21	478	86.4	64.8
WMC(4N)	399	418	0	61	438	77	548	222	543	92	22	480	86.1	64.5
WMC(4N)+DE	399	0	0	61	0	0	548	600	1100	92	500	0	86.1	80.2
WMC(4N)+DE+FAQ	488	0	0	65	0	0	498	600	1100	49	500	0	89.6	81.4

Table 4. Effects of multi-tenancy on semantic caching

Method	T	Correct Hit			Incorrect Hit			Correct Miss			Incorrect Miss			Acc(%)	
		Reg	Cde	Tmp	Reg	Cde	Tmp	Reg	Cde	Tmp	Reg	Cde	Tmp	Reg	All
Oracle	1	250	250	0	0	0	0	300	300	550	0	0	0	100	100
	2	250	250	0	0	0	0	300	250	550	0	0	0		
GPTCache	1	199	215	0	55	288	59	250	36	245	46	11	246	82	57
	2	202	207	0	52	291	61	251	36	240	45	16	249		
WMC(4N)+DE+FAQ	1	211	0	0	55	0	0	249	300	550	35	250	0	83.8	79.4
	2	212	0	0	54	0	0	250	300	550	34	250	0		
WMC(4N)+DE+FAQ+MT	1	244	0	0	33	0	0	247	300	550	26	250	0	89.5	81.3
	2	244	0	0	32	0	0	249	300	550	25	250	0		

In this subsection, we explore what are the effects of the various constituents of waLLMartCache with respect to GPTCache, and an Oracle that makes no incorrect cache hit or miss – this Oracle is used to benchmark the performance of GPTCache and our waLLMartCache. For these experiments, we chose a set of 100 prompts containing code (Cde), 100 textual prompts containing temporal context (Tmp) and 100 (regular) textual prompts without any temporal context (Reg). For each prompt we create 5 semantically similar prompts (which should ideally result in correct cache hits) and 5 dissimilar prompts (which should ideally result in correct cache misses) – note that the 5 dissimilar prompts need to be pairwise semantically different as well to ensure cache misses. We use GPT-4 [21] to generate the semantically similar and dissimilar prompts from a given textual prompt. For generating similar prompts, we instruct GPT-4 to make changes such as, active to passive voice, compound to multiple simple sentences, positive to double negative (e.g., "present" to "not absent"), replace a single or multiple

words by their synonyms; similarly, for producing dissimilar prompts, we make changes such as, positive to negative, replace a single or multiple words by their antonyms, replace the entire prompt by some random unrelated Wikipedia sentence(s). We have manually checked whether the resulting prompts were indeed similar or dissimilar. In case of prompts containing code, to generate prompts with similar meaning, we replaced equations by their mathematical equivalent ones (e.g., $y + y$ to $2 * y$), or added some constant and later subtracted the same constant, etc. To produce dissimilar prompts with code, we deliberately changed some of the operators, or removed equations partially or totally. Thus, each prompt led to 10 additional prompts, and overall we had 3300 prompts (including the original ones). The entire set is randomly shuffled before invoking the LLM caches.

Table 3 underlines the effects of Redis, distributed cache, decision engine and pre-loading FAQs (i.e., all the constituents except multi-tenancy) on waLLMartCache (WMC). Note that we report the correct and the incorrect cache hits and misses similar to earlier literature [5,26]; we further show the break ups for all three types of prompts: Reg, Cde and Tmp. To the best of our knowledge, all related previous works [5,14,18,26,33] only focused on Reg(ular) textual prompts, and did not delve into prompts containing codes or having a specific type of context such as, temporal in our case, that may require specific caching strategies. Accordingly, we report accuracy (Acc) for Reg prompts only (to maintain consistency with earlier literature) along with all types of prompts together (All). For retail business use-cases, typically prompts having temporal context should not be cached, for example, when dynamic pricing [8] is employed. Therefore, as per our strategy, all textual prompts having temporal context should result in cache miss; consequently, the Oracle in Table 3 has zero correct hits and 1100 correct misses for Tmp category of prompts. For the other two categories, the ideal number of correct cache hits is 500 (because of similar prompts) while the ideal number of correct cache misses is 600 (because of dissimilar prompts and the original prompts which should result in misses when executed for the first time). All experiments involving GPTCache and waLLMartCache were carried out five times – here we report their means rounded to nearest integer. In Table 3, WMC(1N) represents waLLMartCache deployed to only a single node – in this case, it differs from GPTCache only in its use of Redis as L2 storage. As expected, there is practically no difference between GPTCache and WCM(1N) on semantic caching because these have the same Similarity Evaluator although Redis contributes in reaching much higher RPS as mentioned in Section 4.1. waLLMartCache distributed to four nodes is represented as WCM(4N) which sees negligible drop in accuracies but it aids in scaling the system drastically and boosts RPS further. However, when it is augmented with the decision engine (WCM(4N)+DE), the DE forces the cache hits (both

correct and incorrect) to zero for Cde and Tmp categories of prompts. Since ideally all Tmp prompts should result in cache misses, the positive effect of DE is obvious; in case of Cde prompts, we found DE to be beneficial because although the Similarity Evaluator helped in getting some correct cache hits, it produced larger number of incorrect cache hits thus bringing the overall accuracy down. Therefore, although the accuracy for Reg remains identical for WCM(4N)+DE, the accuracy for All improves considerably. It may be noteworthy that from the results of Table 3, it may seem that our DE can identify codes and temporal contexts perfectly from a given text; however, the designed DE may need further enhancements especially in case of identifying temporal contexts. We believe that identifying code snippets is not difficult if one builds a dictionary of all the keywords supported by the programming languages of interest along with looking for curly braces, sentences ending with semi-colon, etc. However, identifying temporal context may be challenging because humans sometimes write English in convoluted fashions – the 100 Tmp prompts chosen for this experiment were not too complex and our DE could identify these in all cases even for their similar and dissimilar counterparts. Finally, pre-loading FAQs has a beneficial effect because now potentially even the first executions of the prompts can get a correct cache hit if it is part of the FAQ list as can be seen from the additional improvements in the accuracies for both Reg and, in turn, for All. Note that, in practice, the advantage of having a pre-loaded FAQ can be even more based on the length of the FAQ and the frequency in which these prompts are repeated as evidenced in Section 4.5.

To measure the effect of supporting multi-tenancy, we distribute the prompts from the earlier experiments to two equal halves and assign these to Tenant (T) 1 and 2 – the results are captured in Table 4. Since GPTCache and WMC(4N)+DE+FAQ cannot distinguish between prompts from the different tenants, the number of incorrect cache hits is proportionately much higher compared to the previous scenario. We refrain from mentioning the numbers for WMC(1N), WMC(4N) and WMC(4N)+DE in Table 4 for brevity and also because we believe that these numbers can be derived from the other rows in this table. Lastly, WMC(4N)+DE+FAQ+MT which has multi-tenancy support is able to do the necessary distinction and is able to achieve accuracies similar to that of WMC(4N)+DE+FAQ in Table 3 which did not have multiple tenants and therefore, their accuracies should be equivalent. Usually, the positive effect of multi-tenancy increases linearly with the increase in the number of tenants.

5 Conclusion

LLMs have recently had profound effect on our lives. Programming, story telling, reporting, auditing, chatting – all these activities can possibly be enhanced by many folds by using LLMs effectively. Consequently, all industries are presently in a race to leverage this new AI technology to increase their customer experiences and revenues as quickly as possible. However, LLMs can be quite costly and for large industries their costs can easily shoot up given their massive requirements. Furthermore, too many invocations of the proprietary LLMs often lead to

service outages which are resumed only after some cool-down period. Therefore, caching the responses of the LLMs can be an effective way to not only reduce the costs but ensure better availability of the services. Additionally, returning cached responses can reduce the latency significantly and provide a more stable system. In [5], the authors develop a caching mechanism for LLMs called GPTCache. However, when we tried to use this tool off-the-shelf, we found it to be lacking in various aspects that made its adoption in an industrial setting difficult. Accordingly, we enhanced this tool and developed an in-house LLM caching system called waLLMartCache that can additionally (i) support Redis database (this contribution of ours is already integrated with the open source GPTCache), (ii) scale to multiple nodes with support for a distributed eviction manager, (iii) handle multiple individual tenants while maintaining semantic separations, (iv) handle better semantic caching based on our decision engine, and (v) pre-load the cache with (non-volatile) FAQs, if needed, that boost cache hits significantly. In future, we plan to incorporate more LLMs and vector databases in our system, and explore alternate cache eviction policies. We also plan to check if switching LLMs in the interim based on historical data is a good idea or not, i.e., use a more powerful LLM initially so that our cache is populated with richer responses and then transition to a less powerful LLM if we believe that most responses in the foreseeable future will be returned from the cache.

References

1. Agarwal, S., Mitra, S., Chakraborty, S., Karanam, S., Mukherjee, K., Saini, S.K.: Approximate caching for efficiently serving text-to-image diffusion models. In: USENIX NSDI (2024)
2. Almazrouei, E., Alobeidli, H., Alshamsi, A., Cappelli, A., Cojocaru, R., Debbah, M., Goffinet, É., Hesslow, D., Launay, J., Malartic, Q., Mazzotta, D., Noune, B., Pannier, B., Penedo, G.: The falcon series of open language models. CoRR **abs/2311.16867** (2023)
3. Anil, R., Borgeaud, S., Wu, Y., Alayrac, J., Yu, J., Soricut, R., Schalkwyk, J., Dai, A.M., Hauth, A., Millican, K., Silver, D., Petrov, S., Johnson, M., Antonoglou, I., Schrittwieser, J., Glaese, A., Chen, J., Pitler, E., Lillicrap, T.P., Lazaridou, A., Firat, O., Molloy, J., Isard, M., Barham, P.R., Hennigan, T., Lee, B., Viola, F., Reynolds, M., Xu, Y., Doherty, R., Collins, E., Meyer, C., Rutherford, E., Moreira, E., Ayoub, K., Goel, M., Tucker, G., Piqueras, E., Krikun, M., Barr, I., Savinov, N., Danihelka, I., Roelofs, B., White, A., Andreassen, A., von Glehn, T., Yagati, L., Kazemi, M., Gonzalez, L., Khalman, M., Sygnowski, J., et al.: Gemini: A family of highly capable multimodal models. CoRR **abs/2312.11805** (2023)
4. Anthropic: Introducing Claude. https://www.anthropic.com/news/introducing-claude, accessed: 2024-04-05
5. Bang, F.: GPTCache: An open-source semantic cache for LLM applications enabling faster answers and cost savings. In: Proceedings of the 3rd Workshop for Natural Language Processing Open Source Software (NLP-OSS 2023). pp. 212–218. Association for Computational Linguistics (2023). https://doi.org/10.18653/v1/2023.nlposs-1.24

6. Barnett, S., Kurniawan, S., Thudumu, S., Brannelly, Z., Abdelrazek, M.: Seven failure points when engineering a retrieval augmented generation system. In: CAIN. pp. 194–199 (2024)
7. Bengio, Y., Ducharme, R., Vincent, P.: A neural probabilistic language model. In: NeurIPS (2000)
8. Bertsimas, D., Perakis, G.: Dynamic Pricing: A Learning Approach, pp. 45–79. Springer US (2006)
9. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D.: Language models are few-shot learners. In: NeurIPS (2020)
10. Cheng, K., Hu, W., Wang, Z., Du, P., Li, J., Zhang, S.: Enabling efficient batch serving for lmaas via generation length prediction. CoRR **abs/2406.04785** (2024)
11. Dar, S., Franklin, M.J., Jónsson, B.Þ., Srivastava, D., Tan, M.: Semantic data caching and replacement. In: VLDB. pp. 330–341 (1996)
12. Databricks: Dolly. https://github.com/databrickslabs/dolly, accessed: 2024-04-05
13. Gallegos, I.O., Rossi, R.A., Barrow, J., Tanjim, M.M., Kim, S., Dernoncourt, F., Yu, T., Zhang, R., Ahmed, N.K.: Bias and fairness in large language models: A survey. CoRR **abs/2309.00770** (2023)
14. Gill, W., Elidrisi, M., Kalapatapu, P., Anwar, A., Gulzar, M.A.: Privacy-aware semantic cache for large language models. CoRR **abs/2403.02694** (2024)
15. Handy, J.: The cache memory book. Academic Press Professional, Inc (1993)
16. Jiang, A.Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D.S., de Las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L.R., Lachaux, M., Stock, P., Scao, T.L., Lavril, T., Wang, T., Lacroix, T., Sayed, W.E.: Mistral 7b. CoRR **abs/2310.06825** (2023)
17. Lee, D., Chu, W.W.: Semantic caching via query matching for web sources. In: CIKM. pp. 77–85 (1999)
18. Li, J., Xu, C., Wang, F., von Riedemann, I.M., Zhang, C., Liu, J.: SCALM: Towards semantic caching for automated chat services with large language models. CoRR **abs/2406.00025** (2024)
19. Mazmudar, M., Humphries, T., Liu, J., Rafuse, M., He, X.: Cache me if you can: Accuracy-aware inference engine for differentially private data exploration. Proc. VLDB Endow. **16**(4), 574–586 (2022)
20. OpenAI: Pricing. https://openai.com/pricing, accessed: 2024-04-04
21. OpenAI, Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., Avila, R., Babuschkin, I., Balaji, S., Balcom, V., Baltescu, P., Bao, H., Bavarian, M., Belgum, J., Bello, I., Berdine, J., Bernadett-Shapiro, G., Berner, C., Bogdonoff, L., Boiko, O., Boyd, M., Brakman, A.L., Brockman, G., Brooks, T., Brundage, M., Button, K., Cai, T., Campbell, R., Cann, A., Carey, B., Carlson, C., Carmichael, R., Chan, B., Chang, C., Chantzis, F., Chen, D., Chen, S., Chen, R., Chen, J., Chen, M., Chess, B., Cho, C., Chu, C., Chung, H.W., Cummings, D., Currier, J., Dai, Y., Decareaux, C., Degry, T., Deutsch, N., Deville, D., Dhar, A., Dohan, D., Dowling, S., Dunning, S., Ecoffet, A., Eleti, A., Eloundou, T., Farhi, D., Fedus, L., Felix, N., Fishman, S.P., Forte, J., Fulford, I., Gao, L., Georges, E., Gibson, C., Goel, V., Gogineni, T., Goh, G., Gontijo-Lopes, R., Gordon, J., Grafstein, M., Gray, S., Greene, R., Gross, J., Gu, S.S., Guo, Y., Hallacy, C., Han, J., Harris, J., He, Y., Heaton, M.,

Heidecke, J., Hesse, C., Hickey, A., Hickey, W., Hoeschele, P., Houghton, B., Hsu, K., Hu, S., Hu, X., Huizinga, J., Jain, S., Jain, S., Jang, J., Jiang, A., Jiang, R., Jin, H., Jin, D., Jomoto, S., Jonn, B., Jun, H., Kaftan, T., Łukasz Kaiser, Kamali, A., Kanitscheider, I., Keskar, N.S., Khan, T., Kilpatrick, L., Kim, J.W., Kim, C., Kim, Y., Kirchner, J.H., Kiros, J., Knight, M., Kokotajlo, D., Łukasz Kondraciuk, Kondrich, A., Konstantinidis, A., Kosic, K., Krueger, G., Kuo, V., Lampe, M., Lan, I., Lee, T., Leike, J., Leung, J., Levy, D., Li, C.M., Lim, R., Lin, M., Lin, S., Litwin, M., Lopez, T., Lowe, R., Lue, P., Makanju, A., Malfacini, K., Manning, S., Markov, T., Markovski, Y., Martin, B., Mayer, K., Mayne, A., McGrew, B., McKinney, S.M., McLeavey, C., McMillan, P., McNeil, J., Medina, D., Mehta, A., Menick, J., Metz, L., Mishchenko, A., Mishkin, P., Monaco, V., Morikawa, E., Mossing, D., Mu, T., Murati, M., Murk, O., M/'ely, D., Nair, A., Nakano, R., Nayak, R., Neelakantan, A., Ngo, R., Noh, H., Ouyang, L., O'Keefe, C., Pachocki, J., Paino, A., Palermo, J., Pantuliano, A., Parascandolo, G., Parish, J., Parparita, E., Passos, A., Pavlov, M., Peng, A., Perelman, A., de Avila Belbute Peres, F., Petrov, M.: Gpt-4 technical report (2024)

22. Pinecone: Build knowledgeable ai. https://www.pinecone.io/, accessed: 2024-04-08
23. Qdrant: qdrant. https://github.com/qdrant/qdrant, accessed: 2024-04-08
24. Qiu, H., Mao, W., Patke, A., Cui, S., Jha, S., Wang, C., Franke, H., Kalbarczyk, Z.T., Basar, T., Iyer, R.K.: Efficient interactive LLM serving with proxy model-based sequence length prediction. CoRR **abs/2404.08509** (2024)
25. Qiu, H., Mao, W., Patke, A., Cui, S., Jha, S., Wang, C., Franke, H., Kalbarczyk, Z.T., Basar, T., Iyer, R.K.: Power-aware deep learning model serving with μ-serve. In: USENIX ATC (2024)
26. Rasool, Z., Barnett, S., Willie, D., Kurniawan, S., Balugo, S., Thudumu, S., Abdelrazek, M.: Llms for test input generation for semantic applications. In: CAIN. pp. 160–165 (2024)
27. Redis: Get the world's fastest in-memory database from the ones who built it. https://redis.io/, accessed: 2024-04-09
28. Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Canton-Ferrer, C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P.S., Lachaux, M., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E.M., Subramanian, R., Tan, X.E., Tang, B., Taylor, R., Williams, A., Kuan, J.X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., Scialom, T.: Llama 2: Open foundation and fine-tuned chat models. CoRR **abs/2307.09288** (2023)
29. Wang, J., Yi, X., Guo, R., Jin, H., Xu, P., Li, S., Wang, X., Guo, X., Li, C., Xu, X., Yu, K., Yuan, Y., Zou, Y., Long, J., Cai, Y., Li, Z., Zhang, Z., Mo, Y., Gu, J., Jiang, R., Wei, Y., Xie, C.: Milvus: A purpose-built vector data management system. In: SIGMOD. pp. 2614–2627 (2021)
30. Weaviate: Welcome to weaviate docs. https://weaviate.io/developers/weaviate, accessed: 2024-04-08
31. Xu, D., Zhang, D., Yang, G., Yang, B., Xu, S., Zheng, L., Liang, C.: Survey for landing generative ai in social and e-commerce recsys – the industry perspectives. CoRR **abs/2406.06475** (2024)

32. Ye, J., Chen, X., Xu, N., Zu, C., Shao, Z., Liu, S., Cui, Y., Zhou, Z., Gong, C., Shen, Y., Zhou, J., Chen, S., Gui, T., Zhang, Q., Huang, X.: A comprehensive capability analysis of GPT-3 and GPT-3.5 series models. CoRR **abs/2303.10420** (2023)
33. Zhu, H., Zhu, B., Jiao, J.: Efficient prompt caching via embedding similarity. CoRR **abs/2402.01173** (2024)

ReeSPOT: Reeb Graph Models Semantic Patterns of Normalcy in Human Trajectories

Bowen Zhang, S. Shailja, Chandrakanth Gudavalli(✉), Connor Levenson, Amil Khan, and B. S. Manjunath

Electrical and Computer Engineering Department, University of California Santa Barbara, Santa Barbara, USA
chandrakanth@ucsb.edu

Abstract. This paper introduces ReeSPOT, a novel Reeb graph-based method to model patterns of life in human trajectories (akin to a fingerprint). Human behavior typically follows a pattern of normalcy in day-to-day activities. This is marked by recurring activities within specific time periods. In this paper, we model this behavior using Reeb graphs where any deviation from usual day-to-day activities is encoded as nodes in the Reeb graph. The complexity of the proposed algorithm is linear with respect to the number of time points in a given trajectory. We demonstrate the usage of ReeSPOT and how it captures the critically significant spatial and temporal deviations using the nodes of the Reeb graph. Our case study presented in this paper includes realistic human movement scenarios: visiting uncommon locations, taking odd routes at infrequent times, uncommon time visits, and uncommon stay durations. We analyze the Reeb graph to interpret the topological structure of the GPS trajectories. Potential applications of ReeSPOT include urban planning, security surveillance, and behavioral research.

Keywords: Reeb Graphs · Graph Networks · Trajectory Analysis

1 Introduction

Recently, there has been an increase in location-aware devices that use the Global Positioning System (GPS) for many applications such as finding efficient routes [17], fitness apps, understanding the progression of infectious diseases [6], and predicting demographic information [19]. This collection of movements, and thus vast amounts of raw trajectories, spotlights the need for a scalable representation of these trajectories that preserves and highlights the structure and topologically important movement patterns (Figure 1).

Human movement analysis is the core component of behavioral research, urban planning, and computational sociology [3], which helps in better modeling human behavior and predicting human movement patterns. Similarly, modeling normal human behavior can also help identify abnormal human behavior. In

B. Zhang and S. Shailja—Equal Contributors

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 249–264, 2025.
https://doi.org/10.1007/978-3-031-78183-4_16

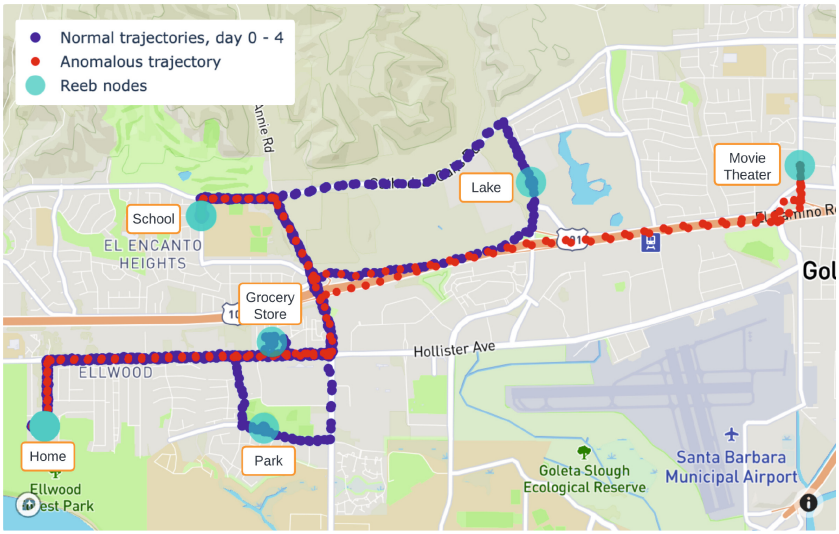


Fig. 1. Map overlay of normal and anomalous trajectories from scenario 2 of the case study, annotated with semantic labels for points of interest (POIs).

particular, given a set of movement patterns for a week, month, or year, we want to capture any change in the semantic "patterns of life". In this paper, we model routine behaviors and movements that characterize daily human activities in a given city using a concept from topology, Reeb graphs.

Traditional trajectory analysis methods are largely based on hand-crafted geometric features and statistical techniques. Such features include traveling distance, mean velocity [22], frequencies of areas or moving patterns [4]. Statistical approaches analyze the temporal patterns with respect to the frequency of trajectory data to identify patterns such as traveling modes [8] and periodic patterns [21]. These approaches are effective for handling structured and less complex data sets but fail to generalize with high-dimensional data or the dynamic nature of human mobility patterns.

Given the amount of GPS data that can be generated by one human on a single day, another obvious direction to look at would be toward data-driven learning methods. Specifically, sampling a single agent's movement data, sampled at a 1Hz frequency over a month, accumulates roughly 2 million data points.

Extrapolating these figures to a population of a small city like Santa Barbara, with approximately 97,000 agents, results in a dataset comprising an immense 194 billion data points. This scale poses substantial challenges in terms of computational resources and data management, and extrapolating to larger cities, such as New York City, would significantly magnify these challenges. Recent advances in deep learning have significantly enhanced the capability to model human mobility patterns by performing the next-location prediction [10]. Particularly, long short-term memory networks (LSTMs) [7] and attention-based models like Transformers [18] are good at capturing temporal regularities and

anomalies in movement patterns. However, these black-box models lack interpretability, thus limiting their applicability in real-time scenarios [20].

Towards interoperability along with large-scale modeling, Graph-based methods are very popular due to their ability to represent complex spatial relationships and movement patterns efficiently. We need models that can succinctly summarize an agent's trajectory data—retaining essential information while discarding redundancies. Transforming GPS data into graph data structures with nodes as significant geographic locations and edges as the movement information between enables intuitive models for pattern-of-life. Research directions include, Guo et al. [5]'s graph model to establish precise topological relationships among trajectories and geographic locations. Qi et al. [11] incorporate hybrid methods that blend graph-based approaches with statistical models to improve the accuracy of trajectory searches and predictions. Another such work focuses on hierarchical clustering based on graph similarity measures[12], further supporting the need for computational geometry.

In this paper, we use Reeb graphs to cluster the common behavior pattern for a given agent. Our research is motivated by and related to previous research on the construction of Reeb graphs for trajectory data [2,13]. A Reeb graph captures the connectivity of level sets of a scalar function defined over a space, effectively summarizing the topological features of the space. In the context of trajectory data, scalar functions could represent attributes such as speed, direction, semantics, or geographical points of interest. Reeb graphs can thus map complex trajectories into more interpretable topological constructs. This abstraction facilitates the detection of anomalies by comparing the topological signatures of trajectories and identifying those that differ significantly from the norm. Our main contributions are summarized below:

- We propose a novel Reeb graph-based approach to model the day-to-day activities of a given agent. To the best of our knowledge, this is the first demonstration of Reeb graphs to fingerprint an agent's behavior.
- We discuss the algorithm and its time complexity demonstrating the scalability of the proposed method.
- We design normal and anomalous scenarios, describe the methods for trajectory generation and present detailed experiments on the interpretation and analysis of Reeb graphs.

2 Methodology

2.1 Previous work on Reeb graphs

Reeb graph was first proposed to study the topology of a manifold [16]. Nodes of the Reeb graph encode the evolution of the level sets of a real-valued function on a manifold. The location of the node is the average location of the points of the trajectories that constitute the node. Reeb graphs have been extensively used in shape analysis for diverse datasets [1]. The first study of Reeb graphs for trajectory group evolvement encodes the merging and splitting structure between

different moving entities [2]. Similarly, the spatial subtrajectory clustering algorithm presented a stricter problem [13–15] but discovers geometric and topological substructure. This is a computationally challenging problem because the initialization step involves an exhaustive search of an agent's events. Motivated by these challenges, the central focus of this paper is to develop a method for fingerprinting the behavior of an agent over time such as days, weeks, and months. Our approach encodes significant spatio-temporal points of interest-specifically, locations and durations that define critical aspects of an agent's behavior. We redefine the grouping definitions used in our adapted Reeb graph model. The constructed Reeb graphs effectively partition a set of GPS points into meaningful nodes and edges, thereby quantifying and identifying path deviations.

2.2 Reeb graph models agent pattern of normalcy

A trajectory T is defined as a dictionary (key: value) containing an ordered sequence of time points and their associated GPS coordinates:

$$T = \{t_0 : p_0, t_1 : p_1, t_2 : p_2, \ldots, t_m : p_m\}, \tag{1}$$

where m is chosen according to the desired resolution to sample the pattern of the agent. Here m denotes the total number of points in a given trajectory T. The frequency of GPS data sampling decides m. For example, to model the weekdays of an agent's activities, the raw GPS data is sampled every second, giving us $m = 86400$ which is the total number of seconds in a day. Similarly, if the data is sampled every hour, then $m = 24$ points per day. We define n as the total number of trajectories for a given agent. For example, to model month-long data, $n = 30$ and for weekdays, $n = 5$. The common setting used throughout the paper for our problem definition is $m = 24$ and $n = 5$. Each time point t_i corresponds to a GPS coordinate p_i representing the position of the agent at time t_i. $p_i = (\text{lat}_i, \text{lon}_i)$, where lat_i represents the latitude and long_i represents the longitude. The Euclidean distance between two GPS coordinates p_i and $p_{i'}$ is calculated at time t_i as follows:

$$d(p_i, p_i') = \sqrt{(\text{lat}_i - \text{lat}_i')^2 + (\text{lon}_i - \text{lon}_i')^2}, \tag{2}$$

where lat_i and lon_i are the latitude and longitude of the first point, and lat_i' and lon_i' are those of the second point. $d(p_i, p_i')$ gives the 2-norm distance between two points on the Euclidean plane. This approximates the geographic distance of the points. The algorithm is defined with respect to a distance threshold ϵ within which the points are considered sufficiently close together i.e. within a small geographical area. This is the inter-trajectory distance that guides the granularity of the Reeb graphs according to the problem definition.

Algorithm 1 Find *connect* and *disconnect* events

Input: Trajectories T and T', threshold ϵ
Output: Dictionary of *connect*/*disconnect* events, $events_{T,T'}$
Initialize $events_{T,T'}$ as an empty dictionary
Initialize $k \leftarrow 0$
Initialize $connect\_flag \leftarrow$ False
while $k < m$ **do**
    if $d(T[t_k], T'[t_k]) < \epsilon$ **then**
        $events_{T,T'}[t_k] \leftarrow connect$
        $connect\_flag \leftarrow$ True
        while $k < m$ and $d(T[t_k], T'[t_k]) < \epsilon$ **do**
            $k \leftarrow k + 1$
        if $k < m$ **then**
            $events_{T,T'}[t_k] \leftarrow disconnect$
    $k \leftarrow k + 1$
return $events_{T,T'}$

Human behavior typically follows a pattern of normalcy in day-to-day activities. This is marked by recurring activities within specific time periods. In order to discover the large-scale spatio-temporal patterns, we represent the bundling structure of trajectories as a *Reeb graph* $R(V, E)$. Nodes of the Reeb graph will pinpoint critical GPS points of the agent's pattern. Intuitively, if a continuous portion of a behavior of the agent happens at the same time and within the same spatial distance (ϵ) every day then they present a pattern of normalcy. We formalize this by introducing the concept of "bundles" to characterize normal behavior through consistent daily subtrajectory events. Each trajectory begins with an *appear* event at the first index and concludes with a *disappear* event at the last index of T. Deviations from this norm by more than ϵ are classified as *disconnect* events, while a return to the norm is labeled a *connect* event. Formally, for a given ϵ and $m = 23$ i.e. sampled every hour, let's take two trajectories T and T':

- At time t_0: p_0 and p'_0 are the *appear* events.
- At time t_{23}: p_{23} and p'_{23} are the *disappear* events.
- If $d(p_0, p'_0) \leq \epsilon, (p_1, p'_1) \leq \epsilon, \ldots, d(p_k, p'_k) \leq \epsilon$, but $d(p_{k+1}, p'_{k+1}) > \epsilon$, then t_{k+1} represents a *disconnect* event between T and T'.
- If $d(p_0, p'_0) > \epsilon, (p_1, p'_1) > \epsilon, \ldots, d(p_k, p'_k) > \epsilon$, but $d(p_{k+1}, p'_{k+1}) \leq \epsilon$, then t_{k+1} represents a *connect* event between T and T'.

2.3 Construction of Reeb graphs and analysis of time complexity

Reeb graph construction (illustrated in Figure 2) can be divided into the following major steps: event computation, construction of dynamic graphs (Gs), connectivity query in the dynamic graph for bundle partition (P), and construction of the Reeb graphs (R) from bundles partition as shown in Figure 2. The first step of Reeb graph construction involves computing the *connect* and *disconnect* events. Algorithm 1 outlines the steps of computing events. The event

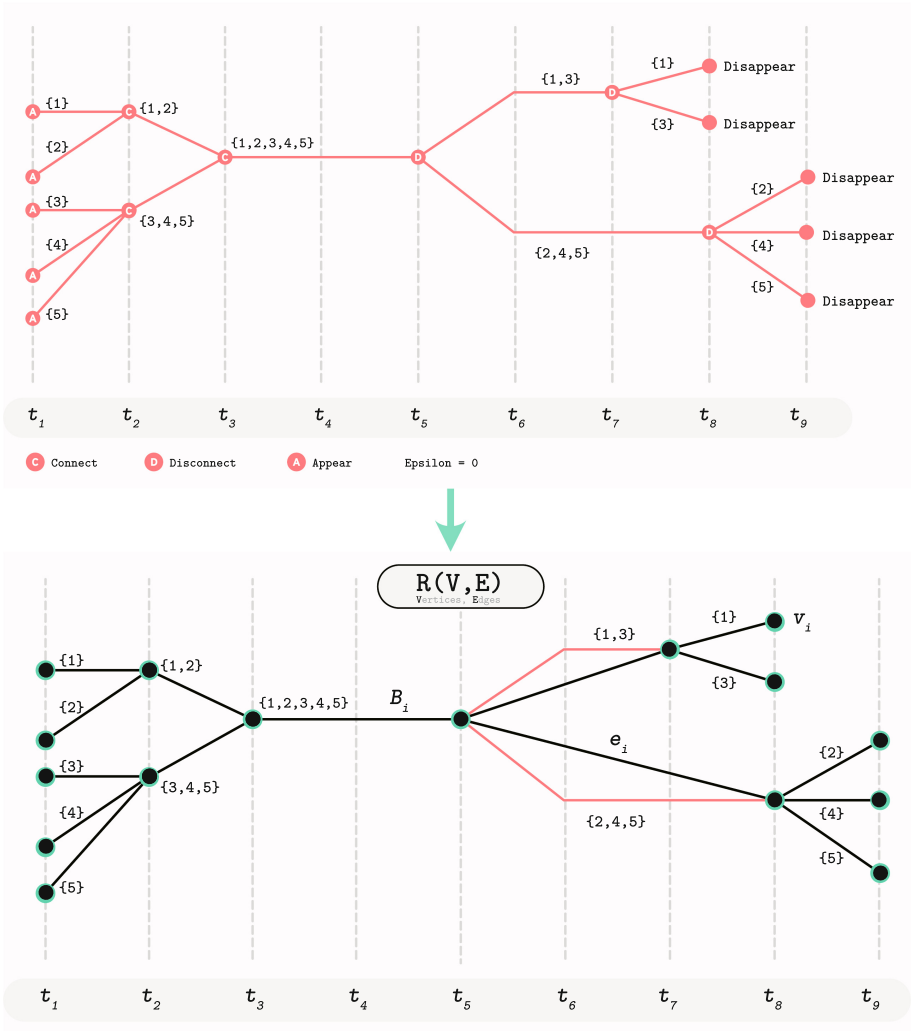


Fig. 2. Reeb Graph Construction Over Time. We show the construction of Reeb graphs $R(V, E)$ for a set of five trajectories. The *appear, disappear, connect,* and *disconnect* events are shown on the left-hand side. Changes in the grouping of trajectories due to these events are encoded as nodes on the right-hand side. Nodes of the Reeb graph $\mathcal{R}$ on the right-hand side are shown in red color and the edges are shown in black color throughout the paper.

computation takes $\mathcal{O}(m)$ time, where m represents the number of time points in the trajectories T and T'. At each time point, the algorithm looks for $\mathcal{O}(5 \times 5)$ possibilities of potential events. The second step of the Reeb graph involves handling the events to construct dynamic graph Gs. The nodes of G represent the daily trajectories and the edges of the G represent the ϵ-connectivity between

them. The total number of nodes in G is 5 representing one trajectory for each day of the agent. The connected component of the G will give us the $\epsilon-$step bundle partition of subtrajectories denoted by $P = \{B1, B2, \ldots, Bk\}$ such that every segment in T_0, T_1, T_2, T_3, T_4 is uniquely assigned to exactly one bundle. The final step is to construct the Reeb graph from these bundles. Reeb graph R can be constructed from P by connecting adjacent bundles with nodes and bundles as edges similar to the described construction in [13]. So, the time complexity of the Reeb graph construction step would be $\mathcal{O}(m)$ because in the worst case, all the time points will have events. At each time, the connectivity query to the dynamic graph with 5 nodes takes constant time. The more detailed steps can be found in the Algorithm 2.

Algorithm 2 Construction of Reeb Graph

```
function ConstructReebGraph(set of events for all pairs of trajectories (E))
    for all steps k from 0 to |E| do                          ▷ Dynamic Graphs
        if appear event of T then
            insert new node T to G_k
        if disappear event of T then
            delete node T from G_k
        if connect event between T_x and T_y then
            insert edge (T_x, T_y) to G_k
        if disconnect event of trajectories T_x and T_y then
            delete edge (T_x, T_y) from G_k
        P ← empty bundle partition                            ▷ Bundle Partition
        Query G_{k-1} and G_k to get the connected components C_{k-1} and C_k respectively;
        for all connected component c_k ∈ C_k do
            if c_k ∈ C_{k-1} then
                assign the same bundle id B_i to the points for trajectories in c_k;
            else
                create a new bundle id B_{i+1} and assign it to the points for trajectories in c_k;
                Add B_{i+1} to P
    Construct Reeb graph R from P by connecting adjacent bundles with nodes and bundles as edges;
                                                              ▷ Construct Reeb graph
    return R
```

3 Experimentation/Case Study

3.1 Data generation

We model the pattern of life of a single agent over different trajectories. Each trajectory is simulated using the SUMO software package [9] and represents realistic behavior and movement patterns over the course of one week. We construct the Reeb graph for each trajectory and show how it sufficiently represents the trajectory's information with significantly fewer nodes.

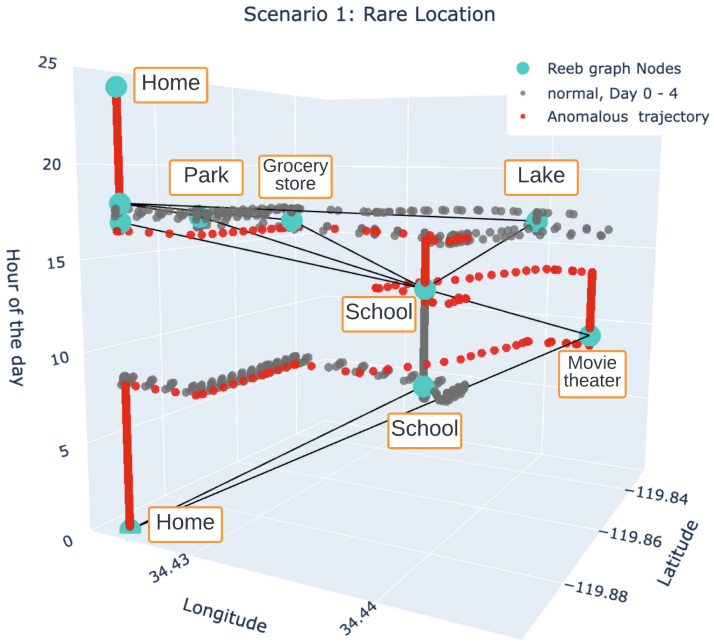


Fig. 3. 3D trajectory plots with computed Reeb graph nodes for scenario 1 in Section 3, where day 0 to day 4 are normal trajectories, and the anomalous trajectory is in red.

In this case study, we analyze the behavioral patterns of a simulated high-school student from the city of Santa Barbara, California (Figure 1), using trajectory data that includes multiple points of interest (POIs), such as the student's home, school, park, grocery store, and lake. The student's daily routine typically consists of attending school from approximately 8:00 AM to 9:00 AM, concluding at around 4:00 PM to 5:00 PM, followed by visits to recreational sites before returning home. To thoroughly investigate both normal and anomalous behavioral patterns, we generated five days of normal trajectory data, complemented by additional days tailored to each specific scenario described earlier. Each trajectory entry is recorded with timestamps, latitude, and longitude coordinates. Figure 1 displays the student's trajectories across different POI locations for the rare location scenario, illustrating the distribution of both routine and deviant movements. Figure 3 displays the same data as a 3D plot, providing a clear spatio-temporal visualization of the student's stay locations, duration, and revisit frequencies.

3.2 Definition of anomalous behavior

We define L as a set of normal POIs and their corresponding time points,

$$L = \{(lat_1, lon_1, t_1), (lat_2, lon_2, t_2), \ldots, (lat_n, lon_n, t_n)\}$$

where (lat_i, lon_i) represents the geographic coordinates with $lat_i \in [-90, 90]$ and $lon_i \in [-180, 180]$, and t_i is the time at which these coordinates were recorded. Relative to this definition, all the anomaly behaviors for a given agent are defined as follows:

Algorithm 3 Trajectory Generation

Inputs:
  POIs – List of Points of Interest as coordinates on a map.
  $TimeList_n$ – Dictionary mapping each POI to normal visit times.
  $TimeList_a$ – Dictionary mapping each POI to abnormal visit times.
  Road Network – Road network graph for route generation.
Output:
  T – A list of normal trajectories of an agent visiting specified POIs.
  T^* – A list of abnormal trajectories of an agent visiting specified POIs.
Initialize Trajectories list
for each POI in the POIs list **do**
    Select $TimeList$ based on a decision rule (normal vs abnormal)
    for each time in $TimeList$ **do**
        Generate a starting point for the agent
        Use duarouter to calculate the shortest path from the starting point to the POI at the given time
        Pass the list of edges to SUMO for movement simulation
        Collect the output trajectory from SUMO
        Append to T or T^* based on decision rule
return T, T^*

Scenario 1 (S1): Rare Location Anomaly Rare location anomaly refers to a scenario when an agent visits a new location $(lat^*, lon^*, t_i) \notin L$. (lat^*, lon^*) is spatially different from their normal spatial geographical points of interest such as school or work. Reeb graph will encode this rare location by creating a new node localizing the abnormality.

Scenario 2 (S2): Rare Route Visit Anomaly In this scenario, the agent visits the same POI locations multiple times but utilizes a uniquely different route on a single journey. This introduces *disconnect* event from their normal movement pattern, resulting in a new node in the Reeb graph. More formally, if $(lat^*, lon^*, t_{k:l}) \notin (lat, lon, t_{1:k-1})$ and $\notin (lat, lon, t_{l+1:m})$, then nodes v_k and v_l will be added to R.

Scenario 3 (S3): Uncommon Time Visit This is a case of time violation where the agent visits a familiar location at an uncommon time t^* i.e, $(lat_i, lon_i, t^*) \neq (lat_i, lon_i, t_i)$

Scenario 4 (S4): Uncommon Stay Duration Anomaly In this scenario the agent stays for an abnormal duration (Δ) at a specific location $(lat^*, lon^*, t_{i+\Delta})$. This results in a *disconnect* event for the agent's trajectory from the normal pattern of life at t_i.

3.3 Reeb Graph Generation

We use a down-sampling rate of one hour for Reeb graphs. This setting helps us to monitor changes in location grouping states at each hour. The threshold ϵ for

spatial connect and disconnect events is set to 0.0005 GPS degrees (5.56 meters). Initially, we construct a Reeb graph from the normal activity trajectories of days 0 to 4 to model the student's typical pattern of life.

As depicted in Figure 1 and Figure 3, ReeSPOT successfully identifies all normal POIs as a part of the Reeb graph nodes, demonstrating its efficacy in reflecting the spatial distribution of the student's activities. Notably, an anomalous scenario depicted in Figure 1 and Figure 3 shows the student visiting a movie theater during school hours which is defined as a deviation from the normal. This is captured by a new Reeb graph node, highlighting its potential for identifying critical spatial anomalies.

3.4 Analysis and interpretation of scenarios using Reeb graphs

To better understand the formation of Reeb graph nodes and demonstrate the utility of the Reeb graph across all six scenarios, we generated time-latitude plots (Figure 4). These plots, with the hour of day on the x-axis and latitude on the y-axis, include trajectory points sampled every 10 seconds alongside Reeb graph nodes. Each plot provides a visual representation of different behavioral patterns and anomalies and illustrates ReeSPOT's effectiveness in capturing anomalous trajectories for all scenarios. We explain the scenarios one by one below:

- **Figure 4(a)** illustrates the student's normal routine pattern, with stays at home, school, and visits to various recreational spots. Notable events include *appear* and *disappear* at the beginning and end of each day. There are three *disconnect* events around hour 17 which indicates divergences to different locations after school. *Connect* event shows trajectories getting merged back on the way home at hour 18.
- **Figure 4(b) for S1** depicts a rare location (lat^*, lon^*) where we visualize an abnormal visit to the movie theater, showing three additional Reeb nodes and altered connectivity events at hour 9 and 14.
- **Figure 4(c) for S2** captures an alternative route to school. At hour 9, instead of following the normal route, the student deviates towards a direction with a lower latitude and then returns to school. This deviation is captured by the bottom Reeb graph node at hour 9. Additionally, a *disconnect* event occurs at 9, followed by a *connect* event at hour 10 when all trajectories converge at the school.
- **Figure 4(d) for S3** reveals an uncommon time anomaly, where the student attends school at hour 2 and travels to the park at around hour 10, significantly deviating from the typical schedule, but with the same POIs.
- **Figure 4(e) for S4** shows another time-related anomaly with a prolonged stay at home until almost hour 12, and similarly, 3 new nodes appear for the reeb graph because of *disconnect* event from the usual trajectory.
- **Figure 4(f) for S4** presents a detailed look at scenario 4, from hour 16 to hour 17. Since the reeb graph sample rate is one hour, the reeb graph nodes appear at hour 17 to represent the *disconnect* events in the past hour.

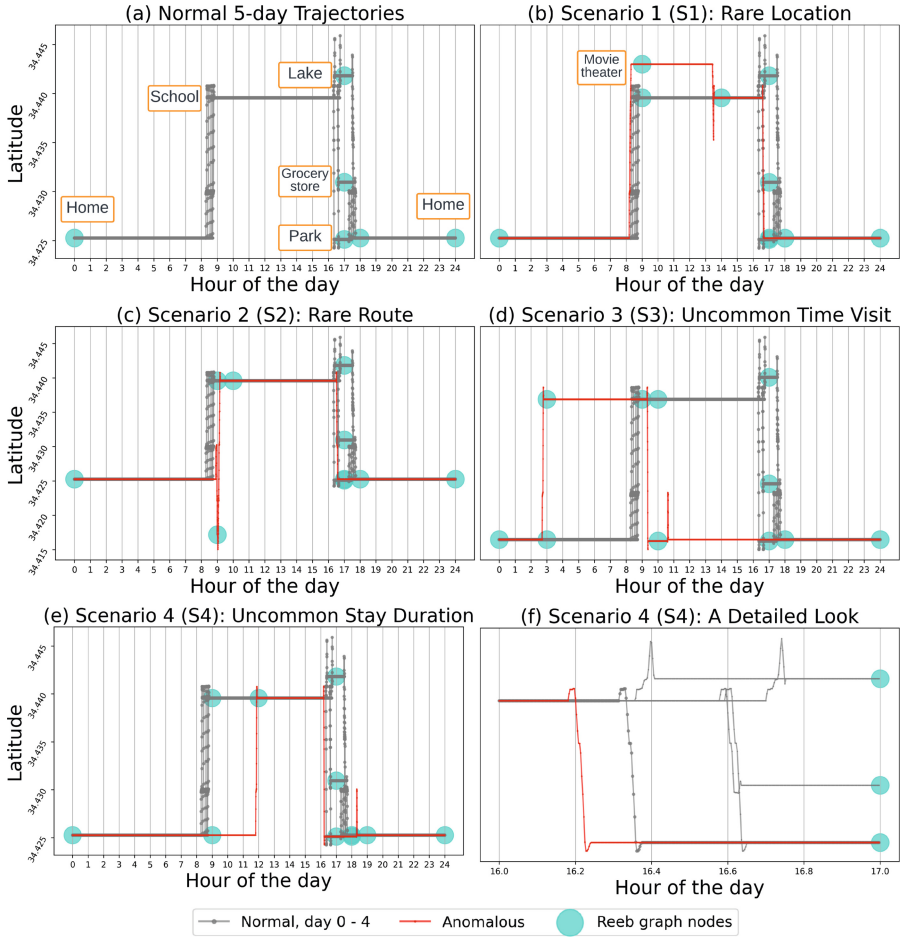


Fig. 4. 2D Trajectory plots displaying time and latitude dimensions alongside computed Reeb graph nodes. These plots illustrate both normal and anomalous scenarios as outlined in Section 3.2. The detailed discussions on node generation and behavioral analysis can be found in Section 3.4.

3.5 Reeb graph iteratively detects anomalous behavior of an agent

In the context of detecting anomalous trajectories within real-life data (test dataset), we iteratively construct Reeb graphs on the test dataset to identify daily anomalous trajectories. An initial Reeb graph is constructed using training data with all normal trajectories. Subsequently, for each daily trajectory in the test dataset, the Reeb graph is iteratively updated day by day. To detect anomalous behaviors effectively, we compute the distance between the existing Reeb graph and every updated version that includes the additional daily

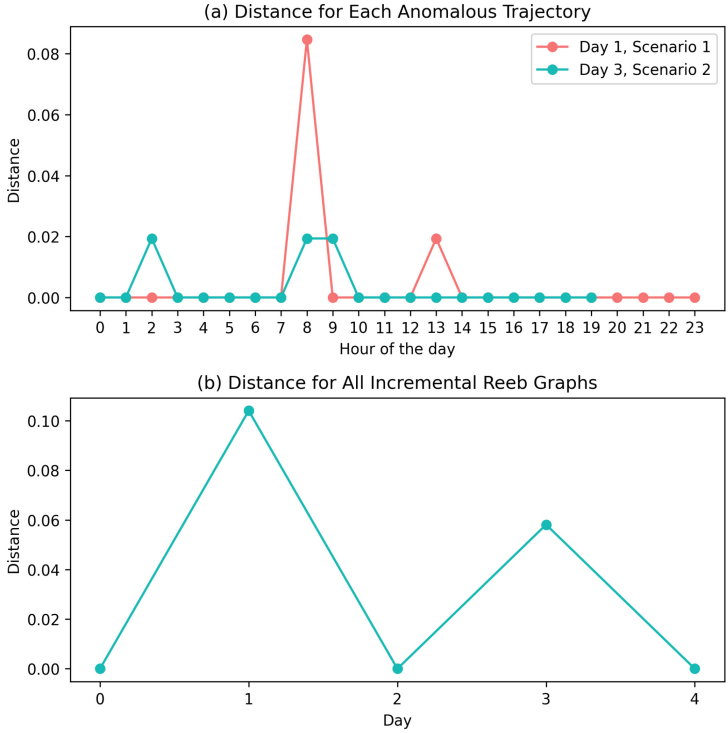


Fig. 5. (a) illustrates the Reeb graph node-level distances for both anomalous days. (b) shows the day-level anomaly scores.

trajectory. The subsequent section details our methodology for calculating this distance and presents the results derived from our case study.

3.6 Quantifying the distance between Reeb graphs

Given two Reeb graphs, a normal Reeb graph R_1 and a Reeb graph with one anomalous trajectory R_2, each containing data points across dimensions of time (0 to 23 hours), the following rules are used to calculate the distance between Reeb graphs defined as $d(R_1, R_2)$:

1. For each hour, if nodes exist in both R_1 and R_2, calculate the Euclidean distance between the nodes.
2. If only one of the Reeb nodes graphs, R_1 or R_2, has a node at a particular hour, calculate the distance to the temporally closest node from the other Reeb graph.
3. If neither Reeb graph has a node for a given hour, the distance is 0.

Specifically, in point 2 above, we have a case where a node at time t_k in Reeb graph R_1 has no corresponding node in R_2. We find the Euclidean distance to

the nodes in R_2 at t_{k+1}. If there are multiple nodes in R_2 at t_{k+1} or t_{k-1}, then we select the one with the minimum distance. $d(R_1, R_2)$ is the sum of the distances computed every hour using the above rules.

Results In this case study, we created a synthetic test dataset to investigate both spatial anomalies (Scenario 1, see Figure 4(b)) and temporal anomalies (Scenario 3, see Figure 4(d)). The dataset comprises three days of randomly simulated normal behavior and two days of anomalous behavior. Figure 5(a) illustrates the node-level distances for both anomalous days. On Day 1, new anomalous nodes appear at hour 8 (movie theater) and hour 13 (coming back). Anomalous events on Day 3 occur at hours 2, 8, and 9. Figure 5(b) depicts the day-level anomalies; the anomalous distance for Day 1 is higher than for Day 3, reflecting the student's travel to a more distant location on Day 1, whereas, on Day 3, the anomalies involve the same POIs.

3.7 Scalability with Reeb Graphs

We successfully applied ReeSPOT to a simulated dataset that is closer to a real-life distribution. This data is an extended version of the data that we described in this paper for proof-of-concept. Here, instead of modeling weekdays of data sampled every hour, we model the patterns over a month sampled at every 15-second interval. This results in $m = 5760$ and $n = 30$. For this dataset, ReeSPOT models the patterns of daily activities for a simulated population of 800,000 agents. Each agent is processed independently, and the Reeb graphs for the entire dataset were constructed within 7.2 hours, parallel processed across 384 CPU cores (AMD EPYC 9654 @ 3.7 GHz). We also implemented the spatial Reeb graph, ReeBundle as proposed in [13] but the quadratic time complexity with respect to m made it computationally challenging. More specifically, for $n = 7$ and $m = 5760$, the Reeb graph construction took around 4 minutes for an agent. ReeSPOT is linear with respect to m and thus for the same problem setting it was able to construct Reeb graphs in approximately 12 seconds on one CPU core. This is an important advantage over spatial Reeb graphs which helps us to apply our method on large-scale datasets. Multi-processing across 384 cores enabled us to construct Reeb graphs in less than 8 hours. We also tested ReeSPOT on medium-sized data with 10,000 agents over a period of one week, Reeb Graphs were computed in approximately 5.5 minutes. The above experiments show the applicability of ReeSPOT in modeling agent's data at different resolutions (weekly, monthly, yearly) and also emphasize the scalability of the proposed algorithm.

4 Discussion and Future Work

In this paper, we proposed a Reeb graph-based approach (ReeSPOT) to model the patterns of normalcy using day-to-day human trajectory data. The proposed Reeb graphs abstract large-scale spatio-temporal data into a comprehensible

topological construct. We design distinct real-life anomalous scenarios, develop trajectory generation methods, and provide a thorough interpretation of Reeb graph results. The parameters of ReeSPOT can control the granularity of the model according to different applications. On the other hand, ReeSPOT depends on the quality of the trajectory, so false positives can impact the accuracy of the model. One explanation for this is the inherent stochasticity of general human behavior.

Another application is a quantifiable sanity check for raw trajectory data such as teleports. We synthesized such scenarios and observed additional nodes in the Reeb graphs. Our experiment setting in this paper is based on the assumption that each agent is independent and the activities conducted by one agent are not related to the other. However, agents in a given population influence the behavior of each other. Such correlations could serve as additional features to our existing model. ReeSPOT has the flexibility to introduce more parameters and features to robustly support the data abstraction. Geo-foundational features describe the nature of each location the agent visited such as residential, commercial, recreational, etc. Nodes of the Reeb graphs can be labeled with such domain-specific information. Such representation can be used as an input to data-driven methods instead of directly using deep learning methods on raw GPS trajectories.

Acknowledgement. We would like to thank Kin Gwn Lore for the invaluable insights and assistance throughout this project. This work is supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/ Interior Business Center (DOI/IBC) contract number 140D0423C0057 The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DOI/IBC, or the U.S. Government.

References

1. Biasotti, S., Giorgi, D., Spagnuolo, M., Falcidieno, B.: Reeb graphs for shape analysis and applications. Theoret. Comput. Sci. **392**(1–3), 5–22 (2008)
2. Buchin, K., Buchin, M., van Kreveld, M., Speckmann, B., Staals, F.: Trajectory grouping structure. In: Workshop on Algorithms and Data Structures. pp. 219–230. Springer (2013)
3. Eagle, N., Pentland, A.: Reality mining: sensing complex social systems. Pers. Ubiquit. Comput. **10**, 255–268 (2006)
4. Giannotti, F., Nanni, M., Pinelli, F., Pedreschi, D.: Trajectory pattern mining. In: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 330–339 (2007)
5. Guo, D., Liu, S., Jin, H.: A graph-based approach to vehicle trajectory analysis. Journal of Location Based Services **4**(3–4), 183–199 (2010)

6. Hast, M., Searle, K.M., Chaponda, M., Lupiya, J., Lubinda, J., Sikalima, J., Kobayashi, T., Shields, T., Mulenga, M., Lessler, J., et al.: The use of gps data loggers to describe the impact of spatio-temporal movement patterns on malaria control in a high-transmission area of northern zambia. Int. J. Health Geogr. **18**, 1–18 (2019)
7. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
8. Kinoshita, A., Takasu, A., Aihara, K., Ishii, J., Kurasawa, H., Sato, H., Nakamura, M., Adachi, J.: Gps trajectory data enrichment based on a latent statistical model. In: International Conference on Pattern Recognition Applications and Methods. vol. 2, pp. 255–262. SCITEPRESS (2016)
9. Lopez, P.A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., Wießner, E.: Microscopic traffic simulation using sumo. In: The 21st IEEE International Conference on Intelligent Transportation Systems. IEEE (2018), https://elib.dlr.de/124092/
10. Luca, M., Barlacchi, G., Lepri, B., Pappalardo, L.: A survey on deep learning for human mobility. ACM Computing Surveys (CSUR) **55**(1), 1–44 (2021)
11. Qi, S., Bouros, P., Sacharidis, D., Mamoulis, N.: Efficient point-based trajectory search. In: International Symposium on Spatial and Temporal Databases. pp. 179–196. Springer (2015)
12. Sabarish, B., Karthi, R., Kumar, T.G.: Graph similarity-based hierarchical clustering of trajectory data. Procedia Computer Science **171**, 32–41 (2020)
13. Shailja, S., Bhagavatula, V., Cieslak, M., Vettel, J.M., Grafton, S.T., Manjunath, B.: Reebundle: a method for topological modeling of white matter pathways using diffusion mri. IEEE Transactions on Medical Imaging (2023)
14. Shailja, S., Chen, J.W., Grafton, S.T., Manjunath, B.: Retrace: Topological evaluation of white matter tractography algorithms using reeb graphs. In: International Workshop on Computational Diffusion MRI. pp. 177–191. Springer (2023)
15. Shailja, S., Zhang, A., Manjunath, B.: A computational geometry approach for modeling neuronal fiber pathways. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 175–185. Springer (2021)
16. Shinagawa, Y., Kunii, T.L., Kergosien, Y.L.: Surface coding based on morse theory. IEEE Comput. Graphics Appl. **11**(05), 66–78 (1991)
17. Ta, N., Zhao, Y., Chai, Y.: Built environment, peak hours and route choice efficiency: An investigation of commuting efficiency using gps data. J. Transp. Geogr. **57**, 161–170 (2016)
18. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
19. Wu, L., Yang, L., Huang, Z., Wang, Y., Chai, Y., Peng, X., Liu, Y.: Inferring demographics from human trajectories and geographical context. Comput. Environ. Urban Syst. **77**, 101368 (2019)
20. Zeng, J., He, X., Tang, H., Wen, J.: A next location predicting approach based on a recurrent neural network and self-attention. In: Collaborative Computing: Networking, Applications and Worksharing: 15th EAI International Conference, CollaborateCom 2019, London, UK, August 19-22, 2019, Proceedings 15. pp. 309–322. Springer (2019)

21. Zhang, D., Lee, K., Lee, I.: Mining hierarchical semantic periodic patterns from gps-collected spatio-temporal trajectories. Expert Syst. Appl. **122**, 85–101 (2019)
22. Zheng, Y., Liu, L., Wang, L., Xie, X.: Learning transportation mode from raw gps data for geographic applications on the web. In: Proceedings of the 17th international conference on World Wide Web. pp. 247–256 (2008)

Label Disambiguation-Based Feature Selection for Partial Multi-label Learning

Fankang Xu[1], Wenbin Qian[1(✉)], Xingxing Cai[1], Wenhao Shu[2], Jintao Huang[3], Yiu-Ming Cheung[3], and Weiping Ding[4]

[1] Jiangxi Agricultural University, Nanchang, China
xufankang9956@126.com, qianwenbin1027@126.com, caixx@jxau.edu.cn
[2] East China Jiaotong University, Nanchang, China
shuwenhao@126.com
[3] Hong Kong Baptist University, Hong Kong SAR, China
{jthuang,ymc}@comp.hkbu.edu.hk
[4] Nantong University, Nantong, China
dwp9988@163.com

Abstract. Partial multi-label learning (PML) addresses the issue of training a multi-label predictor in the context of inaccurate supervision. Objects in PML are relevant to multiple semantics, but only a subset of them are valid. Besides false positive labels that mislead the learning procedure, high dimensionality also acts as a stumbling block for boosting PML. In this paper, a two-stage label disambiguation-based feature selection method, LDFS-PML, is presented for partial multi-label learning. At first, to avoid false positive labels from misleading the feature selection, a label disambiguation technique is devised based on the granular ball, which is the first attempt at multi-label disambiguation from the perspective of cognition computing. By using the label disambiguation technique, label-specific information concealed in the distribution of data is captured, which is conducive to estimating the confidence of candidate labels. In the second stage of LDFS-PML, a feature selection algorithm is proposed which utilizes labeling confidence and simultaneously incorporates cognition computing from both global and local perspectives. Experiments are conducted on various PML datasets, and the superiority of the proposed LDFS-PML is demonstrated.

Keywords: Feature selection · Partial multi-label learning · Granular ball

1 Introduction

In the paradigm of partial multi-label learning (PML), model training is confronted with imprecise supervised information generated by unreliable crowdsourcing annotators [1]. Specifically, in partial multi-label data, samples are linked with multiple labels, of which some are ground-truth labels and the remainder are noise labels, also known as false positive labels [2]. For instance, a

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 265–279, 2025.
https://doi.org/10.1007/978-3-031-78183-4_17

scenario of PML can be illustrated in Fig. 1, where the object is tagged with six relevant labels, but only *athlete*, *sky*, *snowfield*, and *snowmobiling* are ground-truth labels while the remaining are noise ones, including *tree* and *house*. Similar to multi-label learning [3], high dimensionality has emerged as a prevalent trend in PML, which necessitates the development of feature selection techniques [4–7] to deliver a compact feature subset consisting of discriminative characteristics while simultaneously improving the performance and efficiency of PML. However, the selection process of discriminative features is misled by the false positive labels in PML, which is a tricky problem that has attracted increasing attention recently.

To avoid the pitfall of imprecise supervised information in the partial multi-label problem, much research has been conducted to tackle the ambiguity of candidate labels [8,9]. The primitive technique is to treat the association labels equally and then feed them into any off-the-shelf learning model. This deviates from the goal of enhancing the performance of PML, as such a strategy does not fundamentally resolve the negative impact of noise labels on PML. As a result, many researchers have embraced the label disambiguation strategy, wherein the confidence of each candidate label being the ground-truth is estimated [10]. A commonly used method is label propagation, which delivers the confidence of candidate labels using neighborhood information iteratively. Nonetheless, the method has been pointed out to be error-prone when label space is contaminated seriously.

Fig. 1. An example of a PML scenario. In the set consisting of ambiguous labels, only four of them in blue are ground-truth ones, while two in black are noise labels. (Color figure online)

Besides noise labels, the high feature dimension also acts as a stumbling block to improving the performance of PML. However, in the context of PML, typical feature selection algorithms cannot be applied directly, as the procedure of feature selection would be misled by false positive labels, which necessitates the development of the label disambiguation technique[11,12]. Indeed, with the label information distilled by the label disambiguation technique, one could consider any feature selection approach suitable for multi-label data. Based on the interaction with the classifier, the feature selection techniques can be classified

as filter, wrapper, or embedded methods [13]. Distinct from the wrapper-based approach that reflects the quality of selected feature subsets by the outcomes of a specific classifier and the embedded-based approach that integrates the feature selection process into the model learning, the filter-based method selects informative features without the help of any predictor, which is considered to have better generalization ability. Drawing inspiration from cognition computing, this work introduces a feature selection method based on the filter strategy, which considers both global and local perspectives. This method forms the second part of the proposed LDFS-PML algorithm.

Moreover, the emergence of granular ball computing has introduced a powerful data granulation technology to boost the robustness of model learning [14,15]. The main idea is to characterize samples using hyper-balls, where the label is defined using the majority label within the ball. As a result, noise labels with a minor proportion are omitted, and at the same time, the robustness of the model is enhanced. However, the label-specific information concealed in the distribution of samples has not been captured, which could be beneficial for measuring the probability of samples being noise under a specific label. With this in mind, this work proposes a granular ball-guided label disambiguation technique for partial multi-label data, which comprises the first part of LDFS-PML.

The main contributions of this work are summarized as follows:

- To select discriminative features in the context of partial multi-label data, a specific granular ball (SGB) computing technique is designed for label disambiguation. With SGB computing, the label-specific information is first captured by characterizing the distribution of samples to determine the labeling confidence.
- To obtain a compact feature subset for PML, a filter-based feature selection algorithm is designed from both global and local aspects. The global sample similarity relation is considered to be preserved, which is applied to quantify the quality of the feature subset. Additionally, motivated by cognition computing, a term of local consistency is integrated.
- The effectiveness of our proposed LDFS-PML method, which comprises two components: label disambiguation and feature selection, has been validated on eight synthetic datasets and two real-world partial multi-label datasets. Ablative studies are performed to analyze the effectiveness of each module.

2 Related Work

PML is predicated on the assumption that only a subset of pertinent labels correspond to the ground truth [16]. The ubiquitous learning scenario has attracted increasing attention from scholars. From the perspective of labels and features, considerable research effort has been devoted to label disambiguation and feature selection. From the aspect of label disambiguation, PARTICLE [17] estimates labeling confidences using a label propagation technique. PML-GAN [18] proposes a novel disambiguation strategy using an adversarial mode, which includes

an encoder-decoder framework. PENAD [19] introduces a label enhancement-based PML framework that takes into account the varying significance of candidate labels for each instance. To address the problem that iteratively optimizing the disambiguation model is error-prone when false positive labels occupy a large proportion of the label space, PML-LMNNE [20] is developed via the integration of large margin nearest neighbor and embedding. Focusing on the aspect of selecting informative features for PML, several approaches have been developed recently. PMLFS [11] develops a feature selection method for PML regarding the noise and ground-truth labels as latent variables, and the corresponding model is optimized iteratively. CLLFS [12] introduces a two-stage feature selection approach for PML in which credible labels are initially distilled through label structure reconstruction, and a joint model is then devised to select both label-specific and common features. However, in these feature selection frameworks for PML, an embedded method is used, which may deteriorate the generalization ability of the model.

Granular ball computing, a novel technology for knowledge discovery and data mining, has recently garnered increasing attention as it provides a method for knowledge granulation based on the concept of large-scale priority [21]. Granular ball computing has enriched research in various domains. Ball k-means [14] utilizes the granular ball to develop an accelerated exact k-means algorithm for fast and adaptive clustering. In terms of classification tasks, several scholars devote their efforts to enhancing the efficiency, robustness, and scalability of the model by leveraging the advantages of granular ball computing. For instance, ILAGBRS [22] explores an improved granular ball rough set model for classification, taking into account the dynamic nature of data. In the area of feature selection, several approaches based on granular ball computing have been proposed, some of which possess significant denoising capabilities. GBNRS [15] first introduces the granular ball into the neighborhood rough set for the feature selection task. The methods integrated with granular ball computing have made great achievements, especially in improving the robustness of original models. However, label-specific information hidden in the distribution of balls is omitted, which could be captured for label disambiguation.

3 Proposed Approach

Formally, $F = \{f_1, f_2, \ldots, f_m\}$ denotes a feature set, and $L = \{l_1, l_2, \ldots, l_q\}$ denotes a candiate label set. A training dataset in PML is represented by $TS = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{n}$, where $\boldsymbol{x}_i = \left\{x_i^1, x_i^2, \ldots, x_i^m\right\}$ is the feature vector for the i-th instance and $\boldsymbol{y}_i = \left\{y_i^1, y_i^2, \ldots, y_i^q\right\}$ corresponding to the label vector contaminated by noise false positive labels. $y_i^j = 1$ indicates i-th instance is labeled with l_j, otherwise, $y_i^j = 0$. Thus, the feature space is denoted as $X \in \mathbb{R}^{n \times m}$, and the label space is denoted as $Y \in \{0,1\}^{n \times q}$. For label disambiguation, a labeling confidence matrix is formalized as $P \in [0,1]^{n \times q}$.

The framework of the presented LDFS-PML method is illustrated in Figure 2. Under the guidance of a specific label, SGBs are generated with different sizes.

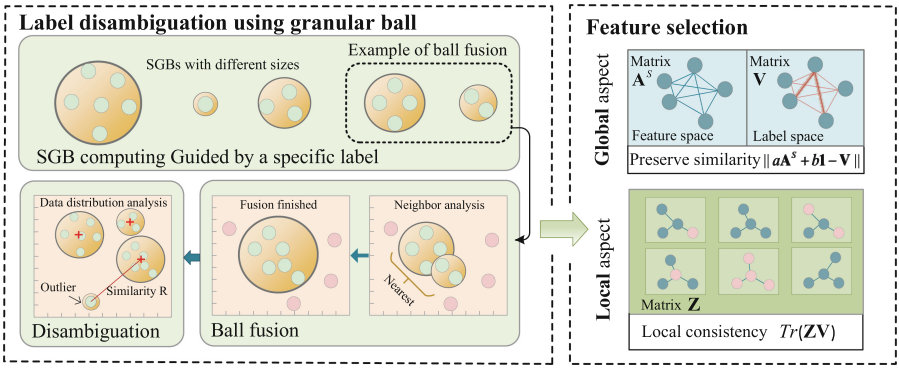


Fig. 2. The overall framework of LDFS-PML. The first stage is label disambiguation, and the second stage is feature selection for PML.

After ball fusion, a ball with a tiny size is more likely to be an outlier or noise. The delivered labeling confidence is applied to feature selection. The feature significance index considers similarity preservation from a global perspective and local consistency from a local perspective.

3.1 Label Disambiguation Using Granular Ball

A collection of SGB generated on TS deduced by $l_j \in L$ is expressed as $\mathcal{SGB}_{l_j} = \{sgb_1, sgb_2, \ldots, sgb_r\}$ $(r \leq n)$. The center c_i and radius r_i for $\forall sgb_i \in \mathcal{SGB}_{l_j}$ are defined as

$$\boldsymbol{c}_i = \frac{1}{\mid X_i' \mid} \sum_{\boldsymbol{x} \in X_i'} \boldsymbol{x}, \quad r_i = \max\{\Delta(\boldsymbol{c}_i, \boldsymbol{x}) \mid \boldsymbol{x} \in X_i'\} \tag{1}$$

where $\mid \cdot \mid$ denotes the cardinality of a set, Δ calculates the Euclidean distance between two vectors, and $X_i^{'} \subseteq X$ is a subdataset on which sgb_i is calculated. Objects covered by the ball form a collection denoted as $\mathcal{O}_i = \{\boldsymbol{x} \mid \Delta(\boldsymbol{x}, \boldsymbol{c}_i) \leq r_i, \boldsymbol{x} \in X\}$. The association $\mathcal{O}_i$ for $\forall sgb_i \in \mathcal{SGB}_{l_j}$ can be partitioned into two parts by l_j, represented as $\mathcal{O}_i / l_j = \left\{O_i^+, O_i^-\right\}$, in which O_i^+ consists of objects relevant to l_j, and vice versa for O_i^-. Thus, the label d_i and label-specific purity p_i for sgb_i are defined respectively as

$$d_i = [\![\mid O_i^+ \mid \geq \mid O_i^- \mid]\!], \quad p_i = \frac{\mid O_i^+ \mid}{\mid \mathcal{O}_i \mid} \tag{2}$$

in which $[\![\cdot]\!]$ is an indicator that returns 1 if predicate $\cdot$ holds and 0 otherwise. Here, for $\forall sgb_i \in \mathcal{SGB}_{l_j}$, $d_i = 1$ denotes that sgb_i is labeled with l_j, and $d_i = 0$ denotes that l_j is irrelevant to sgb_i. The label-specific purity p_i is directly proportional to the number of positive samples labeled with l_j in sgb_i. Thus, $\forall$ $sgb_i \in \mathcal{SGB}_{l_j}$ can be expressed as a quintuple $sgb_i = (\mathcal{O}_i, c_i, r_i, d_i, p_i)$.

In partial multi-label data, under the guidance of $l_j \in L$, the label-specific information can be naturally captured using SGB computing, which is beneficial

for mitigating the impact of false positive labels on downstream feature selection. Concretely, by requiring the label-specific purity of SGBs to reach 1, the resulting balls can precisely characterize the distribution of all samples labeled with l_j. Meanwhile, a $sgb_i \in \mathcal{SGB}_{l_j}$ with $d_i = 1$ and a small size is more likely to contain noise objects. To reduce the influence of the randomness of the 2-means used for ball splitting and thereby improve the effectiveness of label disambiguation, the granular ball covering model can be constructed by considering two perspectives. (1) Overlap degree, balls close to each other may share common objects as the radius of the ball defined in Eq. (1) uses a maximum operator, which is not conducive to determining the true aggregation degree of local positive samples. The overlap degree is encouraged to be as small as possible. (2) Average coverage degree, the more objects covered by a ball labeled with l_j, the lower the probability that those objects are noise ones with respect to l_j. The optimization goal of SGB computing is formulated by

$$\begin{aligned} &Min \left(\sum_{sgb_i \in \mathcal{SGB}^+_{l_j}} (\mid \mathcal{O}_i \mid) - \mid D_{l_j} \mid \right) / \mid D_{l_j} \mid + \mid \mathcal{SGB}^+_{l_j} \mid / \mid D_{l_j} \mid \\ &\text{s.t. } p_i \geq T \end{aligned} \tag{3}$$

where T is a threshold set to be 1 or can be searched in the range of [0.5,1], $\mathcal{SGB}^+_{l_j} = \left\{ sgb_i \in \mathcal{SGB}_{l_j} \mid d_i = 1 \right\}$, and $D_{l_j} = \left\{ \boldsymbol{x}_h \in X \mid y^j_h = 1 \right\}$. The first term reflects the overlap degree, and the second term is associated with the average coverage degree. A ball fusion strategy is presented as follows to deliver an ideal ball covering model for label disambiguation:

$$\begin{aligned} &\forall sgb_i, sgb_k \in \mathcal{SGB}^+_{d_j} : sgb^*_i \\ &= \begin{cases} \mathcal{F}(sgb_i, sgb_k) & \text{if } \mid \mathcal{O}_k \mid < \mid \mathcal{O}_i \mid, p^*_i > T \\ & \text{and } \Delta(\boldsymbol{c}_i, \boldsymbol{c}_k) < r_i + r_k \\ sgb_i, & \text{otherwise} \end{cases} \end{aligned} \tag{4}$$

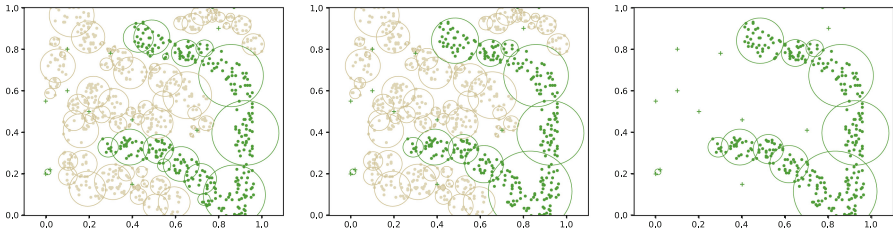
where $\mathcal{F}(sgb_i, sgb_k)$ is a fusion operator that merges sgb_k into sgb_i, sgb^*_i is a new ball generated on $X^{'}_i \cup X^{'}_k$ if fusion conditions are satisfied, and p^*_i is the label-specific purity of sgb^*_i.

The label-specific information provided by label l_j can be naturally captured by characterizing the distribution of positive objects using SGB computing. Figure3 visualizes the generation process of SGBs, in which a 2-means algorithm is applied to split the balls, and the splitting process is terminated when the label-specific purity of the balls reaches 1 or 0. As observed in Fig.3(c), the distribution of the positive objects is characterized and balls with a tiny size are more likely to consist of noisy data.

Accordingly, a label disambiguation technique using label-specific information captured by SGB is expressed as follows:

$$\forall 1 \leq i \leq n, 1 \leq j \leq q : p_i^j = \begin{cases} \sup\max\limits_{sgb_k \in \mathcal{SGB}_{l_j}^+} \{R(x_i, sgb_k), \ [\![x_i \in \mathcal{O}_k \wedge \mid \mathcal{O}_k \mid \geq N]\!]\} & \text{if } y_i^j = 1 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where parameter N is used to determine the size of reliable balls that consist of only non-noise samples, and $R(x_i, sgb_k) = \exp\left(-\Delta(x_i, c_k)^2\right)$. $p_i^j \in P$ reflects the confidence of label l_j with respect to the i-th object. The confidence value of labels for each sample is normalized to [0,1]; then, $\forall \boldsymbol{x}_i \in X, \sum_{j=1}^{q} p_i^j = 1$.



(a) Final iteration. (b) Granular ball fusion. (c) Identify noise samples.

Fig. 3. The visualization of SGB computing on the *fourclass* dataset disturbed by noisy data. Points colored bright green represent positive objects with respect to label l_j, while olive points represent negative ones related to l_j. Noise points are plotted as "+".

3.2 Feature Selection Using Labeling Confidence

Intuitively, labeling confidence can reflect the relative importance of related labels, which enriches the semantic information and may be conducive to the learning procedure. A filter-based feature selection method is presented for the labeling confidence matrix.

The principle of feature significance measure is to preserve the topology of the sample space. An already selected feature set is denoted as S. Thus, an affinity matrix on feature space S is expressed as $\mathbf{A}^S = \left\{A_{ij}^S\right\}_{n \times n}$, which can be calculated as

$$A_{ij}^S = \exp\left(-\frac{\parallel \boldsymbol{x}_i - \boldsymbol{x}_j \parallel_2^2}{\sigma}\right) \tag{6}$$

Similarly, an affinity matrix can be constructed on the labeling confidence matrix, which is defined as $\mathbf{V} = \{V_{ij}\}_{n \times n}$, where $V_{ij} = 1 - \frac{1}{2} \parallel \boldsymbol{p}_i - \boldsymbol{p}_j \parallel_1$. To preserve the global similarity, a function is formulated as follows

$$\min_{S,a,b} \parallel a\mathbf{A}^S + b\mathbf{1} - \mathbf{V} \parallel_F^2 \tag{7}$$

where a and b are variables presented for linear approximation to consider the order of sample similarities. Moreover, inspired by cognition computing, a discriminative feature subset can be quantified by the quality of knowledge granulation. Specifically, an object is encouraged to be consistent with its neighbors under feature space S. In light of this, we considered the consistency term from a local perspective. An edge matrix is denoted as $\mathbf{E}^S = \left\{E_{ij}^S\right\}_{n\times n}$, where E_{ij}^S is calculated as

$$E_{ij}^S = \begin{cases} 1, & \text{if } \boldsymbol{x}_j \in \mathcal{N}_k\left(\boldsymbol{x}_i\right) \text{ or } \boldsymbol{x}_i \in \mathcal{N}_k\left(\boldsymbol{x}_j\right) \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

where $\mathcal{N}_k\left(\cdot\right)$ consists of k-nearest neighbors of the object. The local consistency term can be constructed as

$$\sum_{i=1}^{n}\sum_{j=1}^{n} Z_{ij}^S \cdot V_{ij} = Tr\left(\mathbf{ZV}\right) \tag{9}$$

where $Z_{ij}^S = A_{ij}^S \cdot E_{ij}^S$. Thus, the final object function for the feature significance measure can be formulated as

$$\begin{aligned} &\min_{S,a,b} \parallel a\mathbf{A}^S + b\mathbf{1} - \mathbf{V} \parallel_F^2 + \alpha Tr\left(\mathbf{Z}\left(\mathbf{1}-\mathbf{V}\right)\right) \\ &\text{s.t. } \mathbf{Z} = \mathbf{A}^S \circ \mathbf{E}^S \end{aligned} \tag{10}$$

where $\circ$ is a dot product operator.

To obtain solutions for a and b, a strategy is given assuming S is known. The function (10) can be reformed as

$$\begin{aligned} \Theta\left(a,b\right) =& \parallel a\mathbf{A}^S + b\mathbf{1} - \mathbf{V} \parallel_F^2 + \alpha Tr\left(\mathbf{Z}\left(\mathbf{1}-\mathbf{V}\right)\right) \\ =& Tr\left(a^2\mathbf{A}^S\mathbf{A}^S + 2ab\mathbf{A}^S\mathbf{1} + b^2\mathbf{1}^\mathbf{2} - 2a\mathbf{A}^S\mathbf{V}\right. \\ & \left. -2b\mathbf{1}\mathbf{V} + \mathbf{V}^2\right) + \alpha Tr\left(\mathbf{Z}\left(\mathbf{1}-\mathbf{V}\right)\right) \end{aligned} \tag{11}$$

The partial derivatives of Θ w.r.t a and b are set to 0, thus

$$\frac{\partial\Theta}{\partial a} = aTr\left(\mathbf{A}^S\mathbf{A}^S\right) + bTr\left(\mathbf{A}^S\mathbf{1}\right) - Tr\left(\mathbf{A}^S\mathbf{V}\right) = 0 \tag{12}$$

$$\frac{\partial\Theta}{\partial b} = aTr\left(\mathbf{A}^S\mathbf{1}\right) + bTr\left(\mathbf{1}^\mathbf{2}\right) - Tr\left(\mathbf{1}\mathbf{V}\right) = 0 \tag{13}$$

Based on the above two formulations, we obtain

$$\begin{pmatrix} Tr\left(\mathbf{A}^S\mathbf{A}^S\right) & Tr\left(\mathbf{A}^S\mathbf{1}\right) \\ Tr\left(\mathbf{A}^S\mathbf{1}\right) & Tr\left(\mathbf{1}^\mathbf{2}\right) \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} Tr\left(\mathbf{A}^S\mathbf{V}\right) \\ Tr\left(\mathbf{1}\mathbf{V}\right) \end{pmatrix} \tag{14}$$

According to Cauchy's inequality, $\left(Tr\left(\mathbf{A}^S\mathbf{1}\right)\right)^2 < Tr\left(\mathbf{A}^S\mathbf{A}^S\right)Tr\left(\mathbf{1}^\mathbf{2}\right)$ (in general, $\mathbf{A}^S \neq \mathbf{1}$). Thus, $Tr\left(\mathbf{A}^S\mathbf{A}^S\right)Tr\left(\mathbf{1}^\mathbf{2}\right) - Tr\left(\mathbf{A}^S\mathbf{1}\right)^2 \neq 0$. The unique solution for a and b can be calculated as

$$a = \frac{Tr\left(\mathbf{A}^S\mathbf{V}\right)Tr\left(\mathbf{11}\right) - Tr\left(\mathbf{A}^S\mathbf{1}\right)Tr\left(\mathbf{1}\mathbf{V}\right)}{Tr\left(\mathbf{A}^S\mathbf{A}^S\right)Tr\left(\mathbf{11}\right) - Tr\left(\mathbf{A}^S\mathbf{1}\right)Tr\left(\mathbf{A}^S\mathbf{1}\right)} \tag{15}$$

$$b = \frac{Tr\left(\mathbf{A}^S\mathbf{A}^S\right)Tr\left(\mathbf{1V}\right) - Tr\left(\mathbf{A}^S\mathbf{1}\right)Tr\left(\mathbf{A}^S\mathbf{V}\right)}{Tr\left(\mathbf{A}^S\mathbf{A}^S\right)Tr\left(\mathbf{11}\right) - Tr\left(\mathbf{A}^S\mathbf{1}\right)Tr\left(\mathbf{A}^S\mathbf{1}\right)} \tag{16}$$

Then, with the idea of global preservation and local consistency, a feature significance measure for PML can be obtained:

$$\mathcal{L}\left(S\right) = \| \, a\mathbf{A}^S + b\mathbf{1} - \mathbf{V} \, \|_F^2 + \alpha Tr\left(\mathbf{Z}\left(\mathbf{1} - \mathbf{V}\right)\right) \tag{17}$$

The procedure of the LDFS-PML method is characterized in Algorithm 1.

Algorithm 1. LDFS-PML Algorithm

Input: PML training set TS; parameters N, k, and α.
Output: The sorted set of features.

Initialize selected feature set $S \leftarrow \emptyset$;
for each $l_i \in L$ **do**
  Specific granular ball computing guided by l_i;
  for each $\boldsymbol{x}_j \in X$ **do**
    Updating labeling confidence p_i^j by Eq.(5);
  end for
end for
Normalization for labeling confidence matrix P;
Calculate label similarity matrix $\mathbf{V}$ using P;
repeat
  for each $f \in F - S$ **do**
    Calculate $\mathbf{A}^S$, $\mathbf{E}^S$, and $\mathbf{Z}$;
    Calculate a and b by Eq.(15) and Eq.(16);
    Calculate $\mathcal{L}\left(S \cup \{f\}\right)$;
  end for
  Select an f that minimizes the $\mathcal{L}\left(S \cup \{f\}\right)$;
  $S \leftarrow S \cup \{f\}$;
until $\mid S \mid = \mid F \mid$;
Return a sorted set of features S.

4 Experiments

This section presents the performance evaluation of the proposed LDFS-PML method. An ablation experiment is conducted to validate the efficacy of the label disambiguation.

4.1 Datasets

The experiments are conducted on two real-world (Music_emotion and Music_style) and eight synthetic datasets (the remaining ones). The details of the datasets are outlined in Table 1. To generate synthetic PML datasets, we adopt a random strategy, considering the original label count of each sample. Specifically, for a sample tagged with d labels, $d \times t$ false positive labels are

randomly added, where t is set to $\{0.2, 0.4, 0.6, 0.8\}$. For brevity, experiment results on $t = 0.6$ and $t = 0.8$ are reported. To achieve high efficiency, datasets containing over five thousand instances are sampled at a ratio of 0.2.

Table 1. Statistics of PML datasets.

Dataset	#Inst.	#Feat.	#Classes	Domain
flags	194	19	7	Image
water	1060	16	14	Chemistry
virus	207	440	6	Biology
reuters	294	1000	6	Text
guardian	302	1000	6	Text
bbc	352	1000	6	Text
cal500	502	68	174	Music
emotions	593	72	6	Music
music_emotion	6833	98	11	Music
music_style	6839	98	10	Music

4.2 Baselines

Comparative experiments are conducted by comparing LDFS-PML with six state-of-the-art feature selection algorithms. Among these, PMLFS [11] is customized for PML feature selection, and the others are presented for multi-label feature selection, including GLFS from [23], MFSJMI from [24], MCDM from [25], PMFS from [26], and SSFS from [27]. All parameters are tuned within the range suggested by the existing literature. For our proposed LDFS-PML, N is fixed as 3, k is searched in $\{3, 5, 7, 9, 11\}$, and α is searched in $\{0.01, 0.1, 1, 5, 10\}$

For the sake of fairness, all comparative algorithms choose the leading r features for training. Here, r is configured at 20%, 30%, and 40%, corresponding to datasets with feature counts exceeding 500, 100, and 0, respectively. The quality of the selected feature subsets for PML is manifested in the performance of a unified predictor, MLKNN [12].

4.3 Experimental Results

Five commonly used evaluation metrics are applied to the feature significance measure, including average precision ranking loss, hamming loss, one error, and coverage. Since the performance of the later four indexes is similar, only one is reported. The comparison results are reported in Tables 2-3. In general, the proposed LDFS-PML method outperforms the other six algorithms. It's interesting to note that the performances provided by LDFS-PML do not always deteriorate as the ratio of false positive labels increases. This can be attributed

Table 2. Experimental results of LDFS-PML with six state-of-the-art comparison methods on real-world as well as synthetic PML datasets in terms of average precision (the larger, the better).

Data set	t	LDFS-PML	PMLFS	GLFS	MFSJMI	MCDM	PMFS	SSFS
flags	0.6	.807 ± 033	.807 ± 033	.771 ± 025	.786 ± 036	.799 ± 031	.802 ± 039	**.816 ± 026**
	0.8	**.831 ± 035**	.789 ± 034	.792 ± 017	.826 ± 032	.812 ± 032	.809 ± 042	.816 ± 026
water	0.6	**.682 ± 017**	.671 ± 012	.658 ± 015	.650 ± 016	.668 ± 015	.675 ± 011	.662 ± 018
	0.8	**.688 ± 022**	.671 ± 012	.666 ± 014	.657 ± 018	.682 ± 011	.682 ± 016	.662 ± 018
virus	0.6	**.692 ± 057**	.640 ± 065	.669 ± 050	.660 ± 061	.665 ± 041	.666 ± 061	.661 ± 060
	0.8	**.712 ± 049**	.672 ± 052	.672 ± 047	.644 ± 050	.655 ± 074	.662 ± 059	.661 ± 060
reuters	0.6	**.521 ± 072**	.495 ± 060	.511 ± 059	.497 ± 061	.473 ± 058	.491 ± 061	.498 ± 084
	0.8	**.534 ± 092**	.495 ± 052	.512 ± 064	.494 ± 052	.497 ± 085	.500 ± 070	.496 ± 070
guardian	0.6	**.538 ± 064**	.503 ± 052	.492 ± 050	.496 ± 065	.502 ± 059	.489 ± 050	.494 ± 066
	0.8	**.544 ± 055**	.491 ± 069	.513 ± 039	.495 ± 069	.489 ± 063	.511 ± 069	.494 ± 063
bbc	0.6	**.525 ± 036**	.495 ± 045	.517 ± 053	.495 ± 028	.500 ± 036	.489 ± 036	.497 ± 033
	0.8	**.538 ± 049**	.490 ± 053	.520 ± 058	.490 ± 041	.497 ± 037	.487 ± 036	.495 ± 038
cal500	0.6	**.498 ± 014**	.491 ± 015	.493 ± 017	.496 ± 014	.491 ± 016	.492 ± 016	.492 ± 014
	0.8	**.496 ± 011**	.495 ± 014	.494 ± 013	.493 ± 015	.491 ± 016	.493 ± 015	.492 ± 014
emotions	0.6	**.817 ± 041**	.794 ± 038	.800 ± 030	.800 ± 027	.782 ± 033	.777 ± 035	.777 ± 024
	0.8	**.817 ± 035**	.804 ± 040	.788 ± 041	.791 ± 035	.767 ± 031	.776 ± 031	.772 ± 026
mus. em.	-	**.737 ± 003**	.730 ± 003	**.737 ± 003**	**.737 ± 003**	.732 ± 002	.733 ± 002	.733 ± 003
mus. st.	-	**.762 ± 003**	.757 ± 002	.759 ± 002	.761 ± 003	.759 ± 002	.760 ± 001	.760 ± 002

to the fact that LDFS-PML has captured rich label-specific information for label disambiguation, and the information is retained and directly applied for the downstream learning task. The more false positive labels there are, the more label-specific information is captured, which is conducive to PML feature selection.

Out of the 32 cases that are tested using eight synthetic PML datasets, two noise ratios, and two evaluation metrics, LDFS-PML performs better in 29 of them, accounting for 90.6%. The proposed LDFS-PML consistently outperforms the other six comparison algorithms on seven datasets, including water, virus, reuters, guardian, bbc, cal500, and music_style. In some cases, LDFS-PML obtained suboptimal performance, such as on flags and bbc datasets. Specifically, on the music_emotion dataset, LDFS-PML is comparable to GLFS and MFSJMI in terms of average precision. According to the results of experiments, LDFS-PML obtained satisfactory performance.

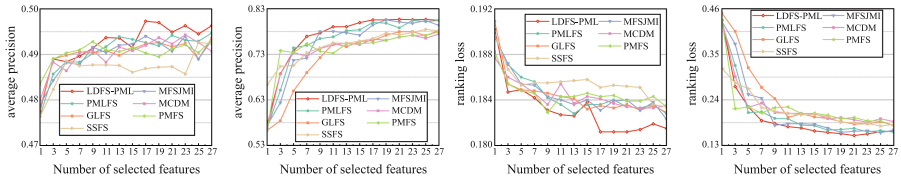
As the above-reported experimental results are based on a fixed number of selected features, the performance of other dimensions might not be adequately reflected. Consequently, we alter the size of the selected feature subset to observe variations in classification performance. Figure 4 shows the outcomes of the cal500 and emotions datasets in terms of average precision and ranking loss. It can be observed that the proposed LDFS-PML method delivers a

Table 3. Experimental results of LDFS-PML with six state-of-the-art comparison methods on real-world as well as synthetic PML datasets in terms of ranking loss (the smaller, the better).

Data set	t	LDFS-PML	PMLFS	GLFS	MFSJMI	MCDM	PMFS	SSFS
flags	0.6	**.208 ± 033**	.218 ± 035	.259 ± 033	.245 ± 045	.221 ± 031	.223 ± 042	.211 ± 024
	0.8	.196 ± 035	.235 ± 038	.236 ± 025	**.188 ± 036**	.211 ± 029	.227 ± 043	.211 ± 024
water	0.6	**.263 ± 008**	.278 ± 013	.285 ± 013	.296 ± 014	.285 ± 017	.271 ± 011	.288 ± 015
	0.8	**.262 ± 011**	.278 ± 013	.278 ± 012	.286 ± 011	.267 ± 013	.271 ± 007	.288 ± 015
virus	0.6	**.203 ± 044**	.233 ± 044	.215 ± 039	.222 ± 045	.216 ± 018	.209 ± 044	.228 ± 035
	0.8	**.203 ± 043**	.226 ± 033	.211 ± 034	.226 ± 045	.232 ± 051	.216 ± 037	.228 ± 035
reuters	0.6	**.386 ± 076**	.402 ± 064	.391 ± 062	.418 ± 071	.443 ± 066	.412 ± 065	.418 ± 086
	0.8	**.363 ± 099**	.413 ± 055	.390 ± 072	.412 ± 055	.406 ± 083	.405 ± 065	.415 ± 074
guardian	0.6	**.381 ± 057**	.417 ± 066	.413 ± 053	.410 ± 059	.407 ± 052	.427 ± 057	.420 ± 070
	0.8	**.380 ± 059**	.423 ± 082	.397 ± 050	.418 ± 070	.432 ± 053	.405 ± 073	.431 ± 071
bbc	0.6	.399 ± 046	.418 ± 054	**.374 ± 050**	.406 ± 036	.401 ± 042	.412 ± 040	.410 ± 040
	0.8	**.376 ± 050**	.436 ± 048	.392 ± 068	.422 ± 045	.422 ± 039	.416 ± 039	.411 ± 042
cal500	0.6	**.182 ± 006**	.183 ± 008	.183 ± 008	.182 ± 006	.183 ± 007	.184 ± 008	.183 ± 007
	0.8	**.181 ± 006**	.182 ± 007	.183 ± 007	.182 ± 007	.183 ± 007	.183 ± 008	.183 ± 007
emotions	0.6	**.148 ± 030**	.168 ± 025	.165 ± 028	.167 ± 023	.181 ± 031	.186 ± 022	.181 ± 021
	0.8	**.154 ± 032**	.164 ± 033	.179 ± 037	.173 ± 028	.192 ± 024	.182 ± 019	.189 ± 024
mus. em.	-	.313 ± 002	.323 ± 002	**.311 ± 002**	.312 ± 003	.319 ± 002	.318 ± 002	.318 ± 002
mus. st.	-	**.403 ± 004**	.409 ± 003	.408 ± 001	.405 ± 004	.409 ± 002	.406 ± 001	.407 ± 004

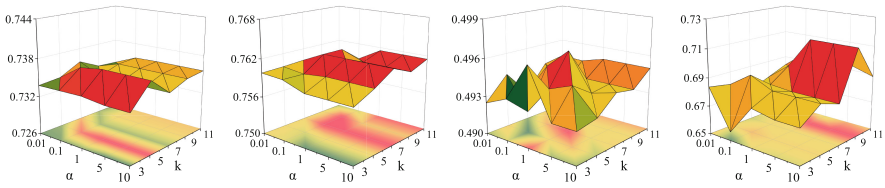
Table 4. Ablation study for LDFS-PML by comparing it with its variant that drops the granular ball-based label disambiguation.

Method	flags	water	virus	reute.	guard.	bbc	cal500	emot.	mu.em.	mu.st.
Average precision (the larger, the better)										
LDFS-PML	**.831**	**.688**	**.712**	**.534**	**.544**	**.538**	**.496**	**.817**	**.737**	**.762**
LDFS-PML-I	.815	.678	.706	.529	.531	.534	.492	.813	.737	.760
Ranking loss (the smaller, the better)										
LDFS-PML	**.196**	**.262**	.203	**.363**	**.380**	**.376**	**.181**	**.154**	**.313**	**.403**
LDFS-PML-I	.218	.269	**.202**	.384	.385	.383	.184	.158	.314	.407



(a) cal500 (b) emotions (c) cal500 (d) emotions

Fig. 4. Classification performance for seven comparison algorithms by varying the number of selected features.



(a) music_emotion (b) music_style (c) cal500 (d) virus

Fig. 5. Performance of LDFS-PML with varying parameter configurations in termsof average precision.

satisfactory result in most cases. The superiority of LDFS-PML becomes increasingly apparent as the size of selected features grows. The superiority of LDFS-PML can be demonstrated compared to the other six algorithms.

Parameter sensitivity is analyzed as shown in Fig. 5. Two real-world PML datasets are used to validate the effectiveness of parameters α and k in terms of average precision. The classification result is more sensitive to parameter k than α. Results from other datasets show similar observations.

4.4 Ablation Study

The proposed LDFS-PML method consists of two stages: ball-based label disambiguation for partial multi-label data, and feature study. To validate the contribution of the label disambiguation, the ablation study is conducted by comparing LDFS-PML with its variant LDFS-PML-I, which drops the label disambiguation stages. The label vectors of PML samples are normalized to $[0, 1]$, with equal significance assigned to the pertinent labels. The results of the ablation study on ten PML datasets in terms of two evaluation metrics are reported in Table 4. It can be observed from the outcomes that, compared to its variant LDFS-PML-I, the full model produces superior values in most cases. This implies that the ball-based label disambiguation technique can mitigate the influence of false positive labels and enhance the feature selection performance in the context of PML.

5 Conclusion

This paper introduces LDFS-PML, a two-stage feature selection approach specifically designed for partial multi-label data. This approach pioneers a unique perspective on label disambiguation by capturing label-specific information. In the first stage of LDFS-PML, a specific granular ball computing technique is devised to capture the label-specific information that is concealed in the data distribution. The label-specific information is applied to construct a labeling confidence matrix. In the second stage of LDFS-PML, labeling confidences containing rich semantics are directly employed in feature selection. To fully utilize the information in labeling confidence, the feature significance measure of LDFS-PML is

devised based on global similarity preservation and local consistency inspired by cognition computing. Interestingly, the method demonstrates enhanced performance as the ratio of noise labels increases. Comprehensive experiments show the superiority of LDFS-PML compared to six state-of-the-art methods on ten datasets.

Acknowledgments. This work is supported in part by the National Natural Science Foundation of China under Grant 62366019 and Grant 62262028; in part by the Natural Science Foundation of Jiangxi Province, China, under Grant 20232BAB202052 and Grant 20224BAB202020; in part by the Jiangxi Province Postgraduate Innovation Special Fund Project under Grant YC2022s390; in part by the Double Thousand Plan of Jiangxi Province, China; in part by the NSFC/Research Grants Council (RGC) Joint Research Scheme under Grant N_HKBU214/21; in part by the General Research Fund of RGC under Grant 12201321, Grant 12202622, and Grant 1220323; in part by RGC Senior Research Fellow Scheme under Grant SRFS2324-2S02.

References

1. Z. Li, G. Lyu, and S. Feng, "Partial multi-label learning via multi-subspace representation," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, C. Bessiere, Ed., 7 2020, pp. 2612–2618
2. Gong, X., Yuan, D., Bao, W., Luo, F.: A unifying probabilistic framework for partially labeled data learning. IEEE Trans. Pattern Anal. Mach. Intell. **45**(7), 8036–8048 (2023)
3. Liu, W., Wang, H., Shen, X., Tsang, I.W.: The emerging trends of multi-label learning. IEEE Trans. Pattern Anal. Mach. Intell. **44**(11), 7955–7974 (2022)
4. Xie, X., Chen, T., Chu, A., Stork, W.: "Efficient network pruning via feature selection," in. International Conference on Pattern Recognition **2022**, 1843–1850 (2022)
5. Yuan, A., Huang, J., Wei, C., Zhang, W., Zhang, N., You, M.: "Unsupervised feature selection via feature-grouping and orthogonal constraint," in. International Conference on Pattern Recognition **2022**, 720–726 (2022)
6. Barlaud, M., Chambolle, A., Caillau, J.-B.: "Classification and feature selection using a primal-dual method and projection on structured constraints," in. International Conference on Pattern Recognition **2021**, 6538–6545 (2020)
7. Khoder, A., Dornaika, F.: "Feature extraction and selection via robust discriminant analysis and class sparsity," in. International Conference on Pattern Recognition **2021**, 7258–7264 (2020)
8. T. Pu, T. Chen, H. Wu, and L. Lin, "Semantic-aware representation blending for multi-label image recognition with partial labels," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, Jun. 2022, pp. 2091–2098
9. T. Chen, T. Pu, H. Wu, Y. Xie, and L. Lin, "Structured semantic transfer for multi-label recognition with partial labels," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, Jun. 2022, pp. 339–346
10. H. Wang, W. Liu, Y. Zhao, C. Zhang, T. Hu, and G. Chen, "Discriminative and correlative partial multi-label learning," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, 7 2019, pp. 3691–3697
11. J. Wang, P. Li, and K. Yu, "Partial multi-label feature selection," in *2022 International Joint Conference on Neural Networks*, Jul. 2022, pp. 1–9

12. T. Xu, Y. Xu, S. Yang, B. Li, and W. Zhang, "Learning accurate label-specific features from partially multilabeled data," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2023
13. You, D., Wang, Y., Xiao, J., Lin, Y., Pan, M., Chen, Z., Shen, L., Wu, X.: Online multi-label streaming feature selection with label correlation. IEEE Trans. Knowl. Data Eng. **35**(3), 2901–2915 (2023)
14. Xia, S., Peng, D., Meng, D., Zhang, C., Wang, G., Giem, E., Wei, W., Chen, Z.: Ball k-means: Fast adaptive clustering with no bounds. IEEE Trans. Pattern Anal. Mach. Intell. **44**(1), 87–99 (2022)
15. Xia, S., Zhang, H., Li, W., Wang, G., Giem, E., Chen, Z.: GBNRS: A novel rough set algorithm for fast adaptive attribute reduction in classification. IEEE Trans. Knowl. Data Eng. **34**(3), 1231–1242 (2022)
16. Xie, M.-K., Huang, S.-J.: Partial multi-label learning with noisy label identification. IEEE Trans. Pattern Anal. Mach. Intell. **44**(7), 3676–3687 (2022)
17. Zhang, M.-L., Fang, J.-P.: Partial multi-label learning via credible label elicitation. IEEE Trans. Pattern Anal. Mach. Intell. **43**(10), 3587–3599 (2021)
18. Y. Yan and Y. Guo, "Adversarial partial multi-label learning with label disambiguation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, May 2021, pp. 10 568–10 576
19. Xu, N., Liu, Y.-P., Zhang, Y., Geng, X.: Progressive enhancement of label distributions for partial multilabel learning. IEEE Transactions on Neural Networks and Learning Systems **34**(8), 4856–4867 (2023)
20. X. Gong, D. Yuan, and W. Bao, "Partial multi-label learning via large margin nearest neighbour embeddings," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, Jun. 2022, pp. 6729–6736
21. S. Xia, X. Dai, G. Wang, X. Gao, and E. Giem, "An efficient and adaptive granular-ball generation method in classification problem," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–13, 2022
22. Q. Zhang, C. Wu, S. Xia, F. Zhao, M. Gao, Y. Cheng, and G. Wang, "Incremental learning based on granular ball rough sets for classification in dynamic mixed-type decision system," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–14, 2023
23. Zhang, J., Wu, H., Jiang, M., Liu, J., Li, S., Tang, Y., Long, J.: Group-preserving label-specific feature selection for multi-label learning. Expert Syst. Appl. **213**, 118861 (2023)
24. Zhang, P., Liu, G., Song, J.: MFSJMI: Multi-label feature selection considering join mutual information and interaction weight. Pattern Recogn. **138**, 109378 (2023)
25. Hashemi, A., Dowlatshahi, M.B., Nezamabadi-pour, H.: MFS-MCDM: Multi-label feature selection using multi-criteria decision making. Knowl.-Based Syst. **206**, 106365 (2020)
26. A. Hashemi, M. Bagher Dowlatshahi, and H. Nezamabadi-pour, "An efficient pareto-based feature selection algorithm for multi-label classification," *Information Sciences*, vol. 581, pp. 428–447, Dec. 2021
27. Gao, W., Li, Y., Hu, L.: Multilabel feature selection with constrained latent structure shared term. IEEE Transactions on Neural Networks and Learning Systems **34**(3), 1253–1262 (2023)

Neural Encoding of Odors: Translating Odors into Unique Digital Representation with EEG Signals

Archana Yadav[1], Vishakha Pareek[2(✉)], Akshay Agarwal[1], and Santanu Chaudhury[3]

[1] Indian Institute of Science Education and Research, Bhopal, India
akagarwal@iiserb.ac.in
[2] TIH iHuB Drishti, Indian Institute of Technology, Jodhpur, India
vishakhapareek94@gmail.com
[3] Indian Institute of Technology, Jodhpur, India
santanuc@iitj.ac.in

Abstract. Electroencephalography (EEG) plays a pivotal role as a non-invasive method for measuring brain activity, offering insights into cognitive processes such as perception, attention, and memory. Its significance lies in its ability to capture real-time neural dynamics with high temporal resolution, making it particularly valuable for studying the complexities of sensory perception, including olfaction. The human olfactory system is crucial for the sense of smell; understanding how the brain processes olfactory stimuli is complex due to the diverse nature of odor molecules and individual variability in perception. This work presents a novel pipeline for encoding unique digital representations of odors. These techniques uncover unique odor representations, providing valuable insights into the intricacies of neural processing. These representations, embedded in matrices, have applications in healthcare, multisensory immersion, and e-commerce. The work explores comprehension of human cognition, while its practical implications underscore its relevance across diverse domains.

Keywords: EEG · Encoder · Olfactory

1 Introduction

The human sense of smell is essential, shaping how humans understand the world, remember experiences, and even feel emotions. Several researches have shown that delving into smell can unlock deep insights into how the human brain works, impacting everything from neuroscience to their understanding of cognition and perception [1], [2]. By studying how the human brain interprets different odors, scientists gain a richer understanding of the brain's sensing principles [3], [4]. This has led to breakthroughs in understanding the complex connections between the

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 280–295, 2025.
https://doi.org/10.1007/978-3-031-78183-4_18

sense of smell and brain activity, paving the way for advancements in various fields.

Odor sampling in humans elicits brain-wide neural activity that EEG devices can capture, a non-invasive method for monitoring brain activity in real-time, which is key to this research. It offers detailed insights into how the human brain responds to smells with high precision. A comprehensive understanding of neural mechanisms underlying odor processing has far-reaching impacts on things like disease diagnosis, healthcare and lifestyle choices, emotions and memory. Despite its powerful influence on emotions, memories, and actions, work on smell perception often receives less attention compared to its other senses.

Despite the well-established quantification methods for visual stimuli, such as RGB values for images, the task remains volatile for olfactory sensations. This challenge arises due to the subject variations in olfactory perception and the complicated nature of the olfactory system, which comprises more than 300 different types of receptors [5]. Understanding how the olfactory system captures odor information and converts it into digital representations within the brain is an interesting and difficult subject within neuroscience. This response will offer an outline of how odors are encoded neurologically and the process by which this data is translated into distinctive digital formats. EEG offers a window into how our brains respond to different odors, enhancing our understanding of olfactory processing. This insight can fuel personalized experiences, creating digital odor encounters in real-time based on individual preferences or physiological reactions, with potential applications spanning marketing, therapy, and entertainment.

Despite much development in understanding the neural mechanisms of perception of the olfactory and processing of the olfactory input into higher brain centers, lots of gaps exist. (Section 2). Various studies have been done related to odor perception. Most studies have determined the specific neural substrates and focused on the identification, but notably, they have not attempted to find or map the unique neural representation of odor. Encoding brain patterns for odors in humans is still significantly unexplored.

Encoded EEG signals will help in the efficient processing and analysis of signals, hence speeding up processing to make research and clinical applications more efficient in the future. Properly encoded EEG signals can be appropriately used in highly advanced signal processing techniques and machine learning algorithms. These included synthetic processes, systems integration with hardware, artificial intelligence, neuromorphic modelling, and computer simulation of neuroprostheses. Accordingly, interpretation and visualization of the encoded EEG signals enable researchers and clinicians to get a description of the underlying neural dynamics and patterns.

To find the odor's encoded pattern, we have proposed two-step approach as follows-

– **Developing Encoder Architecture:** Designing a novel encoder architecture for analyzing EEG responses to different odors, aiming to enhance the precision and efficiency of odor representation in neural data.

– **Creating Unique Digital Representation of Odors:** Formulating a method to translate EEG signals into a distinctive digital format that covers the neural responses associated with each odor.

This unique digital representation encapsulates the neural responses associated with each odor, providing a framework for further analysis and application. Moreover, EEG provides an objective measure of smell's impact on cognitive and emotional states, aiding in the integration of aromas for therapeutic purposes, stress relief, and mood enhancement. The fusion of EEG and odors holds promise for innovative diagnostic tools in healthcare, where specific olfactory stimuli could assist in diagnosing or monitoring neurological disorders by detecting distinct changes in brain activity. Through this approach, we aim to contribute to the advancement of olfactory research and its potential implications in diverse domains, ranging from neuroscience to healthcare and sensory-driven experiences (Figure 1).

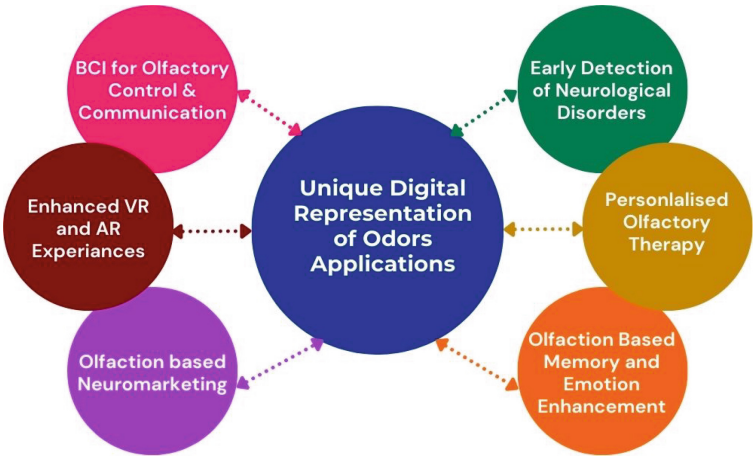


Fig. 1. Application of unique digital representation of odors

2 Related Work

Odors play a vital role in our daily lives, affecting our behaviors, emotions, memories, and overall well-being [6]. Understanding how humans perceive and process odors goes beyond just experiencing smells. Studies have revealed that odors have the power to stir strong emotions and memories, transporting us to specific moments in our past [7]. The human sense of smell also helps us detect potential dangers like spoiled food or harmful chemicals, contributing to our survival instincts [8].

Exploring olfactory information through EEG, this paper delves into the intricate neural dynamics of the human sense of smell. EEG studies offer a unique perspective, capturing the rapid and detailed processes involved in olfactory perception with millisecond precision. These investigations unveil distinct neural activation patterns at different stages of odor processing, from initial sensory detection to higher-order cognitive and emotional responses [9]. Through EEG, researchers have delineated the temporal sequence of events as the brain swiftly identifies and distinguishes odors, engaging memory and emotional centers in the process [7]. Furthermore, EEG-based inquiries highlight the impact of attention, expectation, and context on olfactory processing, revealing the intricate interplay between cognitive factors and sensory experiences. By integrating EEG with advanced analytical techniques like machine learning, new avenues emerge for deciphering previously inaccessible neural signatures, deepening our understanding of olfactory perception's neural foundations.

There have been different studies going on the human olfactory system related to EEG like a study that explores EEG responses to olfactory stimuli during task-related delays [10], discusses the detection of specific EEG components linked to olfactory signals [11], describes an experimental setup for discriminating odors using EEG [12], and utilizes a self-attention network to classify EEG patterns related to olfactory processing [13].

Many researchers worked on using an autoencoder for dimensionality reduction, and noise filtering for EEG data in other application scenarios. Research by Chien et al. designed a new masked autoencoder network using semi-supervised learning [14]. One research used a deep convolutional autoencoder for EEG noise filtering [15]. On the other hand, researchers also worked on the identification of odors using different techniques [16], [17], [18], [19]. However, encoding brain patterns for odors in humans is still significantly unexplored.

This cooperative approach holds immense promise for advancing our understanding of the brain's mechanisms underlying smell perception. Recent studies employing artificial neural networks to investigate the olfactory systems of both mice and flies offer intriguing insights, suggesting convergent evolution [20]. Another study proposes a novel mechanism in the insect olfactory system's mushroom body, utilizing slow lateral excitation between Kenyon cells to decode spatiotemporal information from the antennal lobe, offering insights into temporal decoding in neuronal systems [21]. Decoding olfactory information has been a focus of research for years. Zhu et al. conducted a study mapping olfactory receptors across the olfactory bulb and examining their relationships in the mouse [22].

Most of the significant work has been done for either odor identification or molecule-based odor digitization. Molecule-based digitalization of odors presented [23] makes the approach appealing since it holds a significant promise of high specificity and reproducibility, as assured by existing knowledge in analytical chemistry and flavor science, but human neural perception is missing. Mugihiko Kato et al.'s work on EEG spatiotemporal dynamics of odor representations significantly advanced decoding olfactory information, showing rapid transmission from olfactory areas to regions linked with emotion, semantics,

and memory, with distinct dynamics for different perceptual aspects [24]. In contrast, EEG-based digitization of odors offers a neurophysiological approach, neural representations of odors are accessible during processing and encoding in the brain, in response to the stimuli directly within the electrical activity of the brain. This approach promises to capture, in principle, the subjective, multidimensional nature of olfactory perception since neural responses in the brain inherently reflect the integrated cross-modally processed information about odors.

The human olfactory system commences with the detection of odor molecules by specialized olfactory receptor neurons (ORNs) in the nasal cavity. These ORNs express diverse odorant receptors, each sensitive to particular odor molecules, initiating electrochemical signals transmitted to the olfactory bulb, the primary processing center. In the olfactory bulb, ORN axons converge onto glomeruli, forming synaptic connections with mitral and tufted cells, which then relay olfactory information to higher brain regions. The neural encoding of odors hinges upon the distinctive pattern of activation across ORNs, where each odor molecule elicits a unique activation pattern, meticulously preserved through subsequent neural processing stages, yielding a distinctive digital representation of the odor in the brain. In this work, the EEG-based methods approach is considered to understand how to mimic the pattern of the brain to encode this complex world of odors.

3 Methodology

The human olfactory system is responsible for perceiving, discriminating, and identifying different odors. In this manner, processing the olfactory inputs at the olfactory bulb in a distributed and hierarchical way, the system can develop more and more abstract and finally invariant representations of the raw sensory input, which therefore forms the basis for successful perception and recognition of odors.

An autoencoder is a pair of the encoder network compressing the input data to a lower-dimensional latent space and the decoder network reconstructing the original input from the latent representation [25]. During this process, the autoencoder learns to extract the most salient and informative features of the input data efficiently for an encoding of the input into a compact, yet meaningful, representation.

The conceptual parallelism of the human olfactory system with that of the encoder architecture is relevant to the processing and encoding that the EEG signals will be subjected to within the context of digitizing odors. It means, therefore, that the high temporal resolution of the EEG data and the distributed patterns of activation are related to odorant processing and may be thought of as a complex multidimensional representation of the olfactory processing mechanism of the brain. The overall workflow for this experiment is depicted in Fig. 2. Initially, the EEG data undergoes preprocessing, followed by the generation of encoding representations utilizing an encoder. Subsequently, Singular

Value Decomposition (SVD) is employed to create digital representations for each odor.

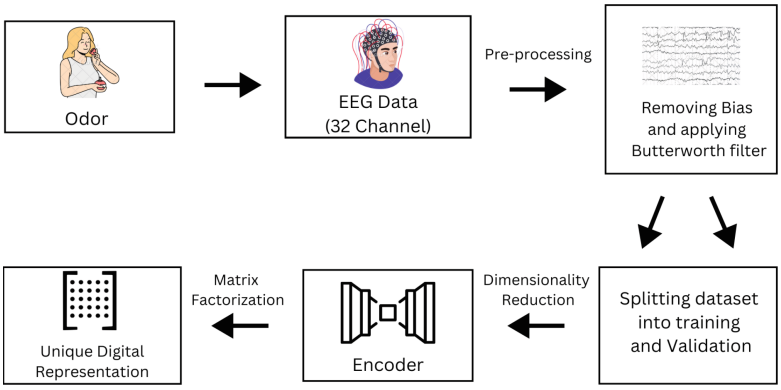


Fig. 2. Sequential Workflow

3.1 Data Collection

This paper uses the dataset, which is available on open-source. This data consists of 11 healthy postgraduates aged 24.9 ± 3.0, including 8 males and 3 females [1]. For each participant, 35 EEG samples were gathered while they smelled each of the thirteen different odors. These samples were recorded for approximately 10 seconds using 32 electrodes. Five of the odors, including the odors of rose, caramel, rotten, canned peach, and excrement, were selected from the T&T olfactometer from Daiichi Yakuhin Sangyo Co., Ltd., Japan, while the remaining eight were sourced from essential oils, such as mint, tea tree, coffee, rosemary, jasmine, lemon, vanilla, and lavender. In total, 5005 EEG samples were collected, resulting from the combination of 11 subjects, 13 odor types, and 35 samples each.

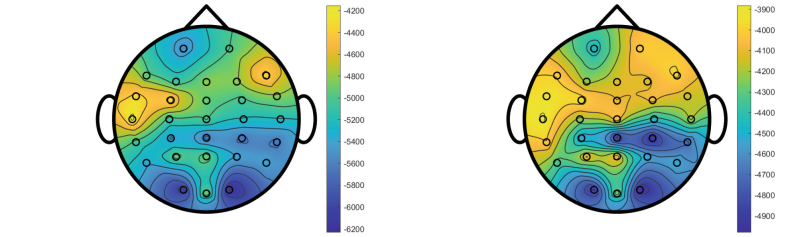
3.2 Data Pre-processing

The data is converted from NS2 format to TXT format using MATLAB, and each file has a 32 x 10010 dimension. There are a total of 5005 files. To mitigate bias, the reference electrodes underwent averaging, while the values of other electrodes were subsequently subtracted from these averaged values. Brain waves are classified into five types: alpha, beta, gamma, delta, and theta, where each one is related to a certain mental state [26]. The butterworth filter is a type of signal-processing filter designed to have a frequency response [16]. We used

[1] https://ieee-dataport.org/documents/olfactory-eeg-datasets-eegdot-and-eegdoc.

butterworth filtering from 10 to 50 Hz, which is alpha, beta, and gamma [27]. The data is then normalized between -1 to 1.

Topographical brain mapping via EEG analyzes brain activity patterns evoked by odors: rose and excrement for this dataset are shown in Figure 3 at a particular time. Color gradients on scalp regions represent EEG intensity, with cooler hues indicating higher activity and warmer hues indicating lower activity. These maps reveal distinct neural activity patterns for each odor, offering insights into how the brain processes and responds to different odors.



(a) (b)

Fig. 3. Topographical Map for (a) Rose (b) Excrement

The dataset is partitioned into training and validation sets, considering subject-specific and odor-specific distinctions. Each subject's data is segregated based on the type of odor, ensuring an equitable representation across both sets. With 30 files allocated per subject per odor for training and 5 files for validation, this approach ensures robust model evaluation and generalization in olfactory research.

3.3 Encoder Network Architecture

In the context of neural encoding of odors, these encoders mimic the early stages of the olfactory pathway to transform complex activation patterns of olfactory receptors into a compact digital representation. By training on diverse olfactory data, the model can extract salient features of odors, effectively translating analog signals into a unique digital code for reconstruction.

The pre-processed data is passed into an autoencoder network consisting of CNN and dense layers. The autoencoder network, illustrated in table 1, comprises an encoder, bottleneck, and decoder. The encoder employs convolutional layers with tanh activation and max-pooling for feature extraction and dimensionality reduction. The bottleneck layer contains dense nodes with tanh activation, facilitating compression. The decoder mirrors the encoder's architecture to reconstruct the input. This hierarchical design enables effective unsupervised feature learning and dimensionality reduction. The autoencoder model is trained on a train-validation dataset for 50 epochs using a batch size of 32 of the training set. The training process utilizes the Adam optimizer with a fixed learning rate

of 0.001. once the autoencoder is trained bottleneck layer is extracted for further experiment.

Table 1. Convolutional autoencoder network architecture

Autoencoder Layer	Configration
Encoder	Conv2D(32x7x7), tanh, MaxPooling(3,7)
	Conv2D(64x5x5), tanh, MaxPooling(5, 5)
	Conv2D(128x3x3), tanh, MaxPooling(2, 2)
Bottleneck	Dense(13), tanh
	Dense(286), tanh
Decoder	Conv2D(128x3x3), tanh, MaxPooling(2, 2)
	Conv2D(64x5x5), tanh, MaxPooling(5, 5)
	Conv2D(32x7x7), tanh, MaxPooling(3,7)
	Conv2D(1x3x3), tanh

3.4 Unique Digital Representation

The mathematical modeling of unique digital representation for each odor is depicted in Figure 4. Encoding representations are extracted from the bottleneck layer of the trained autoencoder model. In Figure 4, the matrix extracted from the bottleneck layer is of size $m \times f$, where m represents the total number of data points, and f denotes the number of features obtained from the bottleneck layer. For each odor, the matrix size is $n \times f$, where n signifies the data points representing that particular odor. The unique representation of each odor is generated through Singular Value Decomposition (SVD), resulting in a matrix of size $s \times f$, where s corresponds to the calculated size of the Σ matrix.

In this study, the encoding representations are utilized for matrix factorization through singular value decomposition (SVD) [28]. Singular value decomposition is a mathematical technique used to decompose a matrix into three constituent matrices, representing the singular values, left singular vectors, and right singular vectors. It is a powerful tool in linear algebra and data analysis, commonly employed for dimensionality reduction, noise reduction, and feature extraction. By identifying principal components capturing the most variance, SVD reduces dimensionality while preserving crucial information. Lower-dimensional embeddings reveal odor similarities, differences, and underlying patterns. Singular values offer insights into principal components' importance, aiding analysis of neural mechanisms in odor perception and recognition.

Let $\mathbf{A}$ be an $m \times n$ real or complex matrix. Then there exist orthogonal matrices $\mathbf{U}$ of size $m \times m$ and $\mathbf{V}$ of size $n \times n$, and a diagonal matrix Σ of size $m \times n$ with non-negative real numbers on the diagonal, such that:

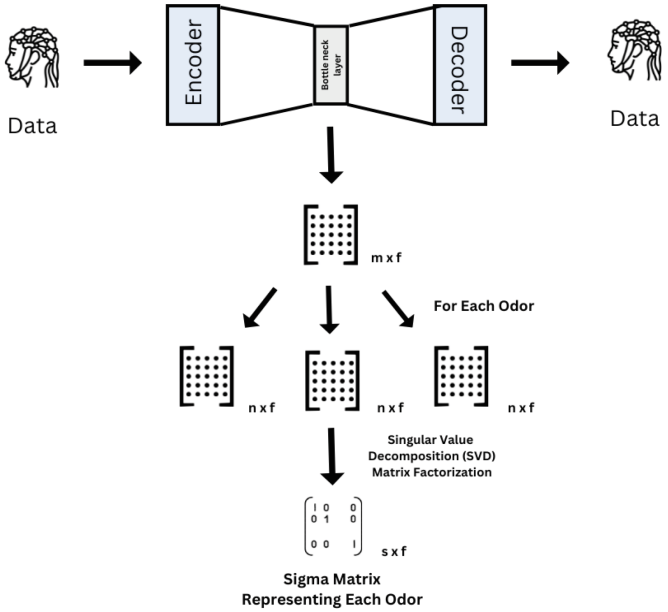


Fig. 4. Sequential workflow from encoder to unique representation

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$$

where $\mathbf{V}^T$ denotes the conjugate transpose of $\mathbf{V}$. The diagonal elements of Σ are the singular values of $\mathbf{A}$, and the columns of $\mathbf{U}$ and $\mathbf{V}$ are called the left and right singular vectors of $\mathbf{A}$, respectively.

Through this method, each odor is uniquely represented by a matrix derived from the encoding representations via Singular Value Decomposition. This approach not only enables the extraction of essential features from EEG data but also unveils complex patterns underlying olfactory responses and odor perception.

4 Result and Discussion

The utilization of encoder networks coupled with matrix factorization techniques has provided insightful results in our study of olfactory data analysis. Through a comprehensive approach encompassing data preprocessing, feature learning via encoders, and subsequent matrix factorization using SVD, we've gained valuable insights into odor perception and underlying patterns within the dataset. Our initial data preprocessing steps were crucial for mitigating biases and ensuring the integrity of the dataset. By averaging reference electrodes and applying a Butterworth filter to the remaining electrodes, we standardized the data and minimized noise, enhancing the robustness of subsequent analyses.

4.1 Encoding Representations Analysis

Following the training of the autoencoder, the entire dataset underwent encoding, with the subsequent extraction of output from the bottleneck layer. The autoencoder network, with its hierarchical architecture comprising convolutional and dense layers, proved effective in capturing essential features and reducing dimensionality. The encoder facilitated feature extraction and dimensionality reduction, while the bottleneck layer compressed the data into a latent space representation. Subsequently, a thorough analysis of Kernel Density Estimation (KDE) plots was conducted on encoding representations. It visually represents the probability density function of a continuous variable, smoothing out data distribution by placing a kernel on each data point and summing them to estimate the overall density. All the unique odor categories got similar KDE plots. This observation is visually depicted in the accompanying KDE plots in Fig. 5. Table 2 shows the statistical study of encoding representations for each smell. The heatmap for the encoding representation is depicted in Figure 6.

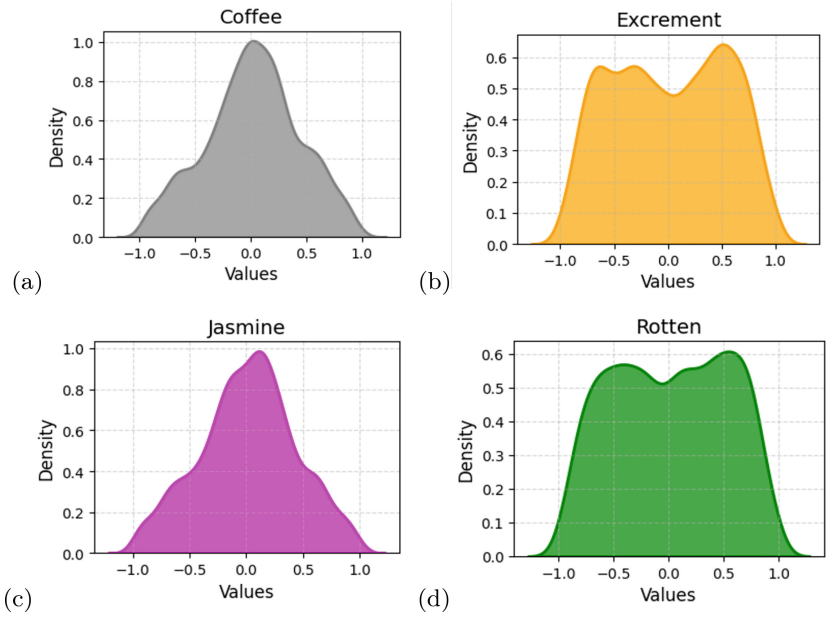


Fig. 5. KDE plots for (a) Coffee (b) Excrement (c)Jasmine (d) Rotten

4.2 Odor Matrix Analysis

We utilize Singular Value Decomposition (SVD) to construct a matrix representation for each odor. Starting from the encoding representations, which encompass values for the entire dataset with each data point comprising 13 features

Table 2. Statistical study of extracted encoding representation

Odor	Mean	Variance	Skewness	Kurtosis	Bandwidth
Jasmine	-0.02584	0.2136	-0.02786	-0.7090	0.1490
Excrement	-0.01818	0.2711	0.03314	-1.2545	0.1678
Lemon	0.00535	0.2084	0.02983	-0.8021	0.1471
Rotten	0.00493	0.2781	-0.02922	-1.2098	0.1699
Rosemary	0.00705	0.2228	-0.15106	-0.7642	0.1521
Tea Tree	-0.02069	0.2122	0.03755	-0.7201	0.1485
Vanilla	0.02390	0.2270	-0.05445	-0.9325	0.1536
Canned Peach	0.06488	0.2858	-0.18891	-1.2583	0.1723
Coffee	0.00085	0.2101	-0.04736	-0.5814	0.1477
Lavender	0.02642	0.2189	-0.08759	-0.6691	0.1508
Rose	0.01704	0.2691	-0.07140	-1.1245	0.1603
Caramel	0.01122	0.2863	-0.04488	-1.1694	0.1632
Mint	0.01523	0.2040	-0.05732	-0.5463	0.1549

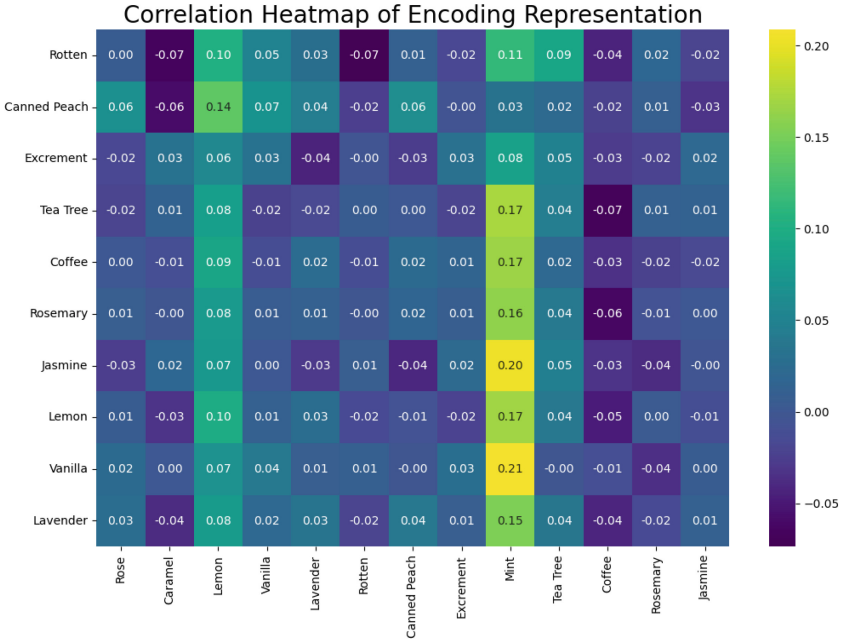


Fig. 6. Correlation heatmap of different odors for encoding representation

(Dense layer with 13 features) of size (5005x13), we generate a matrix for each odor. This matrix, denoted as A, possesses dimensions (385, 13), reflecting the number of data points (385) and features (13) for each odor.

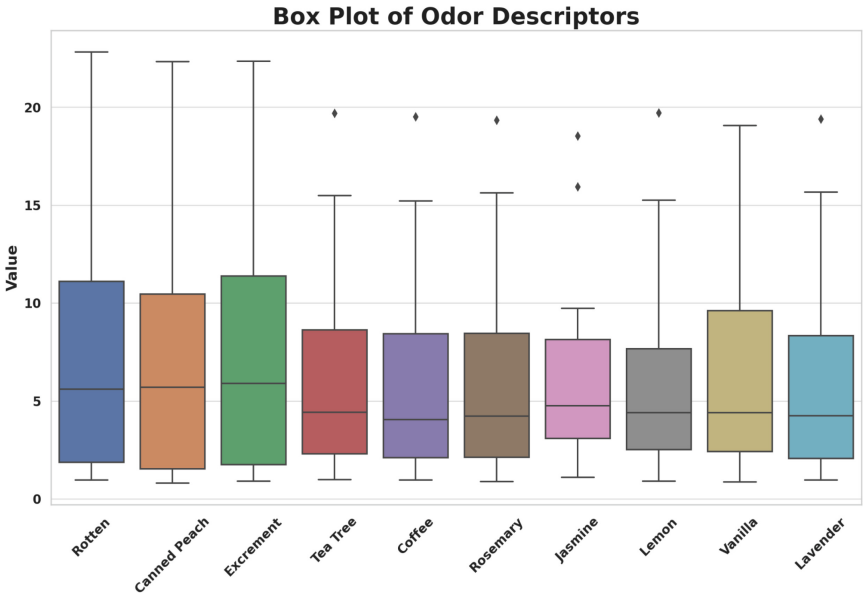


Fig. 7. Box plot for different odors unique representations

Table 3. Statistical study on unique representation generated after applying SVD

Group	Mean	Variance	Skewness	Kurtosis	Bandwidth
Rose	7.65351	45.13421	0.88991	-0.41555	0.08323
Caramel	7.85219	48.60879	0.86518	-0.45977	0.07928
Rotten	7.71066	46.21828	0.87083	-0.45648	0.08591
Canned Peach	7.68822	50.72823	0.79675	-0.74323	0.08990
Excrement	7.73746	47.73887	0.78233	-0.69375	0.08848
Mint	6.76761	32.82482	1.04018	-0.07834	0.07905
Tea Tree	6.38282	29.98615	1.26795	0.47188	0.08976
Coffee	6.23291	29.41055	1.27533	0.53741	0.08975
Rosemary	6.12963	30.18061	1.27750	0.44488	0.09372
Jasmine	6.57075	26.29381	1.22415	0.35686	0.08680
Lemon	6.28444	29.34584	1.31927	0.68870	0.08582
Vanilla	6.49909	29.57909	1.08774	0.01161	0.08619
Lavender	6.22435	29.58958	1.29141	0.52465	0.09166

Upon applying SVD to matrix A, we obtain three constituent matrices: U, Σ, and $\mathbf{V}^T$. The matrix U, sized (385, 13), captures the left singular vectors corresponding to the data points, embodying essential characteristics of each odor. Σ, a diagonal matrix with dimensions (13, 13), contains the singular values,

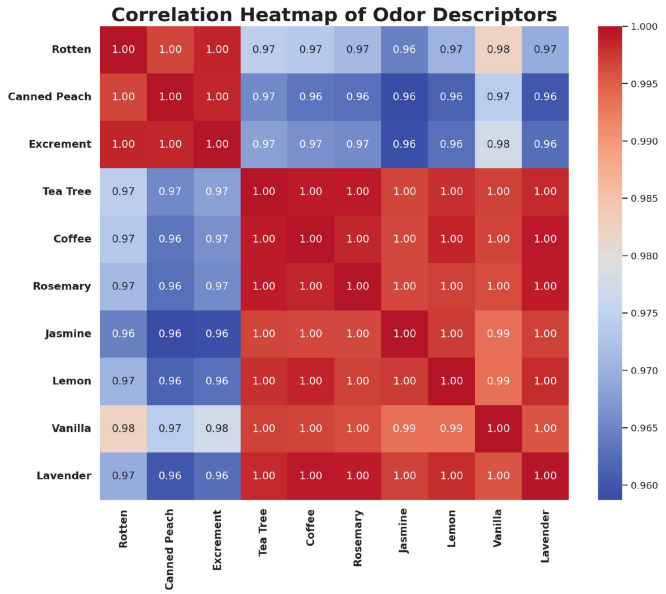


Fig. 8. Correlation heatmap for different odors unique representations

signifying the significance of each feature in the decomposition process. Finally, $\mathbf{V}^T$, with dimensions (13, 13), represents the right singular vectors, offering insights into the underlying structure and relationships within the feature space. This method allows us to effectively extract and represent distinct odor profiles through a structured mathematical framework, facilitating further analysis and interpretation of olfactory data.

Statistical study for the unique representation is provided in Table 3, which shows the difference before applying SVD in Table 2. Figure 7 illustrates the statistical analysis of the diagonal values in the Σ matrix for 10 distinct odors. In contrast, Figure 8 demonstrates that upon applying SVD, the correlation between pleasant and unpleasant odors diminishes as compared to Figure 6. Figure 9 depicts the Euclidean distance between matrices, showcasing a wide range of values. Lower values signify closer or more similar odor profiles, while higher values indicate greater differences between profiles. This analysis provides valuable insights into the relative proximity or disparity between pairs of odor descriptors, crucial for understanding odor similarity and dissimilarity.

The SVD matrix facilitates a comprehensive analysis of Odor A's unique characteristics. Through odor reconstruction, essential features defining its digital signature are revealed. Feature extraction from left and right singular vectors uncovers key olfactory attributes and neural mechanisms shaping Odor A's perception. Dimensionality reduction via principal components enables visualization, potentially unveiling inherent patterns. Furthermore, comparison with other odors using row vectors assesses similarity or dissimilarity, aiding in tasks

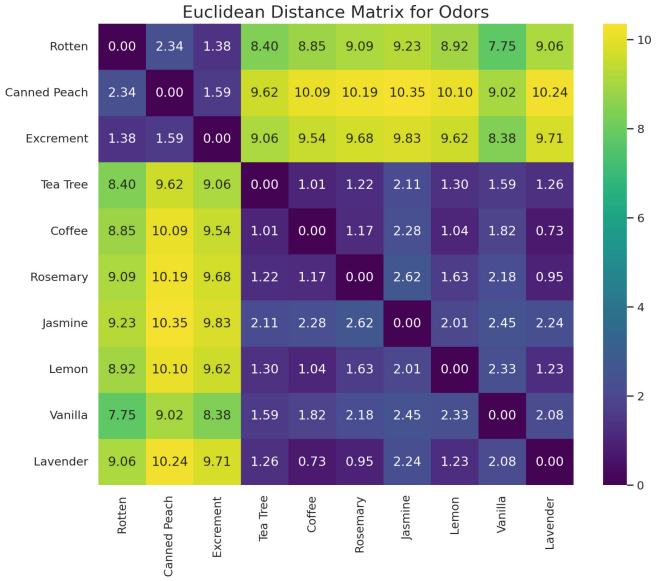


Fig. 9. Euclidian distance for 10 different odors unique representations

like identification and categorization. Insights from this analysis inform various applications including fragrance and flavor development, advancing understanding of olfactory perception's neural basis. We wanted to benchmark our process to demonstrate its value and efficacy. To our knowledge, no work explicitly addresses the task of converting odors into distinct digital representations using EEG signals. This gap in the literature prevents direct comparison with state-of-the-art techniques. In the future, we will likely benchmark the process with other odor datasets. We are providing the code at this link [2].

5 Conclusion

This paper presents a combined approach of encoder-based feature learning and matrix factorization through SVD has enabled us to uncover valuable insights into olfactory data analysis. While our results demonstrate promising capabilities in odor perception and feature extraction, further refinement and expansion of the dataset will enhance the model's discriminative power and generalizability. This study lays the foundation for future research endeavors aimed at explaining complex olfactory phenomena and advancing our understanding of sensory perception. The encoding of olfactory information holds immense potential across various sectors. The convergence of EEG with olfactory research yields diverse applications. In medical studies, EEG explores how odors affect brain activity, aiding early disease detection like Alzheimer's. Therapeutically, EEG tailors

[2] https://github.com/A-2809/Neural-Encoding-of-Odors.

olfactory interventions for mental health, alleviating conditions such as anxiety. Consumer research benefits from EEG's neuroanalysis of odor responses, refining product development in industries like perfumery. Interactive learning leverages EEG-Odor integration to adapt scenarios, enhancing engagement. Entertainment, notably in virtual reality and gaming, utilizes EEG to create immersive environments, adjusting scents based on player decisions. Aromachology benefits from EEG's quantification of odor impacts on emotions and cognition, enriching understanding. Overall, encoding olfactory information has promising applications that span from healthcare to marketing and environmental monitoring.

References

1. Yang Chen. Olfactory display: Development and application in virtual reality therapy. In 16th International Conference on Artificial Reality and Telexistence–Workshops (ICAT'06), pages 580–584, 2006
2. Sankaran, S., Khot, L.R., Panigrahi, S.: Biology and applications of olfactory sensing system: A review. Sens. Actuators, B Chem. **171–172**, 1–17 (2012)
3. Paul E Keller, Richard T Kouzes, Lars J Kangas, and Sherif Hashem. Transmission of olfactory information for telemedicine. In Interactive technology and the new paradigm for healthcare, pages 168–172. IOS Press, 1995
4. Georgescu M Perederco C Sarafoleanu C, Mella C. The importance of the olfactory sense in the human behavior and evolution. Journal of medicine and life, Apr-June 2009
5. Kim S Sowndhararajan K. Influence of fragrances on human psychophysiological activity: With special reference to human electroencephalographic response. Scientia pharmaceutica, 84:724–751, 11 2012
6. Rachel Herz. Aromatherapy facts and fictions: A scientific analysis of olfactory effects on mood, physiology and behavior. The International Journal of Neuroscience, 119:263–90, 02 2009
7. Boakes R. A. Stevenson, R. J. A mnemonic theory of odor perception. psychological review. National Library of Medicine, 110, 04 2003
8. Gérard, B., Charlotte, B., Laurence, J.: Odor hedonic profile (ohp): a self-rating tool of everyday odors. Front. Neurosci. **17**, 04 (2023)
9. Jonas Olofsson, Nicholas Bowman, Katherine Khatibi, and Jay Gottfried. A time-based account of the perception of odor objects and valences. Psychological science, 23:1224–32, 09 2012
10. Ivan, Ninenko., Daria, Kleeva., Nikita, Bukreev., Mikhail, A., Lebedev. (2022). EEG correlates of olfactory processing during an instructed-delay task. bioRxiv, https://doi.org/10.1101/2022.12.06.519284
11. Ninenko, Ivan, et al. "Brain-computer interface for olfaction: Detecting olfactory related EEG components." 2022 Fourth International Conference Neurotechnologies and Neurointerfaces (CNN). IEEE, 2022
12. Ivan, N., Daria, K., Nikita, B., Mikhail, A., Lebedev.: An experimental paradigm for studying EEG correlates of olfactory discrimination. Front. Hum. Neurosci. **17**,(2023). https://doi.org/10.3389/fnhum.2023.1117801
13. Chengxuan, Tong., Yi, Ding., Kevin, Lim, Jun, Liang., Zhuo, Zhang., Haihong, Zhang., Cuntai, Guan. (2022). TESANet: Self-attention network for olfactory EEG classification. 1-7. https://doi.org/10.1109/IJCNN55064.2022.9892920

14. Hsiang-Yun Sherry Chien, Hanlin Goh, Christopher M. Sandino, and Joseph Y. Cheng. Maeeg: Masked auto-encoder for eeg representation learning, 2022
15. Niago Moreira Nobre Leite, Eanes Torres Pereira, Edmar Candeia Gurjao, and Luciana Ribeiro Veloso. Deep convolutional autoencoder for eeg noise filtering. In 2018 IEEE international conference on Bioinformatics and biomedicine (BIBM), pages 2605–2612. IEEE, 2018
16. Maham Saeidi, Waldemar Karwowski, Farzad V. Farahani, Krzysztof Fiok, Redha Taiar, P. A. Hancock, and Awad Al-Juaid. Neural decoding of EEG signals with machine learning: A systematic review. Brain Sciences, 11(11), 2021
17. Zhang, M.Z.H.-R.H.X.-N., Meng, Q.-H.: Decoding olfactory EEG signals for different odor stimuli identification using wavelet-spatial domain feature. J. Neurosci. Methods **363**, 109355 (2021)
18. Aydemir, O.: Olfactory recognition based on eeg gamma-band activity. Neural Comput. **29**(6), 1667–1680 (2017)
19. 30 Ming Zeng Xiao-Nei Zhang, Qing-Hao Meng. A novel channel selection scheme for olfactory eeg signal classification on Riemannian manifolds. Journal of Neural Engineering, 19(4), 2017
20. Peter Y. Wang, Yi Sun, Richard Axel, L.F. Abbott, and Guangyu Robert Yang. Evolving the olfactory system with machine learning. Neuron, 109(23):3879–3892.e5, 2021
21. Nowotny, T., Rabinovich, M.I., Huerta, R., Abarbanel, H.D.: Decoding temporal information through slow lateral excitation in the olfactory system of insects. J. Comput. Neurosci. **15**, 271–281 (2003)
22. Burton Shawn D. Nagai Maira H.-Silverman Justin D. de March Claire A. Wachowiak Matt Matsunami Hiroaki Zhu, Kevin W. Decoding the olfactory map through targeted transcriptomics links murine olfactory receptors to glomeruli. Nature Communications, 13, 2022
23. Mayhew E. J. Sanchez-Lengeling B.-Wei J. N. Qian W. W. Little K. A. Andres M. Nguyen B. B. Moloy T. Yasonik J. Parker J. K. Gerkin R. C. Mainland J. D. Wiltschko A. B. Lee, B. K. A principal odor map unifies diverse tasks in olfactory perception. Science (New York, N.Y.), 381(6661):999–1006, 2023
24. Kato, M., Okumura, T., Tsubo, Y., et al.: Spatiotemporal dynamics of odor representations in the human brain revealed by EEG decoding. Proc. Natl. Acad. Sci. U.S.A. **119**(21), e2114966119 (2022). https://doi.org/10.1073/pnas.2114966119
25. Zhai, Junhai, et al. "Autoencoder and its various variants." 2018 IEEE international conference on systems, man, and cybernetics (SMC). IEEE, 2018
26. Abhang, Priyanka, Gawali, Bharti, Mehrotra, Suresh. (2016). Technological Basics of EEG Recording and Operation of Apparatus. https://doi.org/10.1016/B978-0-12-804490-2.00002-6
27. Hui-Rang Hou, Rui-Xue Han, Xiao-Nei Zhang, and Qing-Hao Meng. Pleasantness recognition induced by different odor concentrations using olfactory electroencephalogram signals. Sensors, 22(22), 2022
28. Zhang, Y., Xu, G. (2009). Singular Value Decomposition. In: LIU, L., ÖZSU, M.T. (eds) Encyclopedia of Database Systems. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-39940-9

Robust Feature Space Organization with Distillation for Few-Shot Object Detection

Vineet R. Shenoy(✉) and Rama Chellappa

Johns Hopkins University, Baltimore, MD 21210, USA
{vshenoy4,rchella4}@jhu.edu

Abstract. Few-Shot Object Detection has received strong interest recently, especially as collecting annotated training data for new and varied problems becomes time-consuming and expensive. In Few-Shot Object Detection, after training on ample "base" class data, the detector is finetuned on few examples of "novel" classes such that the inference performance is maximized over all classes in a test set. Most previous works aim to learn novel classes independently, without regard to interactions with semantically similar classes. We take a different view, integrating contrastive learning with synthesized positive samples from similar but distinct classes. We generate new positive samples via a convex combination between a novel class feature and its closest classes' features; then, we encourage the novel class feature to be close to the generated features using contrastive learning. Our method achieves state-of-the-art results on the standard PASCAL-VOC and MS-COCO benchmarks, and does so with minimally extra learnable parameters during training and no extra parameters during inference.

1 Introduction

On the task of Few-shot Object Detection, object detectors are first trained on ample "base" class data that contains a subset of all classes to be recognized during inference; then, the detector is further trained on a set of "novel" class examples – for which only limited training samples exist – such that performance over a test set of both base and novel examples is maximized. This problem has received increased attention, especially for tasks in which sufficient labeled data is not available. This frequently occurs when the data domain shifts (i.e. ground-based camera to aerial cameras), privacy laws prevent the release of data (i.e. medical imaging), or data is prohibitively expensive to collect and annotate.

Early methods to solve this problem simply train on ample base data, and later fine-tune on the novel examples [39]; this method, while exhibiting good performance, *overfits* on the novel examples and compromises its performance on base classes. During inference time, these detectors are biased towards recognizing base classes, but do so less effectively than base-only training. Recent works

Partially supported by an ONR MURI grant N00014-20-1-2787.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 296–312, 2025.
https://doi.org/10.1007/978-3-031-78183-4_19

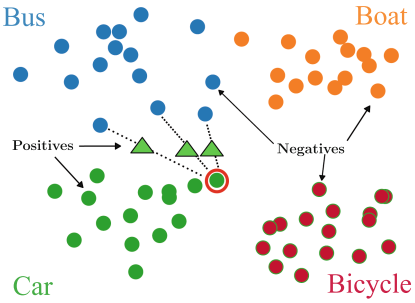


Fig. 1. A visualization of the features we generate. We find the semantically closest class to green (in this case, blue), and generate new positives (light green triangles). The remaining samples, from the closest class and other classes, are used to mine negatives for contrastive learning. (Color figure online)

have alleviated some of these issues; gradient optimization methods [11,34], contrastive learning among different categories [38], and knowledge distillation between k-closest base and novel categories [42] have each pushed the state-of-the-art forward. Contemporary works such as FSCE [38] and MFDC [42] use contrastive proposal encoding and base-novel distillation, respectively, to improve few-shot object detection. One of the main drawbacks of contrastive proposal-based encoding is that its supervised contrastive loss only considers intra-class samples as positives, but not the relationship among classes; and MFDC considers relations among the classes but not the interaction of samples within a class.

In the data-scarce scenario, it is critical to integrate these ideas. We exploit the key concepts of contrastive learning and knowledge distillation; noting that the closest classes to a feature of interest are relevant to feature space, we propose to sample features along the line from a novel class feature to the closest class using MixUp techniques [24,35,50], and add these feature as positives for contrastive learning. Our method, compared to new generative methods [12], requires no additional trainable modules to sample the new features. The only parameters that are required are light-weight key and query encoders parameters in the style of MoCo contrastive learning [18] during training; during inference, we no longer need these modules and their associated parameters. Through our algorithm, we address three key issues in few-shot object detection: 1) we add new samples to address the data-scarcity issue; 2) we address the overfitting issues in few-shot learning by improving the diversity of samples through mixup of features; and 3) we separate classes more effectively in feature space by leveraging these mixed-up features in a contrastive learning framework.

We test our method on the standard PASCAL-VOC and MS-COCO benchmarks, and show performance increases on each benchmark. In addition, we visualize feature space and show that our method organizes the feature space to optimize decision boundaries.

To summarize, we make the following contributions:

- We generate new features between similar classes using mixup techniques and demonstrate that this feature generation helps alleviates the data scarcity and data diversity challenges in few-shot object detection.
- We integrate the aforementioned synthesized positives samples using a supervised contrastive loss to improve few-shot object detection performance. We show that feature space is organized optimally for few-shot detection.
- We achieve state-of-the-art results on the standard PASCAL-VOC and MS-COCO benchmarks to prove the effectiveness of our method.

2 Related Works

2.1 Few-Shot Object Detection

Few-Shot object detection methods can be split into Meta-Learning-based approaches [6,8,14–16,26,33,45,49] and Transfer learning approaches [28,39,40,47,51]. Meta-Learning methods train in an episodic manner; during "base" training, e episodes are sampled from a task distribution which contains N-classes and K-shots (the support set), and a model learns to distinguish the N classes. During the "novel" fine-tuning stage, episodes of the N novel classes are presented, and a network is expected to quickly adapt to this task. Many of these methods follow dual-branch architectures, where a "support branch" is designed to extract information that can be used to generate predictions in the "query" branch through some aggregation procedure. Attention RPN [8] observed that a detector trained on base classes struggles to generate proposals for novel classes, and therefore deployed methods such as average-pooling and depth-wise cross-correlation of query and support features before feeding them to the Region Proposal Network. Work such as FsDetView [45] concatenated support and query features multiplied channel-wise, the difference of these features, and the query features themselves for detection and viewpoint estimation. Meta Faster R-CNN [15] expanded on [45] by passing the features generated by FsDetView through small convolutional networks before computing the loss. Other works exploited spatial information during aggregation [27], attention maps [2,20], and multi-level aggregation [16]. While Meta-Learning approaches have shown good promise, we build upon Transfer Learning-based methods, and direct readers to a thorough review of both Meta-Learning (and Transfer-Learning) methods presented in [25].

Transfer Learning-based methods train a detector on copious base data during its first stage and train on novel examples during a second stage. Among the first Transfer Learning methods is TFA [39], which showed that simply fine-tuning on the novel examples resulted in good performance. TFA trained the base detector identically to the traditional Faster-RCNN [10] (using only the base data and classes), and later, froze the entire network except classifier and regressor during the novel fine-tuning stage. DeFRCN [34] adopted a similar framework, but unfroze all the network components; noting that the objectives of the RPN and RoI heads are competing, the authors decided to stop the gradient from the RPN and scale the gradient from RoI head, achieving state-of-the-art

performance. Built upon DeFRCN is CFA [11], an optimization method that backpropagates gradients if the angle between the base and novel gradients is acute; otherwise the base and novel gradients are projected to the averaged gradient. While we draw inspiration from [38], we make significant and important changes to the Supervised Contrastive Learning framework: we consider similar (i.e. cosine-similarity) but different classes to contain "positive" examples for contrastive learning, and mix the novel features with samples from this different class. We generate these features without any additional trainable modules. The details are presented in Sect. 3.

2.2 Contrastive Learning

The key goal of contrastive learning is to organize feature space to optimize similarity and dissimilarity among data samples, and has been widely used in self-supervised and unsupervised learning [5,19,44]. After early work in Face Recognition to keep latent vectors of "positives" samples closer than those of "negatives" in pairs or triplets [4,36,37], one of the most widely used loss function is InfoNCE [32], which aims to recognize a positive example in the presence of $N-1$ noise examples. This loss is built upon the family of noise-contrastive losses [13,31]. Self-supervised learning architectures adopted similar losses: in SimCLR [3], two augmented versions of a sample were passed through encoder and projection functions, and the similarity of these samples were maximized in the presence of the remaining negatives samples from the batch. Learning strong representations requires large batch sizes, which can exceed training memory; the MoCo [18] framework avoided this by storing previous embeddings in a queue, the encoder of which is a momentum update of the key encoder. Barlow Twins [48] also passed two augmented versions of samples through networks, but unlike SimCLR, seeks to make the cross-correlation between the representations to be the identity, thus reducing redundancy.

Unlike self-supervised approaches, few-shot object detection has access to class labels, which naturally allow applications of the Supervised Contrastive Loss [23]. We show that our novel use of newly-generated samples integrated into a distillation-based few-shot learning scheme improves performance without requiring any additional learnable parameters.

3 Method

3.1 Problem Formulation

Formally, consider the sets of base classes C_b and novel classes C_n from dataset $\mathcal{D}$ such that $C = C_b \cup C_n$. Given at least M examples per base class C_b, and at most K examples per novel class C_n with $K << M$, the goal is to find a model $F(\cdot)$ with highest performance on the test set of $\mathcal{D}$. During the base training stage, all previous methods train on the same base dataset $\mathcal{D}_{\text{base}}$; however, during fine-tuning on novel classes, assumptions differ as to what subset of novel and base

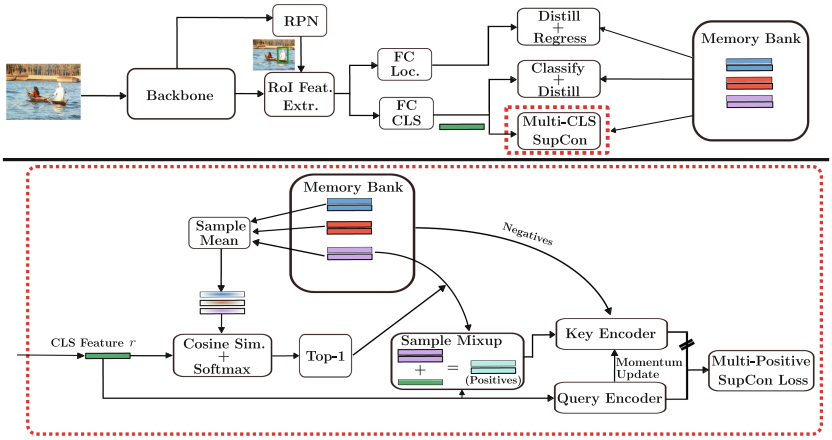


Fig. 2. The main pipeline of our work. We add a Supervised Contrastive branch to the [42] (top, red box); unlike FSCE [38], we consider the closest class – defined via cosine similarity – to contain positives, and generate new samples via a convex combination of the proposal of interest and the most similar features of the proposal's closest class (bottom). (Color figure online)

classes are available. We follow the work of MFDC [42], which assumes that the full base dataset is available during novel fine tuning in addition to K novel examples per-class.

3.2 Approach

Our goal is to address three issues in few-shot object detection: 1) data scarcity; 2) overfitting, and 3) feature space organization. Noting that distillation of base-novel commonalities as well contrastive learning on proposals show performance increases, we seek to integrate them to address the issues listed above. Our contribution involves contrastive learning on proposals in which positive examples not only belong to the same class, but also belong to a mixture of features from the closest class. Previous work [35] has shown that mixing features improves few-shot learning, and [21] showed mixing for hard negatives improves contrastive learning. Our intuition is that not only should features from the same class be close together (and further from unrelated classes), but they should be relatively close to different but related classes. Note that this differs from FSCE [38] as that work only considered positives to be those examples from the same class; we include mixed samples from the top related class, similar to the distribution distillation in MFDC [42]. Unlike previous methods that generate new samples through an intensive generative process [12], our method requires no generative training.

Formally, consider a novel-class foreground region proposal r, its extracted RoI feature $\mathbf{v}_r$, and its associated ground-truth class as c_{gt} as shown in Fig. 2, bottom. We compute the cosine similarity between this feature and the prototype

of each class c in the dataset given by $\mu_{\mathbf{c}} = \frac{1}{N_c}\Sigma_{i=1}^{N_c} f_c^i$, where f_c^i are obtained from a memory bank $\mathcal{M}$.

$$d_r^c = \frac{\mathbf{v}_{r,c_{\text{gt}}} \cdot \mu_c}{\|\mathbf{v}_{r,c_{\text{gt}}}\|\|\mu_c\|}, c \in \mathcal{C}_n \cup \mathcal{C}_b \setminus \{c_{\text{gt}}\} \tag{1}$$

Next, we seek the class with which $\mathbf{v}_r$ shares the highest similarity. To do so, we first softmax-normalize $\mathbf{d_r}$, representing scores for all classes, to create a score vector $\mathbf{q}_r$.

$$q_{r,c} = \frac{\exp(d_r^c)}{\Sigma_{i=1}^{C}\exp(d_r^i)}, c \in \mathcal{C}_n \cup \mathcal{C}_b \setminus \{c_{\text{gt}}\} \tag{2}$$

The scores $\mathbf{q}_r$ are sorted from largest to smallest, signifying the most similar to least similar classes in terms of cosine similarity. Our next goal is to alleviate data scarcity and diversity issues by generating new features, and arranging feature space such that similar classes are closer than different classes.

We implement this idea in a Supervised Contrastive Learning [23] framework. Unlike previous works, which only consider as "positive" those features belonging to c_{gt}, we consider the closest classes' features as candidates to mine more positives. Let $\mathbf{f}_c^i \in \mathcal{C}_b \cup \mathcal{C}_n / \{c_{\text{gt}}\}, i = 1, \ldots, N_c$ be features from the most similar class $c = \arg\max(\mathbf{q}_r)$ in the memory bank $\mathcal{M}$. We compute the cosine similarity $s_{r,i}$ between $\mathbf{f}_c^i$ and $\mathbf{v}_{r,c_{\text{gt}}}$ for $i = 1, \ldots, N$ and select P features $\mathbf{f}_c^i$ with greatest similarity to $\mathbf{v}_{r,c_{\text{gt}}}$, where P is a hyperparameter which we ablate in Sect. 4.4. We use MixUp [50] to blend features from $\mathbf{v}_{r,c_{\text{gt}}}$ and $\mathbf{f}_c^i, i = 1, \ldots, P$, weighted by its cosine similarity. Formally, this becomes

$$\mathbf{g}_r^i = s_{r,i} \cdot \mathbf{v}_{r,c_{\text{gt}}} + (1 - s_{r,i}) \cdot \mathbf{f}_c^i \tag{3}$$

Here, we impose that $s_{r,i} > 0.5$ to ensure that the synthesized feature $\mathbf{g}_r^i$ remains closer to $\mathbf{v}_{r,c_{\text{gt}}}$ than $\mathbf{f}_c^i$. The new features $\mathbf{g}_r^i$ are labeled as $(\mathbf{g}_r^i, c_{\text{gt}})$, i.e. labeled as the same class as $\mathbf{v}_{r,c_{\text{gt}}}$ which ensures that the synthesized feature is treated as a "positive" during contrastive learning. Through this feature generation, we have addressed two drawbacks in few-shot object detection: 1) we have added new samples, alleviating the data scarcity issues, and 2) we have improved the diversity of novel samples, alleviating the overfitting issue.

The supervised contrastive learning framework requires negatives to be effective; among the remaining memory bank samples $\mathbf{f}^i \in \mathcal{M}$ that have not already been labeled as positive, we compute the cosine similarity between $\mathbf{v}_{r,c_{\text{gt}}}$ and $\mathbf{f}^i$ and obtain scores $n_{r,i}$ and take as negatives the Q most similar vectors. We therefore add the hard negatives for robust feature space learning. The supervised contrastive learning paradigm is implemented in a MoCo [18] framework; given an input sample, referred to as the "query", we seek to keep this close to all other "keys" that are of the same label. As in the MoCo framework, we pass the "query" features through a query encoder and "key" features through the "key" encoder, where the key encoder is updated via a momentum update of the trainable query encoder. Let $\mathbf{z}_i = \text{Proj}(\mathbf{v}_{r,c_{\text{gt}}})$ be the projected feature from the query encoder and $\mathbf{z}_p = \text{Proj}(\mathbf{g}_r^i)$ be the projected features from the

key encoder; $\mathbf{z}_a$ are also from the key encoder. These features are passed to a Supervised Contrastive Loss [23], which is defined as

$$\mathcal{L}_{\text{supcon}} = \frac{-1}{|P(i)|} \Sigma_{p \in P(i)} \mathcal{L}(\mathbf{z}_i, \mathbf{z}_p) \tag{4}$$

$$\mathcal{L}(\mathbf{z}_i, \mathbf{z}_p) = \log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_p / \tau)}{\Sigma_{a \in A(i)} \exp(\mathbf{z}_i \cdot \mathbf{z}_a / \tau)} \tag{5}$$

where $a \in A(i)$ indexes the set of all samples not including i and $P(i)$ is the set of positives. In the limited data scenario, where many stored proposals are similar to each other, the addition of these positives add variation for contrastive learning, and as shown empirically in [23], increasing the number of positives is important to increase the loss for learning. In this way, we have addressed the third issue of few-shot object detection: we separate the classes more effectively in feature space by adding synthesized positives and hard negatives in the contrastive loss. Denoting the traditional Faster-RCNN [10] loss as $\mathcal{L}_{\text{det}}$ and the MFDC [42] loss as $\mathcal{L}_{\text{mfdc}} = \lambda_c \mathcal{L}_{distill-cls} + \lambda_l \mathcal{L}_{distill-loc} + \lambda_d \mathcal{L}_{distill-dist}$ we are left with a final loss given by

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{det}} + \mathcal{L}_{\text{mfdc}} + \lambda_{\text{conmix}} \mathcal{L}_{\text{supcon}} \tag{6}$$

where λ_{conmix} is a hyperparameter. In Sect. 4, we discuss implementation details and experimental results to show the effectiveness of our method.

4 Implementation Details and Experimental Results

Table 1. nAP50 for all splits of VOC. Best performing methods are in red, while second best are in blue

Methods	Split 1					Split 2					Split 3				
	1	2	3	5	10	1	2	3	5	10	1	2	3	5	10
TFA w/cos (ICML '20) [39]	39. 8	36.1	44.7	55.7	56.0	23.5	26.9	34.1	35.1	39.1	30.8	34.8	42.8	49.5	49.8
FSCE (CVPR '21) [38]	44.2	43.8	51.4	61.9	63.4	27.3	29.5	43.5	44.2	50.2	37.2	41.9	47.5	54.6	58.8
Rententive RCNN (CVPR '21) [9]	42.4	45.8	45.9	53.7	56.1	21.7	27.8	35.2	37.0	40.3	30.2	37.6	43.0	49.7	50.1
DeFRCN (ICCV '21) [34]	53.3	47.4	58.7	58.8	59.6	33.0	37.0	49.5	53.8	48.5	47.1	45.8	52.7	52.8	52.6
CFA (CVPRW '22) [11]	59.0	63.5	66.4	68.4	68.3	37.0	45.8	50.0	54.2	52.5	54.8	58.5	56.5	61.3	63.5
MFDC (ECCV '22) [42]	63.4	66.3	67.7	69.4	68.1	42.1	46.5	53.4	55.3	53.8	56.1	58.3	59.0	62.2	63.7
VFA (AAAI '23) [17]	57.7	64.6	64.7	67.2	67.4	41.4	46.2	51.1	51.8	51.6	48.9	54.8	56.6	59.0	58.9
Norm-VAE (CVPR '23) [46]	62.1	64.9	67.8	69.2	67.5	39.9	46.8	54.4	54.2	53.6	58.2	60.3	61.0	64.0	65.5
Ours	65.53	66.27	69.41	70.00	68.12	42.67	47.27	53.04	56.24	54.13	58.00	62.01	60.78	64.52	65.61
MFDC (ECCV '22) [42]	51.34 ±6.62	63.86 ±4.45	64.07 ±3.82	68.57 ±1.54	68.46 ±1.65	**36.57 ±4.61**	**46.78 ±2.28**	49.23 ±2.06	**52.87 ±1.84**	55.13 ±1.67	45.96 ±6.74	**58.71 ±3.46**	**58.57 ±2.90**	62.79 ±1.71	65.09 ±0.71
Ours	**51.98 ±6.44**	**64.21 ±4.10**	**64.88 ±3.59**	**68.91 ±1.52**	**68.54 ±1.21**	36.20 ±5.13	46.13 ±2.56	**49.32 ±2.17**	52.67 ±1.50	**55.35 ±1.46**	**46.06 ±6.63**	58.61 ±3.62	58.55 ±3.22	**63.01 ±1.51**	**65.10 ±0.60**

4.1 Benchmark Datasets and Evaluation Metrics

Following the evaluation protocols designed by previous methods, we evaluate our approach on the PASCAL-VOC [7] and MS-COCO datasets [29]. The PASCAL-VOC dataset, containing a total of twenty classes, is divided into fifteen base classes and five novel classes (i.e. $|\mathcal{C}_b| = 15, |\mathcal{C}_n| = 5$); during evaluation, three different base-novel splits are used that shuffle the categories used for base training and novel fine tuning; and for each split, various seeds are evaluated

Table 2. Overall and Base AP for all splits of VOC. Best results are **boldface**

Metric	Methods	Split 1					Split 2					Split 3				
		1	2	3	5	10	1	2	3	5	10	1	2	3	5	10
Overall mAP	MFDC [42]	46.15 ±0.78	47.89 ±0.74	47.50 ±0.57	49.16 ±0.41	49.27 ±0.39	43.99 ± ±0.48	45.20 ±0.46	45.23 ±0.25	**46.72 ±0.29**	47.17 ±0.41	45.54 ±1.21	**47.71 ±0.72**	47.63 ±0.66	49.23 ±0.34	49.75 ±0.34
	Ours	**46.23 ±0.84**	**47.90 ±0.80**	**47.50 ±0.62**	**49.23 ±0.45**	**49.30 ±0.3**	**44.06 ±0.62**	**45.34 ±0.53**	**45.47 ±0.31**	46.54 ±0.32	**47.24 ±0.53**	**45.81 ±1.23**	47.70 ±0.75	**47.86 ±0.62**	**49.37 ±0.26**	**49.82 ±0.22**
Base mAP	MFDC [42]	51.27 ±0.57	51.17 ±0.31	50.70 ±0.52	51.67 ±0.49	**51.68 ±0.59**	52.37 ±0.31	51.96 ±0.23	51.72 ±0.26	**52.58 ±0.31**	52.71 ±0.33	52.07 ±0.59	52.34 ±0.53	52.41 ±0.75	53.31 ±0.24	53.53 ±0.36
	Ours	**51.55 ±0.54**	**51.22 ±0.56**	**50.91 ±0.81**	**51.68 ±0.55**	51.6 ±0.42	**52.46 ±0.37**	**52.21 ±0.19**	**51.88 ±0.29**	52.37 ±0.31	**52.64 ±0.52**	**52.39 ±0.62**	**52.37 ±0.75**	**52.66 ±0.50**	**53.43 ±0.20**	**53.54 ±0.23**

denoting the subset of K instances used for novel fine tuning. The MS-COCO dataset is divided into sixty base classes and twenty novel classes, but unlike PASCAL-VOC, standard evaluation protocols do not shuffle the categories for different splits. Similar to PASCAL-VOC, however, the ten seeds – identifying which K instances out of a possible M instances are used for training – are included to average results.

We evaluate on the standard mean Average Precision (mAP) metric, which integrates the precision and recall of the detector into a single metric; for the PASCAL-VOC dataset, we report the novel AP50 over all three splits and for shots {1,2,3,5,10} as in most previous works, as well as the overall mAP and base class mAP. For the MS-COCO dataset, we report the overall AP, base AP (bAP), and novel AP(nAP). In this paper, "AP" refers to the mAP on the base and novel classes tested together, "bAP" refers to the mAP on the base classes only, and "nAP" refers to the mAP reported on novel classes only. Following TFA, we believe it is critical to report performance averaged over multiple seeds; therefore, we evaluate on the first five-seeds of each dataset and report those results in addition to the standard metrics on seed 0 of each dataset.

4.2 Implementation Details

We implement our algorithm in the Detectron2 library [43]. We follow the same base training as MFDC and DeFRCN, with gradient decoupling between the RoI heads and the RPN; note that we do not implement our contributions during base training.

During novel fine-tuning, we implement the supervised contrastive loss in a MoCo framework. The key and query encoders are of identical architecture with a linear layer followed by a ReLU non-linearity followed by another linear layer. The output features of the key and query encoders are 128-dimensional, and it is in this feature space that the Supervised Contrastive Loss is implemented. The momentum update parameter is set to 0.999. To compute the final supervised contrastive loss, we average the loss for each foreground feature, which has its own set of generated positive examples and negative examples. For each positive novel foreground example, we generate an additional 200 positive features and select 1600 additional samples as negatives from the larger memory bank for contrastive learning. We set the Contrastive Loss temperature at 0.07. As in MFDC, we use a batch size of 16 where 8 images come from the novel set and 8 come from the base set. During training, we start with a learning rate of 5×10^{-3}, and after training for 80% of the maximum number of iterations, we decrease the learning rate by a factor of 0.1. The optimizer's momentum is set to 0.9.

4.3 Results

Table 3. Evaluation metrics for all {5,10,30} shots of COCO. Best performing methods are in red, while second best are in blue

Methods	5-shot			10-shot			30-shot		
	AP	bAP	nAP	AP	bAP	nAP	AP	bAP	nAP
TFA w/cos (ICML '20) [39]	28.1	34.7	8.3	28.7	35.0	10.0	30.3	35.8	13.7
MPSR (ECCV 2020) [41]	-	-	-	15.3	17.1	9.7	17.1	18.1	14.1
FSCE (CVPR '21) [38]	-	-	-	-	-	11.9	-	-	16.4
DeFRCN (ICCV '21) [34]	28.7	33.1	15.3	30.6	34.6	18.6	31.6	34.7	22.5
FADI (NeurIPS '21) [1]	-	-	10.1	-	-	12.2	-	-	16.1
FCT (CVPR '22) [16]	-	-	14.0	-	-	17.1	-	-	21.4
CFA w/fc (CVPRW '22) [11]	30.1	37.1	9.0	30.8	37.6	10.5	31.9	37.7	14.7
CFA w/cos (CVPRW '22) [11]	29.7	36.3	9.8	30.3	36.6	11.3	31.7	37.0	15.6
CFA-DeFRCN (CVPRW '22) [11]	30.1	35. 0	15.6	31.4	35.5	19.1	32.0	35.0	23.0
LVC (CVPR '22) [22]	-	-	-	-	31.9	17.8	-	33.0	24.5
MFDC (ECCV '22) [42]	31.51	36.38	16.4	31.88	35.98	19.4	32.75	36.12	22.7
VFA (AAAI '23)	-	-	16.2	-	-	-	-	-	18.9
Norm-VAE (CVPR'23)	-	-	15.9	-	-	18.7	-	-	22.5
Ours	31.85	36.79	17.03	32.36	36.59	19.68	32.73	36.10	22.7
MFDC [42] (ECCV '22)	31.91 ±0.18	36.82 ±0.20	17.17 ±0.26	32.51 ±0.30	36.86 ±0.39	19.47 ±0.46	33.35 ±0.26	36.89 ±0.34	22.74 ±0.3
Ours	**32.09 ±0.21**	**37.05 ±0.24**	**17.20 ±0.27**	**32.67 ±0.32**	**37.00 ±0.36**	**19.67 ±0.44**	**33.40 ±0.30**	**36.92 ±0.37**	**22.85 ±0.37**

We report the nAP50 on the three splits of the PASCAL VOC dataset, as shown in Table 1. As shown in Table 1, we improve on previous state-of-the-art methods on nearly all shots and all splits of the data for seed 0. We specifically note that we outperform previous methods in the extremely low 1-shot scenario, where we increase performance by +2.1 points on split 1, and +0.56 points on split 2; we believe this occurs because our additional samples serve as pivots to arrange feature space when adequate labeled samples are not present. We note that we have either the best or second best results for all shots on split 3, and four out of five shots on split 1 and split 2. While Table 1 reports the seed 0 performance as in all previous literature, we believe that it is helpful to report the averaged results over many seeds; we show the nAP50 averaged over seeds zero to four in Table 1. In all scenarios for split 1, we see an improvement in the novel AP50, as well as a reduction in the error. This shows that our method organizes features space more appropriately for both detection and localization.

In addition to the nAP50 scores reported on three splits of the VOC data, we report the overall and base AP scores in Table 2. While most papers only report the novel AP50, we show that our method not only improves novel class performance but also improves the overall and base class performance of our detector. On all five shots of split 1, we increase or meet the performance of MFDC, and for splits 2 and 3 we improve the overall AP in four our of five cases. We see similar gains for the base AP, showing the effectiveness of our method.

We also report our results on the more challenging MS-COCO dataset. We compare the 5-shot, 10-shot, and 30-shot performance in Table 3, as enumerated in previous literature. When evaluating on seed 0, as previous literature reports, we note increases in the overall AP, base AP, and novel AP in both the 5-shot and 10-shot scenario. As in Table 1, we also report the average performance over

five seeds of the data. In all scenarios, we get improvements over the baseline. We note an interesting phenomena; while we only use novel-class foreground region proposals and their extracted features for supervised contrastive learning, improvements are seen in the overall AP *and* base AP. While it is expected to see improvements in overall AP, we posit that improvements in base AP are due directly to the sample mixup; when a novel feature is mixed with a base feature, we generate a hard example for both base and novel classes, and expect our network to learn a boundary that is more robust. This has the downstream effect of improving the decision boundaries on the base class.

4.4 Ablation Studies

Table 4. Number of additional (synthesized) positive samples for contrastive learning in the COCO 5-shot scenario. We note that increasing the number of positives past a certain threshold hampers base and overall performance while improving novel-class performance.

Number of samples	5-shot		
	AP	bAP	nAP
10	31.78	36.77	16.79
50	**31.94**	**36.94**	16.95
100	31.84	36.82	16.92
200	31.85	36.79	**17.03**

In addition to benchmark performance increases, we conduct ablation studies to understand the benefit of mixing samples in contrastive learning. First, we experiment with the number of *positive* mixed-up samples to add for contrastive learning (while keeping negatives constant), and report the results in Table 4. We notice dual and competing phenomena; as we increase the number of positive samples from ten to fifty, the overall AP, base AP, and novel AP increase. However, as we increase from fifty samples to one hundred samples, all three metrics decrease. When we increase the number of samples to two hundred additional positives, the overall AP and base AP decrease compared to adding fifty additional samples; however, the novel AP is at a maximum. Intuitively, we note that increasing the number of positive examples essentially increases the data on novel classes to improve learning; however, these positives are not pure positives, which can confuse the network when learning decision boundaries. This implies that there is a mixup-positive tradeoff; while increasing the number of positives improves learning, there is a limit at which overall, base, and novel performance continue to increase. For real-world systems, it is important to carefully tune the number of positives to be included in contrastive learning.

Secondly, we test the strength of our contribution by comparing the inclusion, or lack thereof, of data generation for contrastive learning in Table 5. The

Table 5. Supervised contrastive learning versus data generation+ contrastive learning, COCO dataset seed 0.

Shots	Generation?	AP	bAP	nAP
5-shot	✗	31.75	36.67	16.98
	✓	**31.85**	**36.79**	**17.03**
10-Shot	✗	32.05	36.15	**19.77**
	✓	**32.36**	**36.59**	19.68
30-Shot	✗	**32.84**	**36.16**	**22.88**
	✓	32.73	36.10	22.7

Table 6. Effects of synthesizing 200 new samples on the COCO dataset, seed 0

Synthesized Type	5-shot		
	AP	bAP	nAP
Positives	**31.85**	**36.79**	**17.03**
Negatives	30.70	36.66	16.84
Pos.+ Neg	31.79	36.75	16.91

Table 7. "Vanilla" Mixup vs Our Contribution for {5,10,30} shots of COCO seed 0. Best performing methods are **boldface**

Methods	5-shot			10-shot			30-shot		
	AP	bAP	nAP	AP	bAP	nAP	AP	bAP	nAP
Vanilla Mixup	31.73	36.63	17.00	32.01	36.20	19.44	32.72	36.04	**22.74**
Ours	**31.85**	**36.79**	**17.03**	**32.36**	**36.59**	**19.68**	**32.73**	**36.10**	22.7

experimental settings mimic those used for Table 3, except that including "generation" generates an extra two hundred positive mixed samples, while "without generation" denotes using only positives (i.e. same class as foreground sample) from the memory bank for learning – traditional supervised contrastive learning. Table 5 shows that, in the 5 and 10-shot scenarios, generation is important to improving performance on all three metrics compared to standard contrastive learning. In the 30-shot scenario we see a conflicting effect: while data generation still improves baseline results from Table 3, the extent is not as great when using standard contrastive learning. The 30-shot scenario benefits from more "pure" labeled data (i.e. no mixing) than the 5 and 10-shot case, which are better positives for contrastive learning than the mixed samples. Overall, a clear trend emerges identifying the benefits of sample mixup for few-shot detection – especially in low-shot scenarios.

Next, we test whether generating mixed positives *and* negatives improves learning and metrics, and report the results in Table 6. In this experiment, "positives" refer to generating two hundred additional positives from mixed samples of the semantically similar class without including mixed negatives, while "negatives" refers to mixing the two hundred hardest negatives (independent of class) without including mixed positives. The entry "positives + negatives" refers to using both positives and negatives. For all experiments, we use 1600 true negatives in contrastive learning. We notice that when only positives samples are added for sample mixup, we achieve the best performance overall. We posit that, in the few-shot scenario, learning decision boundaries on positive samples alleviates the dearth of labeled data; the plentiful negatives from the memory bank,

which still provide adequate information and improve results, do not compensate for the lack of labeled examples. Based on the results in Table 6, the majority of observed performance increase is the result of the positives; when we add negatives to the positives, the overall AP and base AP remain steady, while the novel AP decreases. This shows that the positives have a stronger effect on learning decision boundaries.

Additionally, we evaluate our use of mixup by comparing our contribution to "vanilla" mixup [50] in Table 7. Vanilla Mixup, instead of mixing samples from the most similar class as in our contribution, only mixes samples from the same class; all other parameters, including the number of samples generated, the supervised contrastive loss, etc. remain the same. As shown in Table 7, the proposed method improves over vanilla mixup in nearly all metrics. Our intuition suggests that vanilla mixup, which only mixes intra-class samples, is not very effective in the few-shot scenario since the variation among samples is not great. Vanilla mixup only addresses the data scarcity issue but not the data diversity issue, which could lead to overfitting.

Table 8. Using most-similar vs least similar classes for feature generation, COCO dataset seed 0

Shots	Selection?	AP	bAP	nAP
5-shot	MFDC [41]	31.51	36.38	16.4
	Least Similar	31.67	35.78	16.90
	Most Similar	**31.85**	**36.79**	**17.03**
10-Shot	MFDC [41]	31.88	35.98	19.4
	Least Similar	31.99	36.13	19.62
	Most Similar	**32.36**	**36.59**	**19.68**
30-Shot	MFDC [41]	32.75	36.12	22.7
	Least Similar	32.61	36.01	22.67
	Most Similar	**32.73**	**36.10**	**22.7**

Finally, we answer the question: what happens if there are no similar classes to novel classes for feature generation? We simulate this experiment by, instead of selecting the most similar class for feature generation, *we select the most dissimilar class* for data generation. We report the results of our experiment in Table 8. While we do see minor performance degradations which are expected when using the *most dissimilar class*, our model still maintains very good performance. Surprisingly, we still improve performance as compared to MFDC in the 5-shot and 10-shot scenarios (except for 5-shot base AP, in which we see a minor performance decrease) even when using the least similar class. The reason this occurs is that, in addition to hard negative mining, any mixup for feature generation that occurs creates a noisy version of the novel feature; this addresses the data diversity issue of few-shot object detection.

4.5 Qualitative Results

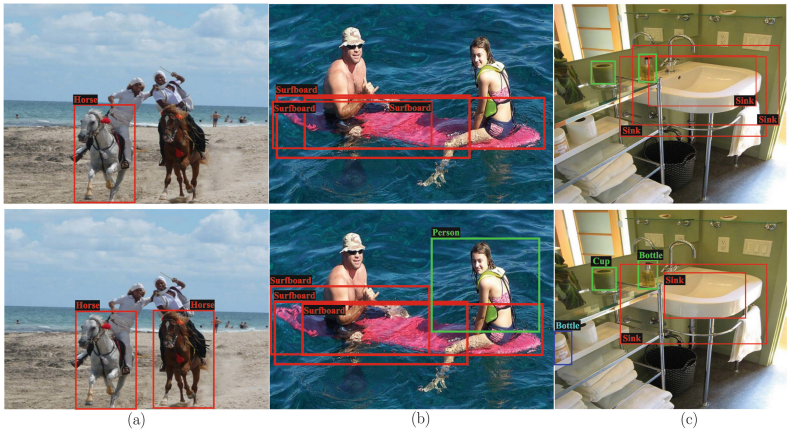


Fig. 3. Qualitative examples on the MS-COCO. The top row is MFDC while the bottom row is our method. In (a), we are able to recognize one of the horses which MFDC does not. In (b), we recognize one "person" (a novel class) where MFDC does not. Panel (c) shows that our method reduces the number of false-positive "sinks" in the image, while also recognizing one of the bottles in the left corner that MFDC does not.

The effects of the proposed contrastive learning method can be seen directly on the data itself, as shown in Fig. 3. The top row shows results of our baseline, the MFDC [42] model, while the bottom row shows results after training our model. In Fig. 3(a), MFDC clearly misses the detection of one of the horses, while our method is able to correctly detect the horse. In Fig. 3(b), our method is able to recognize one "person" (a novel class) when MFDC is not able to do so. In both examples, however, we still see limitations of state-of-the-art few-shot detection: novel classes such as "person" are still difficult to recognize. Nevertheless, our method improves results in a simple and effective way, without complicated data generation procedures. Figure 3(c) clearly shows this; our method recognizes all the objects that MFDC does, as well as recognizing the small "bottle" in the lower left corner.

We can qualitatively understand feature space alignment through t-SNE [30] analysis of feature space, as shown in Fig. 4, in which blue is "person", red is "bicycle", green is "car", and black is "motorcycle". Figure 4(a) shows the features for each class under the 5-shot scenario at the end of training for MFDC [42], while (b) shows feature space after training with supervised contrastive learning and feature generation. Based on the t-SNE visualization, we can clearly see our learned decision boundaries more naturally separate the data between classes, as compared to the small and tight clusters that are present in MFDC's learning paradigm. We also see the semantic nature of our contrastive learning

framework; the "car" class is semantically similar to the "motorcycle" class, and our contrastive framework imposes a feature space alignment that brings these classes closer together. Before training, car (green) and motorcycle (black) were interspersed in feature space among the other classes; our method arranges feature space to bring these classes closer together, as well as clustering these classes effectively. This clustering is clearly visible as the five distinct clusters of the "person" class is collapsed into one cluster.

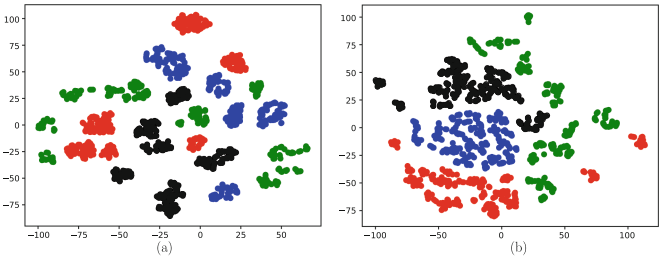


Fig. 4. t-SNE visualization of features for the COCO 5-shot seed 0 model, where blue = "person", red = "bicycle", green = "car", and **black = "motorcycle"**. Subfigure (a) shows' feature space trained like MFDC [42], and subfigure (b) shows features after training with our contrastive loss + generation. Our method learns that "car" (green) is most semantically similar to "motorcycle" (black), and arranges feature space for these classes are closer together. (Color figure online)

5 Conclusion

In this paper, we present a novel use of contrastive learning with mixed positives for few-shot object detection. Not only should features from a class be close to other features within that class, but features from a similar class should "close" in features space to other similar classes. We therefore generate features that mix between the class of interest and its most similar class, and treat them as positives for supervised contrastive learning. With our algorithm, we alleviate many issues in few-shot object detection: 1) we alleviate *data scarcity* by generating synthesized positive samples; 2) we limit *overfitting* by generating diverse samples; and 3) we organize *feature space* more effectively through contrastive learning with synthesized samples. With limited extra parameters during training and none during inference, our algorithm improves few-shot object detection on standard benchmarks.

References

1. Cao, Y., et al.: Few-shot object detection via association and discrimination. In: NeurIPS, vol. 34, pp. 16570–16581 (2021)
2. Chen, D.J., Hsieh, H.Y., Liu, T.L.: Adaptive image transformer for one-shot object detection. In: CVPR 2021. pp. 12242–12251 (2021). https://doi.org/10.1109/CVPR46437.2021.01207
3. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: ICML, Proceedings of Machine Learning Research, vol. 119, pp. 1597–1607. PMLR (2020). https://proceedings.mlr.press/v119/chen20j.html
4. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: 2005 CVPR'05, vol. 1, pp. 539–546 (2005). https://doi.org/10.1109/CVPR.2005.202
5. Chuang, C.Y., Robinson, J., Lin, Y.C., Torralba, A., Jegelka, S.: Debiased contrastive learning. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) NeurIPS 2020, vol. 33, pp. 8765–8775 (2020)
6. Demirel, B., Baran, O.B., Cinbis, R.G.: Meta-tuning loss functions and data augmentation for few-shot object detection. In: CVPR 2023, pp. 7339–7349 (2023)
7. Everingham, M., Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. Int. J. Comput. Vis. **88**(2), 303–338 (2010). https://doi.org/10.1007/s11263-009-0275-4
8. Fan, Q., Zhuo, W., Tang, C., Tai, Y.: Few-shot object detection with attention-RPN and multi-relation detector. In: CVPR 2020, pp. 4012–4021 (2020). https://doi.org/10.1109/CVPR42600.2020.00407
9. Fan, Z., Ma, Y., Li, Z., Sun, J.: Generalized few-shot object detection without forgetting. In: CVPR 2021, pp. 4527–4536 (2021)
10. Girshick, R.: Fast R-CNN. In: Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), pp. 1440–1448. ICCV '15, USA (2015). https://doi.org/10.1109/ICCV.2015.169
11. Guirguis, K., Hendawy, A., Eskandar, G., Abdelsamad, M., Kayser, M., Beyerer, J.: CFA: constraint-based finetuning approach for generalized few-shot object detection. In: CVPR Workshops 2022, pp. 4038–4048 (2022). https://doi.org/10.1109/CVPRW56347.2022.00449
12. Guirguis, K., Meier, J., Eskandar, G., Kayser, M., Yang, B., Beyerer, J.: NIFF: alleviating forgetting in generalized few-shot object detection via neural instance feature forging (2023)
13. Gutmann, M., Hyvärinen, A.: Noise-contrastive estimation: a new estimation principle for unnormalized statistical models. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, vol. 9, pp. 297–304. PMLR (2010)
14. Han, G., He, Y., Huang, S., Ma, J., Chang, S.F.: Query adaptive few-shot object detection with heterogeneous graph convolutional networks. In: ICCV 2021, pp. 3263–3272 (2021)
15. Han, G., Huang, S., Ma, J., He, Y., Chang, S.F.: Meta faster R-CNN: towards accurate few-shot object detection with attentive feature alignment. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36(1), pp. 780–789 (2022). https://doi.org/10.1609/aaai.v36i1.19959
16. Han, G., Ma, J., Huang, S., Chen, L., Chang, S.F.: Few-shot object detection with fully cross-transformer. In: CVPR 2022, pp. 5321–5330 (2022)

17. Han, J., Ren, Y., Ding, J., Yan, K., Xia, G.S.: Few-shot object detection via variational feature aggregation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, pp. 755–763 (2023)
18. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: 2020 (CVPR), pp. 9726–9735 (2020). https://doi.org/10.1109/CVPR42600.2020.00975
19. Hjelm, R.D., et al.: Learning deep representations by mutual information estimation and maximization. In: ICLR (2019). https://openreview.net/forum?id=Bklr3j0cKX
20. Hsieh, T.I., Lo, Y.C., Chen, H.T., Liu, T.L.: One-shot object detection with co-attention and co-excitation. In: NeurIPS 2019, vol. 32 (2019)
21. Kalantidis, Y., Sariyildiz, M.B., Pion, N., Weinzaepfel, P., Larlus, D.: Hard negative mixing for contrastive learning. In: NeurIPS (2020)
22. Kaul, P., Xie, W., Zisserman, A.: Label, verify, correct: a simple few-shot object detection method. In: IEEE CVPR (2022)
23. Khosla, P., et al.: Supervised contrastive learning. In: NeurIPS 2020. NIPS'20, Red Hook, NY, USA (2020)
24. Kim, S., Lee, G., Bae, S., Yun, S.Y.: MixCo: mix-up contrastive learning for visual representation. arXiv preprint arXiv:2010.06300 (2020)
25. Köhler, M., Eisenbach, M., Gross, H.M.: Few-shot object detection: a comprehensive survey (2022)
26. Li, B., Yang, B., Liu, C., Liu, F., Ji, R., Ye, Q.: Beyond max-margin: class margin equilibrium for few-shot object detection. In: CVPR 2021, pp. 7359–7368 (2021). https://doi.org/10.1109/CVPR46437.2021.00728
27. Li, X., Zhang, L., Chen, Y.P., Tai, Y.W., Tang, C.K.: One-shot object detection without fine-tuning (2020)
28. Lin, S., Wang, K., Zeng, X., Zhao, R.: An effective crop-paste pipeline for few-shot object detection. In: CVPR Workshops 2023, pp. 4819–4827 (2023)
29. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) Computer Vision – ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V, pp. 740–755. Springer International Publishing, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
30. van der Maaten, L., Hinton, G.: Visualizing data using T-SNE. J. Mach. Learn. Res. **9**(86), 2579–2605 (2008). http://jmlr.org/papers/v9/vandermaaten08a.html
31. Mnih, A., Kavukcuoglu, K.: Learning word embeddings efficiently with noise-contrastive estimation. In: Burges, C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K. (eds.) NeurIPS 2013, vol. 26 (2013)
32. van den Oord, A., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding (2019)
33. Perez-Rua, J.M., Zhu, X., Hospedales, T.M., Xiang, T.: Incremental few-shot object detection. In: CVPR 2020 (2020)
34. Qiao, L., Zhao, Y., Li, Z., Qiu, X., Wu, J., Zhang, C.: DeFRCN: decoupled faster R-CNN for few-shot object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 8681–8690 (2021)
35. Roy, A., Shah, A., Shah, K., Dhar, P., Cherian, A., Chellappa, R.: FeLMi : few shot learning with hard mixup. In: NeurIPS 2022 (2022). https://openreview.net/forum?id=xpdaDM_B4D
36. Schroff, F., Kalenichenko, D., Philbin, J.: FaceNet: a unified embedding for face recognition and clustering. In: 2015 Conference on Computer Vision and Pattern

Recognition (CVPR), pp. 815–823 (2015). https://doi.org/10.1109/CVPR.2015.7298682
37. Sohn, K.: Improved deep metric learning with multi-class n-pair loss objective. In: Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 29 (2016)
38. Sun, B., Li, B., Cai, S., Yuan, Y., Zhang, C.: FSCE: few-shot object detection via contrastive proposal encoding. In: CVPR 2021, pp. 7352–7362 (2021)
39. Wang, X., Huang, T.E., Darrell, T., Gonzalez, J.E., Yu, F.: Frustratingly simple few-shot object detection. In: ICML 2020, ICML'20, JMLR.org (2020)
40. Wu, A., Han, Y., Zhu, L., Yang, Y.: Universal-prototype enhancing for few-shot object detection. In: ICCV 2021, pp. 9547–9556 (2021). https://doi.org/10.1109/ICCV48922.2021.00943
41. Wu, J., Liu, S., Huang, D., Wang, Y.: Multi-scale positive sample refinement for few-shot object detection. In: Computer Vision – ECCV 2020: 16th European Conference, pp. 456–472 (2020). https://doi.org/10.1007/978-3-030-58517-4_27
42. Wu, S., Pei, W., Mei, D., Chen, F., Tian, J., Lu, G.: Multi-faceted distillation of base-novel commonality for few-shot object detection. In: Computer Vision – ECCV 2022: 17th European Conference, pp. 578–594 (2022). https://doi.org/10.1007/978-3-031-20077-9_34
43. Wu, Y., Kirillov, A., Massa, F., Lo, W.Y., Girshick, R.: Detectron2 (2019). https://github.com/facebookresearch/detectron2
44. Wu, Z., Xiong, Y., Yu, S.X., Lin, D.: Unsupervised feature learning via non-parametric instance discrimination. In: CVPR 2018 (2018)
45. Xiao, Y., Marlet, R.: Few-shot object detection and viewpoint estimation for objects in the wild. In: Computer Vision – ECCV 2020, pp. 192–210 (2020). https://doi.org/10.1007/978-3-030-58520-4_12
46. Xu, J., Le, H., Samaras, D.: Generating features with increased crop-related diversity for few-shot object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 19713–19722 (2023)
47. Yang, Z., Wang, Y., Chen, X., Liu, J., Qiao, Y.: Context-transformer: tackling object confusion for few-shot detection. In: AAAI Conference on Artificial Intelligence (2020)
48. Zbontar, J., et al.: Barlow twins: self-supervised learning via redundancy reduction. In: ICML 2021, Proceedings of Machine Learning Research, vol. 139, pp. 12310–12320. PMLR (2021). http://proceedings.mlr.press/v139/zbontar21a.html
49. Zhang, G., Luo, Z., Cui, K., Lu, S., Xing, E.P.: Meta-DETR: image-level few-shot detection with inter-class correlation exploitation. IEEE Trans. Pattern Anal. Mach. Intell. (2022). https://doi.org/10.1109/TPAMI.2022.3195735
50. Zhang, H., et al.: mixup: beyond empirical risk minimization. In: International Conference on Learning Representations (2018). https://openreview.net/forum?id=r1Ddp1-Rb
51. Zhang, W., Wang, Y.X.: Hallucination improves few-shot object detection. In: CVPR 2021, pp. 13003–13012 (2021). https://doi.org/10.1109/CVPR46437.2021.01281

Image Domain Translation for Few-Shot Learning

Liangyuan Chen, Zhenan He, and Hai Zhang(✉)

College of Computer Science, Sichuan University, Chengdu, China
zhanghi@stu.scu.edu.cn

Abstract. Few-shot learning is crucial in machine learning and computer vision. It enables models to recognize new objects with limited labeled data, addressing the challenge of data scarcity and expanding the application of machine learning to domains with scarce data. Previous methods built metric space using labeled data from the base set and then classified queried images from the novel set by finding the nearest class prototype. However, due to the presence of poor-quality data in the novel set, the class prototype often exhibits instability. In response to this challenge, this paper proposes a Semantic Conditional Translation Network for reconstructing stable class prototypes. Specifically, images are first divided into edge domain (i.e., images at the cluster edge) and prototype domain (i.e., images at the cluster center). Then, an Enhanced Generative Adversarial Network is introduced to learn the translation from edge toward prototype, where a Non-parametric Classification Regularizer is designed to enlarge the discriminability of the translated samples. Meanwhile, class definitions are exploited as semantics providing precise descriptions and enhancing translation performance. Experimental results demonstrate that the proposed method obtains competitive results on four benchmark datasets.

Keywords: Few-Shot Learning · Generative Adversarial Networks · Multimodal Learning

1 Introduction

Deep learning models have made dramatic improvements through exploiting a huge amount of annotated data [12]. However, in many real-world scenarios, there are not enough labeled data provided for models, which reduces the application range of traditional deep learning techniques. In contrast, humans possess a remarkable cognitive ability to learn new concepts and recognize categories from just a few examples. This exceptional capability enables humans to generalize acquired knowledge and make accurate predictions even when confronted with limited training instances [5]. Motivated by this, Few-Shot Learning (FSL) [51] is proposed to mimic the humans' ability to learn from a few labeled samples.

In the standard FSL setting, the models first rely on a well-annotated base set to establish a robust metric space. This set provides rich and representative

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 313–329, 2025.
https://doi.org/10.1007/978-3-031-78183-4_20

annotated samples to facilitate model training and prior knowledge acquisition. Subsequently, the model encounters a substantial challenge: identifying images in a novel set that do not overlap with the categories in the base set. Notably, for each category in the novel set, only a few labeled examples are available, which significantly increases the difficulty for the model to generalize to new instances. A natural and effective strategy involves mapping samples from the novel set into the metric space and utilizing the few labeled samples to compute class prototypes. Classification tasks are then accomplished by seeking the nearest neighbor class prototype for a given query sample. However, inferring class prototypes from a limited sample set tends to deviate from the genuine class prototypes owing to the presence of low-quality data within the limited samples, such as occlusion, incompleteness, or the inclusion of multiple targets, thereby resulting in a deterioration of classification performance. Consequently, accurately and efficiently capturing and representing category features for new classes in environments with poor-quality data emerge as pivotal concerns in FSL.

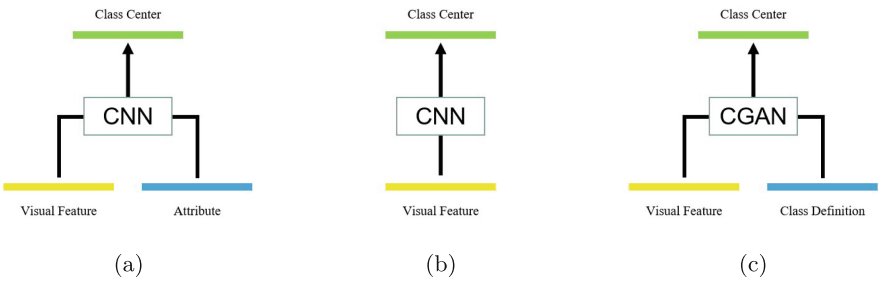


Fig. 1. Different paradigms for reconstructing stable class prototypes.

To tackle this issue, previous studies [57,59] have proposed diverse methods for reconstructing stable class prototypes, as depicted in Fig. 1. Specifically, RestoreNet [57] suggested learning the mapping from edge samples to the class center, as illustrated in Fig. 1b. Nevertheless, mastering the direct mapping from edges to the center without external prior knowledge poses a significant challenge. ProtoComNet [59], as shown in Fig. 1a, improved upon RestoreNet by designing complex nonlinear networks to integrate attribute labels. However, nonlinear networks may encounter difficulties in accurately capturing the relationships between distributions due to the high-dimensional attributes inherent in distributions. Moreover, attribute labels, being a limited data resource, are unavailable for the majority of datasets.

Based on this, we propose the Semantic Conditioned Translation Network (SCTN), aiming to leverage the powerful distribution learning capability of the Conditional Generative Adversarial Network (CGAN) to establish a distribution mapping from edge samples to the center, thereby reconstructing class prototypes. This learning paradigm is illustrated in Fig. 1c. Specifically, we first

design an image domain division mechanism to divide the training data into two image domains: the edge image domain and the prototype image domain. The edge image domain comprises samples that are distant from the class centers, with blurry class characteristics making them difficult to identify accurately, as depicted in Fig. 2. Conversely, the prototype image domain encompasses samples that are close to the class centers and exhibit distinctive class characteristics, thus considered instances with strong class representativeness. Furthermore, we utilize CGAN to establish and optimize the mapping between these two image domains for reconstructing class prototypes. In this process, the conditional information is derived from class definitions extracted from WordNet based on class labels. Compared to simple class names, class definitions entail richer class features and are easily accessible. Additionally, to enhance the discrimination of the reconstructed class prototypes, we introduce a Non-parametric Classification Regularizer (NCR). The entire system integrates these components, forming a learning framework named Feature Enhanced Generative Adversarial Network (FEGAN), the structure of which is shown in Fig. 2. The experimental results demonstrate that SCTN achieves competitive results across four diverse FSL datasets, confirming the effectiveness of this approach in reconstructing class prototypes.

The contribution can be summarized in the following three points:

- The proposed image domain division mechanism separates strong and weak class representations, where the information inside the weak class representation is further explored to help construct robust prototypes.
- The designed Semantic Conditioned Translation Network enhances the representation of edge images by capturing the distribution of class prototypes, thereby improving the model's recognition performance on poor-quality data.
- A Non-parametric Classification Regularizer is introduced to constrain the category of translated samples, thereby enlarging their discriminability and improving the overall classification performance.

2 Related Work

In recent years, FSL has made significant developments in three main directions: metric-based, optimization-based, and semantic-based.

Metric-based methods aim to learn a metric space in which inter-class distances are maximized and intra-class distances are minimized. In the early stage, researchers devised various kinds of distance metrics to achieve precise estimation of the distances between samples, including the L1 distance [30], the Euclidean distance [51], the cosine distance [53], and the Mahalanobis distance [3]. After that, some improvements were proposed. For example, in data augmentation [49] approaches, hallucinated samples can be generated by deformation [9] and synthesis [2,20,35,48,54,60] to overcome the difficulty of labeled data lacking. The representation learning [4,25] approaches suggested to add a supervised [6,52,58]

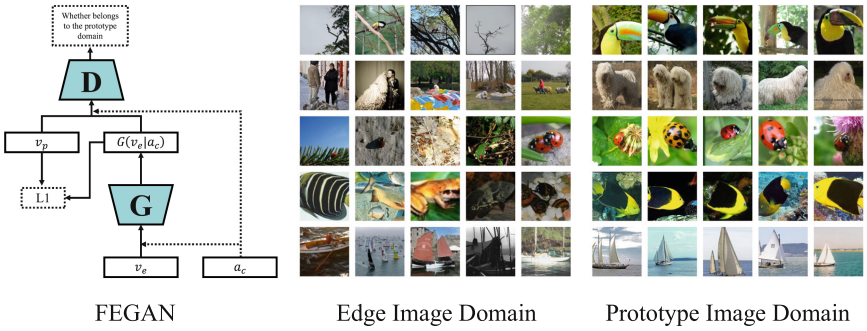


Fig. 2. FEGAN with two image domains. The left figure is the structure of the FEGAN, where the solid rectangular indicates the feature embedding, and the dashed rectangular represents the objective function. v_e and v_p denote the feature embedding sampled from the edge and prototype image domains, respectively, i.e., $v_e \sim D_{edge}$ and $v_p \sim D_{proto}$. Here, v_e and v_p are from the same class and a_c denotes their class semantic information. The middle and right figures contain images sampled from the edge and prototype image domains, respectively.

or self-supervised [14,19,39] training stage to pre-train a robust feature extraction network.

Optimization-based methods [1,44] aim at enabling the model to adapt rapidly to a novel task using only a few labeled data through several steps of learning. For example, MAML [18] proposed to learn a set of initialization model parameters and then fine-tune them to novel tasks. However, updating the entire model parameters with a few labeled data can lead to meta-overfitting. CAVIA [61] and LEO [47] reduced the number of updated parameters at test time. Specifically, CAVIA partitioned the model parameters into context parameters and shared parameters, while LEO employed low-dimensional latent space optimization. On the other hand, TAML [27] suggested to learn an unbiased initial model to avoid this problem. MetaNAS [17] further improved the performance of MAML by integrating Neural Architecture Search (NAS) [16] into MAML.

Semantic-based methods intended to combine vision and language information to improve performance in visual recognition tasks. For instance, in [59], attribute knowledge and unlabelled data are utilized to complete a representative prototype. In [55,56], it is suggested to use the class name to get a text prototype and then mix a visual prototype to enhance the representativeness of the prototype. In [33], a task-relevant adaptive margin loss based on the semantic similarity between class names in the task is proposed to increase the discriminative power of the embedding space. In [10], multiple types of semantic spaces are used to synthesize instance features to achieve data augmentation.

3 Methodology

This section specifies how the prototype instability issue is mitigated by the proposed method through image domain translation. The definition of FSL and the required background knowledge are presented first, followed by the description of the proposed SCTN.

3.1 Background

First, let's provide the problem definition for the classification task in few-shot learning. The dataset includes two components: base set $\mathcal{D}_{base} = \{(x, y)|x \in \mathcal{X}_{base}, y \in \mathcal{C}_{base}\}$ for training and novel set $\mathcal{D}_{novel} = \{(x, y)|x \in \mathcal{X}_{novel}, y \in \mathcal{C}_{novel}\}$ for testing, where x denotes the image and y represents the label. Furthermore, $\mathcal{X}_{base}$ and $\mathcal{X}_{novel}$ refer to the sets of images within the base and novel components, respectively. It is important to note that the label space for both sets is disjoint, that is, $\mathcal{C}_{base} \cap \mathcal{C}_{novel} = \emptyset$. During the testing process, the support set $\mathcal{S} = \{(x_i, y_i)\}_{i=0}^{N \times K}$ is randomly sampled from $\mathcal{D}_{novel}$, which contains N classes with each class consisting of K samples. Then, the model is required to correctly classify images in the query set $\mathcal{Q} = \{(x_i, y_i)\}_{i=0}^{N \times M}$ into one of N classes in the support set $\mathcal{S}$, where M is the number of query samples in Q. Generally, this type of classification task is named an N-way K-shot task.

Prototypical Network is a simple but effective FSL method. It first calculates the prototype of each class. For instance, the prototype of class c, labeled as r_c, is the mean value of all samples in the support set belonging to class c, which is calculated in (1).

$$r_c = \frac{1}{|\mathcal{S}_c|} \sum_{x_i \in \mathcal{S}_c} f(x_i), \tag{1}$$

where $\mathcal{S}_c$ denotes the set of all samples belonging to class c in the support set $\mathcal{S}$ and f denotes the backbone network. Labels are then assigned based on the distance from query samples to each class prototype. Specifically, the probability of a query sample q belonging to class c is calculated by (2).

$$P(y = c) = \frac{\exp d(q, r_c)}{\sum_{j=1}^{N} \exp d(q, r_j)}, \tag{2}$$

where $d(\cdot, \cdot)$ denotes the negative Euclidean distance, i.e., $d(i, j) = -||i - j||_2$, and N denotes the number of classes.

Conditional Generative Adversarial Network (CGAN) is an impressive generative model composed of a generator G and a discriminator D [41]. The generator generates images follow class-specific distribution from a random Gaussian noise z and class label y, while the discriminator aims to separate the generated image from the real one. Suppose x denotes the input image, p_{data} stands for the input data distribution, and p_z represents the noise distribution.

The objective function of CGAN [41] is:

$$\begin{aligned}\max_D \min_G V(D,G) =& \mathbb{E}_{x\sim p_{data}(x)}[\log D(x|y)]+ \\ & \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z|y)))],\end{aligned} \tag{3}$$

where y contains auxiliary information, i.e., class labels.

3.2 Overall Framework of SCTN

For the sake of convenience and ease of understanding in subsequent discourse, it is assumed that image translation is implemented on class c. The samples belong to class c in the base set D_{base} are divided into two image domains, namely the edge image domain $D_{edge} = \{(v_e, c)\}$ and the prototype image domain $D_{proto} = \{(v_p, c)\}$. Here, v represents feature embedding extracted from image x by backbone network f, i.e., $v = f(x)$, v_e and v_p denote the embedding of the edge image domain and prototype image domain, respectively.

Image Domain Division. The image domain division operates on a per-class basis, wherein each class c undergoes independent processing within the base set D_{base}. Firstly, a subset D_c is extracted from D_{base} containing all samples labeled as c, followed by the computation of the class center t_c. This computation can be performed using either the feature mean method or the k-means clustering algorithm [21]. Subsequently, employing metrics such as cosine or Euclidean distance, we measure the distances between each sample in D_c and its corresponding class center t_c to quantify the representational strength of each sample. Based on this analysis, the top W closest samples, exhibiting stronger class representation, are assigned to the prototype image domain $\mathcal{D}proto$, while samples distant from the class center with weaker representation are allocated to the edge image domain $\mathcal{D}edge$. Here, the parameter W is a manually set hyper-parameter, typically configured to one-tenth the size of D_c. By implementing the image domain division strategy, distinct groups of samples with varying degrees of class representation can be effectively identified and distinguished.

Feature Enhanced Generative Adversarial Network. To learn the translation from weak class representative samples to strong class representative samples, Feature Enhanced Generative Adversarial Network (FEGAN) is designed in SCTN to construct the translation from the edge distribution p_{edge} to the prototype distribution p_{proto}. Specifically, FEGAN comprises two components: the Conditional Generative Adversarial Network (CGAN) and the Non-parametric Classification Regularizer (NCR).

The CGAN comprises a generator G and a discriminator D. The generator G aims to enhance the feature representation of edge samples to emulate the feature distribution of the prototype image domain, while the discriminator D is responsible for accurately distinguishing whether input features originate from the edge

distribution p_{edge} or the prototype distribution p_{proto}, thereby driving the generator to continuously optimize its performance. An illustration of FEGAN can be found in Fig. 2. Initially, image samples for class c are extracted separately from p_{edge} and p_{proto}, and these images are input to a pre-trained feature extractor f to generate corresponding feature embeddings v_e and v_p. Subsequently, the discriminator D is tasked with distinguishing whether input features originate from p_{edge} or p_{proto}. On the other hand, the generator G endeavors to generate prototype features for class c from the edge distribution p_{edge} conditioned on the semantic embedding a_c, attempting to deceive the discriminator D. Here, a_c is derived from the class definition extracted from WordNet. It is noteworthy that the input to the generator G is feature embedding v rather than Gaussian noise z, distinguishing this method from existing data augmentation strategies [35,48,56,60]. Through this approach, FEGAN not only leverages visual information but also integrates higher-level semantic knowledge.

Based on the preceding discussion, the objective function of CGAN in FEGAN is:

$$\begin{aligned}\mathcal{L}_{CGAN}(D,G) =& \mathbb{E}_{v_p \sim p_{proto}(v_p)}[\log D(v_p|a_c)] + \\ & \mathbb{E}_{v_e \sim p_{edge}(v_e)}[\log(1 - D(G(v_e|a_c)))],\end{aligned} \tag{4}$$

where $D(G(v_e|a_c))$ and $D(v_p|a_c)$ represent the probabilities of the input v_e and v_p belonging to p_{proto} given the semantic embedding a_c, respectively.

The optimization process encompasses adversarial training between the discriminator D and the generator G. Here, the discriminator aims to maximize the loss value of the objective function $\mathcal{L}_{CGAN}(D,G)$, while the generator endeavors to minimize this value. The training of the discriminator and generator occurs in alternating steps. The global optimum is reached when the generator effectively establishes a mapping from the edge image domain to the prototype image domain. At this stage, the discriminator assigns probabilities of 0.5 to the prototype images and the edge images translated by the generator. This signifies the achievement of a Nash equilibrium state, where real samples and translated samples cannot be effectively distinguished by the discriminator.

Although CGAN, by introducing the class label a_c as a conditional variable, aids in constraining the category of translated samples, the inherent similarity between semantics may pose a risk to the discriminability of translated samples. Thus, FEGAN integrates a Non-parametric Classification Regularizer (NCR), which explicitly establishes alignment relationships between samples of the same category in both the prototype image domain and the edge image domain. This mechanism ensures that translated samples accurately capture the distinctive properties of their respective categories. The objective function of NCR is formulated as follows:

$$\mathcal{L}_{NCR}(G) = \mathbb{E}[\|G(v_e|a_c) - v_p\|_1], \tag{5}$$

where v_e and v_p are samples from the edge and prototype image distributions, respectively, i.e., $v_e \sim p_{edge}$ and $v_p \sim p_{proto}$. Hence, the final objective of FEGAN

is shown in (6), where β is a manually controlled hyper-parameter.

$$\max_{D} \min_{G} V(D,G) = \mathcal{L}_{CGAN}(D,G) + \beta \mathcal{L}_{NCR}(G). \tag{6}$$

Prototype Fusing. During the testing phase, for each class c in the support set $\mathcal{S}$, two types of class prototypes are constructed: the mean prototype r_c^m and the translated prototype r_c^t. The mean prototype is derived by computing the average of all sample features belonging to class c (Formula 7). The translated prototype is generated by combining the mean prototype r_c^m with the corresponding semantic information a_c, which is then input into the pre-trained generator G to generate a new representation (Formula 8).

$$r_c^m = \frac{1}{|S_c|} \sum_{x_i \in S_c} f(x_i), \tag{7}$$

where $\mathcal{S}_c$ denotes the set of all samples belonging to class c in the support set $\mathcal{S}$.

$$r_c^t = G(r_c^m | a_c). \tag{8}$$

$$r_c = k_1 r_c^m + k_2 r_c^t, \tag{9}$$

where $k_1 > 0$, $k_2 > 0$, and $k_1 + k_2 = 1$.

The mean prototype r^m encapsulates task-specific visual features, whereas the translated prototype r^t integrates additional semantic information. To leverage the strengths of both, robust class prototypes r_c are constructed through convex combination using (9). Subsequently, labels are assigned to query samples q using (2), where $d(\cdot,\cdot)$ is implemented as the cosine distance.

4 Experiments

4.1 Datasets

The proposed method was evaluated on four widely used FSL datasets: MiniImageNet [53], TieredImageNet [45], CIFAR-FS [32], and FC100 [42].

MiniImageNet, derived from ImageNet [12], consists of 100 classes with 600 images per class, where 64 classes are designated for training, 16 for validation, and the remaining 20 for testing. The image resolution for the ResNet is set to 84×84, while for the Transformer, it is 224×224.

TieredImageNet shares similarities with MiniImageNet as it is also derived from ImageNet. However, it boasts a larger scale and a hierarchical structure, encompassing 608 classes distributed as 351/97/160 for training/validation/testing, while maintaining the same image resolution as MiniImageNet.

CIFAR-FS, a variant of CIFAR100 [31], comprises 60,000 images across 100 classes, with 600 images per class. Adhering to a similar split as MiniImageNet, 64 classes are allocated for training, 16 for validation, and the remaining 20

for testing. The image size for the ResNet is set to 32×32, whereas for the Transformer, it is 224×224.

FC100, an FSL version derived from the CIFAR100, includes 100 classes with 600 images per class. It employs a superclass partition strategy, aimed at mitigating information overlap. Following the partitioning scheme outlined in [42], it is divided into 12/4/4 superclasses for training/validation/testing, maintaining the same image resolution as CIFAR-FS.

4.2 Implementation Details

Architecture. In experiments, we utilized two feature extractors: ResNet12 [22] and Swin Transformer [36]. Notably, the output dimension for ResNet12 is configured to 640, while Swin Transformer's output dimension is set to 768. The generator G is a three-layer MLP with a LeakyReLU activation function [37], where the two hidden layers of G have dimensions of 4,096 and 2,048, respectively. The discriminator D is a two-layer MLP where the activation function in the first layer is LeakyReLU and in the second layer is Sigmoid [46]. The hidden layer of D has a dimension of 1,024.

Training Details. A two-step training procedure is performed in experiments. In step one, the feature extractor f is pre-trained using the cross-entropy loss function [24] and meta-tuned [8]. Next, G and D are trained 100 epochs with a batch size of 128 using an Adam optimizer [29] with a learning rate of 1e-4. The hyper-parameter β is set to one, $[k_1, k_2]$ are set as [0.25, 0.75] for 1-shot tasks and [0.9, 0.1] for 5-shot tasks. Class definitions are given from WordNet [40] based on class labels. No other data augmentation strategies are used during training, except for color jitter and random flipping.

Evaluation protocol. The proposed method is evaluated under 5-way 1-shot and 5-way 5-shot settings. For each setting, 600 classification tasks are uniformly sampled from the novel set. In each task, there are 15 query samples per class. The mean and 95% confidence interval of the accuracy are reported.

4.3 Results

Tables 1, 2, 3 and 4 provide a detailed comparison of the proposed method (SCTN) and existing methods on the 5-way 1-shot and 5-way 5-shot tasks. Notably, in the 1-shot learning scenario, SCTN demonstrates a significant competitive advantage. Specifically, SCTN, based on the ResNet architecture, has surpassed several Transformer-based methods [7,15,23]. Moreover, when integrating the Transformer into SCTN to form the SCTN-Trans version, this model achieves the highest performance among all compared methods. In the 5-shot task, SCTN demonstrates the best performance on CIFAR series datasets. However, on ImageNet series datasets, its performance closely follows the state-of-the-art method FewTURE [23]. This observation can be attributed to FewTURE's implementation of an online optimization strategy called "re-weighting" during the inference stage of the 5-shot setting, which significantly enhances its performance in 5-shot tasks but also substantially increases the computational

Table 1. Comparison on miniImageNet dataset.

Method	Venue	Backbone	5-way 1-shot	5-way 5-shot
MatchNet [53]	NeurIPS'16	ResNet-12	65.64 ± 0.20	78.72 ± 0.15
ProtoNet [51]	NeurIPS'17	ResNet-12	62.39 ± 0.21	80.53 ± 0.14
MAML [18]	ICML'17	ResNet-12	49.24 ± 0.21	58.05 ± 0.10
CTM [34]	CVPR'19	ResNet-18	64.12 ± 0.82	80.51 ± 0.13
AM3 [55]	NeurIPS'19	ResNet-12	65.30 ± 0.49	78.10 ± 0.36
RestroreNet [57]	AAAI'20	ResNet-18	61.14 ± 0.22	-
FEAT [58]	CVPR'20	ResNet-12	66.78 ± 0.20	82.05 ± 0.14
Meta-Baseline [8]	ICCV'21	ResNet-12	63.17 ± 0.23	79.26 ± 0.17
SVAE-Proto [56]	CVPR'22	ResNet-12	74.84 ± 0.23	83.28 ± 0.40
SUN [15]	ECCV'22	ViT-S	67.80 ± 0.45	83.25 ± 0.30
FewTURE [23]	NeurIPS'22	Swin-T	72.40 ± 0.78	**86.38 ± 0.49**
FGFL [11]	ICCV'23	ResNet-12	69.14 ± 0.80	86.01 ± 0.62
Meta-AdaM [50]	NeurIPS'23	ResNet-12	59.89 ± 0.49	77.92 ± 0.43
SP-CLIP [7]	CVPR'23	Visformer-T	72.31 ± 0.40	83.42 ± 0.30
SCTN (Ours)	Ours	ResNet-12	74.30 ± 0.67	81.88 ± 0.54
SCTN-Trans (Ours)	Ours	Swin-T	**75.86 ± 0.74**	85.91 ± 0.51

Table 2. Comparison on tieredImageNet dataset.

Method	Venue	Backbone	5-way 1-shot	5-way 5-shot
MatchNet [53]	NeurIPS'16	ResNet-12	68.50 ± 0.92	80.60 ± 0.71
ProtoNet [51]	NeurIPS'17	ResNet-12	68.23 ± 0.23	84.03 ± 0.16
MAML [18]	ICML'17	ResNet-12	67.92 ± 0.17	72.41 ± 0.20
TADAM [42]	NeurIPS'18	ResNet-12	62.13 ± 0.31	81.92 ± 0.30
AM3 [55]	NeurIPS'19	ResNet-12	69.08 ± 0.47	82.58 ± 0.31
CAN [26]	NeurIPS'19	ResNet-12	69.89 ± 0.51	84.23 ± 0.37
CTM [34]	CVPR'19	ResNet-18	68.41 ± 0.39	84.28 ± 1.73
RFS [52]	ECCV'20	ResNet-12	69.74 ± 0.72	84.41 ± 0.55
FEAT [58]	CVPR'20	ResNet-12	70.80 ± 0.23	84.79 ± 0.16
Meta-Baseline [8]	ICCV'21	ResNet-12	68.62 ± 0.27	83.29 ± 0.18
SVAE-Proto [56]	CVPR'22	ResNet-12	76.98 ± 0.65	85.77 ± 0.50
SUN [15]	ECCV'22	ViT-S	72.99 ± 0.50	86.74 ± 0.33
FewTURE [23]	NeurIPS'22	Swin-T	76.32 ± 0.87	**89.96 ± 0.55**
FGFL [11]	ICCV'23	ResNet-12	73.21 ± 0.88	87.21 ± 0.61
Meta-AdaM [50]	NeurIPS'23	ResNet-12	65.31 ± 0.48	85.24 ± 0.35
SP-CLIP [7]	CVPR'23	Visformer-T	78.03 ± 0.46	88.55 ± 0.32
SCTN (Ours)	Ours	ResNet-12	74.20 ± 0.87	83.86 ± 0.67
SCTN-Trans (Ours)	Ours	Swin-T	**78.35 ± 0.82**	88.59 ± 0.57

burden during inference. Comparison with the latest semantic-based methods [7] reveals SCTN's significant performance enhancement. Specifically, in the 1-shot task evaluation across four datasets, SCTN achieves an average accuracy improvement of 1.90%. For the 5-shot task, its average performance improvement reaches 1.42%, affirming the effectiveness of the proposed SCTN in few-shot learning.

Table 3. Comparison on CIFAR-FS dataset.

Method	Venue	Backbone	5-way 1-shot	5-way 5-shot
ProtoNet [51]	NeurIPS'17	ResNet-12	72.20 ± 0.70	83.50 ± 0.50
MetaOptNet [32]	CVPR'19	ResNet-12	72.80 ± 0.70	84.30 ± 0.50
MABAS [28]	ECCV'20	ResNet-12	73.51 ± 0.92	85.65 ± 0.65
RFS [52]	ECCV'20	ResNet-12	71.50 ± 0.80	86.00 ± 0.50
SUN [15]	ECCV'22	ViT-S	78.37 ± 0.46	88.84 ± 0.32
FewTURE [23]	NeurIPS'22	Swin-T	77.76 ± 0.81	88.90 ± 0.59
SP-CLIP [7]	CVPR'23	Visformer-T	82.18 ± 0.40	88.24 ± 0.32
SCTN (Ours)	Ours	ResNet-12	81.41 ± 0.75	86.72 ± 0.62
SCTN-Trans (Ours)	Ours	Swin-T	**83.56 ± 0.72**	**88.95 ± 0.57**

Table 4. Comparison on FC100 dataset.

Method	Venue	Backbone	5-way 1-shot	5-way 5-shot
ProtoNet [51]	NeurIPS'17	ResNet-12	41.54 ± 0.76	57.08 ± 0.76
TADAM [42]	NeurIPS'18	ResNet-12	40.10 ± 0.40	56.10 ± 0.40
MetaOptNet [32]	CVPR'19	ResNet-12	47.20 ± 0.60	55.50 ± 0.60
MABAS [28]	ECCV'20	ResNet-12	42.31 ± 0.75	58.16 ± 0.78
RFS [52]	ECCV'20	ResNet-12	42.60 ± 0.70	59.10 ± 0.60
FewTURE [23]	NeurIPS'22	Swin-T	47.68 ± 0.78	63.81 ± 0.75
Meta-AdaM [50]	NeurIPS'23	ResNet-12	41.12 ± 0.49	56.14 ± 0.49
SP-CLIP [7]	CVPR'23	Visformer-T	48.53 ± 0.38	61.55 ± 0.41
SCTN (Ours)	Ours	ResNet-12	48.56 ± 0.76	59.99 ± 0.73
SCTN-Trans (Ours)	Ours	Swin-T	**50.86 ± 0.76**	**64.00 ± 0.75**

4.4 Ablation Study

Ablation Studies for Feature Enhanced Generative Adversarial Network. A series of ablation experiments were conducted to assess the effectiveness of different components of FEGAN on MiniImageNet and CIFAR-FS, with

detailed results provided in Table 5. Initially, a baseline experiment was designed using a non-linear network to capture the mapping between the prototype and edge image distributions. However, the results showed poor performance, indicating that relying solely on a non-linear network is inadequate for modeling the distribution of the prototype image domain. Subsequently, the introduction of CGAN during the modeling process significantly improved the overall performance, affirming CGAN's superiority in distribution learning over non-linear networks. Moreover, the integration of the proposed NCR module into FEGAN led to a more substantial performance enhancement. Specifically, in the 1-shot learning task, FEGAN improved accuracy by 8.95% to 9.33% compared to the baseline, while in the 5-shot scenario, it increased by 0.94% to 1.01%. This experiment strongly supports the effectiveness of FEGAN.

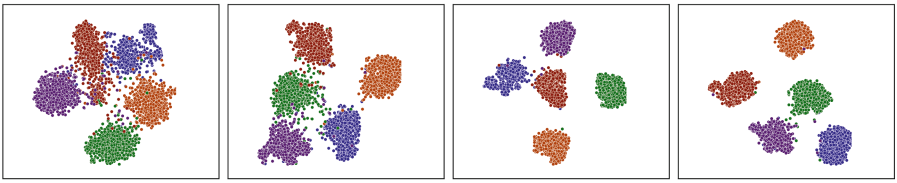
Table 5. Ablation studies for Feature Enhanced Generative Adversarial Network.

Method	MiniImageNet		CIFAR-FS	
	5W1S	5W5S	5W1S	5W5S
Baseline model	64.97 ± 0.81	80.87 ± 0.58	72.46 ± 0.90	85.78 ± 0.64
+CGAN	69.30 ± 0.74	81.24 ± 0.57	75.96 ± 0.83	85.90 ± 0.65
+CGAN+NCR	74.30 ± 0.67	81.88 ± 0.54	81.41 ± 0.75	86.72 ± 0.62

Ablation Studies for the Semantic. The choice of text type and encoder significantly influences the acquisition of semantic embeddings. Therefore, we conducted experiments to compare and analyze the impact of these factors on SCTN. Initially, we utilized two text encoders: BERT [13] (a natural language pre-training model) and CLIP [43] (a multi-modal pre-training model) to generate semantic embeddings. Regarding text type, we focused on the differentiation between class definitions and names. Specifically, we transformed class names into sentences using templates and then inputted them into the encoder to extract semantic embeddings. The experimental results, presented in Table 6, indicate that in both 1-shot and 5-shot learning scenarios, SCTN employing CLIP as the encoder exhibited superior performance, with accuracy enhancements ranging from 3.77% to 7.61% and from 0.66% to 5.87%, respectively. This suggests that CLIP demonstrates stronger adaptability in visual tasks. Moreover, irrespective of whether it was in a 1-shot or 5-shot scenario, SCTN utilizing class definitions instead of names demonstrated superior performance, particularly with an average accuracy improvement of 0.98% in the 1-shot setting. This underscores that detailed class definitions help reduce the inherent ambiguity in class names, leading to enhanced accuracy.

Table 6. Comparison using different semantics.

Text Encoder	Text Type	MiniImageNet		CIFAR-FS	
		5W1S	5W5S	5W1S	5W5S
BERT	Name	65.66 + 0.79	80.87 ± 0.58	75.59 ± 0.87	86.00 ± 0.64
	Definition	66.69 ± 0.80	80.98 ± 0.58	77.64 ± 0.81	86.15 ± 0.65
CLIP	Name	73.57 ± 0.71	81.71 ± 0.56	81.29 ± 0.75	86.70 ± 0.62
	Definition	74.30 ± 0.67	81.88 ± 0.54	81.41 ± 0.75	86.72 ± 0.62



(a) Mini, O (b) Tiered, O (c) Mini, T (d) Tiered, T

Fig. 3. Samples from five novel classes are sampled from miniImageNet and tieredImageNet visualized by t-SNE algorithm. The labels "Mini" and "Tiered" denote the dataset, while "O" and "T" denote the original and translated feature space distribution, respectively.

4.5 Visualization Analysis

To further validate the effectiveness of the proposed method, a sampling of five novel classes is conducted from the test sets of the miniImageNet and tieredImageNet datasets. The distribution of these classes in the feature space is visualized using t-SNE [38], a widely-used visualization technique. The resulting visualizations can be observed in subfigures 3a and 3b of Fig. 3. Subsequently, feature enhancement is applied to all the samples using the trained FEGAN model. The distribution of the enhanced samples in the feature space is depicted in subfigures 3c and 3d of Fig. 3. Upon careful examination of the distributions before and after feature enhancement, notable changes can be observed. Specifically, the enhanced feature space exhibits a more compact intra-class distance, indicating tight clustering of instances belonging to the same class. Furthermore, the inter-class distance becomes more distinct and discernible, indicating a clearer separation between different classes. This visual comparison provides compelling evidence for the effectiveness of the proposed method. By rendering the intra-class distances more compact and the inter-class distances more distinguishable, the proposed SCTN method enables the construction of stable class prototypes, ultimately leading to enhanced few-shot classification performance.

5 Conclusion

In this paper, the Semantic Conditional Translation Network (SCTN) is proposed to tackle the problem of unstable class prototypes in few-shot learning. To tackle this problem, the base set is divided into a prototype image domain with strong class representation and an edge image domain with weak class representation. Then, the Feature Enhanced Conditional Generative Adversarial Network (FEGAN) is used to learn the translation function between both domains. Specifically, the generator in FEGAN is employed to learn a mapping from the edge image domain to the prototype image domain under the control of semantics, and the discriminator is employed to determine the class representation of an image. Importantly, a Non-parametric Classification Regularizer is proposed to enlarge the discriminability of the mapped samples, thus enhancing the translation performance. The trained FEGAN is used to improve the class representation of the support set samples and thus stable class prototypes are available. The experiment results show that SCTN achieves competitive performance on four widely-used FSL datasets. In future, we intend to explore additional potential applications of semantic information in enhancing visual recognition tasks.

Acknowledgements. This work was supported by CNPC Innovation Found (2021DQ02-0903), the National Natural Science Foundation of China under Grant NSFC-62076172, the National Key Research and Development Program of China under Grant 2023YFF1204901, and the Key Research and Development Program of Sichuan Province under Grant 2023YFG0116.

References

1. Andrychowicz, M., et al.: Learning to learn by gradient descent by gradient descent. In: NIPS, vol. 29 (2016)
2. Antoniou, A., Storkey, A., Edwards, H.: Data augmentation generative adversarial networks. arXiv preprint arXiv:1711.04340 (2017)
3. Bateni, P., Goyal, R., Masrani, V., Wood, F., Sigal, L.: Improved few-shot visual classification. In: CVPR, pp. 14493–14502 (2020)
4. Bengio, Y., Courville, A., Vincent, P.: Representation learning: a review and new perspectives. PAMI **35**(8), 1798–1828 (2013)
5. Biederman, I.: Recognition-by-components: a theory of human image understanding. Psychol. Rev. **94**(2), 115 (1987)
6. Chen, W.Y., Liu, Y.C., Kira, Z., Wang, Y.C.F., Huang, J.B.: A closer look at few-shot classification. arXiv preprint arXiv:1904.04232 (2019)
7. Chen, W., Si, C., Zhang, Z., Wang, L., Wang, Z., Tan, T.: Semantic prompt for few-shot image recognition. In: CVPR, pp. 23581–23591 (2023)
8. Chen, Y., Liu, Z., Xu, H., Darrell, T., Wang, X.: Meta-baseline: exploring simple meta-learning for few-shot learning. In: ICCV, pp. 9062–9071 (2021)
9. Chen, Z., Fu, Y., Wang, Y.X., Ma, L., Liu, W., Hebert, M.: Image deformation meta-networks for one-shot learning. In: CVPR, pp. 8680–8689 (2019)
10. Chen, Z., Fu, Y., Zhang, Y., Jiang, Y.G., Xue, X., Sigal, L.: Multi-level semantic feature augmentation for one-shot learning. TIP **28**(9), 4594–4605 (2019)

11. Cheng, H., Yang, S., Zhou, J.T., Guo, L., Wen, B.: Frequency guidance matters in few-shot learning. In: ICCV, pp. 11814–11824 (2023)
12. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: CVPR, pp. 248–255. IEEE (2009)
13. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
14. Doersch, C., Gupta, A., Zisserman, A.: CrossTransformers: spatially-aware few-shot transfer. In: NIPS, vol. 33, pp. 21981–21993 (2020)
15. Dong, B., Zhou, P., Yan, S., Zuo, W.: Self-promoted supervision for few-shot transformer. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XX, pp. 329–347. Springer Nature Switzerland, Cham (2022). https://doi.org/10.1007/978-3-031-20044-1_19
16. Elsken, T., Metzen, J.H., Hutter, F.: Neural architecture search: a survey. JMLR **20**(1), 1997–2017 (2019)
17. Elsken, T., Staffler, B., Metzen, J.H., Hutter, F.: Meta-learning of neural architectures for few-shot learning. In: CVPR, pp. 12365–12375 (2020)
18. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: ICML, pp. 1126–1135. PMLR (2017)
19. Gidaris, S., Bursuc, A., Komodakis, N., Pérez, P., Cord, M.: Boosting few-shot visual learning with self-supervision. In: ICCV, pp. 8059–8068 (2019)
20. Hariharan, B., Girshick, R.: Low-shot visual recognition by shrinking and hallucinating features. In: ICCV, pp. 3018–3027 (2017)
21. Hartigan, J.A., Wong, M.A.: Algorithm AS 136: a K-means clustering algorithm. RSSC **28**(1), 100–108 (1979)
22. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR, pp. 770–778 (2016)
23. Hiller, M., Ma, R., Harandi, M., Drummond, T.: Rethinking generalization in few-shot classification. NIPS **35**, 3582–3595 (2022)
24. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Comput. **18**(7), 1527–1554 (2006)
25. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (2006)
26. Hou, R., Chang, H., Ma, B., Shan, S., Chen, X.: Cross attention network for few-shot classification. In: NIPS, vol. 32 (2019)
27. Jamal, M.A., Qi, G.J.: Task agnostic meta-learning for few-shot learning. In: CVPR, pp. 11719–11727 (2019)
28. Kim, J., Kim, H., Kim, G.: Model-agnostic boundary-adversarial sampling for test-time generalization in few-shot learning. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I, pp. 599–617. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-58452-8_35
29. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
30. Koch, G., Zemel, R., Salakhutdinov, R., et al.: Siamese neural networks for one-shot image recognition. In: ICML Workshop, vol. 2. Lille (2015)
31. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)

32. Lee, K., Maji, S., Ravichandran, A., Soatto, S.: Meta-learning with differentiable convex optimization. In: CVPR, pp. 10657–10665 (2019)
33. Li, A., Huang, W., Lan, X., Feng, J., Li, Z., Wang, L.: Boosting few-shot learning with adaptive margin loss. In: CVPR, pp. 12576–12584 (2020)
34. Li, H., Eigen, D., Dodge, S., Zeiler, M., Wang, X.: Finding task-relevant features for few-shot learning by category traversal. In: CVPR, pp. 1–10 (2019)
35. Li, K., Zhang, Y., Li, K., Fu, Y.: Adversarial feature hallucination networks for few-shot learning. In: CVPR, pp. 13470–13479 (2020)
36. Liu, Z., et al.: Swin transformer: hierarchical vision transformer using shifted windows. In: ICCV, pp. 10012–10022 (2021)
37. Maas, A.L., Hannun, A.Y., Ng, A.Y., et al.: Rectifier nonlinearities improve neural network acoustic models. In: Proceedings of ICML, vol. 30, p. 3. Atlanta, Georgia, USA (2013)
38. Van der Maaten, L., Hinton, G.: Visualizing data using T-SNE. JMLR **9**(11) (2008)
39. Mangla, P., Kumari, N., Sinha, A., Singh, M., Krishnamurthy, B., Balasubramanian, V.N.: Charting the right manifold: manifold mixup for few-shot learning. In: WAVC, pp. 2218–2227 (2020)
40. Miller, G.A.: Wordnet: a lexical database for English. CACM **38**(11), 39–41 (1995)
41. Mirza, M., Osindero, S.: Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014)
42. Oreshkin, B., Rodríguez López, P., Lacoste, A.: TADAM: task dependent adaptive metric for improved few-shot learning. In: NIPS, vol. 31 (2018)
43. Radford, A., et al.: Learning transferable visual models from natural language supervision. In: ICML, pp. 8748–8763. PMLR (2021)
44. Ravi, S., Larochelle, H.: Optimization as a model for few-shot learning. In: ICLR (2017)
45. Ren, M., et al.: Meta-learning for semi-supervised few-shot classification. arXiv preprint arXiv:1803.00676 (2018)
46. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature **323**(6088), 533–536 (1986)
47. Rusu, A.A., et al.: Meta-learning with latent embedding optimization. arXiv preprint arXiv:1807.05960 (2018)
48. Schwartz, E., et al.: Delta-encoder: an effective sample synthesis method for few-shot object recognition. In: NIPS, vol. 31 (2018)
49. Shorten, C., Khoshgoftaar, T.M.: A survey on image data augmentation for deep learning. JBD **6**(1), 1–48 (2019)
50. Siyuan Sun, H.G.: Meta-AdaM: a meta-learned adaptive optimizer with momentum for few-shot learning. In: NIPS (2023)
51. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. In: NIPS, vol. 30 (2017)
52. Tian, Y., Wang, Y., Krishnan, D., Tenenbaum, J.B., Isola, P.: Rethinking few-shot image classification: a good embedding is all you need? In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12359, pp. 266–282. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58568-6_16
53. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. In: NIPS, vol. 29 (2016)
54. Wang, Y.X., Girshick, R., Hebert, M., Hariharan, B.: Low-shot learning from imaginary data. In: CVPR, pp. 7278–7286 (2018)
55. Xing, C., Rostamzadeh, N., Oreshkin, B., O Pinheiro, P.O.: Adaptive cross-modal few-shot learning. In: NIPS, vol. 32 (2019)

56. Xu, J., Le, H.: Generating representative samples for few-shot classification. In: CVPR, pp. 9003–9013 (2022)
57. Xue, W., Wang, W.: One-shot image classification by learning to restore prototypes. In: AAAI, vol. 34, pp. 6558–6565 (2020)
58. Ye, H.J., Hu, H., Zhan, D.C., Sha, F.: Few-shot learning via embedding adaptation with set-to-set functions. In: CVPR, pp. 8808–8817 (2020)
59. Zhang, B., Li, X., Ye, Y., Huang, Z., Zhang, L.: Prototype completion with primitive knowledge for few-shot learning. In: CVPR, pp. 3754–3762 (2021)
60. Zhang, R., Che, T., Ghahramani, Z., Bengio, Y., Song, Y.: MetaGAN: an adversarial approach to few-shot learning. In: NIPS, vol. 31 (2018)
61. Zintgraf, L., Shiarli, K., Kurin, V., Hofmann, K., Whiteson, S.: Fast context adaptation via meta-learning. In: ICML, pp. 7693–7702. PMLR (2019)

Towards Adversarial Robustness and Reducing Uncertainty Bias through Expert Regularized Pseudo-Bidirectional Alignment in Transductive Zero Shot Learning

Abhishek Kumar Sinha[1(✉)], Deepak Mishra[2], and S. Manthira Moorthi[1]

[1] Signal and Image Processing Area, Space Applications Center, Ahmedabad, India
{aks,smmoorthi}@sac.isro.gov.in
[2] Indian Institute of Space Science and Technology, Thiruvananthapuram, India
deepak.mishra@iist.ac.in

Abstract. Transductive zero-shot learning (TZSL) aims to minimize the domain shift between the learned and true distribution of the unseen classes by allowing access to the unpaired samples from unseen classes. While many distribution alignment based methods attempt to align both visual and semantic spaces to train the classifier, their performance is still limited by confirmation bias. Additionally, bidirectional alignment approaches are based on the strong assumption that the intrinsic dimensions of visual and semantic spaces are the same, which is rarely true. In this work, we first highlight the limitations of bidirectional alignment in terms of intrinsic dimensionality. We then present a pseudo-bidirectional approach that, without any underlying assumptions on these spaces, utilizes the learned visual-to-attribute mapping to minimize the distribution shift between learned and true unseen visual feature distributions. We further utilize an entangled loss between semantic and visual space to minimize the confirmation or uncertainty bias and improve the adversarial robustness. We, theoretically and empirically, show the performance gain in addition to the adversarial robustness under the proposed setting.

Keywords: Zero-shot learning · Bias · Adversarial robustness.

1 Introduction

The aim of zero-shot learning is to recognize and classify objects or concepts for which they have not been explicitly trained. In most of the practical scenarios, the computer vision models are required to be trained on a set of large number of training examples paired with their corresponding labels, known as seen classes. The trained model are then used to infer the labels for which there are no available training examples, referred to as target labels. In inductive zero-shot learning, samples from the target (or unseen) classes are not provided

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 330–345, 2025.
https://doi.org/10.1007/978-3-031-78183-4_21

for training. However, a sufficient number of paired examples are provided for the seen categories. This approach requires the classifier to learn the relation between visual and semantic spaces using the seen classes and transfers this knowledge to the unseen classes assuming that such relevant knowledge exists. This knowledge sharing requires annotated data such as vector embedding of labels, attribute features and so on. However, transfer learning without access to unseen labels can be quite challenging due to domain shift problem [3]. To simplify the problem, Transductive zero-shot learning (TZSL) [4,13,14] utilizes the unlabeled examples of the targeted classes for training. This allows access to the collective target data distribution without correspondences to ease off the burden of distribution shift.

Most of the approaches are influenced by generative modelling that intend to align the distribution of real examples and generated examples followed by training the classifier on the generated examples. Depending on the discrepancy in the learned data distribution, the classifier may suffer from confirmation bias, which means that the classifier is trained on the generated samples assuming that they are correctly paired to their target labels. Additionally, many previous graph based approaches model the attribute relation using Word2Vec or GloVe embeddings but rely on corpora training which may not provide necessary characteristics to distinguish between the classes [19]. Moreover, knowledge graph based methods [14] also has its own challenges. For example, knowledge graphs may struggle with handling ambiguous concepts or entities with multiple senses. Different classes or concepts can share similar or overlapping features, making it challenging to disambiguate them solely based on the information in the knowledge graph.

To address these limitations, we propose pseudo-bidirectional alignment that utilizes expert information to learn bidirectional-like mapping. The contribution of the proposed work is as follows:

1. We introduce pseudo-bidirectional alignment using Expert guided VAEGAN that, unlike bidirectional adversarial learning, learns the semantic-to-visual mapping based on the additional knowledge from an expert model, which is visual-to-attribute mapping in our case.
2. The proposed model improves the semantic-to-visual mapping by incorporating knowledge from an expert model to learn distribution shift in a low intrinsic dimensional space, contributing to a more robust and effective learning process.
3. A new entangled loss function is introduced for classifier training, by integrating generated visual features and pseudo-labels. This leads to reduction of confirmation bias and shows its effectiveness in terms of adversarial robustness and, providing a novel and impactful contribution to the training process.
4. Both theoretical and experimental evidence were presented, showcasing the remarkable performance of pseudo-bidirectional alignment. Finally, implicit robustness is achieved through the proposed approach, contributing to the model's resilience in the face of various challenges. The method's ability to

overcome uncertainty is highlighted, making it a notable and impactful contribution to the field of Transductive zero-shot learning.

2 Related Works

2.1 Zero-Shot Learning

Zero-shot learning has garnered a lot of interest in the past few years due to its practical applications in many vision and language-related problems. Inductive zero-shot entity recognition has previously been addressed in which most of them tend to learn semantic to visual space mapping using projection mapping [32]. This approach of transferring knowledge from seen to unseen classes suffers from domain shift due to non-overlapping distributions of seen and unseen classes. Subsequently, some works utilize two networks to align the distributions in both semantic and visual spaces by using generative modelling such as VAE and GANs. For example, Cycle-WGAN [2] uses a new multi-modal cycle consistency loss which constrains the optimization problem to generate useful visual features for the training of classifier. Another method is to exploit the expert knowledge for domain alignment by leveraging the expertise of a domain expert to constrain the learning process, and it closely resembles our idea in this work. For example, Norouzi et al. [15] proposes a convex combination mapping approach for zero-shot learning. It incorporates expert knowledge in the form of semantic attributes and enforces a regularization term to constrain the model's predictions to be a convex combination of attribute vectors. Since the model is regularized to align its predictions with the provided attributes, noisy or misleading attributes might negatively impact the learning process and lead to erroneous predictions. Field-Guide-Inspired Zero-Shot Learning ([12]) is another interesting approach which directly involves a human expert to interact with the learner. In this approach, the learner is first trained on a set of base classes followed by interaction with an expert annotator to seek minimal guidance on the attributes to classify the unseen classes. The method may suffer from the knowledge gap between attribute understandings of humans and the neural net, and it is relatively difficult to align their knowledge due to variations in the human experts. Nevertheless, the absence of knowledge of unseen classes serves as the performance bottleneck and restricts the performance of inductive ZSL.

Contrary to an inductive setting, transductive ZSL allows the learner to utilize the knowledge of examples from unseen classes without correspondence. Generative models have been adopted by most of the state-of-the-art as adversarial training allows to align the distribution. Marmoreo et al. [13] proposed the idea of decoupled feature generation by encapsulating the visual patterns into structured prior to boost the performance of conditional visual feature synthesis. It uses DecGAN to capture the distribution of visual features and generate realistic descriptors. The pioneering work in zero-VAE-GAN [4] is the first to attempt the coupled Variational Autoencoder (VAE) and generative model for this task. It uses generative methods to synthesize visual features conditioned on semantic side information and learn a conventional supervised classifier from generated

sample. However, when generative models are trained with seen classes, there are inherently biased when it comes to generalization to unseen classes.

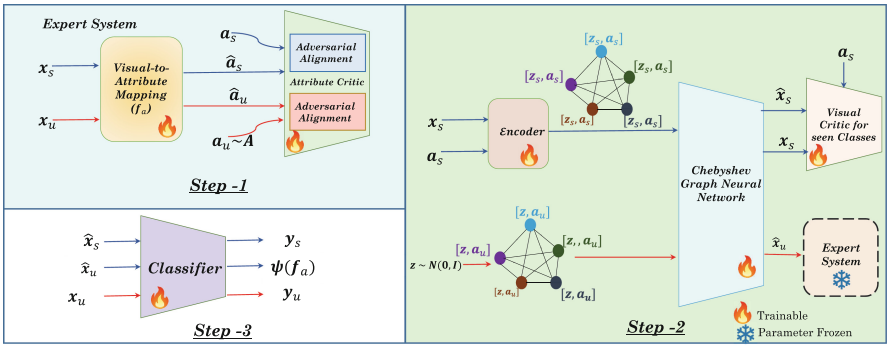


Fig. 1. Overall training methodology for proposed Transductive ZSL. Step 1 involves adversarial training of visual-to-attribute mapper (f_a) for both seen and unseen classes. Step 2 utilizes f_a from previous step for aligning distributions of true and generated visual features of unseen labels. In step 3, we utilize the paired generated feature-label and original feature-pseudolabel to train the classifier.

2.2 Graph Neural Networks in ZSL

Unlike multi-layer perceptron that has fully connected layers, graph neural nets learn the node embeddings based on the node strength and its connections to other nodes of the graph. Similar to self-attention [20–22], it is capable of learning global representations based on the node connections and strengths. The recent studies [7,14,29] have demonstrated the effectiveness of utilizing the graph structure in zero shot learning. Xiel et al. [29] proposed a region graph embedding network to capture the relationships between various parts of the image using graph convolutions. The graph nodes consist of local regions of the image and are connected by the edges depending on the pairwise nodes' similarity. Since the regionwise features of the image may fail to capture the extent of the relation, it translates to the edges' strength resulting in misleading interaction between patches. Similarly, the Visual-Semantic Entanglement network in [7] learns the graph embeddings of visual features and maps it to the semantic attributes using the knowledge graph. Additionally, it uses a multi-path entangled path network which feeds the visual features from CNN to GCN to learn the semantic relations resulting in self-consistent regression for graph modelling. Liu et al. [14] also exploits the knowledge graph through a transformer to learn class representations by embedding nodes in the knowledge graph. [10] exploits graph relation for attribute propagation to refine the features in semantic space based on the information aggregated from the neighbouring nodes. This approach does not add any constraint to align the learned attribute features and therefore makes

a strong assumption that attribute propagation does not affect the associated attribute labels, and thus the label for the propagated attributes is the same as the original associated label before propagation. While the knowledge graph embeddings are useful in natural language-related problems, the semantic vectors may not be directly applicable to computer vision problems. The limited semantic coverage of knowledge graph vectors introduces a bias towards certain classes restricting its ability to generalize for unseen classes. To avoid this limitation, we directly utilize the attribute vectors associated with class labels to represent the connection between nodes.

3 Limitations of Bidirectional Alignment

We use the idea of Intrinsic dimensionality and Wassertein distance to highlight the limitation of bidirectional alignment [25].

Theorem 1 (Invariance of domain). *If U is an open subset of R^n and $f : U \rightarrow R^n$ is an injective continuous map, then $V = f(U)$ is an open and f is a homeomorphism between U and V.*

Bi-VAEGAN [25] learns adversarial mapping between visual space and attribute space. However, visual features comprise of additional details beyond the attribute features and therefore, the Intrinsic dimension of visual feature space (ID_v) is relatively larger than that of attribute space (ID_a). This limitation may negatively impact the diversity of learned samples in visual space.

Proposition 1. *Let $x \sim \mathcal{P}$ and $x' \sim \mathcal{P}'$ be the samples from true and learned distributions, respectively such that difference between the intrinsic dimension of $\mathcal{P}'$ and $\mathcal{P}$ is δ. If D^* is the intrinsic dimension of $\mathcal{P}$, then normalized Wasserstein distance, conditioned on $[0, w]$, is given by,*

$$\mathcal{W}_2^2(F, G, w) = \frac{2\delta^2}{(D^* + 2)(D^* + \delta + 2)(2D^* + \delta + D^{*2} + D^*\delta)} \tag{1}$$

The proof of Proposition 1 is provided in supplementary material. It shows that the Wasserstein distance is less sensitive to the distribution shift if the underlying intrinsic dimension is large. This means that, for a given shift, the Wasserstein loss in visual space remains relatively lower than that in attribute space, and therefore, in the case of Bi-VAEGAN [25], adversarial learning in visual space is not as effective as attribute space. We, therefore, propose to use only attribute space to learn pseudo-bidirectional alignment.

4 Methodology

4.1 Problem Formulation

Transductive ZSL aims to classify the unseen classes by accessing the unpaired examples from the domain of unseen classes $\mathcal{D}^u$. We denote by $\mathcal{D}^s$ =

$\{(x, y, a_y)|x \in \mathcal{X}^s, y \in \mathcal{Y}^s, a_y \in \mathcal{A}^s\}$ the domain of seen classes, where x is the visual feature, y is the corresponding label and a_y is the attribute of that category. Similarly, $\mathcal{D}^u = \{(x, u, a_u)|x \in \mathcal{X}^u, u \in \mathcal{Y}^u, a_u \in \mathcal{A}^u\}$ is the set of unseen labels, and $\mathcal{A} = \mathcal{A}^s \bigcup \mathcal{A}^u$. The model won't have access to the labels for $x \in \mathcal{X}^u$. For generalized setting, it is assumed that $\mathcal{Y}^s \cap \mathcal{Y}^u = \Phi$. Furthermore, f_a and D_a are used to denote the visual-to-attribute mapper and the attribute critic, respectively. Additionally, $\psi \circ f_a$ assigns the label based on the cosine similarity between the generated attribute vector and attribute from unseen domain. We denote by x^{ug} the unseen generated visual feature vector. $\mathcal{E}$ and $\mathcal{G}$ are the encoders and decoders of the variational autoencoder, where $\mathcal{G}$ is a graph neural network. Let D_v^s denote the visual critic for seen classes only. The goal is to develop a framework to classify the examples from unseen classes correctly in both conventional and generalized setting.

For theoretical analysis, we denote by $\epsilon(h, f) = \mathbb{E}_{(x,a)\in D}[\mathbf{1}_{f(x)\neq h(x,a)}]$ the actual risk and $\hat{\epsilon}(h, f) = \frac{1}{|D|}\sum_{(x,a)\in D}[\mathbf{1}_{f(x)\neq h(x,a)}]$ the empirical risk. $d_{h^*}(\mathcal{D}_1, \mathcal{D}_2)$ and $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_1, \mathcal{D}_2)$ is the generative distance for optimal hypothesis h^* and $\mathcal{H}\Delta\mathcal{H}$ distance [1], respectively. The details are discussed in supplementary material.

4.2 Overall Outline

Figure 1 shows the overall training pipeline of pseudo-bidirectional alignment. The first stage involves training the visual-to-attribute mapper in adversarial fashion for both seen and unseen classes. Subsequently, it is utilized for transferring knowledge to train the VAEGAN architecture to reduce domain shift between true and generated visual features. Furthermore, it also exploits the semantic relationship among the classes through graphical structure to adversarially learn the visual features of unseen classes.

Visual-to-Attribute Mapping The first stage of training involves learning attribute feature from the given visual feature for both seen and unseen classes. It uses a simple multi-layer perceptron architecture which is trained in the supervised fashion with adversarial regularization for seen classes. For the samples from unseen classes, the model is trained only in adversarial fashion. Unlike Bi-VAEGAN [25], the adversarial learning on both seen and unseen classes helps the attribute critic to learn the interaction between their distributions. The optimization objective minimizes the L_1 norm for the examples from $\mathcal{D}^s$ given by,

$$L_{f_a}^s = \min_{f_a} \max_{D_a} ||f_a(x) - a_y||_1 + \lambda_1 L_{adv}^s, \tag{2}$$

where $L_{adv}^s(\mathcal{A}^s, \mathcal{V}^s) = \mathbb{E}[D_a(a^s)] - \mathbb{E}[D_a(f_a(x^s))] + (||\nabla_{\hat{a}^s}\mathbb{E}[D_a(\hat{a}^s)]||_2 - 1)^2$ with $\hat{a} = \alpha a^s + (1-\alpha) f_a(x^s)$. The objective for the unseen classes is similarly defined as,

$$\begin{aligned} L_{f_a}^u = L_{adv}^u = &\mathbb{E}[D_a(a^u)] - \mathbb{E}[D_a(f_a(x^u))] \\ &+ (||\nabla_{\hat{a}^u}\mathbb{E}[D_a(\hat{a}^u)]||_2 - 1)^2, \end{aligned} \tag{3}$$

where $a^u \sim \mathcal{A}^u$. The critic training includes the gradient penalty term [6] to induce better Lipschitz stability.

Pseudo-Bidirectional Alignment for Attribute to Visual Mapping The aim of training Expert-VAEGAN is to align the distributions of synthetic and true visual features. It uses the visual critic D_v^s specifically to align the generated seen visual features conditioned on their corresponding attribute features. Additionally, we use the visual-to-attribute map from the preceding stage as an expert system to transfer the knowledge about the previously learned relationship between visual and attribute space for unseen categories. We call this approach pseudo-bidirectional alignment because it tends to diversify only those components of visual features that are necessary to discriminate them in attribute space. To alleviate this issue, we re-utilize the visual-to-attribute mapper and attribute critic together to minimize the distribution shift between generated and true visual features. Since they have already been trained in preceding step, they together serve as an expert through their learned mapping. Firstly for seen categories, an encoder is explicitly used to learn the latent representations for the visual features conditioned on their respective attribute space. For unseen categories, we randomly sample the latent vector from a standard Gaussian distribution and stack them with a sampled attribute vector. The concatenated attribute and latent vectors serve as the node embeddings for the graph with the node connections defined by cosine similarity between the node attributes. For two nodes with attributes a_i and a_j, the the weight of connecting edge is defined as $e_{ij} = \frac{<a_i, a_j>}{||a_i||_2 ||a_j||_2}$. The constructed graph is then passed to the first order Chebyshev graph net (ChebNet) that computes the visual features corresponding to each node. Since it leverages the Laplacian eigenbasis of the graph to perform convolutions in the spectral domain, ChebNet captures both local and global structural dependency effectively. This, in turn, allows to learn the visual features based on their attribute similarity more effectively. Furthermore, some of elements in the visual feature vector is randomly masked with zero while training so to enhance the model's ability to learn the intra-feature connections.

For training, we apply the VAE objective on the latent space vectors of the seen classes as it is known to prevent the mode-collapse in GAN training. Furthermore, we add L1 loss to minimize the reconstruction error of visual features, and an adversarial regularization to align the synthetic and true visual features distributions. Since the visual features can be paired with its attribute features, the adversarial training aligns the learnt and true feature distribution conditioned on the attribute space. The overall training objective for the seen classes is given by,

$$\begin{aligned} \mathcal{L}^s = & \min_{\mathcal{E},\mathcal{G}} \max_{D_v^s} \mathbb{E}_{z_s \sim \mathcal{E}(x^s, a^s)}[KL(z^s || \mathcal{N}(0, I))] + \\ & \mathbb{E}_{z_s \sim \mathcal{E}(x^s, a^s)}[||\mathcal{G}(z^s, a^s) - x^s||_1] + \lambda_2 \mathcal{L}^s_{D_v^s}, \end{aligned} \tag{4}$$

where $D_v^s = \mathbb{E}[D_v^s(x^s, a^s)] - \mathbb{E}_{\bar{x}^s \sim \mathcal{G}}[D_v^s(\bar{x}^s, a^s)] + (||\nabla_{\hat{x}^s} \mathbb{E}[D_v^s(\hat{x}^s, a^s)]||_2 - 1)^2$.

For unseen examples, we directly utilize the attribute critic for adversarial learning. In this case, the training can be formulated as,

$$\mathcal{L}^u = \min_{\mathcal{G}} \max_{D_a} \mathbb{E}_{\hat{x}^u \sim \mathcal{G}}[||f_a(\hat{x}^u) - a^u||_1] + \lambda_3 \mathcal{L}^u_{adv}(\mathcal{G}, \mathcal{A}^u), \tag{5}$$

where $L^u_{adv}(\mathcal{G}, \mathcal{A}^u) = \mathbb{E}[D_a(a^u)] - \mathbb{E}_{\hat{x}^u \sim \mathcal{G}}[D_a(f_a(\hat{x}^u))] + (||\nabla_{\bar{a}^u} \mathbb{E}[D_a(\bar{a}^u)]||_2 - 1)^2$ and $\bar{a} = \alpha a^u + (1 - \alpha)\hat{a}^u$.

Here, λ_2 and λ_3 are the hyper-parameters. Equation 5 allows to align the knowledge of the graph net with that of visual-to-attribute mapping. Instead of strictly aligning visual features' distribution for unseen labels, we utilize the expert knowledge to learn the visual features discriminative enough to classify them into correct categories.

Algorithm 1 Algorithm for pseudo-bidirectional alignment

$\mathcal{X}^s$,$\mathcal{Y}^s$, $\mathcal{X}^u$, (A^u,A^s), T_1, T_2 **Trained** $\mathcal{G}$, $\mathcal{E}$, f_a, D^s_v, D_a
for *i in range* (T_1) **do**
  Train the visual-to-attribute mapping transductively using equations 2 and 3.
end
for *i in range* (T_2) **do**
  Generate synthetic visual features $\hat{x}^s$ for a sampled $\{x^s,a^s\}$.
  Train $\mathcal{E}$ and $\mathcal{G}$ for seen classes using equation 4.
  Uniformly sample a batch of attributes $a^u \sim \mathcal{A}^u$ and $z \sim \mathcal{N}(0, I)$.
  Estimate the edge weights $E = \{e_{ij}\}$ for $a_i, a_j \in a^u$.
  Generate the corresponding synthetic visual feature $\hat{x}^u \sim \mathcal{G}(z, a^u, E)$ and get $\{\hat{x}^u,a^u\}$.
  Train $\mathcal{E}$ and $\mathcal{G}$ for unseen classes using equation 5.
  For training of classifier, generate a pair of true visual feature and its pseudo-label $\{x^u,\psi \circ f_a\}$.
  Also, generate a pair of synthetic visual feature and original attribute vector $\{\hat{x}^u,a^u\}$.
  Train the classifier using the loss function 6.
end

4.3 Training the classifier

Since transductive setting allows access to unseen classes, it adds another degree of freedom that we exploit in the loss function. It is to be noted that training Expert-VAEGAN involves two modules that can assign a label to the visual feature, one is visual-to-attribute mapping and the other one is the classifier itself. To strengthen the alignment of their predictions, we apply entanglement between true and learnt distributions in both visual and semantic space. For this, we generate pseudo-labels from f_a for the given true visual feature in addition to the paired synthetic visual feature and the attribute vector. The combined training objective for the classifier is given by,

$$\mathcal{L}_{cls} = \beta[-\mathbb{E}_{x \sim \mathcal{X}^u} P(\psi \circ f_a|x; \theta)] + (1 - \beta)[-\mathbb{E}_{\hat{x} \sim \mathcal{G}} P(y|\hat{x}; \theta)], \tag{6}$$

where $P(y|\hat{x};\theta)$ denotes the probability of assigning label y to the synthetic feature $\hat{x}$. Similarly, $P(\psi \circ f_a|x;\theta)$ is the probability of assigning the pseudo-label $\psi \circ f_a$ to the true features. The overall procedure for training is described in the Algorithm 1.

4.4 Theoretical Perspective

In this section, we provide theoretical arguments to support our claims on adversarial robustness and confirmation bias.

Confirmation (or Uncertainty) Bias

Theorem 2. *Let $\mathcal{R}$ and $\mathcal{H}$ denote the hypothesis space of classifier h and visual-to-attribute regressor R, respectively. Without the loss of generality, lets assume that for regressor R, $\psi \circ R$ assigns the label based on the similarity measure between predicted attribute and unseen classes' attributes. If optimal classifier h^* satisfies the condition: $h^* = \underset{h'}{\operatorname{argmin}}\ \hat{\epsilon}_s(h', f) + \hat{\epsilon}_{ug}(h', f) + \hat{\epsilon}_u(h', \psi \circ R)$, Then with probability $1 - \delta$, following inequality holds for N number of samples,*

$$\begin{aligned}\epsilon_u(h, f) \leq \hat{\epsilon}_s(h, f) + d_{h^*}(\mathcal{X}^{ug}, \mathcal{X}^u) + d_{R^*}(\mathcal{X}^u, \mathcal{X}^s) + \\ d_{h^*}(\mathcal{X}^u, \mathcal{X}^s) + \frac{1}{2} d_{H\Delta H}(\mathcal{X}^u, \mathcal{X}^s) + \epsilon_u(h^*, f) \\ + \lambda + \sqrt{\frac{1}{2N} log \frac{2}{\delta}},\end{aligned}$$

where $\lambda = \epsilon_s(h^*, f) + \epsilon_u(h^*, \psi \circ R^*) + \epsilon_{ug}(h^*, f)$

We attempt to theoretically show that uncertainty or confirmation bias has detrimental impact on the overall performance. In transductive zero shot learning, confirmation bias arises in two ways. Firstly, we assume that the synthetic visual features belong to a particular class even though the generated feature may be perturbed enough to change its category. Second, the confirmation bias may get injected into the model through the conflict between the pseudo-label assigned by visual-to-attribute mapping to the true visual features and label predicted by the classifier. If the label predicted by the classifier and the pseudo-label assigned by the mapper do not match with each other, the overall training may converge to a sub-optimal solution. Theorem 2 shows that the loss function in 6 implicitly adds a constraint of on R^* and h^* through $\epsilon_u(h^*, \psi \circ R^*)$. For prediction error to reduce on the unseen labels, the labels assigned by both of them on a given feature must agree. Additionally, $d_{h^*}(\mathcal{X}^{ug}, \mathcal{X}^u)$ constrains the distance between the distributions of $\mathcal{X}^{ug}$ and $\mathcal{X}^u$ to reduce, and second, third and fourth terms are constant for a given problem due to fixed domain shift between seen and unseen classes.

Table 1. Performance comparison with state-of-the-art in both conventional and generalized ZSL. $\mathcal{I}$ and $\mathcal{T}$ refer to inductive and transductive settings, respectively. In generalized ZSL, U and S indicate accuracies for unseen and seen labels, respectively, and H is their harmonic mean. The best and second best results are shown in red and blue, respectively.

	Method	Conventional			Generalized								
		AWA2	CUB	SUN	AWA2			CUB			SUN		
					S	U	H	S	U	H	S	U	H
$\mathcal{I}$	RGEN [29]	73.6	76.1	63.8	67.1	76.5	71.5	60.0	73.5	66.1	44.0	31.7	36.8
$\mathcal{I}$	APNet [10]	68.0	57.7	62.3	83.9	54.8	66.4	55.9	48.1	51.7	40.6	35.4	37.8
$\mathcal{I}$	FG [12]	-	-	-	65.0	65.8	65.4	59.6	52.8	55.8	**61.3**	41.3	49.3
$\mathcal{I}$	LSG [30]	61.1	52.9	53.4	84.9	60.4	70.6	50.4	49.6	50.0	23.1	52.8	32.2
$\mathcal{I}$	Assym. Net [24]	-	55.9	57.6	-	-	-	19.4	56.5	28.9	18.5	28.6	22.5
$\mathcal{T}$	DSRL [31]	72.8	56.8	48.7	-	-	-	25.0	17.7	20.7	39.0	17.3	24.0
$\mathcal{T}$	F-VAEGAN-D2 [28]	-	71.1	70.1	-	-	-	65.1	61.4	63.2	41.9	60.6	49.6
$\mathcal{T}$	Zero-VAEGAN [4]	89.0	69.1	68.4	87.0	70.2	77.6	57.9	64.1	60.8	35.8	53.1	42.8
$\mathcal{T}$	ZSL-KG [14]	78.1	-	-	84.4	66.8	74.6	-	-	-	-	-	-
$\mathcal{T}$	DecGAN [13]	-	-	-	-	-	-	44.3	57.2	49.9	**68.4**	60.9	**63.4**
$\mathcal{T}$	Bi-VAEGAN [25]	95.8	**76.8**	**74.2**	**91.0**	**76.1**	**90.4**	**71.7**	**71.2**	**71.5**	45.4	**66.8**	54.1
$\mathcal{T}$	Ours	**96.4**	**77.2**	**75.2**	**92.6**	**89.6**	**91.1**	**70.3**	**73.9**	**72.1**	58.7	**66.2**	**62.2**

Implicit Adversarial Robustness In self-training based methods, it is quite common to apply label interpolation to improve adversarial robustness. However, [16] provides detailed analysis to contradict this assumption by showing that interpolation in noisy labels is as large an adversarial risk as the poisoning with similar noise rate.We, therefore, refrain from applying such interpolation in our loss function. Instead, we now show that loss function 6 implicitly acts as a weak upper bound on the robustness of the learned classifier. Our analysis for robustness is based on two assumptions: (1) the generated samples already contain adversarial noise, and therefore serves as adversarial examples [9], (2) We follow [18,26] to use population consistency loss as the measure of robustness on the unlabeled features from unseen classes. Based on this, we assume $x^{ug} \in \mathcal{B}_\rho$, where $\mathcal{B}_\rho(x) = \{x' : ||x' - x|| \leq \rho\}$. The population consistency loss is defined as $\mathcal{R}_B(h, x) = \mathbb{E}_{x\sim\mathcal{D}}[\mathbf{1}(\exists\ x' \in \mathcal{B}_\rho(x)\ such\ that\ h(x) \neq h(x'))]$.

Theorem 3. *Let $x \sim \mathcal{D}^u$ and x' be the true and the corresponding adversarial features, respectively. Let $R(x)$ maps the given visual feature x to its corresponding semantic feature, and $\psi \circ R(x)$ produces corresponding label based on semantic feature similarity. The population consistency loss $\mathcal{R}_B$ is weakly bounded by,*

$$\begin{aligned}\mathcal{R}_B \leq & \mathbb{E}_{x'\sim\mathcal{B}_\rho(x), x\sim\mathcal{D}^u}[\mathbf{1}(h(x') \neq f(x))] + \mathbb{E}_{x\sim\mathcal{D}^u}[\mathbf{1}(h(x) \neq \psi \circ R(x))] + \\ & \mathbb{E}_{x\sim\mathcal{D}^u}[\mathbf{1}(\psi \circ R(x) \neq f(x))],\end{aligned} \tag{7}$$

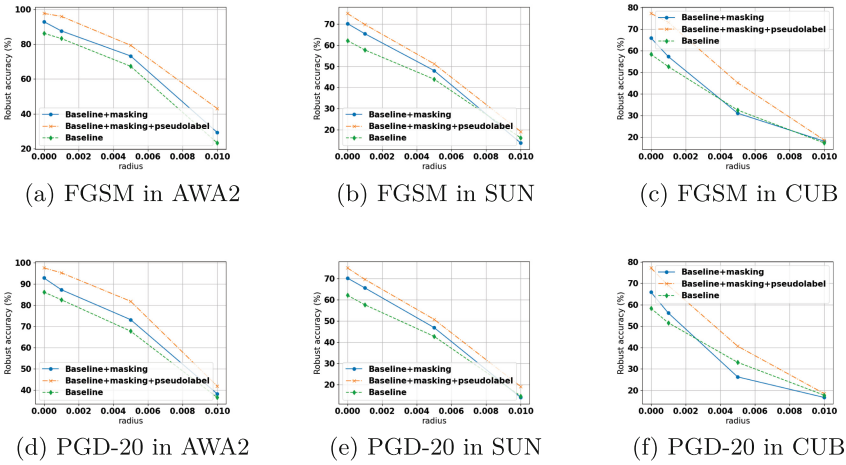
In Theorem 3, the first two terms represent the loss function 6 provided we treat the generated features as the adversarial examples. The bound is apparently weaker in the initial phase of training since we have no prior information about the labels of x for unseen categories, and therefore the third term cannot be explicitly controlled. However, as the model begins to converge, the third term approaches to zero and the bound eventually becomes tighter. In other words, the proposed loss function provides weak guarantee of adversarial robustness.

5 Experiments

In this section, we compare the performance with other state-of-the-art methods using benchmark datasets. Additionally, we provide empirical evidences to support our theoretical analysis. We conduct our experiments on three datasets, including AWA2 [27], CUB [23] and SUN [17]. The visual features of the images are extracted using ResNet-101 pre-trained network. We analyse the performance in both conventional and generalized setting by measuring overall accuracy for all the unseen classes. The dataset and training details along with hyper-parameter settings are provided in the supplementary material. In generalized setting, we measure the accuracy for both seen (ACC^s) and unseen classes (ACC^u) and express them using Harmonic mean given by $H = \frac{2ACC^s \times ACC^u}{ACC^s + ACC^u}$. Additionally, We directly report the results from the published papers.

5.1 Performance Comparison

Since we utilize the graphical structure in VAE-GAN setup, We compare the performance with VAE-GAN setups, including F-VAEGAN-D2 [28], Zero-VAEGAN [4], DecGAN [13] and Field-Guided CADA-VAE [12], and graph based approaches such as RGEN [29], APNet [10], LSG [30], Asymmetric Graph Network [24] and ZSL-KG [14]. Table 1 presents the comprehensive comparison to the aforementioned state-of-the-art models. In Conventional setting, our method achieves the best performance in in all three datasets. In generalized setting, our approach achieves best performance in AWA2 and CUB, and second best in SUN in terms of harmonic mean. We argue that our idea is still competitive for two reasons. Firstly, most of the generative methods apply strong discriminability on both visual and semantic spaces, whereas our model learns the features in visual space by solely transferring knowledge to semantic space and leveraged it to learn the discriminant visual features. This approach reduces the dependency on additional discriminator and aids to training stability. Secondly, it can be observed that the model does not overfit on the seen categories and maintains the decent balance between the observed and unobserved samples. It is to be noted that Field-Guided CADA-VAE [12] achieved second best accuracy on the seen classes of SUN dataset at the cost of unseen ones, and therefore overfits on the seen labels. Overall, the method achieved competitive performance by simply exploiting the knowledge of expert to learn the distribution of visual features.



(a) FGSM in AWA2 (b) FGSM in SUN (c) FGSM in CUB

(d) PGD-20 in AWA2 (e) PGD-20 in SUN (f) PGD-20 in CUB

Fig. 2. Graphical comparison of performance under FGSM and PGD-20 attacks on visual features for AWA2, SUN and CUB datasets.

5.2 Ablation studies

In this section, we analyse several aspects of the method to study their impacts on the overall accuracy of the classifier. We discuss the impact of visual feature masking and how the loss function based on pseudo-labels contributes to reduce the confirmation bias in the network, and we empirically verify the Theorem 2. Additionally, we evaluate its robustness against commonly used adversarial attacks to validate Theorem 3.

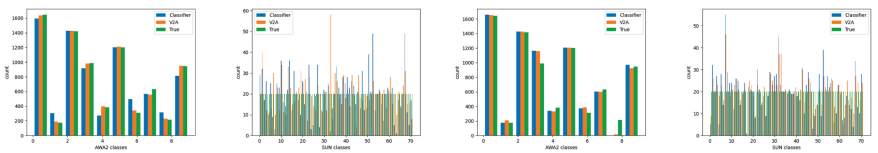
Table 2. Performance comparison with Top-1 accuracy under various settings. Masking-10/20 means that 10/20 random elements in the visual feature vector are masked during training. Methods without pseudolabel setting only uses the first term of loss function for training. Best results are shown in **bold**.

Method	Zero Shot Learning			Generalized Zero Shot Learning					
	AWA2	CUB	SUN	AWA2		CUB		SUN	
				S	U	S	U	S	U
Baseline	85.3	58.7	61.8	92.5	76.8	50.2	49.2	48.8	38.3
Baseline+masking-10	91.9	65.9	69.9	92.8	79.5	69.6	55.4	47.8	52.1
Baseline+masking-20	91.2	65.2	70.7	**93.5**	78.5	**72.8**	61.0	48.4	51.9
Baseline+masking-10+pseudolabel	**96.4**	**77.2**	**75.2**	92.6	**89.6**	70.3	**73.9**	**58.7**	**62.2**

Feature Masking and Pseudo-labels The motivation to learn intra-class feature dependencies is from Kong et. al. [8], which learns a new embedding

to enhance the separability between seen and unseen classes. Instead of adding learning overhead, we mask some of the elements in feature elements randomly and train the model over the masked feature vectors. Since the visual features extracted using ResNet-101 may also contain some redundant information, masking lets the network explore the intra-feature relationships to extract the maximum information and predict the correct attribute. Additionally, The graphical structure allows to share the knowledge and exploit the inter-class relationships. Table 2 shows a significant performance gain when a small fraction of visual features are randomly masked. Moreover, there is no significant advantage when we increase the number of masked features from 10 to 20. Therefore, all the experiments are conducted with 10 masked elements in visual feature vector. The performance is further supplemented by reduced confirmation bias when trained with proposed loss function. The empirical studies on the confirmation bias follows in the next section.

Analysing the robustness We study the robust accuracy of our approach under two adversarial attacks, including FGSM [5] and PGD-20 [11] attacks. For this, the visual features are subjected to these attacks for different perturbation budgets, and the accuracy of classifier is then observed to evaluate the robustness. Figures 2 (a)-(c) and Figures 2 (d)-(f) compare the performances under FGSM and PGD-20 attacks, respectively. It can be observed that the model trained with loss function 6 outperforms by a relatively large margin supporting our claims in Theorem 3.



(a) AWA2@$\beta = 0$ (b) SUN@$\beta = 0$ (c) AWA2@$\beta = 0.5$ (d) SUN@$\beta = 0.5$

Fig. 3. Comparison of categorical distribution learnt by classifier and visual-to-attribute mapper when trained for $\beta = \{0.5, 0\}$

Table 3. Top-1 accuracies on three datasets for different values of β. The best results are obtained when both terms in the loss function are given equal weightage.

β	0	0.1	0.2	0.4	0.5	0.6	0.8	1
AwA2	52.7	94.8	94.0	95.1	96.4	94.9	94.3	91.1
SUN	71.3	71.1	71.7	72.2	75.2	69.2	68.8	64.9
CUB	65.8	70.2	70.6	71.2	77.2	72.7	71.9	62.2

Mitigating Confirmation Bias To further support our claims on Theorem 2, we ablate the value of β to show that minimizing both $-\mathbb{E}_{x\sim\mathcal{X}^u}P(\psi \circ f_a|x;\theta)]$ and $-\mathbb{E}_{\hat{x}\sim\mathcal{G}}P(y|\hat{x};\theta)$ enhances the overall accuracy of the trained classifier. Table 3 shows the accuracies of the classifier when trained with different values of classifier. It is clearly evident that the overall performance increases significantly when both losses are minimized with equal weightage. An intuitive explanation is that when these losses are assigned unequal weights, one of the losses reduces faster than other resulting in imbalance in their accuracy. Additionally, Figure 3 illustrates the histogram distribution of the labels predicted by the classifier and visual-to-attribute mapper (f_a). It can be observed that there is large disagreement between the classes assigned by f_a and the classifier for a given visual feature. Such conditions result in higher uncertainty due to which classifier converges to sub-optimal solution. Furthermore, the histogram distribution are much better aligned for $\beta = 0.5$ showing consistency in the behaviour of both modules.

6 Conclusions

In this work, we proposed an approach for transductive ZSL to tackle the mismatch in intrinsic dimensionality during bidirectional domain alignment. In addition, we highlighted the confirmation or uncertainty bias that were prevalent while training the classifier and compensated it through entangled loss function, and then theoretically and empirically demonstrated the advantage of the proposed loss function in terms of adversarial robustness. This approach results in notable improvement in overall performance as compared to Bi-VAEGAN.

References

1. David, S.B., Blitzer, S., Crammer, K., Kulesza, A., Vaughan, J.W.: A theory of learning from different domains. Machine Learning **79(1-2)**, 151–175 (2010)
2. Felix, R., Vijay Kumar, B., Reid, I., Carneiro, G.: Multi-modal cycle-consistent generalized zero-shot learning. In: Computer Vision – ECCV 2018. ECCV 2018. Lecture Notes in Computer Science (2018)
3. Fu, Y., Hospedales, T.M., Xiang, T., Gong, S.: Transductive multi-view zero-shot learning. IEEE Trans. Pattern Anal. Mach. Intell. **37**(11), 2332–2345 (2015)
4. Gao, R., Hou, X., Qin, J., Chen, J., Liu, L., Zhu, F., Zhang, Z., Shao, L.: Zero-vae-gan: Generating unseen features for generalized and transductive zero-shot learning. IEEE Trans. Image Process. **29**, 3665–3680 (2020)
5. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: International Conference on Learning Representations, ICLR (2015)
6. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 30 (2017)

7. Hu, Y., et al.: Graph-based visual-semantic entanglement network for zero-shot image recognition. IEEE Trans. Multimedia **24**, 2473–2487 (2022)
8. Kong, X., et al.: En-compactness: Self-distillation embedding & contrastive generation for generalized zero-shot learning. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 9296–9305 (2022)
9. Liu, F., Xu, M., Li, G., Pei, J., Shi, L., Zhao, R.: Adversarial symmetric gans: Bridging adversarial samples and adversarial networks. Neural Netw. **133**, 148–156 (2021)
10. Liu, L., Zhou, T., Long, G., Jiang, J., Zhang, C.: Attribute propagation network for graph zero-shot learning. In: AAAI (2020)
11. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. In: International Conference on Learning Representations, ICLR (2018)
12. Mall, U., Hariharan, B., Bala, K.: Field-guide-inspired zero-shot learning. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV). pp. 9526–9535 (2021)
13. Marmoreo, F., Cavazza, J., Murino, V.: Transductive zero-shot learning by decoupled feature generation. In: 2021 IEEE Winter Conference on Applications of Computer Vision (WACV). pp. 3108–3117 (2021)
14. Nayak, N.V., Bach, S.: Zero-shot learning with common sense knowledge graphs. Transactions on Machine Learning Research (2022)
15. Norouzi, M., et al.: Zero-shot learning by convex combination of semantic embeddings. In: arXiv:1312.5650 (2014)
16. Paleka, D., Sanyal, A.: A law of adversarial risk, interpolation, and label noise. In: The Eleventh International Conference on Learning Representations (2023)
17. Patterson, G., Hays, J.: Sun attribute database: Discovering, annotating, and recognizing scene attributes. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. pp. 2751–2758 (2012)
18. Raghunathan, A., Xie, S.M., Yang, F., Duchi, J., Liang, P.: Understanding and mitigating the tradeoff between robustness and accuracy. Proceedings of Machine Learning Research (2020)
19. Senel, L.K., Utlu, I., Yucesoy, V., Koc, A., Cukur, T.: Semantic structure and interpretability of word embeddings. IEEE/ACM Transactions on Audio, Speech, and Language Processing **26**(10), 1769–1779 (oct 2018)
20. Sinha, A.K., Manthira Moorthi, S., Dhar, D.: Nl-ffc: Non-local fast fourier convolution for image super resolution. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 466–475 (2022)
21. Sinha, A.K., S, M.M.: Lips-specformer: Non-linear interpolable transformer for spectral reconstruction using adjacent channel coupling. In: 34th British Machine Vision Conference 2023, BMVC 2023, Aberdeen, UK, November 20-24, 2023. BMVA (2023)
22. Vaswani, A., et al.: Attention is all you need (2023)
23. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: Caltech-ucsd bird. Tech. Rep. CNS-TR-2011-001, California Institute of Technology (2011)
24. Wang, Y., Zhang, H., Zhang, Z., Long, Y.: Asymmetric graph based zero shot learning. Multimedia Tools and Applications (2019)
25. Wang, Z., Hao, Y., Mu, T., Li, O., Wang, S., He, X.: Bi-directional distribution alignment for transductive zero-shot learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 19893–19902 (June 2023)

26. Wei, C., Shen, K., Chen, Y., Ma, T.: Theoretical analysis of self-training with deep networks on unlabeled data. In: International Conference on Learning Representations (2021), https://openreview.net/forum?id=rC8sJ4i6kaH
27. Xian, Y., Lampert, C.H., Schiele, B., Akata, Z.: Zero-shot learning-a comprehensive evaluation of the good, the bad and the ugly. IEEE Trans. Pattern Anal. Mach. Intell. **41**, 225–2265 (2018)
28. Xian, Y., Sharma, S., Schiele, B., Akata, Z.: F-vaegan-d2: A feature generating framework for any-shot learning. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10267–10276 (2019)
29. Xie1, G.S., et al.: Region graph embedding network for zero-shot learning. In: Europena Conference on Computer Vision (2020)
30. Xu, B., Zeng, Z., Lian, C., Ding, Z.: Semi-supervised low-rank semantics grouping for zero-shot learning. IEEE Trans. Image Process. **30**, 2207–2219 (2021)
31. Ye, M., Guo, Y.: Zero-shot classification with discriminative semantic representation learning. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5103–5111 (2017)
32. Zhao, A., Ding, M., Guan, J., Lu, Z., Xiang, T., Wen, J.R.: Domain-invariant projection learning for zero-shot recognition. In: Advances in Neural Information Processing Systems. vol. 31. Curran Associates, Inc. (2018)

Zero-Shot Underwater Gesture Recognition

Sandipan Sarma(✉), Gundameedi Sai Ram Mohan, Hariansh Sehgal, and Arijit Sur

Indian Institute of Technology Guwahati, Assam, India
{sandipan.sarma,m.gundameedi,s.hariansh,arijit}@iitg.ac.in

Abstract. Hand gesture recognition allows humans to interact with machines non-verbally, which has a huge application in underwater exploration using autonomous underwater vehicles. Recently, a new gesture-based language called CADDIAN has been devised for divers, and supervised learning methods have been applied to recognize the gestures with high accuracy. However, such methods fail when they encounter unseen gestures in real time. In this work, we advocate the need for zero-shot underwater gesture recognition (ZSUGR), where the objective is to train a model with visual samples of gestures from a few "seen" classes only and transfer the gained knowledge at test time to recognize semantically-similar unseen gesture classes as well. After discussing the problem and dataset-specific challenges, we propose new seen-unseen splits for gesture classes in CADDY dataset. Then, we present a two-stage framework, where a novel transformer learns strong visual gesture cues and feeds them to a conditional generative adversarial network that learns to mimic feature distribution. We use the trained generator as a feature synthesizer for unseen classes, enabling zero-shot learning. Extensive experiments demonstrate that our method outperforms the existing zero-shot techniques. We conclude by providing useful insights into our framework and suggesting directions for future research. The code is available at: https://github.com/sandipan211/ZSUGR.

Keywords: Underwater gesture recognition · Zero-shot learning · Autonomous underwater vehicles · Transformers · Cross-attention · Generative adversarial networks

1 Introduction

Despite our planet being 70% water, our knowledge regarding underwater ecosystems remains limited. A major reason behind this is the harsh underwater environment and human divers face trouble collecting data from ocean depths due to several concerns like increasing hydrostatic pressure and oxygen depletion. In the past few decades, humans have passed the baton to autonomous underwater vehicles (AUVs) that can assist marine experts in several ways by capturing underwater images/videos, detecting oil spillage, inspecting oil and natural gas

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 346–361, 2025.
https://doi.org/10.1007/978-3-031-78183-4_22

pipelines, and conducting bathymetric surveys – all while being well-resistant to the problems faced by human divers. As a result, underwater exploration has garnered a lot of attention nowadays, with applications in oceanography, marine warfare, information navigation, and marine scene understanding.

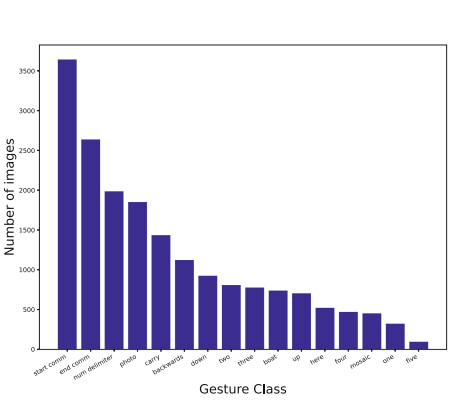
In many underwater missions, AUVs are accompanied by human divers who communicate non-verbally via different gestures. However, underwater gesture recognition is a relatively under explored area in computer vision owing to the lack of annotated datasets. Recently, a new gesture-based communication language called CADDIAN [5] was developed, and an image-based underwater gesture recognition dataset named CADDY [8] (Fig. 1b) was constructed as an effort to facilitate vision research. However, there are primarily two challenges. Firstly, underwater images suffer from problems like low contrast, haziness, color distortion, and blurriness. As a result, traditional gesture recognition methods face problems in analyzing them. Secondly, existing gesture recognition models are predominantly supervised [2,18,20,32–34] and can only recognize gestures from a predefined set used to train the models, failing to interpret "unseen" gestures. This is an inevitable bottleneck as it is impossible to collect thousands of labeled images for every possible gesture that might be used by human divers in the wild. For instance, a new gesture meant to indicate low oxygen levels due to a real-time accident might not be interpreted by an AUV equipped with a supervised gesture recognition model, putting the human diver at risk.

Zero-shot learning (ZSL) has come to the limelight in recent years to alleviate such data scarcity, which mimics the human tendency to learn from other modalities (*semantics* or class attributes) like text or audio in the absence of visual examples. The objective of ZSL is to transfer knowledge about a few "seen" concepts/classes via a visual-semantic association and recognize unseen ones. Although a few recent works explore zero-shot gesture recognition, it is still unexplored in underwater scenarios.

In this paper, we introduce and study the task of *zero-shot underwater gesture recognition* (ZSUGR). Being the first work of its kind, standard seen-unseen zero-shot splits are currently unavailable. Therefore, we propose three new splits by designating seen and unseen classes randomly, following previous splitting strategies [14,31]. Considering the challenges in extracting strong visual features from underwater images and the highly class-imbalanced nature of the CADDY dataset (Fig. 1a), a two-stage framework is devised consisting of a novel transformer and a generative adversarial network. In the first stage, we design a novel Gated Cross-Attention Transformer (GCAT) which is responsible for a strong representation learning. Visual features extracted from a pretrained ResNet-50 are passed to an encoder, and its outputs are transformed into powerful gesture representations by our novel gesture decoder. The trained GCAT is then used as a visual feature extractor for the seen classes. A conditional Wasserstein GAN [1] is then trained with these visual features, with seen class semantic vectors as class conditional variables. Here, we rely on the text encoder of a pretrained visual-language model called CLIP [23] to obtain the semantic vector of a gesture class. Visual features of unseen gestures can then be synthesized using

our trained WGAN, enabling us to train a gesture classifier with data from both seen and unseen classes, mitigating the bias problem in zero-shot settings [22]. Extensive experiments are conducted, showing the failure of supervised models, our model's improved performance compared to state-of-the-art zero-shot classification methods in conventional and generalized zero-shot settings (inspired by previous zero-shot gesture recognition works [14,17,29]), and demonstrating the role of improved visual representations for ZSUGR. To summarize, we make the following contributions in this paper:

- We introduce the problem of zero-shot *underwater* gesture recognition from images for the first time and discuss its real-world applications.
- Seen-unseen splits of gesture classes are proposed for the CADDY dataset for zero-shot training and model evaluation.
- A two-stage network is proposed for ZSUGR consisting of a novel transformer that can be trained as a strong underwater visual feature extractor. The obtained features can be fed to a class-conditional generative adversarial network that can learn to synthesize visual features of unseen gestures.



(a) Distribution of gesture classes

(b) Examples of the 16 gestures

Fig. 1. Properties of the CADDY dataset [8].

2 Related Works

2.1 Hand Gesture Recognition (HGR)

Vision-based HGR methods can be broadly categorized into dynamic (limb movements indicate gesture) and static (hand shape and finger orientations

indicate gesture). Among these, segmenting the hand using skin color models to extract visual gesture features is a common practice [26,35]. With the availability of neural networks pretrained on large-scale datasets, transfer learning approaches [28] have also been successful. However, the extracted visual features are affected by factors such as lighting conditions and background clutter. To enable HGR frameworks to recognize a wider range of gestures, 3D methods are explored, where several sensors and cameras are used to obtain temporal volumes [13] or skeletal joints [6] to analyze the hand shapes.

2.2 Zero-Shot Learning (ZSL)

Image classification remains the most researched vision task exploiting ZSL. The first seen-unseen splits were formally provided for five datasets in the seminal work of Xian et al. [30], who also popularized the concept of *conventional* (only unseen class samples at test time) and *generalized* (samples from both seen and unseen classes at test time) zero-shot learning settings. Meanwhile, gesture recognition as a vision task is less explored. One of the earliest works in zero-shot gesture recognition (ZSGR) unified coordinated natural language, gesture, and context [27] to facilitate human-robot interactions. However, ZSGR was standardized in the work of Madapana and Wachs [15], who used gestures from two gesture-based datasets to define a total of 13 seen and 8 unseen classes. Furthermore, they provided a list of 13 high-level semantic descriptors to characterize each gesture class. Their zero-shot classifiers were inspired by previous image classification methods [12,24]. In an extended work, they further introduced hard zero-shot gesture recognition [17] where there are only a few visual samples for the seen classes as well. Recently in another work, they introduced a gesture attribute dataset [16] with segmented skeletal data and proposed a joint semantic encoder optimizing reconstruction, semantic, and classification losses [14]. Another novel dataset was built by Wu et al. [28] with 16 seen and 9 unseen classes, and then they proposed an end-to-end prototype learning framework [29].

2.3 Underwater Diver Gesture Recognition

Since the arrival of the CADDY dataset [8], several works have been undertaken in the last four years in underwater gesture recognition (UGR). The first such work [32] used pretrained convolutional networks like ResNet, GoogleNet, and others for transfer learning. Several other works that followed [2,18,20] compared the performance of classical and deep learning methods for UGR. On the other hand, DARE [33] demonstrated a hierarchical tree-structured classification scheme. Recently, VT-UHGR [34] used a pretrained ViT as the visual encoder and a pretrained BERT as the text encoder for multimodal underwater gesture feature learning. Data augmentation using generative methods has also been used to improve UGR performance. However, most of these methods use

transfer learning and do not provide any novel architecture for UGR. Additionally, none of them are zero-shot learners. This paper presents the first work that addresses both of these concerns.

3 Methodology

3.1 Problem Definition

Let we have a training set $\mathcal{D}_{train} = \{(x_i, y_i) | \ x_i \in \mathcal{X}_{seen}, y_i \in \mathcal{S}\}$, where x_i is an image from a seen class y_i. A separate set of unseen data $\mathcal{D}_{novel} = \{(x_j, y_j) | \ x_j \in \mathcal{X}_{unseen}, y_j \in \mathcal{U}\}$ is given such that the sets of seen and unseen classes are disjoint, i.e., $\mathcal{S} \cap \mathcal{U} = \phi$. Additionally, we have a semantic vector $a_i \in \mathbb{R}^{512}$ for each gesture class $y \in \mathcal{S} \cup \mathcal{U}$. We work in the more realistic inductive zero-shot setting [30] – where visual examples of unseen classes are unavailable during training – instead of the transductive setting. Then, the task in conventional zero-shot learning is to learn a classifier $f_{zsl} : \mathcal{X}_{unseen} \rightarrow \mathcal{U}$. For generalized zero-shot learning, a small subset of $\mathcal{X}_{seen}$ ($\mathcal{X}^{sub}_{seen}$) is extracted as the set of seen samples at test time, following the suggestions in the benchmark ZSL paper [30]. The objective then changes to learning a classifier $f_{gzsl} : \mathcal{X}^{sub}_{seen} \cup \mathcal{X}_{unseen} \rightarrow \mathcal{S} \cup \mathcal{U}$.

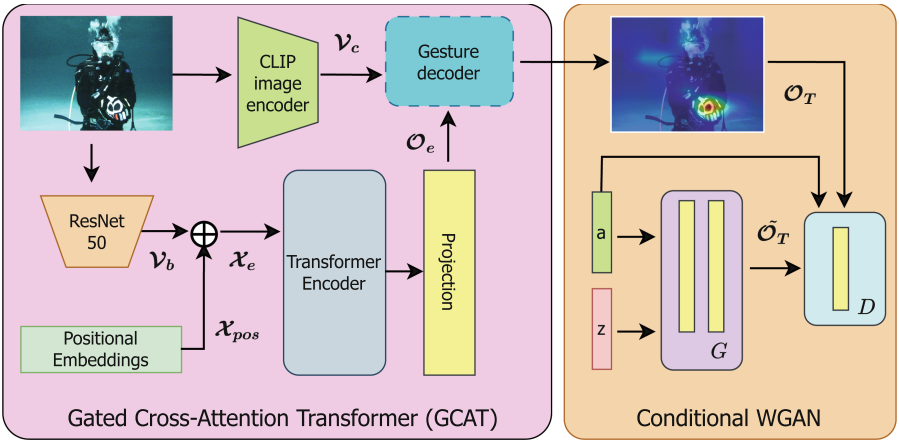


Fig. 2. Proposed two-stage framework for ZSUGR. Here, z denotes a random noise vector, and a denotes semantic vector of a gesture class.

3.2 System Overview

Figure 2 shows our overall two-stage framework, which includes a novel transformer and a Wasserstein GAN (WGAN). Given an underwater image $\mathcal{X} \in \mathbb{R}^{C \times H \times W}$, its visual feature map $\boldsymbol{\mathcal{V}_b} \in \mathbb{R}^{C' \times H' \times W'}$ is first extracted using a

pretrained ResNet-50 backbone. Since a transformer encoder considers each spatial location in $\boldsymbol{\mathcal{V}_b}$ as a token while computing self-attention, it is necessary to keep track of the token positions. We construct positional embeddings $\boldsymbol{\mathcal{X}_{pos}} \in \mathbb{R}^{C' \times H' \times W'}$, add them element-wise with $\boldsymbol{\mathcal{V}_b}$, and pass the resulting feature map to the transformer encoder E. The outputs from E are then decoded via our novel gated cross-attention mechanism guided by powerful visual representations from a pre-trained CLIP. These decoded visual gesture features are used as "real features" for training a WGAN, conditioned on class semantics. The trained WGAN can be used to generate visual features corresponding to unseen gesture classes. A classifier $\Phi_{cls}(.)$ is trained with seen class data and the synthesized unseen class data for zero-shot gesture recognition. We discuss the novel GCAT in Sec. 3.3, followed by the WGAN in Sec. 3.4.

3.3 Gated Cross-Attention Transformer (GCAT)

As mentioned in Sec. 3.2, the encoder E takes the input:

$$\boldsymbol{\mathcal{X}_e} = \boldsymbol{\mathcal{V}_b} + \boldsymbol{\mathcal{X}_{pos}} \tag{1}$$

and performs a self-attention operation on the visual tokens. The output $\boldsymbol{\mathcal{O}_e} \in \mathbb{R}^{C' \times H'W'}$ contains image-wide contextual information that is passed to our gesture decoder *Dec*. Meanwhile, we extract additional visual features $\boldsymbol{\mathcal{V}_c} \in \mathbb{R}^{C'' \times k}$ from a pretrained CLIP image encoder and use them to refine the knowledge coming from the encoder, yielding gesture features. To this end, $\boldsymbol{\mathcal{O}_e}$ is first projected to CLIP dimension, and then a two-branch cross-attention mechanism is proposed. The two branches perform identical operations, with the only difference being in the query (Q), key (K), and value (V) assignments. The branch assignments (Fig. 3) are as follows:

$$Q_L = \boldsymbol{\mathcal{O}_e}, \quad K_L = V_L = \boldsymbol{\mathcal{V}_c} \tag{2}$$

$$Q_R = \boldsymbol{\mathcal{V}_c}, \quad K_R = V_R = \boldsymbol{\mathcal{O}_e} \tag{3}$$

where L and R denote left and right branches. The attention outputs are normalized using layer normalization (LN) operation and added to the branch queries:

$$A_L = LN(Q_L + CrossAtt(Q_L, K_L, V_L)) \tag{4}$$

$$A_R = LN(Q_R + CrossAtt(Q_R, K_R, V_R)) \tag{5}$$

Now, since visual inputs to *Dec* come from two sources – $\boldsymbol{\mathcal{O}_e}$ based on ResNet features and $\boldsymbol{\mathcal{V}_c}$ based on CLIP features – it is beneficial to weight the attention outputs for understanding the contribution of the encoder outputs as well as CLIP context. Consequently, we propose a gating mechanism where we apply a 1×1 convolution first to the attention outputs of each branch and downscale them by a factor of 2. A GELU activation [11] finally produces the gated values. The gating function can hence be formulated as:

$$g(A_L) = GELU(\boldsymbol{w_L} A_L + \boldsymbol{b_L}) \tag{6}$$

$$g(A_R) = GELU(\boldsymbol{w_R} A_R + \boldsymbol{b_R}) \tag{7}$$

where $(\boldsymbol{w_L}, \boldsymbol{b_L})$ and $(\boldsymbol{w_R}, \boldsymbol{b_R})$ denote the weights and biases of the convolutional layer in the left and right branches respectively. The gated outputs indicate how much cross-attention information should be preserved and how much can be learned subsequently. The attention maps from the two branches are fused together, and the source knowledge from the encoder is refined as:

$$\boldsymbol{\mathcal{O}_e} = \boldsymbol{\mathcal{O}_e} \circ (A_L + A_R) \tag{8}$$

where $\circ$ denotes Hadamard product. It is then passed to a feedforward network (FFN), as shown in Fig. 3, to produce the final feature map $\boldsymbol{\mathcal{O}_T}$ containing gesture cues. A channel-wise mean is taken at the end such that $\boldsymbol{\mathcal{O}_T} \in \mathbb{R}^d$.

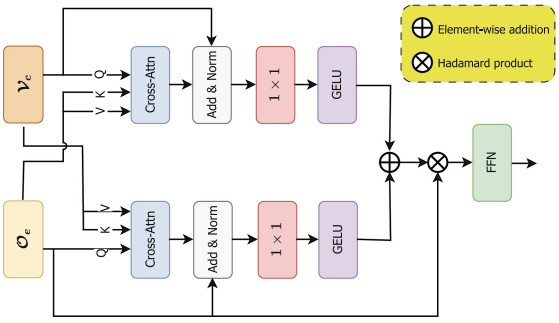


Fig. 3. Architecture of our novel GCAT gesture decoder.

3.4 Transformed-Feature Generating Network

Due to problems like color distortion and haziness that underwater images suffer from, it becomes exceedingly difficult to work in extreme scenarios like zero-shot in the absence of visual images of unseen classes. Moreover, predictions may get highly biased toward the seen classes owing to the imbalanced nature of underwater datasets like CADDY. To alleviate the seen-bias problem, we employ a conditional Wasserstein Generative Adversarial Network (c-WGAN) [1]. As depicted in Fig. 2, it takes the visual gesture features $\boldsymbol{\mathcal{O}_T}$ corresponding to seen classes as input and their semantic vectors (a) as a conditional variable. The generator G attempts to mimic the true distribution of the visual features $\boldsymbol{\mathcal{O}_T}$ while being discriminative in the visual space with guidance received from the class semantics. It generates seen class features $\tilde{\boldsymbol{\mathcal{O}}}_T$, which are then fed to a critic network D along with features from true distribution $\boldsymbol{\mathcal{O}_T}$. D acts as a binary classifier that scores the "realness" of the input data (either coming from the mimicked feature distribution or from the true distribution). During backpropagation, D tries to improve its predictions, whereas G tries to improve

itself based on the feedback from D and synthesizes features closer to the true distribution. For optimization, we use the Wasserstein loss:

$$\mathcal{L}_{\texttt{WGAN}} = \mathbb{E}[D(\boldsymbol{\mathcal{O}_T}, a)] - \mathbb{E}[D(\tilde{\boldsymbol{\mathcal{O}}_T}, a)] + \lambda \mathbb{E}[(||\nabla_{\hat{\boldsymbol{\mathcal{O}}_T}} D(\hat{\boldsymbol{\mathcal{O}}_T}, a)||_2 - 1)^2] \quad (9)$$

where the first two terms represent the critic loss in WGAN and the third term represents a gradient penalty [9], with λ being the penalty coefficient. $\tilde{\boldsymbol{\mathcal{O}}_T} = G(z, a)$ denotes the generated feature, and $\hat{\boldsymbol{\mathcal{O}}_T} = \rho\, \boldsymbol{\mathcal{O}_T} + (1 - \rho)\, \tilde{\boldsymbol{\mathcal{O}}_T}$, with $\rho \sim U(0, 1)$. Additionally, we use a mode-seeking loss [19] to mitigate the problem of mode-collapse in GANs:

$$\mathcal{L}_{\texttt{MS}} = \mathbb{E}[||G(z_1, a) - G(z_2, a)||_1 / ||z_1 - z_2||_1] \quad (10)$$

Here, z_1 and z_2 correspond to two noise vectors that produce different visual features. The idea is to regularize that the generated features should get sampled corresponding to noise vectors in the learned distribution that are far apart from each other, giving rise to discriminative synthetic visual features.

Table 1. Details of proposed seen-unseen splits of the CADDY dataset.

Split	Seen classes	Unseen classes	$\mathcal{X}_{seen}$	$\mathcal{X}^{sub}_{seen}$	$\mathcal{X}_{unseen}$
1	10	6	12505	1389	4584
2	10	6	10251	1139	7088
3	10	6	11713	1301	5464

3.5 Training and Inference

We follow a two-stage training process, as shown in Fig. 2. In the first stage, we train the transformer GCAT to produce visual gesture features. We feed them to a classifier Φ_c whose weights are initialized by the text embeddings (or semantics) of the seen classes. For a class y_i, we obtain its semantic vector by converting the class name to "A photo of a diver gesturing $[y_i]$" via a prompt template $\rho_g(.)$, and then using a pretrained CLIP text encoder [23] $\psi(.)$ as:

$$\boldsymbol{a_i} = \psi(\rho_g(y_i)) \quad \forall\, y_i \in \mathcal{S} \cup \mathcal{U} \quad (11)$$

We minimize a cross-entropy loss ($\mathcal{L}_{\texttt{CE}}$) between the predicted and ground-truth gesture labels, strengthening the visual feature representations for seen gestures.

The trained GCAT is used in the next stage as a feature extractor for underwater images. With the extracted features (seen data), the c-WGAN is trained to synthesize seen data by minimizing the loss (with hyperparameter α):

$$\min_{G} \max_{D} \quad \mathcal{L}_{\texttt{WGAN}} + \alpha\, \mathcal{L}_{\texttt{MS}} \quad (12)$$

Given unseen class semantic vectors, the trained GAN can then be used to generate unseen class visual features. The synthesized unseen features and the visual features for seen classes extracted from the trained GCAT can then be combined and used to train a linear softmax classifier $\Phi_{cls}(.)$ for zero-shot prediction. At test time, visual gesture features of an image are first extracted using GCAT, and the trained classifier $\Phi_{cls}(.)$ predicts the gesture class.

Table 2. Top-1 accuracy (in %) reported for CZSL and GZSL settings. H denotes harmonic mean of seen and unseen accuracy in GZSL. Best results are in **bold**. Codes for CZSL training and evaluation were not provided by DGZ [3].

Method	U_{czsl}	S_{gzsl}	U_{gzsl}	H
TFVAEGAN [21]	41.50 ± 5.47	79.57 ± 13.11	13.49 ± 5.28	22.51 ± 7.40
CNZSL [25]	16.72 ± 0.08	20.27 ± 3.84	11.88 ± 2.78	14.97 ± 3.23
FREE [4]	15.83 ± 3.95	84.88 ± 9.03	14.55 ± 3.36	24.61 ± 4.50
CE-GZSL [10]	39.89 ± 6.49	$\mathbf{94.11 \pm 0.55}$	2.58 ± 0.45	5.01 ± 0.87
DGZ [3]	-	57.89 ± 2.90	15.72 ± 3.32	24.62 ± 4.14
Ours	$\mathbf{45.91 \pm 4.71}$	61.93 ± 5.71	$\mathbf{20.03 \pm 7.14}$	$\mathbf{29.53 \pm 7.06}$

4 Experiments

4.1 Dataset

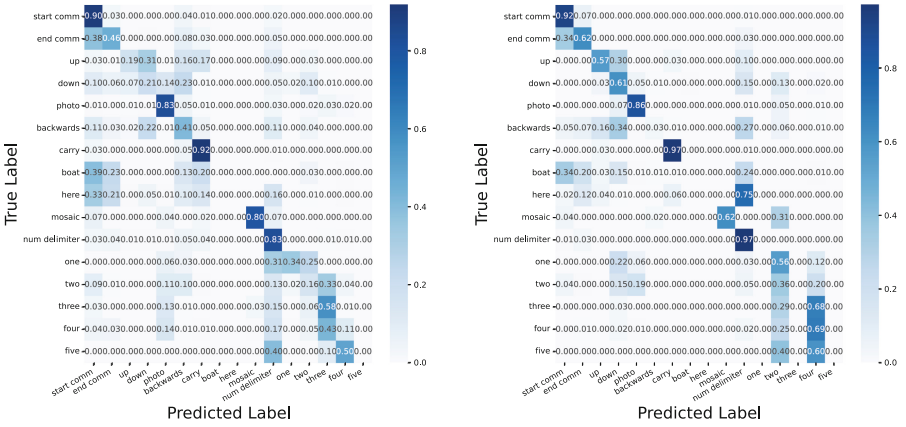
All our experiments are conducted on the CADDY underwater stereo-vision dataset [8], one of the largest publicly available underwater image datasets and the only one for diver gesture recognition. Overall, 18,478 labeled diver gesture images are present in CADDY, belonging to 16 different gesture classes (Fig. 1b). The gestures used by divers correspond to the CADDIAN sign language [5]. Notably, the dataset is highly imbalanced with a long-tail distribution (Fig. 1a), making it very challenging for zero-shot learning.

4.2 Zero-Shot Splits and Evaluation Protocols

Due to the absence of any previous work on ZSUGR, we define new splits of the CADDY dataset, marking the classes to be used as seen/unseen for zero-shot models. In previously undertaken zero-shot vision applications, seen-unseen splits are proposed following two approaches. Some works designate fixed sets of seen/unseen classes [30] for a given dataset. On the other hand, some works randomly split the dataset classes into seen and unseen sets at a 50%/50% proportion, and results are reported over multiple such random splits [31]. We take this second approach to eliminate human bias in picking the seen-unseen split. However, owing to highly imbalanced data, we propose to pick 10 seen and

6 unseen classes randomly from the dataset, and three such splits are obtained. Table 1 provides our split details.

For model evaluation, we follow the benchmark protocol for zero-shot classification [30]. For conventional zero-shot (**CZSL**) evaluation, we report the top-1 accuracy for the unseen classes ($\boldsymbol{U_{czsl}}$). In the case of generalized evaluation (**GZSL**), we compute the top-1 accuracy for both seen ($\boldsymbol{S_{gzsl}}$) and unseen ($\boldsymbol{U_{gzsl}}$) classes and report their harmonic mean ($\boldsymbol{H}$). The harmonic mean encourages the model to perform well on both seen and unseen classes, which is a more realistic metric for zero-shot settings. Since we have multiple random splits, we indicate the mean and standard deviation of obtained accuracy across the three random splits for all the reported results.



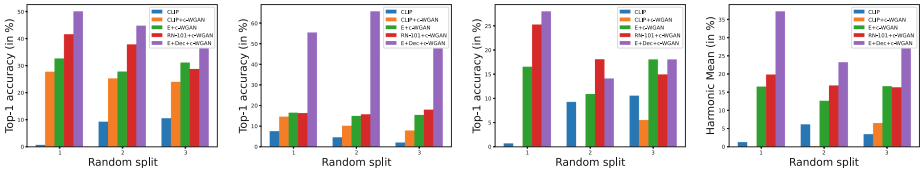
(a) DGZ [3] (b) Ours

Fig. 4. Comparison of our GZSL confusion matrix with the state-of-the-art.

4.3 Implementation Details

All the experiments are undertaken in PyTorch using a single NVIDIA A100 GPU. The raw underwater images are resized to $3 \times 224 \times 224$. We used the ResNet-50 backbone to extract a feature map $\boldsymbol{\mathcal{V}_b}$ of $256 \times 7 \times 7$. Our encoder E contains the same layers as the traditional transformer, consisting of a self-attention mechanism, followed by layer normalization and neural layers. The final encoder output is obtained after 3 blocks of encoder layers. We use 3 blocks of decoder layers, and the final decoder output $\boldsymbol{\mathcal{O}_T} \in \mathbb{R}^{(7\times7)\times512}$ is averaged across the feature maps to yield a 512-dimensional visual gesture feature. The transformer GCAT is optimized using AdamW with a learning rate of 1e-5 and weight decay of 1e-4. The output from the CLIP image encoder $\boldsymbol{\mathcal{V}_c}$ used for refining the encoder features is a feature map of dimension 768×50. As for the

CLIP text encoder, the output embeddings are 512-dimensional vectors used as semantic vectors for every class. In our c-WGAN, the generator (G) and critic (D) networks are essentially two multi-layer perceptions, both optimized using the Adam optimizer with a learning rate of 1e-4. In $\mathcal{L}_{\texttt{MS}}$, we use $\alpha = 1e-4$.



(a) $\boldsymbol{U_{czsl}}$ (b) $\boldsymbol{S_{gzsl}}$ (c) $\boldsymbol{U_{gzsl}}$ (d) $\boldsymbol{H}$

Fig. 5. Component analysis of our framework for the three proposed random splits. Top-1 accuracy is reported in CZSL ($\boldsymbol{U_{czsl}}$) and GZSL settings ($\boldsymbol{S_{gzsl}}$ and $\boldsymbol{U_{gzsl}}$ with harmonic mean $\boldsymbol{H}$). E = GCAT encoder, RN-101 = ResNet-101 as feature extractor, Dec = GCAT decoder, c-WGAN = Conditional WGAN.

4.4 Zero-Shot Results

Since there are no prior works in ZSUGR, we adapted a few zero-shot image classification methods for the ZSUGR task, following previous works [14,17,29]. For a fair comparison, we train these methods with the same semantic vectors as ours. We evaluate these models in both CZSL and GZSL settings and report their mean and standard deviations over the three random splits we proposed.

Conventional Setting (CZSL). DGZ [3] did not provide their implementation for CZSL evaluation. CZSL results for some other methods are reported in Tab. 2. Our method outperforms all of them by a good margin with an average top-1 accuracy of $45.91 \pm 4.71\%$.

Generalized Setting (GZSL). Table 2 suggests that CE-GZSL [10] is extremely capable of recognizing seen gestures during evaluation with an accuracy of $94.11 \pm 0.55\%$ but performs very poorly in case of unseen gesture classes with an accuracy of $2.58 \pm 0.45\%$, yielding low harmonic mean. This indicates that CE-GZSL has low generalizability in ZSUGR for the CADDY dataset, although it exhibited state-of-the-art (SOTA) generalizability in image classification. The other methods suffer from the same issue as well. One of the reasons behind such low generalization ability could be the class imbalance problem in CADDY, which is not as extreme in the image classification datasets on which these methods are usually evaluated. Moreover, the zero-shot methods usually suffer from the bias problem [22], where predictions for unseen class samples are incorrectly biased towards a similar seen class. Together, the bias problem and class

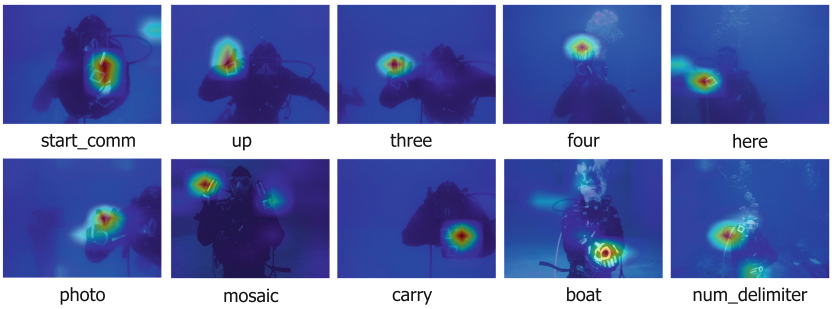


Fig. 6. Attention visualization of GCAT gesture decoder for ten gestures.

imbalance make ZSUGR a challenging task for the SOTA zero-shot classification methods. Meanwhile, our method seemingly finds a better balance between seen and unseen class knowledge. Additionally, it deals with the bias problem better than the other generative methods like TFVAEGAN [21], FREE [4], and DGZ [3], indicating that our GAN has had superior training with gesture features extracted from the novel GCAT, as compared to the ResNet-101 features with which the GANs of other methods are trained. Consequently, we obtain the best harmonic mean of $29.53 \pm 7.06\%$ (Tab. 2). For class-wise comparison, we plot confusion matrices in Fig. 4 to demonstrate how the SOTA classification method DGZ [3] fares against our method.

Table 3. Effect of activation function on top-1 accuracy (in %) during CZSL and GZSL evaluation. H denotes the harmonic mean of seen and unseen accuracy.

Activation	U_{czsl}	S_{gzsl}	U_{gzsl}	H
GELU	45.16 ± 4.64	61.93 ± 5.71	20.03 ± 7.14	29.53 ± 7.06
ELU	44.66 ± 8.11	59.50 ± 12.11	18.05 ± 5.94	26.84 ± 6.54
RELU	45.38 ± 8.74	65.48 ± 2.03	14.73 ± 4.17	23.86 ± 6.65
Sigmoid	40.71 ± 5.25	36.50 ± 5.52	30.02 ± 2.81	32.67 ± 2.15
SiLU	47.71 ± 3.75	65.65 ± 5.09	15.46 ± 0.74	25.03 ± 1.24

Qualitative Analysis. Fig. 6 shows decoder attention visualizations highlighting where our GCAT gesture decoder is focusing at test time while extracting visual gesture features. We observe that our trained GCAT focuses on the hands of the divers, resulting in highly relevant visual features extracted for ZSUGR. Specifically, for relevant gestures like *three*, *photo*, and *four*, the decoder focuses on the fingers, which are more informative. On the other hand, it shifts its focus to both hands whenever relevant, whether the hands are close to each other (like *boat* and *carry*) or distant (such as *mosaic*).

Plight of Supervised Models. Previous works [18,20,32] have shown that pretrained CNN models can be easily adapted to underwater gesture recognition via transfer learning on CADDY. In Tab. 4, we mention the supervised gesture recognition results for CADDY as reported by previous works, with the best accuracy reaching 98% using ResNet-18. We employ these models in the GZSL setting to investigate their performance in recognizing unseen gestures. To this end, we obtain visual features from these pretrained CNNs via transfer learning and compute a cosine similarity with the CLIP semantic vectors to get zero-shot predictions. It can be observed from Tab. 4 that their performance drops for both seen and unseen classes. Specifically, they fail miserably in recognizing unseen gestures, with a best harmonic mean of $2.45 \pm 4.18\%$ achieved using MobileNet-v3. On the contrary, our model achieves a harmonic mean of $29.53 \pm 7.06\%$. Hence, the need for zero-shot models for underwater gesture recognition is evident as the best-performing supervised models are insufficient.

4.5 Ablation Studies

Component Analysis. The baseline zero-shot performance of CLIP is extremely underwhelming, as can be inferred from Fig. 5. Moreover, using raw visual features extracted from CLIP to train the c-WGAN improves CZSL performance to an extent but hurts generalizability, as the GZSL results demonstrate. This reinforces the need for a task-specific framework for ZSUGR and justifies the necessity of our work. To validate the effect of each component of our framework, we conduct an ablation study on CADDY in both CZSL and GZSL settings with the following changes to our framework: (i) removing the novel GCAT gesture decoder and using only the vanilla encoder to get visual features and (ii) removing the GCAT module and using pretrained ResNet-101 visual features to train our c-WGAN. Fig. 5 suggests that our decoder significantly improves performance in both CZSL and GZSL settings across all three random splits. This justifies refining the visual features extracted by the vanilla encoder using our novel decoder, which makes GCAT attend to more critical and discriminative regions of the diver's hand within the image. On the contrary, the raw features from ResNet-101 or the vanilla encoder alone are less effective.

Impact of Activation Functions. In Tab. 3, we study the choice of activation function in the architecture of our novel gesture decoder while training the GCAT. In the CZSL scenario, we see comparable performance of the five different activation functions we studied. Interestingly, in the GZSL setting, we find that different activations have varying impacts on recognizing seen and unseen gestures. For example, the sigmoid function generalizes well, but seen accuracy is low. On the other hand, using SiLU [7] degrades unseen accuracy. The activation we used finally, GELU, performs well in both CZSL and GZSL settings. In summary, it is evident that the activation functions can dictate the quality of visual gesture features which can, in turn, hamper a generative method's ability to generate discriminative visual features for seen and unseen classes.

Table 4. A performance comparison of pretrained CNN models in the supervised (done by previous works) and generalized zero-shot settings (shown by us). Top-1 accuracy (in %) is reported for supervised setting, whereas top-1 accuracy for seen and unseen classes are reported over our proposed splits for GZSL.

Pretrained CNN	Supervised	S_{gzsl}	U_{gzsl}	H
AlexNet [32]	82.89	71.75 ± 2.19	0.76 ± 1.21	1.48 ± 2.35
VGG-16 [32]	95.00	76.20 ± 1.23	0.42 ± 0.57	0.84 ± 1.12
ResNet-18 [18]	98.00	53.54 ± 3.77	0.72 ± 1.25	1.38 ± 2.39
ResNet-50 [20]	97.06	61.94 ± 2.86	0.79 ± 1.27	1.53 ± 2.44
GoogleNet [32]	90.08	53.19 ± 3.46	0.8 ± 1.39	1.53 ± 2.64
MobileNet-v3 [18]	84.32	62.78 ± 5.21	1.31 ± 2.23	2.45 ± 4.18

5 Conclusion

We introduced the task of zero-shot underwater gesture recognition in this paper and discussed its potential for diver-AUV communications. We proposed a two-stage framework consisting of a novel transformer that learns strong visual gesture representations and a conditional generative network that learns the distributions of these features. From the dataset perspective, we proposed three random seen-unseen splits of the CADDY dataset and reported our results in both conventional and generalized zero-shot settings, comparing them with state-of-the-art classification methods. Through extensive experimentations and ablations, we discussed how poorly supervised methods perform in zero-shot settings, why existing zero-shot classification methods suffer so much from class imbalance and bias problems, and how activation functions may impact transformers for gesture recognition. One of the areas for future research could be learning the semantic representations of gestures instead of using handcrafted prompt templates. We hope that this work can serve as a benchmark for the advancement of ZSUGR as a field since there is a huge room for improvement.

Acknowledgements. This work is supported by the Indian Institute of Technology Guwahati (IITG) Technology Innovation & Development Foundation (TI&DF) as a part of the National Mission on Interdisciplinary Cyber-Physical Systems with financial assistance from the Department of Science and Technology, India, through grant number DST/NMICPS/TIH12/IITG/2020.

References

1. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: International conference on machine learning. pp. 214–223. PMLR (2017)
2. Chavez, A.G., Ranieri, A., Chiarella, D., Birk, A.: Underwater vision-based gesture recognition: A robustness validation for safe human-robot interaction. IEEE Robotics & Automation Magazine **28**(3), 67–78 (2021)

3. Chen, D., Shen, Y., Zhang, H., Torr, P.H.: Deconstructed generation-based zero-shot model. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37, pp. 295–303 (2023)
4. Chen, S., Wang, W., Xia, B., Peng, Q., You, X., Zheng, F., Shao, L.: Free: Feature refinement for generalized zero-shot learning. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 122–131 (2021)
5. Chiarella, D., Bibuli, M., Bruzzone, G., Caccia, M., Ranieri, A., Zereik, E., Marconi, L., Cutugno, P.: A novel gesture-based language for underwater human-robot interaction. Journal of Marine Science and Engineering **6**(3), 91 (2018)
6. Devineau, G., Moutarde, F., Xi, W., Yang, J.: Deep learning for hand gesture recognition on skeletal data. In: 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018). pp. 106–113. IEEE (2018)
7. Elfwing, S., Uchibe, E., Doya, K.: Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. Neural Netw. **107**, 3–11 (2018)
8. Gomez Chavez, A., Ranieri, A., Chiarella, D., Zereik, E., Babić, A., Birk, A.: Caddy underwater stereo-vision dataset for human-robot interaction (hri) in the context of diver activities. Journal of Marine Science and Engineering **7**(1), 16 (2019)
9. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. Advances in neural information processing systems **30** (2017)
10. Han, Z., Fu, Z., Chen, S., Yang, J.: Contrastive embedding for generalized zero-shot learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2371–2381 (2021)
11. Hendrycks, D., Gimpel, K.: Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415 (2016)
12. Lampert, C.H., Nickisch, H., Harmeling, S.: Attribute-based classification for zero-shot visual object categorization. PAMI **36**(3), 453–465 (2013)
13. Liang, B.: Gesture recognition using depth images. In: Proceedings of the 15th ACM on International conference on multimodal interaction. pp. 353–356 (2013)
14. Madapana, N.: Zero-shot learning for gesture recognition. In: Proceedings of the 2020 international conference on multimodal interaction. pp. 754–757 (2020)
15. Madapana, N., Wachs, J.: Zsgl: zero shot gestural learning. In: Proceedings of the 19th ACM International Conference on Multimodal Interaction. pp. 331–335 (2017)
16. Madapana, N., Wachs, J.: Database of gesture attributes: Zero shot learning for gesture recognition. In: 2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019). pp. 1–8. IEEE (2019)
17. Madapana, N., Wachs, J.P.: Hard zero shot learning for gesture recognition. In: 2018 24th international conference on pattern recognition (ICPR). pp. 3574–3579. IEEE (2018)
18. Mangalvedhekar, S., Nahar, S., Maskare, S., Mahajan, K., Bagade, A.: Underwater diver gesture recognition. In: 2023 International Conference on Electrical, Electronics, Communication and Computers (ELEXCOM). pp. 1–6. IEEE (2023)
19. Mao, Q., Lee, H.Y., Tseng, H.Y., Ma, S., Yang, M.H.: Mode seeking generative adversarial networks for diverse image synthesis. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1429–1437 (2019)
20. Martija, M.A.M., Dumbrique, J.I.S., Naval Jr, P.C.: Underwater gesture recognition using classical computer vision and deep learning techniques (2020)
21. Narayan, S., Gupta, A., Khan, F.S., Snoek, C.G.M., Shao, L.: Latent embedding feedback and discriminative features for zero-shot classification. In: ECCV. pp. 479–495 (2020)

22. Paul, A., Krishnan, N.C., Munjal, P.: Semantically aligned bias reducing zero shot learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7056–7065 (2019)
23. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021)
24. Romera-Paredes, B., Torr, P.H.S.: An embarrassingly simple approach to zero-shot learning. In: Proceedings of the 32nd International Conference on International Conference on Machine Learning. pp. 2152–2161 (2015)
25. Skorokhodov, I., Elhoseiny, M.: Class normalization for (continual)? generalized zero-shot learning. arXiv preprint arXiv:2006.11328 (2020)
26. Sun, J.H., Ji, T.T., Zhang, S.B., Yang, J.K., Ji, G.R.: Research on the hand gesture recognition based on deep learning. In: 2018 12th International symposium on antennas, propagation and EM theory (ISAPE). pp. 1–4. IEEE (2018)
27. Thomason, W., Knepper, R.A.: Recognizing unfamiliar gestures for human-robot interaction through zero-shot learning. In: 2016 International Symposium on Experimental Robotics. pp. 841–852. Springer (2017)
28. Wu, B.X., Yang, C.G., Zhong, J.P.: Research on transfer learning of vision-based gesture recognition. Int. J. Autom. Comput. **18**(3), 422–431 (2021)
29. Wu, J., Zhang, Y., Zhao, X.: A prototype-based generalized zero-shot learning framework for hand gesture recognition. In: 2020 25th International Conference on Pattern Recognition (ICPR). pp. 3435–3442. IEEE (2021)
30. Xian, Y., Lampert, C.H., Schiele, B., Akata, Z.: Zero-shot learning-a comprehensive evaluation of the good, the bad and the ugly. IEEE Trans. Pattern Anal. Mach. Intell. **41**(9), 2251–2265 (2018)
31. Xu, X., Hospedales, T., Gong, S.: Transductive zero-shot action recognition by word-vector embedding. Int. J. Comput. Vision **123**, 309–333 (2017)
32. Yang, J., Wilson, J.P., Gupta, S.: Diver gesture recognition using deep learning for underwater human-robot interaction. In: Oceans 2019 Mts/Ieee Seattle. pp. 1–5. IEEE (2019)
33. Yang, J., Wilson, J.P., Gupta, S.: Dare: Diver action recognition encoder for underwater human-robot interaction. IEEE Access (2023)
34. Zhang, Y., Jiang, Y., Qi, H., Zhao, M., Wang, Y., Wang, K., Wei, F.: An underwater human-robot interaction using a visual-textual model for autonomous underwater vehicles. Sensors **23**(1), 197 (2022)
35. Zhou, H., Lin, D.J., Huang, T.S.: Static hand gesture recognition based on local orientation histogram feature distribution model. In: 2004 Conference on computer vision and pattern recognition workshop. pp. 161–161. IEEE (2004)

Optic Atrophy Classification from Fundus Images with Few-Shot Learning

Anirban Dasgupta[1(✉)], Anwesha Sengupta[2], and Shubhobrata Bhattacharya[3]

[1] Department of Electronics and Electrical Engineering, IIT, Guwahati, India
anirban.dasgupta@iitg.ac.in
[2] Department of Electrical Engineering, NIT, Rourkela, India
senguptaan@nitrkl.ac.in
[3] MathWorks India Private Limited, Bengaluru, India

Abstract. Optic Atrophy is a significant cause of visual impairment and demands accurate and timely diagnosis for effective treatment. However, diagnosis of this condition is often carried out by manual inspection by ophthalmologists. Recent studies have attempted to automate this diagnosis using machine learning techniques from fundus images. The prime challenge in this approach lies in the limited availability of public datasets and the difficulty in handcrafting distinctive patterns in the fundus images. The present work proposes using a Siamese network architecture for Optic Atrophy detection using fundus images, which would be effective despite the limited number of publicly available samples of the condition. The other contribution of the paper is the use of augmentation techniques, *viz.*, random Gaussian blur, and the addition of salt-and-pepper noise to increase the generalization of the problem. The Siamese network accepts pairs of inputs and classifies the images as similar or dissimilar according to the distance of the two instances in the embedding space. One of the images is taken as a standard image, while the second is the test image. The proposed method provides better results than the existing works.

Keywords: Optic Atrophy · Fundus Images · Few-Shot Learning · Data Augmentation · Siamese Network

1 Introduction

Optic atrophy is a condition where the optic nerve deteriorates or gets damaged, leading to the loss of nerve tissues and causing visual impairment [7]. To diagnose optic atrophy, doctors use methods like Optical Coherence Tomography, Ophthalmoscopy, and Visual Acuity Testing [17], followed by manual inspection. Ophthalmoscopy, also known as fundoscopy, is a popular, non-invasive procedure that involves examining the back of the eye, also known as the fundus, with an ophthalmoscope [12]. This method does not require special preparation

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 362–374, 2025.
https://doi.org/10.1007/978-3-031-78183-4_23

or anesthesia and produces images called fundus images [10]. Optic atrophy can be identified by the pale optic nerve heads seen in these fundus images. However, diagnosing optic atrophy is challenging due to the poor contrast and complex structure of fundus images.

1.1 Motivation

Fundus imaging has emerged as a valuable diagnostic tool, offering detailed insights into the structural changes within the retina and optic nerve head. Traditional methods for detecting optic atrophy involve manual examination of fundus images by ophthalmologists. The main indicators include changes in the optic disc color and shape, thinning of the retinal nerve fiber layer, and the appearance of the optic cup [4]. However, the manual examination is subjective and time-consuming, leading to the development of automated techniques. In recent years, machine learning approaches have shown promise in automating the classification of ocular diseases, including Optic Atrophy, from fundus images [6]. Some fundus images of healthy cases of optic atrophy are shown in Fig. 1, obtained from [6]. In the figure, the main difference between normal healthy patients depicted in the bottom row and patients with optic atrophy given in the top row is the color of the optic disc. The optic disc is the circular region in the fundus image where a number of blood vessels meet. The disks in the bottom row appear reddish, as compared to yellowish color of the disks in the top row. The cup to disc ratio is also normal [0.3 to 0.4] for the bottom row, while it is not so for the optic atrophy cases. However, it is often very difficult to obtain the discriminatory features with visual inspection, as such a clear distinction is not always obtained. This issue makes the development of hand-crafted features difficult for such data. Deep learning models offer vast computational power and ability to learn intricate patterns from data, and hence may be employed to offer a potential solution to enhance the accuracy and efficiency of Optic Atrophy detection using fundus images and hence such approaches may be explored to address the research issue.

1.2 Objectives

The prime objective of this work is to develop a deep neural network model with limited training fundus image samples, and hence to improve the accuracy of the existing methods.

1.3 Literature Survey

Till date, the major focus of automated diagnosis of ocular disorders from fundus images has been on glaucoma detection. The literature on glaucoma detection is relatively rich, while the detection of optic atrophy through fundus images has not received much attention. Very recently, Sneha et al. [13] have proposed the use of empirical wavelet transform (EWT) as features with support vector

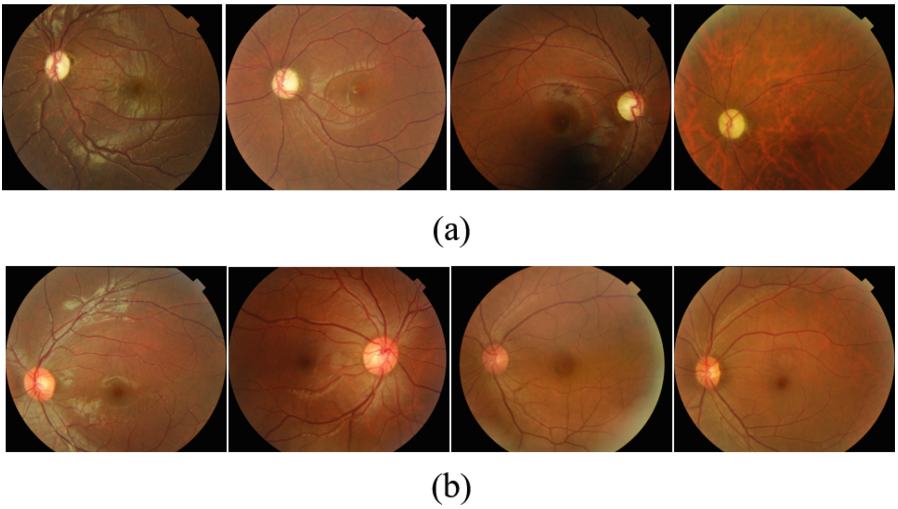


Fig. 1. Sample Fundus Image Taken From the Dataset depicting (a) optic atrophy and (b) normal cases

Table 1. Literature Review

Author, Year	Algorithm	Database	Remarks
Sneha et al., 2019 [13]	EWT with SVM	custom data	- limited accuracy - data not available publicly
Liu et al., 2021 [8]	ResNet-152	custom data	- range of optic disc abnormalities - Spanish ethnicity - data not available publicly
Wang et al., 2023 [14]	VGG-16	custom data obtained from hospitals	Dataset not publicly available
Lee et al., 2024 [6]	ResNet-18	custom data obtained from hospitals	- dataset available - 39 categories
Wang et al., 2024 [6]	DenseNet121	custom data obtained from hospitals	- dataset not publicly available - multiple ocular disorders

machines (SVM) for classifying optic atrophy based on extracted texture features. This method was found to achieve significant accuracy improvements over manual methods. The advent of deep learning has significantly advanced the field of fundus image classification, with a few articles on optic atrophy detection. Liu et al. [8] used ResNet-152 with a custom dataset to detect a range of optic disc abnormalities, focusing on individuals of Spanish ethnicity. However, this data is not publicly available. Wang et al. [14] employed VGG-16 with custom data obtained from hospitals, but similarly, their dataset is not publicly accessible. In contrast, Lee et al. [6] utilized ResNet-18 with hospital-acquired custom data, and their dataset is available. Moreover, this dataset has 39 categories containing a total of 1000 fundus images. The most recent work in 2024 is by Wang et al. [16] utilizing a DenseNet121 model to classify several ocular disorders includ-

ing optic atrophy, achieving robust performance. A summary of the few recent relevant works is provided in Table 1.

The table suggests that there is a scarcity of publicly available datasets, as some researchers could not make the datasets available due to the non-agreement of the participants [14]. The only standard publicly available dataset is by Lee et al., [6] with 39 ocular disorder categories with a total of 1000 images. Specifically, there are only 12 fundus images for the optic atrophy class. Further dividing these images in the ratio 7:3 for training to testing it leaves us with only 8 images for optic atrophy, categorizing it as a few-shot learning problem.

1.4 Research Issues

Examination of the literature reveals the following research issues:

- Limited availability of publicly available datasets for the problem,
- Only publicly available dataset has very few annotated data, with only 12 fundus images for optic atrophy class,
- Generation of synthetic data will require extensive manual inspection by experts,
- Interpretability and explainability of the model's predictions are essential for clinical acceptance and trust.
- Limited scope for generalization to unknown data, thereby restricting real-world applicability.

1.5 Contributions

The present work contributes to a few-shot learned Siamese architecture-based neural network trained on a small-sized dataset. The dataset is generalized with two data augmentation techniques to introduce variability in the dataset concerning blurry and noisy acquisitions. Our assessment of the benchmark dataset demonstrates its superior performance in accuracy, precision, and F1-score when compared to existing methods. Our primary contribution lies in combining both data augmentation and a Siamese network equipped with a contrastive loss function.

2 Dataset

The proposed model is based on the '1000 Fundus images with 39 categories' dataset [3] developed by the Joint Shantou International Eye Centre (JSIEC), Shantou city, Guangdong province, China. The dataset contains a total of 1000 fundus images representing 39 cases of different ocular diseases along with a healthy class. The present work makes use of optic atrophy and healthy classes, containing 12 and 38 images respectively. The images are further augmented with Gaussian blur and addition of salt-and-pepper noise.

2.1 Data Augmentation

Few-shot learning models rely on limited data to generalize well to new, unseen data. By augmenting the training set with various distortions and noises, the model learns to focus on the underlying anatomical structures rather than specific pixel patterns. This leads to better generalization to new images.

Gaussian Blur. By applying Gaussian blur, the model can learn to recognize features even when they are slightly obscured or blurred, mimicking real-world conditions where the quality of fundus images can vary due to factors like patient movement or imperfect focus. This augmentation technique applies Gaussian blur with random kernel sizes and standard deviations to the image [5].

$$I_{\text{blurred}}(x,y) = \frac{1}{2\pi\sigma^2} \sum_{i=-M}^{M} \sum_{j=-M}^{M} \exp\left(-\frac{i^2+j^2}{2\sigma^2}\right) \cdot I(x+i, y+j) \tag{1}$$

This operation yields a blurred version $I_{\text{blurred}}(x,y)$ of the original fundus image $I(x,y)$ by convolving through different Gaussian masks of kernel size M and standard deviations σ. The Gaussian mask defines the weight of each neighboring pixel. The standard deviation σ controls the spread of the blur, while $\frac{1}{2\pi\sigma^2}$ normalizes the Gaussian kernel so that its total sum is 1.
The value of σ was empirically kept in the range $[0.8, 1.2]$, while $M \in \mathbb{Z}, 3 \leq M \leq 9$. In this manner, ten augmented images are obtained for the normal class and 30 for the optic atrophy class, per original image.

Salt-and-Pepper Noise. Salt-and-pepper noise is also added to the images to simulate the presence of random artifacts in the images, such as dust or pixel dropouts. The successful handling of noisy data by the developed model would ensure that the model is robust to such irregularities and enables the model to generalize from limited data. The augmented images $I_{\text{noisy}}(x,y)$ with added noise are obtained as [5]

$$I_{\text{noisy}}(x,y) = \begin{cases} 0 & \text{with probability } p_{\text{salt}} \\ 255 & \text{with probability } p_{\text{pepper}} \\ I(x,y) & \text{otherwise} \end{cases} \tag{2}$$

From the above equation, it is evident that the intensity at some random pixel locations are set to 0 with a probability of p_{salt}, while some are set to 255 with a probability of p_{pepper}, while the remaining are kept intact. We have used $p_{\text{salt}} = p_{\text{pepper}} = 0.05$, which means random 5% pixels are set to 0, while other random 5% are set to 255, and the remaining 90% are kept unchanged. As such, 10 and 30 more images per image were obtained for the normal and atrophy classes, respectively (Figs. 2 and 3).

In this manner, the normal class images were extended to $38(10+10+1) = 798$ images. Likewise, the optic atrophy images were extended to $12(30+30+1) = 732$.

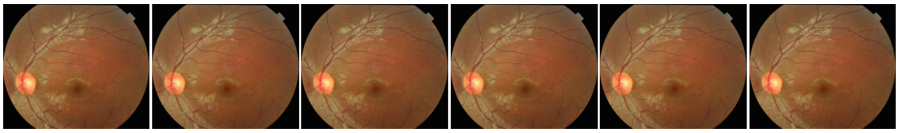


Fig. 2. Sample augmentation of images of a healthy patient with Gaussian blur

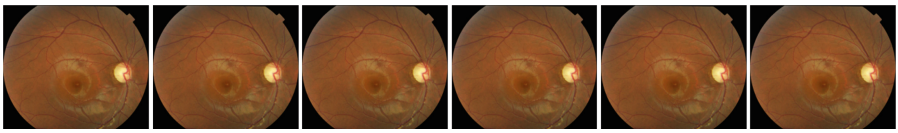


Fig. 3. Sample augmentation of images of a patient having optic atrophy with Gaussian blur

2.2 Dataset Split

The dataset is split into training, validation, and test sets with respective percentages of 70%, 15%, and 15%.

3 Network Architecture

The network is inspired by a Siamese Network, which has gained popularity in many medical-image-based diagnosis tasks due to its few-shot learning capabilities [11]. Siamese networks have emerged as a powerful architectural paradigm within the context of computer vision and similarity learning. These networks are designed to compare and contrast the similarity between inputs, making them ideal for tasks such as face recognition, signature verification, and anomaly detection.

This network takes two images, *viz.* Image 1 and Image 2, as inputs and outputs 0 or 1 depending on whether they belong to the same class or not, as shown in Fig. 4. At the heart of Siamese networks are two identical neural networks, which are parallel to each other. Each network accepts a distinct input, processes the input independently, and computes an output vector. The architecture is designed such that the two networks do not interact directly with each other during the forward pass. Instead, their output vectors are compared using a similarity or distance metric, such as the Euclidean distance, cosine similarity, or contrastive loss. This comparison produces a similarity score that reflects how closely related the two inputs are. As illustrated in Fig. 4, the Siamese Network comprises three convolutional layers succeeded by pooling layers suited for acquiring pairwise similarities. The details of the layers involved are as follows:

- Conv Layer 1: 32 filters, kernel size 3×3, ReLU activation
- Max Pooling Layer 1: Pool size 2×2
- Conv Layer 2: 64 filters, kernel size 3×3, ReLU activation
- Max Pooling Layer 2: Pool size 2×2

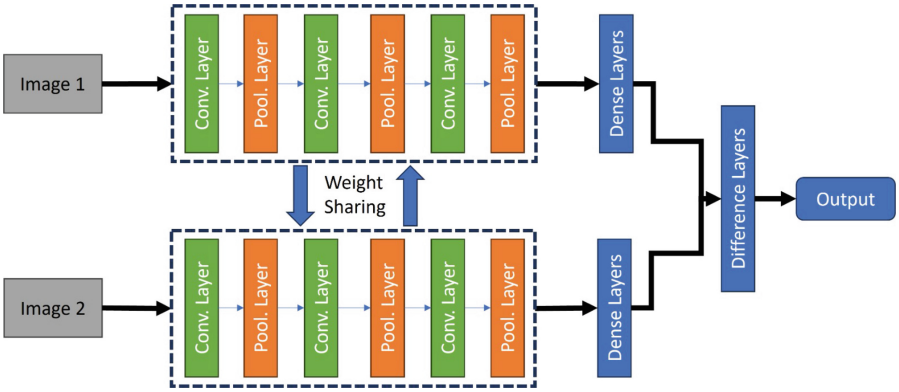


Fig. 4. Architecture of the network

- Conv Layer 3: 128 filters, kernel size 3×3, ReLU activation
- Max Pooling Layer 3: Pool size 2×2

- Dense Layer 1: 256 nodes, ReLU activation
- Dense Layer 2: 128 nodes, ReLU activation
- Dense Layer 3: 64 nodes, ReLU activation
- Output Layer: 1 node, Sigmoid activation

The embedding vectors of the branches are compared by a distance metric.

4 Hyper-parameter Optimization

Hyper-parameter optimization is a critical step in improving the performance and generalization capability of neural networks. For the Siamese Network architecture described, we explored various hyper-parameters including learning rate, batch size, optimizer choice, and regularization techniques. The goal is to identify the optimal set of hyper-parameters that yield the best performance on the validation dataset.

4.1 Hyper-parameter Space

The following hyper-parameters were considered for optimization:

- **Learning Rate:** We experimented with {0.001, 0.01, 0.1}, of which, 0.001 gives the optimal result.
- **Batch Size:** We have taken the candidates for batch size of 16, 32 and 64. The optimal performance is obtained at 32.
- **Optimizer:** Stochastic Gradient Descent [1] is used as the optimizer for this work.
- **Weight Initialization:** We involved Xavier initialization as the method of weight initialization [1].

– **Number of Epochs:** The model performs optimally at 50 epochs.

A combination of grid search and random search techniques was employed to identify the optimal set of hyper-parameters.

4.2 Baseline Model Training

A baseline convolutional neural network (CNN) architecture was trained using the entire dataset for classification into the two classes. Standard loss functions like cross-entropy loss and optimizer algorithms such as Adam were employed to optimize the model parameters. This baseline model served as a starting point for further embedding learning.

Algorithm 1 Baseline CNN Training for Two Classes

Input: Fundus image dataset with two classes (normal, optic atrophy)
Output: Trained CNN model
Initialize CNN architecture
Set hyperparameters: learning rate = 0.1, batch size = 2, number of epochs = 100
Split dataset into training and validation sets
Initialize optimizer = Adam and loss function = binary cross-entropy
Training:
for each epoch **do**
    for each batch in training set **do**
        Forward pass: Compute predicted probabilities for each class
        Calculate loss using predicted probabilities and ground truth labels
        Backward pass: Update model parameters to minimize loss
    end for
    Validate the model on the validation set
end for
Output: Trained CNN model

Algorithm 2 Siamese Network Architecture

Input: Fundus image pairs (x_1, x_2)
Output: Embeddings e_1, e_2
Define CNN architecture CNN_{shared} (shared weights)
Initialize distance metric function $dist$
Forward Pass:
Pass image x_1 through CNN_{shared} to obtain embedding e_1
Pass image x_2 through CNN_{shared} to obtain embedding e_2
Compute Distance:
Calculate distance $d = dist(e_1, e_2)$
Output: Distance d

4.3 Training the Siamese Network

A contrastive loss function $L_{contrastive}$ is defined [15] to penalize the model when the distance between embeddings of images from the same class is large, and the distance between embeddings of images from different classes is small.

$$L_{\text{contrastive}}(y, d) = \frac{1}{2} y \cdot d^2 + \frac{1}{2}(1 - y) \cdot (\max(margin - d, 0))^2 \tag{3}$$

Here, y is a binary label indicating whether the pair of samples are similar ($y = 1$) or dissimilar ($y = 0$). The value d is the distance between the two samples, typically computed using a distance metric, which is explained in Sect. 5.1. The parameter $margin$ defines a radius around the dissimilar pairs. If the distance d is within this margin for dissimilar pairs, it incurs a loss. It helps in creating a margin between similar and dissimilar pairs. For similar pairs ($y = 1$), the term $\frac{1}{2} y \cdot d^2$ encourages the distance d between similar pairs to be as small as possible. Since y=1, this reduces to $\frac{1}{2} d^2$. The factor $\frac{1}{2}$ is typically used to simplify the gradient calculation. For dissimilar pairs ($y = 0$), the term $\frac{1}{2}(1-y) \cdot (\max(margin - d, 0))^2$ is active. It encourages the distance d to be larger than the margin. If d is less than the margin, the term $\max(margin - d, 0)$ will be positive and contribute to the loss. If d is greater than the margin, the term will be zero, indicating no penalty.

The Siamese network was trained using pairs of fundus images sampled from the dataset. For each training iteration, pairs of images were sampled, ensuring an equal distribution of intra-class and inter-class pairs. This was ensured through the data augmentation techniques. A learning rate scheduler was used to adjust the learning rate during training for better convergence.

Algorithm 3 Training the Siamese Network

Input: Fundus image pairs (x_1, x_2), ground truth labels y
Output: Trained Siamese network
Initialize Siamese network architecture
Set hyperparameters: learning rate = 0.1, batch size = 2, number of epochs = 50
Split dataset into training and validation sets
Initialize optimizer = Adam and contrastive loss function $L_{contrastive}$
Training:
for each epoch **do**
  for each batch in training set **do**
    Sample positive and negative image pairs (x_{1p}, x_{2p}) and (x_{1n}, x_{2n})
    Forward pass: Compute embeddings $e_{1p}, e_{2p}, e_{1n}, e_{2n}$
    Calculate contrastive loss $L_{contrastive}$ using embeddings and ground truth labels
    Backward pass: Update model parameters to minimize contrastive loss
  end for
  Validate the model on the validation set
end for
Output: Trained Siamese network

4.4 Embedding Space Visualization

The learned embeddings were visualized using t-SNE dimensionality reduction [2] technique to ensure that the embeddings were well-separated in the embedding space.

5 Results

The performance of the proposed network needs to be compared against that of existing methods. The ability of distance metrics to encapsulate the discriminating capabilities in the embedding space also need to be examined. Hence, we evaluate the performance of the proposed network on the test set using various distance metrics and compare the performance of the same with existing reported methods.

5.1 Distance Metric Evaluation

We have used the following distance metrics to compare the embeddings [9].

Euclidean Distance. The Euclidean Distance between two embeddings $\mathbf{x}_1$ and $\mathbf{x}_2$, in an n-dimensional space, is obtained as

$$d(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{\sum_{i=1}^{n} (x_{1i} - x_{2i})^2} \tag{4}$$

Manhattan Distance. The Manhattan Distance is obtained as the absolute difference as

$$d(\mathbf{x}_1, \mathbf{x}_2) = \sum_{i=1}^{n} |x_{1i} - x_{2i}| \tag{5}$$

Cosine Similarity. This distance signifies the cosine of the angle between the two embedding vectors, computed as

$$d(\mathbf{x}_1, \mathbf{x}_2) = \frac{\mathbf{x}_1 \cdot \mathbf{x}_2}{\|\mathbf{x}_1\| \|\mathbf{x}_2\|} \tag{6}$$

Hamming Distance. This distance is obtained as

$$d(\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{n} \sum_{i=1}^{n} \mathrm{I}(x_{1i} \neq x_{2i}) \tag{7}$$

The performance comparison for these different distances is given in Table 2. We find that the cosine similarity performs better in this regard. This observation may be attributed to the fact that cosine similarity considers only the orientation of the vectors in the embedding space and is unaffected by the magnitude of the vectors.

Table 2. Performance of Siamese Network with Different Distance Metrics

Distance Metric	Performance			
	Accuracy	Precision	Recall	F1 Score
Euclidean Distance	88.45	0.89	0.95	0.919
Manhattan Distance	86.77	0.84	0.88	0.859
Cosine Similarity	91.13	0.92	0.89	0.904
Hamming Distance	87.23	0.87	0.89	0.879

5.2 Comparison With Existing Works

The performance of the proposed Siamese network is compared with that of two existing methods. The results show that the proposed method outperforms the existing methods, except for the recall, where VGG-16 has a better recall. Since the dataset is well-balanced and representative of the two classes, VGG16 was able to achieve high recall by effectively learning discriminative features for each class. This may not be true if a more skewed dataset was used for training (Table 3).

Table 3. Results of Fundus Image Classification

Model	Accuracy (%)	Precision	Recall	F1 Score
Sneha *et al.* (EWT with SVM) [13]	82.853	0.786	0.812	0.794
Liu *et al.* (ResNet-152) [8]	85.137	0.814	0.877	0.838
Wang *et al.* (VGG-16) [14]	87.244	0.837	0.916	0.868
Lee *et al.* (ResNet-18) [6]	83.924	0.833	0.891	0.862
Wang *et al.* (DenseNet-121) [16]	87.242	0.837	0.913	0.868
Siamese (proposed)	91.130	0.921	0.891	0.904

5.3 Ablation Study

To understand the contribution of each component within our deep network architecture, we performed an ablation study by systematically removing or altering specific layers and observing the impact on model performance. The architecture in question comprises three convolutional layers, each followed by max pooling, and several dense layers leading to a final sigmoid-activated output layer. We conducted experiments on a dataset of fundus images for optic atrophy classification. The performance was evaluated using accuracy and F1-score metrics on the validation set.

– Baseline Model: The full model with three convolutional layers achieved an accuracy of 91.13% and an F1-score of 0.90.

- Removing Conv Layer 3: Excluding the third convolutional layer (128 filters) resulted in a significant drop in performance, with accuracy falling to 85% and the F1-score to 0.82. This indicates that deeper features extracted by the third layer are crucial for distinguishing subtle differences in fundus images.
- Removing Conv Layer 2: Omitting the second convolutional layer (64 filters) caused the accuracy to decrease to 78% and the F1-score to 0.75. This layer appears essential for capturing intermediate-level features.
- Removing Conv Layer 1: Excluding the first convolutional layer (32 filters) led to the most drastic reduction in performance, with accuracy plummeting to 65% and the F1-score to 0.60. This suggests that initial feature extraction is vital for the model's overall success.
- Effect of Pooling Layers: Removing all max pooling layers resulted in overfitting, with training accuracy at 95% but validation accuracy dropping to 70%. This highlights the importance of pooling layers in reducing spatial dimensions and preventing overfitting.

The ablation study confirms that each convolutional layer contributes significantly to the model's performance. The first convolutional layer is critical for initial feature extraction, while subsequent layers refine these features to improve classification accuracy. Pooling layers play a crucial role in generalization by preventing overfitting.

6 Conclusion

The paper proposes a Siamese network architecture trained for classifying fundus images as either healthy or indicative of optic atrophy. Due to the limited availability of training data, data augmentation techniques have been employed to expand the dataset, enabling more robust model training. The limited availability of data has also prompted the use of the Siamese network, which has shown significant performance for few-shot learning problems. The proposed network is shown to have outperformed traditional architectures like ResNet and VGG16 in terms of classification metrics.

The results of the work also demonstrate that cosine similarity yielded the best performance when comparing image embeddings generated by the proposed Siamese network.

The results of the present work offer sufficient promise to extend the proposed approach to future research problems such as detection of other ocular anomalies such as glaucoma, optic neuropathy, and diabetic retinopathy. By exploring the broader spectrum of ocular diseases, the authors aim to contribute to the development of more accurate and versatile diagnostic tools for use by ophthalmic healthcare practitioners, thereby offering better opportunities for patient care.

Acknowledgements. The authors acknowledge **Dr. Poulami Roy**, MBBS, M.S, DNB Ophthalmology, for her valuable insights on optic atrophy and fundus image analysis.

References

1. Aghdam, H.H., Heravi, E.J.: Guide to Convolutional Neural Networks. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-57550-6
2. Anowar, F., Sadaoui, S., Selim, B.: Conceptual and empirical comparison of dimensionality reduction algorithms (PCA, KPCA, LDA, MDS, SVD, LLE, ISOMAP, LE, ICA, T-SNE). Comput. Sci. Rev. **40**, 100378 (2021)
3. Cen, L.P., et al.: Automatic detection of 39 fundus diseases and conditions in retinal photographs using deep neural networks. Nat. Commun. **12**(1), 4828 (2021)
4. Chaddah, M., Khanna, K., Chawla, G.: Optic atrophy (review of 100 cases). Indian J. Ophthalmol. **19**(4), 172–176 (1971)
5. Elangovan, P., Nath, M.K.: Glaucoma assessment from color fundus images using convolutional neural network. Int. J. Imaging Syst. Technol. **31**(2), 955–971 (2021)
6. Lee, D.K., Choi, Y.J., Lee, S.J., Kang, H.G., Park, Y.R.: Development of a deep learning model to distinguish the cause of optic disc atrophy using retinal fundus photography. Sci. Rep. **14**(1), 5079 (2024)
7. Lenaers, G., et al.: Dominant optic atrophy. Orphanet J. Rare Dis. **7**, 1–12 (2012)
8. Liu, T.A., et al.: Detection of optic disc abnormalities in color fundus photographs using deep learning. J. Neuroophthalmol. **41**(3), 368–374 (2021)
9. Lotlekar, K.S., Desai, S.D.: Multilevel classification model for diabetic retinopathy. In: 2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS), pp. 326–331. IEEE (2018)
10. Qureshi, A., Virdee, J., Tsermoulas, G., Sinclair, A.J., Mollan, S.P.: Optical coherence tomography confirms shunt malfunction and recurrence of raised intracranial pressure in optic atrophy. Br. J. Neurosurg. **36**(2), 185–191 (2022)
11. Rossi, A., Hosseinzadeh, M., Bianchini, M., Scarselli, F., Huisman, H.: Multi-modal siamese network for diagnostically similar lesion retrieval in prostate mri. IEEE Trans. Med. Imaging **40**(3), 986–995 (2020)
12. Scott, D.A., Wang, M.T., Danesh-Meyer, H.V., Hull, S.: Optic atrophy in prematurity: pathophysiology and clinical features. Clin. Exper. Optomet. 1–10 (2023)
13. Sneha, K., Byjubai, T.: Automated diagnosis of optic atrophy using empirical wavelet transform. In: International Conference on Sustainable and Innovative Solutions for Current Challenges in Engineering and Technology, pp. 271–281. Springer (2019)
14. Wang, C., et al.: Research on assistant diagnosis of fundus optic neuropathy based on deep learning. Curr. Eye Res. **48**(1), 51–59 (2023)
15. Wang, F., Liu, H.: Understanding the behaviour of contrastive loss. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2495–2504 (2021)
16. Wang, J.D., Liu, M.R., Liu, M.L., Zhang, R., Chen, C.X., Cao, K.: An auxiliary diagnostic tool for common fundus diseases based on fundus color photography and light-weight classification models. Graefes Arch. Clin. Exp. Ophthalmol. **262**(1), 223–229 (2024)
17. Xie, J.S., Donaldson, L., Margolin, E.: The use of optical coherence tomography in neurology: a review. Brain **145**(12), 4160–4177 (2022)

Recognition of Online Handwritten Chinese Texts in Any Writing Direction via Stroke Classification Based Over-Segmentation

Yi Chen[1,2], Heng Zhang[1,2], Min-Si Ren[1,2], and Cheng-Lin Liu[1,2(✉)]

[1] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China
{yi.chen,liucl}@nlpr.ia.ac.cn, {heng.zhang,renminsi2021}@ia.ac.cn
[2] State Key Laboratory of Multimodal Artificial Intelligence Systems (MAIS), Institution of Automation, Chinese Academy of Sciences, Beijing 100190, China

Abstract. Online handwritten text recognition technology has been increasingly applied in intelligent touch-based and pen-based devices. Current mainstream methods are mostly designed for horizontally written texts, thus are difficult to handle texts in any writing direction. This paper proposes a recognition framework based on over-segmentation which is applicable to text recognition of any writing direction. It divides text line inclination styles into two cases: texts with the entire line rotated and texts with the line direction rotated while keeping the characters upright. A text line inclination style classification module is introduced in the preprocessing stage to classify these two cases. The former case can be recognized using a horizontal text line recognizer after rotation correction. For the latter case, an improved over-segmentation algorithm is designed based on stroke classification using bidirectional long short-term memory networks (BiLSTM) to achieve text recognition in any writing direction. Experimental results demonstrate that the proposed method is capable of text recognition in any writing direction and achieves highly competitive results on the CASIA-OLHWDB and ICDAR2013-Online datasets.

Keywords: Online Handwritten Chinese Text Recognition · Text Recognition in Any Writing Direction · Improved Over-segmentation

1 Introduction

With the development and progress of deep learning, end-to-end methods based on implicit segmentation have gradually become the mainstream approach for recognizing handwritten Chinese texts [1–5]. Compared to implicit segmentation-based end-to-end methods, explicit segmentation-based recognition methods have the advantage of providing the position of each character in the text along

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 375–391, 2025.
https://doi.org/10.1007/978-3-031-78183-4_24

with the final recognition result. This is of great significance for downstream tasks such as confidence calibration, text editing, and text correction. However, for the recognition problem of texts in any writing direction, neither the explicit segmentation-based methods nor the implicit segmentation-based methods can solve it effectively. Explicit segmentation-based methods have the potential of adapting to any-direction texts because the embedded character classifier there can flexibly recognize segmented characters irrespective of line direction. Therefore, this paper focuses on the recognition of online handwritten texts in any writing direction and proposes an improved method based on the over-segmentation framework.

Currently, in over-segmentation-based frameworks, the segmentation module typically uses methods based on connected component analysis [8], which performs well in segmenting neat and horizontal texts. However, with the widespread use of electronic tablets and interactive electronic screens, people write texts more freely to generate texts of various writing styles, including variable writing directions. Rule-based over-segmentation algorithms only consider horizontal and vertical writing, making it ineffective for accurately segmenting texts in any writing direction. Additionally, when the stroke direction of adjacent characters overlaps significantly, rule-based methods can mistakenly merge strokes, leading to decreased character segmentation and recognition performance. Designing delicate over-segmentation rules cannot deal with highly complicated and variable cases of segmentation. While implicit segmentation-based methods can partially address this problem by collecting or synthesizing a large number of texts in any writing for training, they suffer from the inherent limitation of not segmenting characters and the reliance on huge training data in a similar style with testing data. In summary, the problems of segmentation and recognition of texts in any writing direction remain unresolved.

This paper addresses the recognition of online handwritten Chinese texts in any writing direction. Chinese characters are composed of basic strokes such as horizontal/vertical/diagonal, hooks, and dots, arranged in a specific order. Each text line is formed by a sequence of strokes from the beginning to the end. Regardless of the variation in the writing direction of the texts, as long as the stroke components of each character are merged, the characters in texts of any writing direction can be segmented, thereby addressing the recognition problem of texts in any writing direction.

Building upon our previous work [6], we propose a stroke classification model based on bidirectional long short-term memory (BiLSTM) networks [7]. We can perform character segmentation by classifying the strokes in the text and then merging all the strokes from adjacent starting to ending markers. Since the BiLSTM model allows for end-to-end training, by training it with synthesized text data in any writing direction along with horizontally written data, we can address the recognition problem of texts in any writing direction. Furthermore, this approach further improves the recognition performance of horizontally written texts under the over-segmentation framework. Compared to end-to-end recognition methods, the proposed method only requires training a concise stroke classifica-

tion model without the need to train the entire model end-to-end, significantly reducing training costs and adapting to various application scenes. The main contributions of this work include the proposed framework for online handwritten text recognition of any writing direction, the method to classify the text line inclination style into two cases, and an improved stroke classification-based over-segmentation method.

The rest of this paper is organized as follows: Sect. 2 reviews related work. Section 3 describes our proposed method. Section 4 presents experimental results, and Sect. 5 draws concluding remarks.

2 Related Work

2.1 Text Over-Segmentation

Over-segmentation methods for online handwritten text can be divided into rule-based and deep learning-based approaches. Rule-based approaches can handle both online and offline data segmentation. On the other hand, most deep learning-based approaches convert online sequences into offline images and combine rules with convolutional neural networks for segmentation. However, the segmentation performance is unsatisfactory for online texts with significant writing overlaps due to the lack of temporal information after converting online data into offline images.

In the early stages, Liu et al. [9] proposed a segmentation algorithm based on connected component analysis and contour shape analysis. It was successfully applied to the recognition of Japanese handwritten postal addresses. Building upon this work, Wang et al. [10] were the first to apply this method to Chinese handwritten text recognition tasks, achieving state-of-the-art recognition performance at that time. Xu et al. [11] proposed a novel approach for separating handwritten touch characters based on skeleton and contour analysis. This method achieved good segmentation results for individual connected characters. For online data, it can be converted into offline images and then segmented using the aforementioned methods. Alternatively, connected component analysis can be directly applied to online data for segmentation. Wu et al. [12] proposed a two-step over-segmentation algorithm that utilizes a convolutional neural network (CNN) for secondary filtering of segmentation points to improve the segmentation performance. However, since the algorithm was originally designed to address the offline text line segmentation problem, converting online data into offline images results in the loss of temporal information and degraded segmentation performance.

2.2 Text Recognitions in Any Writing Direction

Currently, a large amount of handwriting recognition work is focused on addressing the recognition of horizontal or vertical texts. Explicit segmentation-based methods face challenges in designing complex rules to handle text segmentation

in any writing direction. Implicit segmentation-based methods can address this issue by synthesizing a large number of samples with various writing styles and directions, but they can only provide the recognition result for texts in a similar style to training data, and cannot output character segmentation positions, these methods require a significant amount of training data in any writing direction, resulting in high training costs.

Nakagawa et al. [13] proposed an algorithm for unconstrained handwritten Japanese text recognition within an over-segmentation framework. The method achieved promising recognition results. Building upon this work, Hao et al. [14] separated handwritten text with arbitrary character direction and text direction into text elements. This approach achieved recognition performance similar to that of normal horizontally written texts. However, the rule design of this method is complex. Long et al. [15] proposed a centroid balance-based analysis and correction method, but it can only handle the overall rotation of texts. Yan et al. [16] introduced a two-step rotation correction method. They determined the inclination angle by fitting a straight to the center of gravity of strokes and then rotated the entire text and each stroke separately. However, this two-step rotation method can cause changes in the shape of the characters. Yang et al. [17] introduced path signature features into character recognition and applied them to rotation-free text recognition. These advancements in research have aimed to address the challenges of recognition in unconstrained and any writing direction, but are still lacking in recognition texts of any writing direction accurately.

3 Methodology

3.1 Overview

To achieve text recognition in any writing direction, we propose an improved method based on the BiLSTM networks, as shown in Fig. 1. This method addresses the issues of traditional over-segmentation methods by transforming the character segmentation problem into a stroke classification problem. It models and learns the categories of character strokes.

To solve the problem of recognizing texts in any writing direction, this chapter defines two text line inclination styles: (1) the overall rotation of the text and (2) the rotation of the writing direction while keeping the characters upright. For the first style, the simplest approach is to synthesize a large amount of data and train a stroke classification model. However, this method significantly increases the training costs and overhead. To reduce reliance on synthesized data, we employ a rule-based approach to determine the orientation of the texts. For cases where the texts are rotated overall, they are simply rotated to a horizontal position, and methods for recognizing horizontally written texts are used for recognition. For the other style, we synthesize the corresponding data and combine it with horizontally written text data to train and optimize the model.

After over-segmentation, the primitive segments are dynamically combined into candidate characters in text line recognition, using our previously proposed

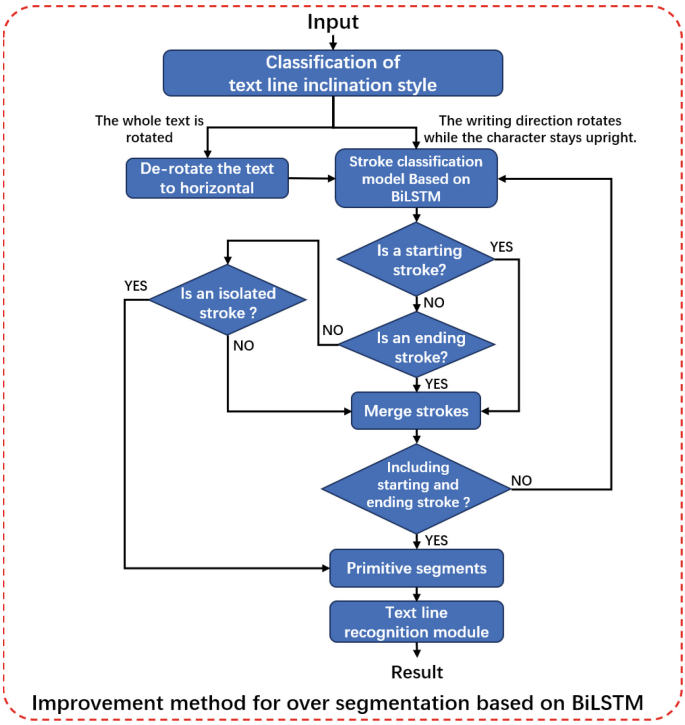


Fig. 1. Overall diagram of the improved over-segmentation based recognition method.

method [6]. For evaluating the candidate paths in text line recognition, the candidate characters are classified using a convolutional prototype network, which is pre-trained on isolated character samples, and then trained end-to-end on string samples.

3.2 Classification of Text Line Inclination Style

Different populations have significant variations in their writing styles due to factors such as age, educational background, and personal habits. When writing online text, even the same individual can produce texts with different shapes due to differences in input speed, carelessness, posture, and devices used. Inspired by the literature [13,14], the direction of handwritten text can be divided into the writing direction of the whole text and the writing direction of individual characters. Based on these two directions, inclined styles of text lines can be classified into two types: texts with the entire line rotated and texts with the line direction rotated while keeping the characters upright, as Fig. 2 shown. In the one type, both the character direction and the line direction have an angle with the horizontal direction. This can be considered as a rotation of horizontal lines in any direction. In the other type, the direction of the characters

is generally aligned with the character in the horizontally written line, with only the line direction having an angle with the horizontal direction. It should be emphasized that vertical writing is a special case of this type, where the direction of characters and text lines are both perpendicular to the horizontal direction.

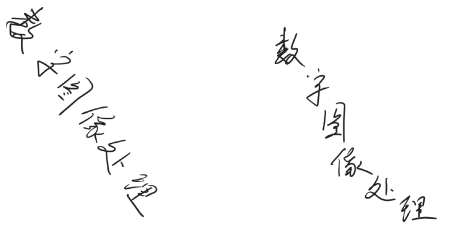


Fig. 2. The two text line inclination styles.

According to the analysis above, we have designed a rule-based and non-training method to determine the directions of text lines and characters to identify the inclination style of text lines. For calculating the direction of the text line, we use the least squares method to fit the centroids of all strokes in the text, and the inclination angle of the fitted line is considered the angle of the text line. For the direction of characters within the text, in standard writing, the direction of writing is usually consistent with the movement direction of the pen tip, that is, from top to bottom and from left to right. Therefore, we define 0∘ as left to right and 90∘ as top to bottom. Then, we calculate the direction of the writing trajectory formed by adjacent points and use it as a feature for each stroke. By analyzing the distribution of these features, we can obtain a histogram of stroke momentum for the text line. The first two peaks of this histogram can determine the writing direction of the characters.

Due to the fact that there is no need to continue using the writing direction of text lines and characters in the subsequent process, our method can achieve an almost indistinguishable classification accuracy compared to separately designed learnable writing direction classification models and does not require additional data for training.

Figure 3 presents an example of determining the inclination style of a text. It specifies that the character writing direction is 0° for left to right, 90° for top to bottom, 180° for right to left, and 270° for bottom to top. In the histogram shown in the figure, The first two peaks of this histogram components correspond to 0° and 90°, respectively. This indicates that the writing direction of the character is from left to right and top to bottom. The calculated angle of the fitted line is −27.14°, indicating that the text is inclined upward by 27.14°. This suggests that the text undergoes rotation of the writing direction while keeping the characters upright.

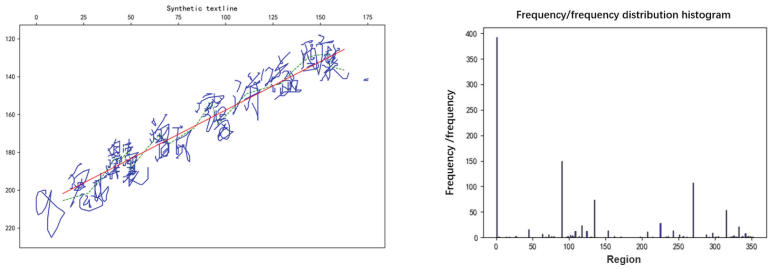


Fig. 3. Example of text line inclination style classification.

3.3 Stroke Classification Model Based on BiLSTM

Rule-based over-segmentation algorithms [22,23] have achieved good results in Chinese and Japanese text recognition but are not robust, especially for Chinese characters with overlapping parts. Current approaches to handling texts in any writing direction involve complex secondary rotation correction and segmentation rules. To address these challenges, we analyze Chinese character writing characteristics and propose treating character segmentation as a stroke classification task. By categorizing strokes into four classes: initial strokes, inserted strokes, ending strokes, and isolated strokes, and training a classification model, we aim to accurately segment characters from the text with a high recall rate for character positions.

Algorithm 1 Stroke merging algorithm

Input: Text stroke sequence Seq, Stroke classification model $Model_{LSTM}$
Output: Primitive fragment set $\mathbb{P} = \emptyset$
for $Strk_i$ in Seq **do**
  $Cls_i = Model_{LSTM}(Strk_i)$
  if $Cls_i ==$ Start stroke **then**
    $Frag_i = \emptyset$, $Frag_i \cup \{Strk_i\}$
  else if $Cls_i ==$ Insert stroke **then**
    $Frag_i \cup \{Strk_i\}$
  else if $Cls_i ==$ End stroke **then**
    $Frag_i \cup \{Strk_i\}$, $\mathbb{P} \cup \{Frag_i\}$
  else if $Cls_i ==$ Isolated stroke **then**
    $\mathbb{P} \cup \{Strk_i\}$
  end if
end for

The model structure consists of two BiLSTM layers, one average pooling layer, and the final classification layer. The model takes input data as $\mathbf{p}_i = (\Delta x_i, \Delta y_i, x_i, y_i, s_i)$, where Δx_i, Δy_i, x_i, y_i, and s_i denote the horizontal offset, vertical offset, horizontal coordinate x, vertical coordinate y, and pen state of

the current sequence point, respectively. The pen state has values of $s = 0$ for the writing state and $s = 1$ for the pen lifted state after writing.

As shown in Algorithm 1, the first step is to input the text line sequence into the BiLSTM model. The model will segment and classify the sequence based on the pen state s, resulting in a classified stroke sequence. After obtaining the category for each stroke, the stroke sequence is divided into multiple sets based on the input order and the predicted positions of isolated strokes. Then, the sets that do not contain isolated strokes are processed. The strokes from the adjacent starting and ending strokes within each set are merged. This forms a primitive unit composed of the strokes in each set. These primitive units, along with the isolated strokes, arranged in order, represent all the primitive fragments of the current text line.

We consider the primitive fragments that only contain their strokes as candidates with $s_i = 1$. Then, we merge adjacent primitives as candidates with $s_i = 2$. Finally, we obtain all the candidate characters for that line.

4 Experiments

4.1 Dataset

The **CASIA-OLHWDB** [18] dataset contains both isolated characters and string samples, divided into a training set with 816 writers and a test set with 294 writers, which consists of $3,129,496$ isolated character samples belonging to $7,356$ classes and $41,710$ handwritten text lines. The string test set includes $10,510$ text lines from $1,020$ text pages, with a total of $269,674$ characters from $2,631$ classes. The **ICDAR2013** [19] Online competition dataset consists of $3,432$ online Chinese handwritten text lines from 60 writers.

We first evaluated the performance of our method on these databases and compared our method with state-of-the-art approaches. Subsequently, we used the ICDAR2013-Online dataset as the source dataset to synthesize text data in any direction and tested the segmentation and recognition performance of our model. Then, we conducted ablation studies to explore the effectiveness of our method. Finally, we evaluated the recognition performance on horizontally written text collected from different devices. The **CH-Textline** dataset is written and collected directly on an electronic screen, divided into two subsets with $1,021$ and $1,108$ text lines, respectively. The **SCUT-TouchScreenLCD** [20] dataset contains $8,809$ text lines written on LCD screens by the South China University of Technology, We randomly selected 1026 lines of text for testing. Additionally, we generated data using the Diffusion Model [21] called **DS-SY**, which has the same semantics as the ICDAR2013-Online dataset, consisting of $1,000$ text lines.

4.2 Performance Metrics

String Recognition Performance Metrics. Following [10,25], the string recognition performance is evaluated by character level correct rate (CR) and accurate rate (AR). The definition is as follows:

$$CR = \frac{N_t - D_e - S_e}{N_t} \times 100\% \tag{1}$$

$$AR = \frac{N_t - D_e - S_e - I_e}{N_t} \times 100\% \tag{2}$$

where N_t is the total number of characters in the test set. The numbers of substitution errors (S_e), deletion errors (D_e), and insertion errors (I_e) are calculated by aligning the recognition result string with the ground truth transcript by dynamic programming.

String Segmentation Performance Metrics. For string segmentation tasks, we use precision, recall rate, and F1-score to evaluate. The definition is as follows:

$$Precision = \frac{True\ positives}{True\ positives + False\ positives} \tag{3}$$

$$Recall = \frac{True\ positives}{True\ positives + False\ negatives} \tag{4}$$

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{5}$$

In the following experiments, "P" indicates the precision, "R" indicates the recall, and "F1" indicates the F1 score.

4.3 Implementation Details

We utilized an NVIDIA Titan 12G GPU to build and train our model based on the PyTorch. We employed two layers of BiLSTM with a hidden size of 100. The batch size was set to 16. We used the RMSprop optimizer for training with an initial learning rate of 0.001. The learning rate was halved every 15 epoch, and a total of 100 epochs were trained. In the inference stage, we used the same Convolutional Prototype Network (CPN) in our previous work [6] as the character classifier, and used a completely consistent dynamic search to complete the recognition.

4.4 Ablation Experiment

To validate the excellent performance of the BiLSTM-based stroke classification model in text line over-segmentation tasks, this study conducted comparative experiments with a connected component-based segmentation method on the CASIA-OLHWDB2.0-2.2 test set. It is important to note that this experiment

Table 1. Comparison of segmentation performance and recognition speed of different methods on the CASIA-OLHWDB test set.

Method	**P**	**R**	**F1**	**Line/S**
Connected Component Analysis	69.49	98.64	81.54	1.75
Ours	**98.32**	**99.38**	**98.85**	**0.93**

"P" indicates the precision, the "R" indicates the recall and the "F1" indicates the F1 score.

evaluated the model's segmentation performance in terms of character boundaries and not the overall character segmentation performance.

The experimental results, as shown in Table 1, indicate that the recall rate of the BiLSTM-based segmentation method is slightly higher than that of the connected component-based method. However, the precision rate and F1 score are significantly higher for the BiLSTM-based method compared to the connected component-based method, demonstrating substantial improvements in both final recognition performance and recognition speed.

4.5 Comparison with the State-of-the-Art Methods in the Horizontal Text Recognition Task

To further demonstrate the effectiveness and competitiveness of the proposed method in this chapter, we conducted a performance comparison with current mainstream methods in the task of horizontal text recognition. It should be noted that in our proposed method, the segmentation model is trained using the CASIS-OLHWDB2.0-2.2 text line training set and randomly synthesized text line data in any writing direction. The recognition model is pre-trained using the CASIS-OLHWDB1.0-1.2 character training set, and weakly supervised fine-tuning is performed using the CASIS-OLHWDB2.0-2.2 text line training set. The performances of other methods are all reported in their corresponding papers.

The detailed comparison results on the CASIA-OLHWB test set are presented in Table 2. Our method achieved a recognition performance of **AR 92.46** and **CR 93.30** without using a language model. When using only the $5-gram$ language model, the recognition performance improved to **AR 96.6** and **CR 96.75**. Our proposed method achieved the best performance in both scenarios, without using any language model and only using the $5-gram$ language model.

It is worth noting that Xie et al. [24] used an additional implicit language model and incorporated extra semantic information, which led to higher recognition performance in their work.

Table 3 provides a detailed performance comparison of the ICDAR2013-Online test set. Our method achieved a recognition performance of **AR 96.08** and **CR 96.19** when using only the $5-gram$ language model. Compared to our previous work, the addition of the BiLSTM-based over-segmentation improvement algorithm further improved the recognition performance. In the experiment settings without using a language model, our method surpassed the work of Xie

Table 2. Comparison of performance between BiLSTM improved method and mainstream methods on CASIA-OLHWDB test set.

Method	Without LM		With LM	
	AR	CR	AR	CR
Shi et al. [1]	87.67	89.58	–	–
Wang et al. [22]	–	–	92.97	93.76
Zhou et al. [23]	–	–	94.69	95.32
Xie et al. [24]	91.38	92.29	95.50	96.09
Xie et al. [24]	**93.31***	**94.47***	**97.23***	**97.50***
Chen et al. [6]	89.66	91.71	94.77	95.47
Ours	**92.46**	**93.30**	**96.64**	**96.75**

The "∗" represents the use of additional implicit language models.
The AR represents the accuracy and the CR represents the correct rate.

et al. [24], but there is still a gap compared to the best method. This is because our proposed method mainly improves the segmentation performance, while the character recognition performance remains unchanged. Even without any data augmentation, our proposed method still demonstrates potential and competitiveness.

4.6 Text Recognition Experiment in Any Writing Direction

To enable the model to segment characters in text lines with any writing direction, we explicitly modeled it by generating training data that includes text lines with writing direction rotation while keeping the character upright. This data was combined with normal data to train the stroke classification model. For the test set, we synthesized text lines with any writing direction using the test set data.

The method we used involves sampling the horizontal spacing ΔH and vertical spacing ΔV of characters from two Gaussian distributions, which are estimated from the training set of text lines. Specifically, we computed the mean and standard deviation of the horizontal and vertical spacing between characters in the training set of text lines. Using these statistical values, we generated two Gaussian distributions, one for sampling the horizontal spacing and the other for sampling the vertical spacing of characters.

To further validate the effectiveness of the proposed method in recognition tasks, we synthesized training data with the same quantity and semantics as the CASIA-OLHWDB training set but with varying angle ranges. Additionally, we synthesized test data with different angle ranges, which have the same quantity and semantics as the ICDAR2013-Online test set.

Tables 4 and 5 display the model's segmentation and recognition performance on data with different skew angle ranges. The analysis reveals that as the skew

Table 3. Comparison of performance between BiLSTM improved method and mainstream methods on ICDAR2013-Online test set.

Method	Without LM		With LM	
	AR	CR	AR	CR
Shi et al. [1]	83.60	85.14	–	–
Xie et al. [24]	86.85	87.82	91.81	92.67
Xie et al. [24]	88.88*	90.17*	96.50	97.15
Peng et al. [2]	91.24	91.81	–	–
Peng et al. [2]	**95.05**†	**95.46**†	97.36†	97.63†
Peng et al. [3]	94.46	94.67	**97.89**†	**98.06**†
Chen et al. [6]	87.62	89.49	95.37	95.55
Ours	90.66	91.23	96.08	96.19

The "$*$" represents the use of additional implicit language models.
The "†" represents the use of additional Transformer language models.

Table 4. The segmentation performance of the model on data with different writing angles.

Writing Angle	Segmentation Performance							
	Start		Insert		End		Isolated	
	P	**R**	**P**	**R**	**P**	**R**	**P**	**R**
0°	0.98	0.98	0.99	0.99	0.98	0.98	0.97	0.97
0° ∼ 3°	0.98	0.98	0.99	0.99	0.98	0.98	0.97	0.97
0° ∼ 5°	0.98	0.97	0.99	0.99	0.98	0.97	0.97	0.96
0° ∼ 10°	0.98	0.97	0.99	0.99	0.98	0.97	0.97	0.96
10° ∼ 45°	0.96	0.96	0.98	0.98	0.97	0.96	0.96	0.96

Table 5. The recognition performance of the model on data with different writing angles.

Writing Angles	Recognition Performance	
	AR	CR
0°	90.66	91.23
0° ∼ 3°	90.62	91.20
0° ∼ 5°	90.55	91.07
0° ∼ 10°	90.38	90.82
10° ∼ 45°	89.84	90.44

angle increases, the segmentation performance is somewhat affected, resulting in a slight decrease in recognition performance. This phenomenon is reasonable because the source data used for synthesis has relatively small skew angles, and the synthesized data mostly falls within the range of 0° to 10°. Therefore, as the angle increases, there is a slight decline in the segmentation and recognition performance. However, since the synthesized data can be integrated into the training process, randomly skewed text line data can be dynamically generated during each mini-batch iteration and trained together with the original data. As a result, the integrated model can still achieve a high level of recognition for text lines with any writing direction.

4.7 Generalization Experiment

To validate the good generalization ability of the proposed BiLSTM improvement algorithm, this study conducted recognition performance tests using horizontally written text line data collected from different devices and weakly labeled horizontally written data generated using a diffusion model-based approach.

Table 6. The recognition performance of the model when tested with different data.

Dateset	Recognition Performance	
	AR	CR
ICDAR2013-Online	90.66	91.23
CH-Textline1.0	85.57	89.10
CH-Textline1.1	83.52	87.29
SCUT-TouchScreenLCD	86.72	87.54
DS-SY	80.67	80.49

From the experimental results as shown in Table 6, it can be observed that the proposed method in this study demonstrates usable recognition performance on data obtained from different acquisition methods, even without any data augmentation. Even for the data generated using the diffusion model [21], the proposed method still achieves usable recognition performance. This indicates that the proposed method is capable of learning the relationships between strokes and the visual features of characters, and it possesses a certain level of generalization ability.

4.8 Further Visualization and Analysis

The visualization results are shown in Fig. 4. The upper part of the figure compares the segmentation results of the same texts. It can be observed that the BiLSTM-based method can achieve more accurate text segmentation and obtain more precise primitive fragments while ensuring segmentation recall.

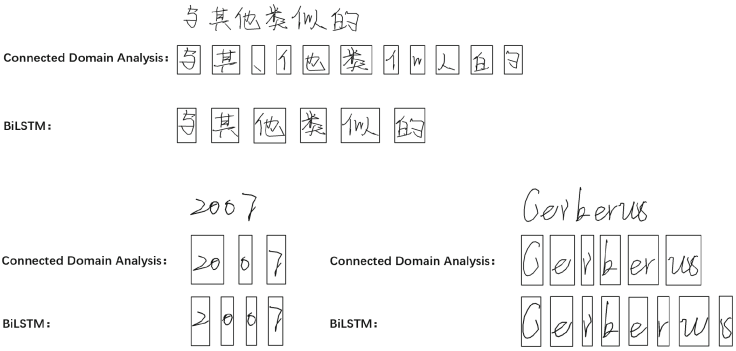


Fig. 4. Visualization experimental results of two over-segmentation methods.

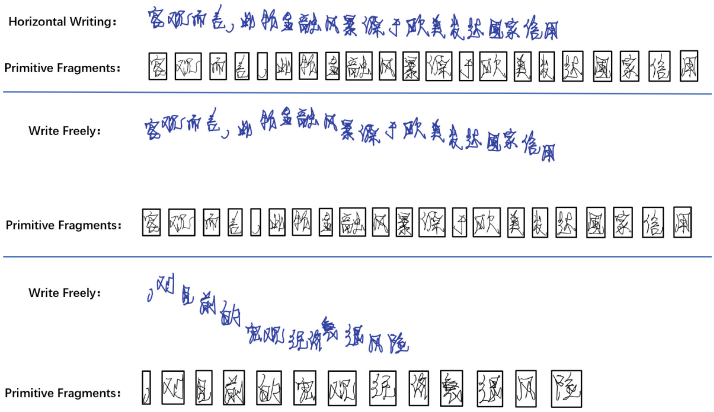


Fig. 5. Visual segmentation results of texts in any writing direction using an improved BiLSTM method.

The lower part of Fig. 4 showcases the segmentation results for digit strings and English in handwritten texts. Due to the presence of overlapping strokes, the method based on connected component analysis tends to incorrectly merge strokes as connected components, leading to irreparable errors in subsequent recognition. In contrast, the improved method demonstrates better segmentation performance when dealing with characters with overlapping strokes. This is because the BiLSTM model can learn the relationships between strokes, enabling more accurate character segmentation. Through the BiLSTM model, the improved method captures the sequential information of strokes within characters and utilizes contextual information for segmentation decisions. Therefore, even in the presence of stroke overlap, this method can accurately segment characters into their respective components.

Figure 5 illustrates the over-segmentation results of the improved method for arbitrarily written texts. Combining it with Fig. 4, it can be observed that

regardless of the writing direction of the texts, the improved method consistently produces good results. This is because BiLSTM itself can capture dependencies within sequences and consider both the forward and backward information of input sequences. Through training with synthesized data, it can learn the underlying patterns and structures of strokes in text sequences of any writing direction, enabling effective modeling of the temporal sequences of texts written in any direction.

5 Conclusions

In this paper, we propose a stroke classification model based on BiLSTM to improve the over-segmentation algorithm. Unlike previous methods based on connected component analysis, our proposed improvement method consequently solves the recognition for text lines written in any direction.

To our best knowledge, our work is the first to utilize deep learning methods to improve online text line over-segmentation and achieve recognition for text lines written in any direction within the over-segmentation framework. Our proposed method partially addresses the recognition of texts written in any direction and achieves competitive recognition results for horizontal text recognition tasks. However, the BiLSTM-based method still struggles with connected characters, Therefore, further exploration and research are needed to address the segmentation of connected characters.

Acknowledgements. This work has been supported by the National Key Research and Development Program Grant 2020AAA0109700, the National Natural Science Foundation of China (NSFC) grants 61936003 and 62276258.

References

1. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Trans. Pattern Anal. Mach. Intell. **39**(11), 2298–2304 (2016)
2. Peng, D., Jin, L., Wu, Y., et al.: A fast and accurate fully convolutional network for end-to-end handwritten Chinese text segmentation and recognition. In: The 15th International Conference on Document Analysis and Recognition (ICDAR), pp. 25–30. IEEE (2019)
3. Peng, D., Jin, L., Ma, W., et al.: Recognition of handwritten Chinese text by segmentation: a segment-annotation-free approach. IEEE Trans. Multimedia **65**, 2368–2381 (2022)
4. Yu, M.M., Zhang, H., Yin, F., et al.: An approach for handwritten Chinese text recognition unifying character segmentation and recognition. Pattern Recogn. **151**, 110373 (2024)
5. Huang, J., Peng, D., Li, H., et al.: SegCTC: offline handwritten Chinese text recognition via better fusion between explicit and implicit segmentation. In: The 17th International Conference on Document Analysis and Recognition (ICDAR), pp. 332–349, vol. 14190. Springer (2023)

6. Chen, Y., Zhang, H., Liu, C.L.: Improved learning for online handwritten Chinese text recognition with convolutional prototype network. In: The 17th International Conference on Document Analysis and Recognition (ICDAR), vol. 14190, pp. 38–53. Springer (2023)
7. Graves, A., Fernández, S., Schmidhuber, J.: Bidirectional LSTM networks for improved phoneme classification and recognition. In: International Conference on Artificial Neural Networks, pp. 799–804. Springer (2005)
8. Cheriet, M., Kharma, N., Liu, C.L., et al.: Character Recognition Systems: A Guide for Students and Practitioners. Wiley (2007)
9. Liu, C.L., Koga, M., Fujisawa, H.: Lexicon-driven segmentation and recognition of handwritten character strings for Japanese address reading. IEEE Trans. Pattern Anal. Mach. Intell. **24**(11), 1425–1437 (2002)
10. Wang, Q.F., Yin, F., Liu, C.L.: Handwritten Chinese text recognition by integrating multiple contexts. IEEE Trans. Pattern Anal. Mach. Intell. **34**(8), 1469–1481 (2011)
11. Xu, L., Yin, F., Wang, Q.F., et al.: Touching character separation in Chinese handwriting using visibility-based foreground analysis. In: The 11th International Conference on Document Analysis and Recognition (ICDAR), pp. 859–863. IEEE (2011)
12. Wu, Y.C., Yin, F., Liu, C.L.: Improving handwritten Chinese text recognition using neural network language models and convolutional neural network shape models. Pattern Recogn. **65**, 251–264 (2017)
13. Nakagawa, M., Onuma, M.: Online handwritten Japanese text recognition free from constrains on line direction and character orientation. In: The 7th International Conference on Document Analysis and Recognition (ICDAR), pp. 519–523. IEEE (2003)
14. Hao, Y., Zhu, B., Nakagawa, M.: A line-direction-free and character-orientation-free on-line handwritten Japanese text recognition system. IEICE Trans. Inf. Syst. **99**(1), 197–207 (2016)
15. Long, T., Jin, L.: A novel orientation free method for online unconstrained cursive handwritten Chinese word recognition. In: The 19th International Conference on Pattern Recognition (ICPR), pp. 1–4. IEEE (2008)
16. Yan, H.Y.: Unconstrained Online Handwritten Chinese Textline Recognition. Dissertation, South China University of Technology (2011)
17. Yang, W., Jin, L., Ni, H., et al.: Rotation-free online handwritten character recognition using dyadic path signature features, hanging normalization, and deep neural network. In: The 23th International Conference on Pattern Recognition (ICPR), pp. 4083–4088. IEEE (2016)
18. Liu, C.L., Yin, F., Wang, D.H., et al.: CASIA online and offline Chinese handwriting databases. In: The 11th International Conference on Document Analysis and Recognition (ICDAR), pp. 37–41. IEEE (2011)
19. Yin, F., Wang, Q.F., Zhang, X.Y., et al.: ICDAR 2013 Chinese handwriting recognition competition. In: The 12th International Conference on Document Analysis and Recognition (ICDAR), pp. 1464–1470. IEEE (2013)
20. Yan, H., Jin, L., Viard-Gaudin, C., et al.: SCUT-COUCH Textline NU: an unconstrained online handwritten Chinese text lines dataset. In: The 12th International Conference on Frontiers in Handwriting Recognition (ICFHR), pp. 581–586. IEEE (2010)
21. Ren, M.S., Zhang, Y.M., Wang, Q.F., et al.: Diff-Writer: a diffusion model-based stylized online handwritten Chinese character generator. In: The 5th Interna-

tional Conference on Neural Information Processing (ICNLP), pp. 86–100. Springer (2023)
22. Wang, D.H., Liu, C.L., Zhou, X.D.: An approach for real-time recognition of online Chinese handwritten sentences. Pattern Recogn. **45**(10), 3661–3675 (2012)
23. Zhou, X.D., Wang, D.H., Tian, F., et al.: Handwritten Chinese/Japanese text recognition using semi-Markov conditional random fields. IEEE Trans. Pattern Anal. Mach. Intell. **35**(10), 2413–2426 (2013)
24. Xie, Z., Sun, Z., Jin, L., et al.: Learning spatial-semantic context with fully convolutional recurrent network for online handwritten Chinese text recognition. IEEE Trans. Pattern Anal. Mach. Intell. **40**(8), 1903–1917 (2017)
25. Su, T.H., Zhang, T.W., Guan, D.J., et al.: Off-line recognition of realistic Chinese handwriting using segmentation-free strategy. Pattern Recogn. **42**(1), 167–182 (2009)

ProFONet: Prototypical Feature Space Optimized Network for Few Shot Classification

Abhijit Das[1(✉)], Vandan Gorade[1], Debesh Jha[1], Koushik Biswas[1], Pethuru Raj[2], and Ulas Bagci[1]

[1] Machine and Hybrid Intelligence Lab, Department of Radiology, Northwestern University, Chicago, USA
aj.das.research@gmail.com
[2] Edge AI Division, Reliance Jio Platforms Ltd., Bangalore, India

Abstract. Conventional Few-shot learning (FSL) techniques often struggle with overfitting and representation collapse, especially with imbalanced datasets. Existing inductive methods try to learn fine-grained features but fail to capture the correlation between the support and query embeddings. This leads to limited discriminative feature learning, resulting in sparse prototypes and overlapping decision boundaries. To address these issues, we propose ***Pro****totypical* ***F****eature* ***O****ptimized* **Net***work* ***ProFONet***, a novel approach for training inductive prototypical networks that emphasizes learning inter-class and intra-class relationships while broadening decision boundaries. ProFONet integrates variance, invariance, and covariance regularization techniques to ***optimize feature space***, resulting in more discriminative prototype clusters and robust representations. The proposed method exhibits adaptability to inductive FSL architectures and demonstrates superior performance on imbalanced datasets. By preventing collapse and reducing false positives, ProFONet outperforms state-of-the-art methods on *CUB* benchmark and a newly curated long-tailed medical imaging dataset *GIF*. The source code of *ProFONet* can be found at https://github.com/aj-das-research/ProFONet.

Keywords: Few Shot Classification · Prototypical Networks · Feature Space Optimization · Distance Function

1 Introduction

To address issues related to expensive manual annotation and the prevalence of highly imbalanced datasets, few-shot learning has emerged as a promising approach [8,26,27,29,30]. In few-shot classification, a classifier is designed to

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78183-4_25.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 392–406, 2025.
https://doi.org/10.1007/978-3-031-78183-4_25

accommodate unseen classes with limited data [32]. Few-shot Matching Networks [29], employs an attention mechanism [6] over a learned embedding space of the support set to predict classes for unlabeled query sets like a weighted nearest-neighbor classifier [29]. Limited data can hinder conventional learning algorithms from grasping underlying patterns effectively, potentially causing overfitting where the model memorizes examples rather than learning generalized representations [26]. Ravi and Larochelle introduced a meta-learning strategy [22], employing LSTM [12] trained in episodes to enhance generalization. Existing methods are susceptible to intensified overfitting when classes demonstrate highly variable characteristics [26]. This can force the model to learn spurious correlations from the training data. To tackle the earlier challenges, Prototypical Networks employ a non-linear mapping of input data into an embedding space, relying on computing class prototypes as the mean of support set embeddings [26]. However, in high-dimensional spaces, the discriminative power of distances between points may diminish, rendering the notion of prototype centroids less meaningful [24]. This issue can arise if the training process fails to effectively optimize the embedding for discriminative clustering [23]. Moreover, in class-imbalanced datasets, prototypes of underrepresented classes may be unreliable due to limited support examples, resulting in biased classification decisions [18,21]. To overcome the problem of unstable representation and poor generalizability techniques like MAML (Model-Agnostic Meta-Learning) [8], triplet network [14], and several regularization techniques [4,15,17] are proposed. However, they may struggle to capture higher-order relationships in the data, leading to representation collapse, particularly evident in structured data, where excessive shrinkage during the learning process can occur [34].

Moreover, methods like Zbontar et al. [33], Ermolov et al. [7] maximize the information content of the embedding; prevent informational collapse by decorrelating every pair of embedding vectors [2]. The Barlow Twins method drives the normalized cross-correlation matrix of the two embeddings towards the identity Zbontar et al. [33], while the Whitening-MSE method whitens and spreads out the embedding vectors on the unit sphere Ermolov et al. [7], signifying the impact of regularization to obtain a discriminative representation of classes in feature space [2,3].

2 ProFONet: Intuition

To address the challenges of overfitting and representation collapse, we introduce ***ProFONet***, a few-shot classification network designed to train prototypical networks by emphasizing the learning of inter-class and intra-class relationships [10]. The core principles are: *Variance regularization* minimizes the spread of embeddings within the same class, promoting consistent representations and closely clustered embeddings. *Invariance regularization* facilitates the learning of robust representations that capture essential data characteristics while remaining invariant to irrelevant variations, enhancing the discriminative nature of prototype clusters. Additionally, *covariance regularization* captures inter-class correlations and penalizes deviations from the desired covariance structure, guiding

the model to learn embeddings reflecting underlying relationships that optimize the feature space.

Contribution. The main contribution of this work is the *variance-invariance-covariance (VIC) injected feature optimization*. Utilizing a weighted loss function *VIC Injected ProtoLoss* $\mathcal{L}_\theta$, *ProFONet* maintains information content within each embedding level. Variance, managed by a hinge loss, preserves the standard deviation across the support set, while invariance, enforced through the mean square distance between class embedding vectors, enhances robustness. Inspired by Barlow Twins [33], covariance regularization combats representation collapse due to embedding redundancy. Recent work such as MLVICX [25] has also demonstrated the benefits of VIC regularization for SSL. Nonetheless, with fewer architectural constraints, *ProFONet* offers a unified framework. To be more precise *ProFONet*:

- Yields compact clusters and encourages separation among class prototypes.
- does not depend on the type of feature extractor.

We demonstrate the effectiveness of feature space optimization by evaluating the representations learned with *ProFONet* on CUB dataset. Furthermore, we also present ***GI-Findings*** (GIF), a *curated dataset for a few shot classifications on gastro-intestinal findings that can support a 5 way- 5 shot scenario.*

3 ProFONet: Detailed Description

ProFONet aligns with contemporary few-shot learning trends, focusing on metric space learning where classification relies on distances to prototype representations for each class concerning a given query. Unlike recent methods, *ProFONet* adopts a simpler inductive bias suitable for limited and highly imbalanced data scenarios. Additionally, our analysis demonstrates that straightforward design choices in distance calculation can lead to significant enhancements.

Pipeline. For a given *support set* $\mathbf{S} = \{(x_i, y_i) \mid x_i, y_i\}$, where each $x_i \in \mathbb{R}^d$ is the d-dimensional feature vector of an example and $y_i \in \{1, \ldots, K\}$ is the corresponding label; *ProFONet* computes a m-dimensional *prototype* $\mathbf{P}_k$ of class k through an embedding function $\mathcal{E}$, followed by a distance function $\mathcal{D}$ between the prototypes and a *query* point $\mathbf{Q}$, as shown in Fig. 1. *ProFONet* learns the distribution over all the K classes and VIC injected feature optimization algorithm back-propagates the weighted loss $\mathcal{L}$, assigning the *query* $\mathbf{Q}$ to the nearest prototype $\mathbf{P}_n$ during each episode.

In *Feature Optimization* (O), the objective is two-fold: first, to eliminate differences between representations of a single class prototype; second, to non-linearly expand dimensions to reduce dependencies among embedding variables. The loss function $\mathcal{L}$ incorporates a term $\mathcal{I}$ for learning invariance to data transformations, regularized by a variance term $\mathcal{V}$ to prevent norm collapse, and a covariance term $\mathcal{C}$ to prevent informational collapse by decorrelating vector dimensions. This optimization felicitates accurate assignment of queries $\mathbf{Q}$ to prototypes $\mathbf{P}_n$.

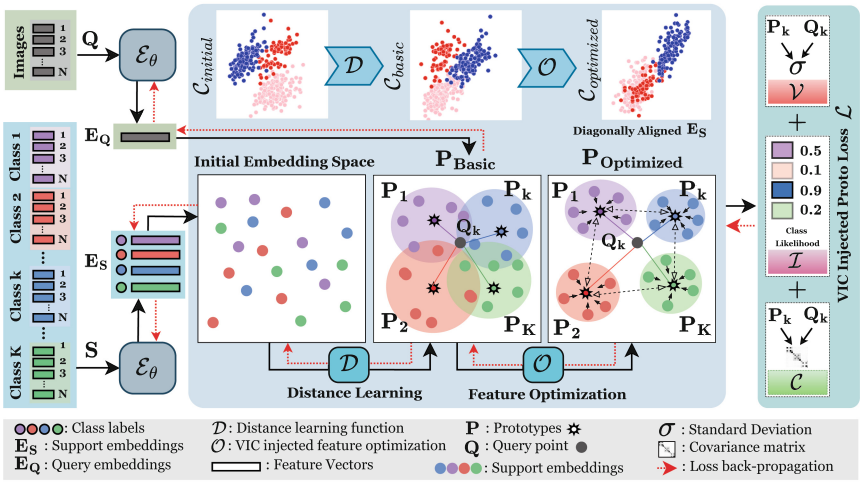


Fig. 1. ProFONet architecture. At the start of each episode of prototypical training, images from Support set **S** and Query set **Q** goes through the shared encoder $\mathcal{E}_\theta$ and produces embedding points. Distance function learns the prototypes ($\mathbf{P_k}$) and forms the clusters from initial embedding. Proposed **Feature Optimization**($\mathcal{O}$) generates denser and well separated clusters of prototypes in $\mathbf{P_{optimized}}$ by inter-class repulsion and intra-class condensation using proposed VIC Injected ProtoLoss $\mathcal{L}_\theta$. **On the top row, the effect of covariance regularization is shown** where embedded points are aligned diagonally after feature optimization (Three classes are selected randomly from a novel set).

3.1 Method

In ProFONet the embedding function $\mathcal{E}_\theta : \mathbb{R}^d \to \mathbb{R}^m$ maps each feature vector x in the feature space $\mathbb{R}^d$ to a point in the latent space $\mathbb{R}^m$, parameterized by θ:

$$\mathcal{E}_\theta : \mathbb{R}^d \to \mathbb{R}^m, \quad x \mapsto \mathcal{E}_\theta(x) \tag{1}$$

Given support set $\mathcal{S}_k$ for class k, the prototype $\mathbf{P}_k$ is computed as the mean vector of embeddings of all support points in $\mathcal{S}_k$:

$$\mathbf{P}_k = \frac{1}{|\mathcal{S}_k|} \sum_{(x_i, y_i) \in \mathcal{S}_k} \mathcal{E}_\theta(x_i) \tag{2}$$

where, $|\mathcal{S}_k|$ denotes the cardinality of $\mathcal{S}_k$. This is followed by the distance function $\mathcal{D} : \mathbb{R}^m \times \mathbb{R}^m \to [0, +\infty)$ that quantifies the dissimilarity between two points in the latent space $\mathbb{R}^m$.

ProFONet produces a distribution over all the K classes for a *query* point $\mathbf{Q}$. The probability of $\mathbf{Q}$ belonging to k is determined by a softmax applied to the negative distances between its embedding $\mathcal{E}_\theta(\mathbf{Q})$ and the prototypes $\mathbf{P}_k$:

$$p_\theta(y = k|\mathbf{Q}) = \frac{\exp(-\mathcal{D}(\mathcal{E}_\theta(\mathbf{Q}), \mathbf{P}_k))}{\sum_{k'=1}^{K} \exp(-\mathcal{D}(\mathcal{E}_\theta(\mathbf{Q}), \mathbf{P}_{k'}))} \tag{3}$$

The objective during training is to minimize the negative log-probability of the true class k for the query $\mathbf{Q}$, expressed as the cross-entropy loss:

$$\mathcal{I} = -\log p_\theta(y = k|\mathbf{Q}) \tag{4}$$

This ensures the invariance regularization $\mathcal{I}$. Additionally, ProFONet utilizes the variance and covariance regularization terms to optimize the embedding space $\mathbb{R}^m$ while calculating the prototypes $\mathbf{P}$ for the support set $\mathcal{S}$.

For a m-dimensional support set embedding $\mathbf{E}_k = \mathcal{E}_\theta(x)$ and the corresponding prototype $\mathbf{P}_k$, we define the variance regularization term $\mathcal{V}$ as a hinge function on the standard deviation of the embeddings:

$$\mathcal{V}(\mathbf{E}) = \frac{1}{m} \sum_{j=1}^{m} \max(0, \gamma - \sigma(\mathbf{E}_j, \epsilon)) \tag{5}$$

where, $\mathbf{E}$ represents the concatenated support set embedding $\mathbf{E}_k$ and prototype embedding $\mathbf{P}_k$, $\mathbf{E}_j$ represents each dimension in $\mathbf{E}$, and σ is the regularized standard deviation $\sigma(\mathbf{E}_j, \epsilon) = \sqrt{\text{Var}(\mathbf{E}_j) + \epsilon}$. γ is a constant target value (fixed to 1 in our experiments), ϵ is a small scalar preventing numerical instabilities. To prevent collapse with all the support set embeddings $\mathbf{E}_k$ mapped on the same vector $\mathbf{P}_k$, we enforce all the classes to be equal to γ along each dimension. If we take $\mathcal{V}(\mathbf{E}) = \text{Var}(\mathbf{E})$ instead of using σ, the gradient of $\mathcal{V}$ with respect to E becomes close to 0 when E is close to $\bar{E}$. Therefore, the gradient becomes close to 0, resulting in representation collapse.

Inspired by Barlow Twins, we define the covariance regularization term $\mathcal{C}$ as the sum of the squared off-diagonal coefficients of $\mathcal{C}(\mathbf{E})$, with a factor $\frac{1}{m}$ that scales the criterion as a function of the support set embedding dimension:

$$\mathcal{C}(\mathbf{E}) = \frac{1}{m-1} \sum_{j=1}^{m} (\mathbf{E}_j - \bar{\mathbf{E}})(\mathbf{E}_j - \bar{\mathbf{E}})^T, \quad \text{where} \quad \bar{\mathbf{E}} = \frac{1}{m} \sum_{i=1}^{K} \mathbf{E}_j \tag{6}$$

This term encourages the off-diagonal coefficients of $\mathcal{C}(\mathbf{E})$ to be close to 0, decorrelating the different dimensions of the embeddings and preventing them from encoding similar information. This ensures decision boundaries are well distinguishable and distinct for each query point $\mathbf{Q}_k$ in the training of ProFONet.

Loss Function. The overall loss function for ProFONet is a weighted and balanced average of all the regularization terms ($\mathcal{V}$, $\mathcal{I}$, and $\mathcal{C}$):

$$\mathcal{L}(\theta) = \lambda_\mathcal{V} \cdot \mathcal{V}(\mathbf{E}) + \lambda_\mathcal{I} \cdot (-\log p_\theta(y = k|\mathbf{Q})) + \lambda_\mathcal{C} \cdot \mathcal{C}(\mathbf{E}) \tag{7}$$

where $\lambda_\mathcal{V}$, $\lambda_\mathcal{I}$, and $\lambda_\mathcal{C}$ are hyper-parameters controlling the importance of each term in the loss optimization and embedding space refinement.

During each training episode, a subset of classes is randomly chosen from the training set. Within each class, a subset of examples forms the support set $\mathbf{S} = \{(x_i, y_i) \mid x_i, y_i\}$ and the remaining as query. The model parameters θ are updated iteratively using Stochastic Gradient Descent (SGD) to minimize the loss function $\mathcal{L}(\theta)$:

$$\theta \leftarrow \theta - \alpha \cdot \nabla \mathcal{L}(\theta) \tag{8}$$

where, α is the learning rate and $\nabla \mathcal{L}(\theta)$ is the gradient of loss with respect to θ.

3.2 Implementation Details

Datasets. We evaluate our proposed method on standard benchmark *Caltech-UCSD Birds 200* (CUB) dataset [31]. Furthermore, we curated a new medical imaging dataset *GI-Findings* dataset[1] (GIF), for few shot classification that can support a 5 way- 5 shot scenario. The CUB dataset includes 200 classes of bird species and has 11,788 images in total. We followed the split suggested in [22] into 100 base classes, 50 validation classes and 50 novel classes. In the GIF dataset, there are a total of 27 classes representing anatomical landmarks, pathological findings, and therapeutic interventions obtained from the gastrointestinal tract, showing distinct findings from the upper and lower GI tract. Images are collected from four open-source datasets GastroVision [16], CVC-ColonDB [28], and PolypGen [1]. We have kept the 4 classes of pathological findings in our novel classes while the other 5 classes are in the validation classes, and in the base class, we have only used anatomical landmarks and therapeutic interventions. This produces a challenging scenario for few shot classification where the model needs to map a novel class and validation class based on representations learned from a different data distribution.

GIF Dataset: Detailed Description. Primarily, the Gastrovision dataset contains only a few images in classes like Angiectasia (17 images), Ulcer (6 images), Esophageal Varices (7 images), and Erythema (15 images). The authors also reported the results in supervised classification tasks by dropping these classes during training, and they have suggested that future researchers solve this classification problem in a few-shot framework. However, the scarcity of data in these aforementioned classes does not support a mere 5-way- 5-shot classification configuration. To ensure that this gastro-intestinal findings dataset is well curated to support standard experiments in few-shot learning, we have included images to GastroVision for the classes that are common in CVC-ColonDB and PolypGen. During curation the images are kept as it is without any resizing. Later preprocessed and formatted as per the standard few shot classification tasks. Curated dataset *Gastro Intestinal Findings* (GIF) contains Angiectasia (517 images), Ulcer (806 images), Esophageal Varices (30 images), and Erythema (18 images). And this is suitable for a 5 way-5 shot classification experiment. Few examples from GIF dataset are shown in Fig. 2.

[1] Curated samples at https://www.kaggle.com/datasets/ad21research/gif-dataset..

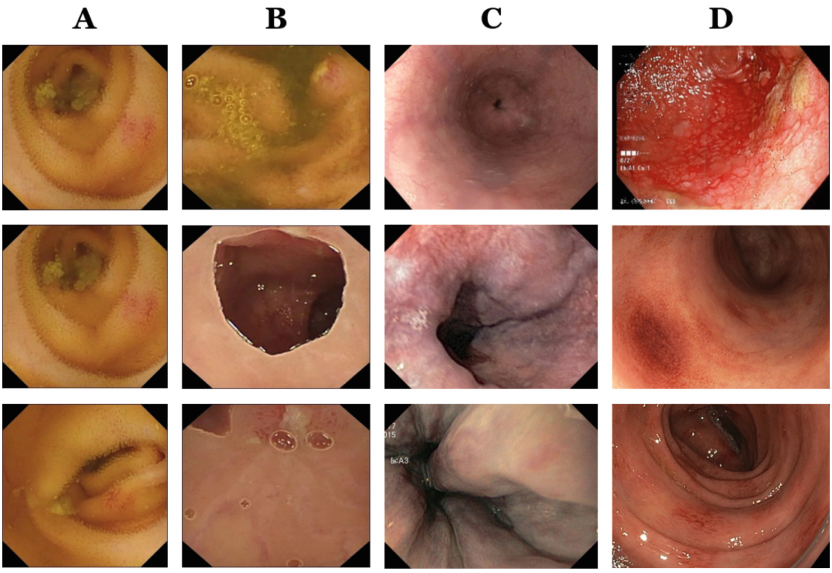


Fig. 2. Examples of the images added to the GastroVision dataset from CVC-ColonDB and PolypGen datasets. (A) Angiectasia, (B) Ulcer, (C) Esophageal Varices and (D) Erythema.

Network Configuration. ProFONet uses a *FeatResNet12* encoder as a feature extractor. Hyperparameters $\lambda_{\mathcal{V}}$, $\lambda_{\mathcal{I}}$, and $\lambda_{\mathcal{C}}$ are set to 0.5, 9 and 0.5, respectively in Eq. 7. At the start of the training, α (Eq. 8) is set to 0.0001, and the K way- N shot training was performed for 200 episodes for all the experiments. A total of 10 query points are used during each training episode. We have used *Mahalanobis* distance metric in our distance function $\mathcal{D}$ in ProFONet.

Techniques of Comparison. We have compared ProFONet with 5 SOTA *inductive* few shot classification methods including MatchingNet [29], RelationNet [18] and ProtoNet [26]. ProFONet is primarily designed as an inductive few-shot learning method, meaning it learns a model from a limited labeled training set and makes predictions on unseen test instances. Whereas, *transductive* few-shot classifiers like LaplacianShot [34] or TIM [5] leverage information from both the labeled training set and the unlabeled test set during inference. This gives a fair platform to evaluate the efficiency of the proposed ProFONet against SOTA methodologies.

4 Results

In this section, we present the quantitative and qualitative results of different algorithms on the CUB and GIF benchmarks.

Table 1. Comparative results of ProFONet against existing SOTA inductive few shot methods. Purple: Second best. Green: Gain over the baseline ProtoNet.

Method	CUB			GIF		
	5w-5s	**5w-3s**	**5w-1s**	**5w-5s**	**5w-3s**	**5w-1s**
SimpleShot [30]	$80.42_{\pm 0.26}$	$68.64_{\pm 0.22}$	$58.02_{\pm 0.24}$	$50.86_{\pm 0.23}$	$45.74_{\pm 0.64}$	$39.44_{\pm 0.91}$
MatchingNet [29]	$82.01_{\pm 0.36}$	$70.25_{\pm 0.22}$	$58.40_{\pm 0.24}$	$52.10_{\pm 0.23}$	$46.12_{\pm 0.64}$	$40.00_{\pm 0.91}$
RelationNet [27]	$82.56_{\pm 0.32}$	$70.00_{\pm 0.22}$	$59.29_{\pm 0.24}$	$54.12_{\pm 0.23}$	$47.02_{\pm 0.64}$	$40.97_{\pm 0.91}$
FEAT [32]	$83.20_{\pm 0.41}$	$71.44_{\pm 0.22}$	$61.53_{\pm 0.24}$	$54.06_{\pm 0.23}$	$47.33_{\pm 0.64}$	$41.21_{\pm 0.91}$
ProtoNet [26]	$84.45_{\pm 0.42}$	$72.50_{\pm 0.85}$	$62.65_{\pm 0.92}$	$56.12_{\pm 0.23}$	$48.86_{\pm 0.64}$	$42.17_{\pm 0.91}$
ProFONet	$\mathbf{88.39}_{\pm 0.38}$	$\mathbf{75.46}_{\pm 0.76}$	$\mathbf{66.02}_{\pm 0.90}$	$\mathbf{63.96}_{\pm 0.32}$	$\mathbf{55.82}_{\pm 0.81}$	$\mathbf{46.42}_{\pm 0.93}$
	03.94 ↑	**02.96** ↑	**3.37** ↑	**07.84** ↑	**06.96** ↑	**04.35** ↑

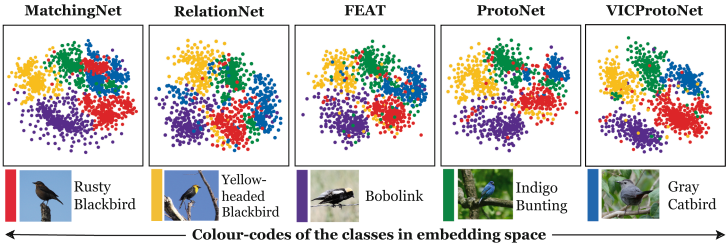


Fig. 3. 2D TSNE plot of SOTA methods vs ProFONet on CUB dataset.

4.1 Evaluation on CUB

On the CUB dataset, ProFONet achieves an accuracy of 88.39%, 75.46%, 66.02%, respectively, for 5 shot, 3 shot, and 1 shot scenarios. As listed in Table 1, the proposed method gains a maximum of 3.94% accuracy in 5 way- 5 shot setting. While all the baselines showcase competitive performance, baselines like ProtoNet and FEAT achieves better results in 1-shot classification. In both FEAT and ProtoNet, there are no explicit *matching* step or *relation score*, unlike in MatchingNet and RelationNet. Instead, they utilize feature embeddings or prototype representations to classify query examples. This also identifies prototype based networks as a superior architecture among these inductive few shot methods. As shown in Fig. 3, in case of ProFONet we can observe denser clusters, separable decision boundary and diagonally aligned embedding space.

4.2 Evaluation on GI-Findings

ProFONet achieves an accuracy of 63.96%, 55.82%, 46.42%, respectively, for 5 shot, 3 shot and 1 shot scenarios, on the GIF dataset. In Table 1, it is evident that the proposed architecture gains 7.84%, 6.96%, 4.35% accuracy. Notably, ResNet18 [13] and ResNet50 [13] only achieve 17% and 19% accuracy in supervised classification setup, whereas ProFONet achieves superior performance

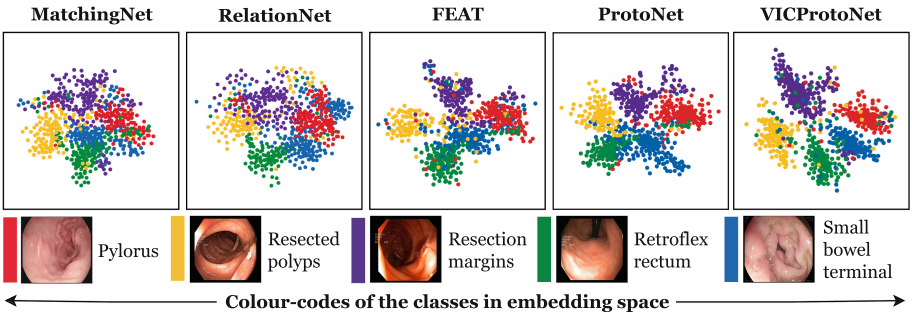


Fig. 4. 2D TSNE plot of SOTA methods vs ProFONet on GIF dataset.

using few shot learning. The difficult task setup of classifying novel pathological findings given anatomical and therapeutic interventions makes our experimental results more effective. As shown in Fig. 4, ProFONet outperforms other methods by optimizing the embedding space using VIC injected ProtoLoss. ProtoNet and FEAT produce a competitive performance, but the decision boundaries between *pylorus* and *small bowel terminal* are not easily separable. Also, ProtoNet produces a sparse cluster for *resection margin* class.

5 Analysis

5.1 Effectiveness of VIC Based Feature Optimization

In Table 2, we can see the improvements achieved by utilizing ProtoLoss and feature space optimization on inductive few shot SOTA methods. SimpleShot employs a nearest-neighbor classifier without meta learning. VIC improves performance by making the decision boundaries among the clusters broader and separable. It's worth mentioning that bidirectional LSTM based MatchingNet learns the dependence of each embedding with the rest. Here, VIC regularization detangles the class embeddings and helps to capture the inter-class relationship well. Moreover, feature optimization leverages classification accuracy for RelationNet and FEAT by a significant margin in both 5 shot and 1 shot settings with 5 query samples per episode. For ProtoNet and FEAT the implementation of VIC ProtoLoss remains identical to what we have done in ProFONet[2]. In MatchingNet, the objective is to minimize the discrepancy between the matching scores and class probabilities. Here, VIC regularization penalizes the divergence between the predicted class distribution and a prior distribution, leading to more robust representations.

[2] Pseudo code implementation is given in supplementary.

Table 2. Effectiveness of feature optimization on different few shot inductive methods.

Method	CUB				GIF			
	w/o VIC		w/ VIC		w/o VIC		w/ VIC	
	5w-5s	5w-1s	5w-5s	5w-1s	5w-5s	5w-1s	5w-5s	5w-1s
SimpleShot [30]	80.42	58.02	82.79	59.92	50.86	39.44	54.23	41.89
			02.37 ↑	01.90 ↑			03.37 ↑	02.45 ↑
MatchingNet [29]	82.01	58.40	83.97	60.41	52.10	40.00	56.78	43.33
			01.96 ↑	02.01 ↑			04.68 ↑	03.33 ↑
RelationNet [27]	82.56	59.29	85.97	61.84	54.12	40.97	57.82	43.62
			03.41 ↑	02.55 ↑			03.70 ↑	02.65 ↑
FEAT [32]	83.20	61.53	86.26	64.51	54.06	41.21	58.94	45.07
			03.06 ↑	02.98 ↑			04.92 ↑	03.86 ↑

5.2 Ablation Study with Different Backbones

As shown in Fig. 5, for both the datasets CUB and GIF FeatResNet is outperforming ResNet12 and ResNet18. FeatResNet is a ResNet12 alike implementation of FEAT model [32] with set-to-set function instantiated with Transformer blocks. FeatResNet learns more task specific and discriminative representations. It is notable that we avoid using heavier and deeper encoders like ResNet50 to keep the computational cost in mind.

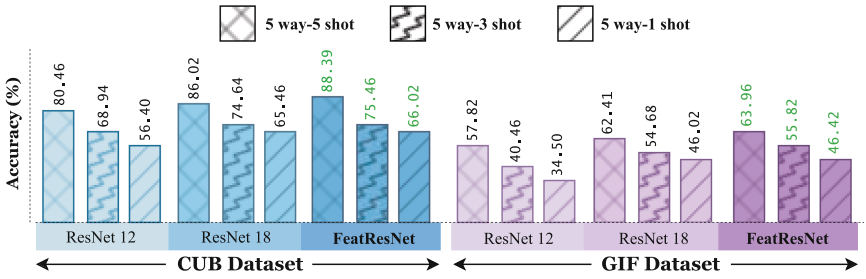


Fig. 5. Ablation with different backbones.

5.3 Ablation Study with Different Distance Functions

In this section, we conduct an ablation study to investigate the impact of different distance functions on the performance of Few-Shot Prototypical Networks. Seven different distance functions mentioned in Fig. 6 are considered. The ablation study reveals that the choice of distance metric significantly influences

the performance of Prototypical Networks on both datasets and across different shot configurations. Notably, the *Mahalanobis* distance [19] function consistently outperforms other metrics, achieving the highest accuracy across most experimental settings. To be precise $88.39\%, 75.46\%, 66.02\%$ (for CUB) and $63.96\%, 55.82\%, 46.42\%$ (for GIF) in 5-shots, 3-shots and 1-shot configurations respectively. Conversely, the *Hamming distance* [20] metric exhibits lower performance compared to other metrics, particularly on the GIF dataset. Unlike *Euclidean* distance, which treats all dimensions equally, *Mahalanobis* distance considers the covariance structure of the data. For non-spherical clusters, *Euclidean* (used in Prototypical Networks) [26] distance may not accurately capture the true distance between points. For datasets like GIF where the data distribution is highly skewed or non-Gaussian, *Mahalanobis* distance provides a more accurate measure of distance compared to other distance functions that assume a uniform distribution inherently.[3]

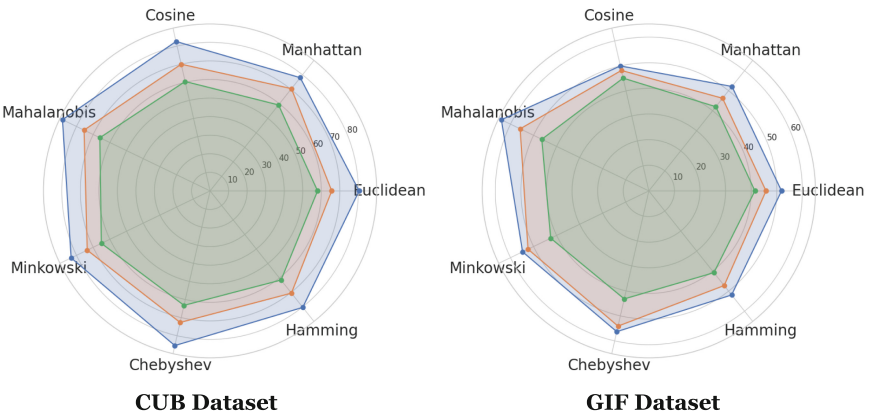


Fig. 6. Radar chat showing the ablation of different distance functions. ProFONet achieves best performance using Mahalanobis distance function with $88.39\%, 75.46\%, 66.02\%$ (for CUB) and $63.96\%, 55.82\%, 46.42\%$ (for GIF) accuracy in 5-shots, 3-shots and 1-shot configurations respectively.

5.4 Effects of Deep Supervision (DS) and VIC Injected ProtoLoss

In this study, we investigate the efficacy of deep supervision (DS) in the context of ProFONet. By providing additional supervision signals during training, we aim to study the impact of DS on feature representation optimization, particularly in tasks with complex decision boundaries. We also ablated with different weighted combinations of loss from the intermediate layers of the encoder and the last

[3] Intuitions of the distance functions used here are discussed in supplementary.

encoder layer. As shown in Figs. 7 and 8, with only Cross Entropy loss in ProFONet, network with no DS outperforms DS enabled settings. This is because the features learned by intermediate layers may not be effectively updated to optimize the task-specific objectives, potentially leading to exploding gradients. With VIC injected ProtoLoss ProFONet reflects the same limitation showing poor performance, as illustrated in Figs. 7 and 8. Deep supervision introduces additional regularization through the auxiliary classification heads, which may conflict with or overshadow the regularization provided by VIC injected ProtoLoss, leading to *gradient misalignment*.

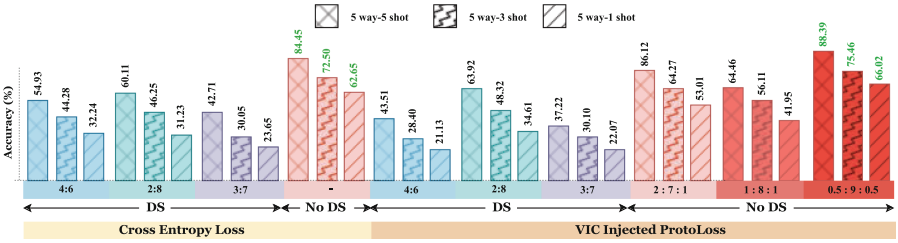


Fig. 7. CUB dataset: Deep supervision vs. VIC Injected ProtoLoss. The ratios inside the deep supervision (DS) section represents the importance given to the intermediate layer loss and the last layer loss while calculating the total loss. And the ratio presented in the 'No DS' section represents the importance of $\lambda_{\mathcal{V}}$, $\lambda_{\mathcal{I}}$, and $\lambda_{\mathcal{C}}$, respectively.

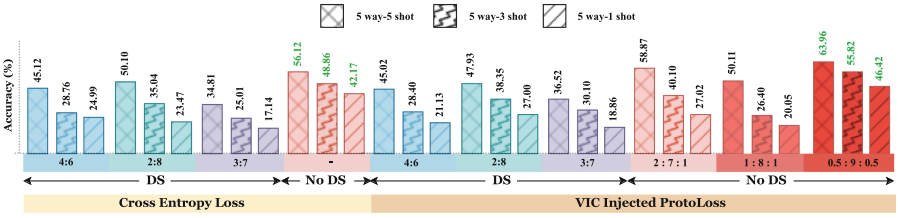


Fig. 8. GIF Dataset: Deep supervision vs. VIC Injected ProtoLoss.

In this study, we also evaluated the performance of ProFONet with different hyperparameters in VIC ProtoLoss. As shown in Figs. 7 and 8 in the right most part we have evaluated with combinations of $\lambda_{\mathcal{V}}$, $\lambda_{\mathcal{I}}$, and $\lambda_{\mathcal{C}}$ and the best combination turns out to be $0.5 : 9 : 0.5$. This signifies that excessive regularization can suppress the model's flexibility and hinder its ability to adapt to task-specific characteristics present in the data. However, the optimal choice of weights may vary depending on the specific characteristics of the task, dataset, and encoder architecture.

6 Conclusion

We introduce *ProFONet*, a novel prototypical network for few shot classification by optimizing feature space for discriminative prototype formation. Learning the inter-class and intra-class relationship produces more distinct and dense embedding for each prototype using variance-invariance-covariance regularization. *ProFONet* mitigates the representation collapse and provides better generalizability for unseen test classes. Extensive experiments and analysis on natural vision and medical imaging datasets demonstrate *ProFONet's* superiority, achieving significant improvements. Our work emphasizes the importance of feature space optimization in few shot learning. In the future, we will explore *ProFONet's* efficacy to enhance model generalization and robustness in various downstream tasks like object detection and segmentation. Also, we will incorporate the concept of learning features in complementary spaces (discrete and continuous) [9,11] for more robust and generalized learning.

Acknowledgment. This project is supported by NIH funding: R01-CA246704, R01-CA240639, U01-DK127384-02S1, and U01-CA268808.

References

1. Ali, S., et al.: A multi-centre polyp detection and segmentation dataset for generalisability assessment. Sci. Data **10**(1), 75 (2023)
2. Bardes, A., Ponce, J., LeCun, Y.: Vicreg: variance-invariance-covariance regularization for self-supervised learning (2022)
3. Bardes, A., Ponce, J., LeCun, Y.: Vicregl: self-supervised learning of local visual features. Adv. Neural. Inf. Process. Syst. **35**, 8799–8810 (2022)
4. Boudiaf, M., et al.: Open-set likelihood maximization for few-shot learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 24007–24016 (2023)
5. Boudiaf, M., Ziko, I., Rony, J., Dolz, J., Piantanida, P., Ben Ayed, I.: Information maximization for few-shot learning. Adv. Neural. Inf. Process. Syst. **33**, 2445–2457 (2020)
6. Das, A., et al.: Pam-unet: shifting attention on region of interest in medical images. arXiv preprint arXiv:2405.01503 (2024)
7. Ermolov, A., Siarohin, A., Sangineto, E., Sebe, N.: Whitening for self-supervised representation learning. In: International Conference on Machine Learning, pp. 3015–3024 (2021)
8. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: Proceeding of the International Conference on Machine Learning, pp. 1126–1135 (2017)
9. Gorade, V., Mittal, S., Jha, D., Bagci, U.: Synergynet: bridging the gap between discrete and continuous representations for precise medical image segmentation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 7768–7777 (2024)
10. Gorade, V., Mittal, S., Singhal, R.: Pacl: patient-aware contrastive learning through metadata refinement for generalized early disease diagnosis. Comput. Biol. Med. **167**, 107569 (2023)

11. Gorade, V., et al.: Towards synergistic deep learning models for volumetric cirrhotic liver segmentation in MRIs. arXiv preprint arXiv:2408.04491 (2024)
12. Graves, A., Graves, A.: Long short-term memory. In: Supervised Sequence Labelling with Recurrent Neural Networks, pp. 37–45 (2012)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition (2015)
14. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. In: Similarity-Based Pattern Recognition: Third International Workshop, SIMBAD 2015, Copenhagen, 12–14 October 2015. Proceedings 3, pp. 84–92 (2015)
15. Huang, S., Zeng, X., Wu, S., Yu, Z., Azzam, M., Wong, H.S.: Behavior regularized prototypical networks for semi-supervised few-shot image classification. Pattern Recogn. **112**, 107765 (2021)
16. Jha, D., et al.: Gastrovision: a multi-class endoscopy image dataset for computer aided gastrointestinal disease detection. In: Workshop on Machine Learning for Multimodal Healthcare Data, pp. 125–140 (2023)
17. Lee, D.H., Chung, S.Y.: Unsupervised embedding adaptation via early-stage feature reconstruction for few-shot classification. In: Proceedings of the International Conference on Machine Learning, pp. 6098–6108 (2021)
18. Li, X., Sun, Q., Liu, Y., Zhou, Q., Zheng, S., Chua, T.S., Schiele, B.: Learning to self-train for semi-supervised few-shot classification. Adv. Neural Inf. Process. Syst. **32** (2019)
19. McLachlan, G.J.: Mahalanobis distance. Resonance **4**(6), 20–26 (1999)
20. Norouzi, M., Fleet, D.J., Salakhutdinov, R.R.: Hamming distance metric learning. Adv. Neural Inf. Process. Syst. **25** (2012)
21. Ochal, M., Patacchiola, M., Vazquez, J., Storkey, A., Wang, S.: Few-shot learning with class imbalance. IEEE Trans. Artif. Intell. (2023)
22. Ravi, S., Larochelle, H.: Optimization as a model for few-shot learning. In: Proceedings of the International Conference on Learning Representations (2016)
23. Rodríguez, P., Laradji, I., Drouin, A., Lacoste, A.: Embedding propagation: smoother manifold for few-shot classification. In: Proceedings of the 16th European Conference on Computer Vision (ECCV 2020), pp. 121–138 (2020)
24. Sendera, M., Przewieźlikowski, M., Karanowski, K., Zieba, M., Tabor, J., Spurek, P.: Hypershot: few-shot learning by kernel hypernetworks. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 2469–2478 (2023)
25. Singh, A., Gorade, V., Mishra, D.: Mlvicx: multi-level variance-covariance exploration for chest x-ray self-supervised representation learning. arXiv preprint arXiv:2403.11504 (2024)
26. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. Adv. Neural Inf. Process. Syst. **30** (2017)
27. Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P.H., Hospedales, T.M.: Learning to compare: Relation network for few-shot learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1199–1208 (2018)
28. Vazquez, D., et al.: A benchmark for endoluminal scene segmentation of colonoscopy images. J. Healthc. Eng. (2017)
29. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. Adv. Neural Inf. Process. Syst. **29** (2016)
30. Wang, Y., Chao, W.L., Weinberger, K.Q., Van Der Maaten, L.: Simpleshot: revisiting nearest-neighbor classification for few-shot learning. arXiv preprint arXiv:1911.04623 (2019)

31. Welinder, P., et al.: Caltech-UCSD birds 200 (2010)
32. Ye, H.J., Hu, H., Zhan, D.C., Sha, F.: Few-shot learning via embedding adaptation with set-to-set functions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8808–8817 (2020)
33. Zbontar, J., Jing, L., Misra, I., LeCun, Y., Deny, S.: Barlow twins: self-supervised learning via redundancy reduction. In: International Conference on Machine Learning, pp. 12310–12320 (2021)
34. Ziko, I., Dolz, J., Granger, E., Ayed, I.B.: Laplacian regularized few-shot learning. In: Proceedings of the International Conference on Machine Learning, pp. 11660–11670 (2020)

Few-Shot Copycat: Improving Performance of Black-Box Attack with Random Natural Images and Few Examples of Problem Domain

Jhonatan Machado Leão(✉), Jacson Rodrigues Correia-Silva, Alberto Ferreira de Souza, Claudine Santos Badue, and Thiago Oliveira-Santos

Universidade Federal do Espírito Santo, Espírito Santo, Brazil
jhonatan.leao@lcad.inf.ufes.br

Abstract. Many companies have developed Convolutional Neural Networks (CNNs) models as a product to offer through APIs to solve various problems. Therefore, the protection of the Intellectual Property of these models from potential attacks is a critical concern for these entities. Regarding these threats, several studies have identified vulnerabilities in these systems, such as model extraction, where the adversary uses Problem Domain (PD) and Non-Problem Domain (NPD) data to generate an imitation of the target model (Oracle). An example of this attack is the Copycat CNN method, where the adversary uses NPD images to train a surrogate model with the Oracle's hard-labels. The surrogate model is then fine-tuned with PD images labeled by the Oracle, significantly improving performance and reducing Oracle queries. However, PD images are generally expensive and scarce on the Internet. In this study, we introduce Few-Shot Copycat, a novel approach to improve Copycat CNN. With just a few PD images from each class of the target problem, our approach improves the performance of the method based on NPD images only. This method requires much fewer queries to copy the target model exposing even more of this threat to companies. The proposed method was evaluated in five real classification problems (Facial Expression Recognition, General Object, Street View House Number, Traffic Sign, and Fashion Image). Results showed that the Few-Shot Copycat can reduce at least 6× the number of images required for extraction (i.e., reducing the number of queries).

Keywords: Deep learning · CNN · Model extraction

1 Introduction

Machine Learning (ML) encompasses a variety of methods that empower computers to find optimal or semi-optimal solutions without being explicitly pro-

This study was financed in part by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - (CAPES, Brazil) - Finance Code 001; Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq (307357/2023-6, 311439/2022-5 e 312817/2022-3); and Fundação de Amparo à Pesquisa do Espírito Santo (FAPES, Brazil) – grant 2021-07KJ2, 2024-DXZ2F, 2022-NGKM5, 2021-GL60J.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 407–422, 2025.
https://doi.org/10.1007/978-3-031-78183-4_26

grammed to do so. ML includes different methods, such as Convolutional Neural Networks (CNNs), to solve a variety of problems including object recognition [7] and disease diagnosis from medical images [32].

As a result, CNNs began to be explored to solve a wide range of problems. Taking advantage of their impressive performance, companies have offered these models as products as Machine Learning as a Service (MLaaS), i.e., already trained models provided to users through Application Programming Interfaces (APIs), like Azure Cognitive Service for Vision and Google Cloud Vision AI. Nevertheless, studies have exposed some vulnerabilities in these models that allow the extraction of their knowledge through techniques called model extraction [4,5,11,24,34]. This approach consists of generating surrogate models with the same or similar performance as the target model (Oracle) [1].

Many studies have delved into exploiting vulnerabilities in black-box models. They exposed the ability to generate surrogate models using only Non-Problem Domain (NPD) images and their corresponding hard-labels [2,5,27,30] or soft-labels [9,14,28,29]. This highlights that the vulnerabilities of CNN models allow different types of attacks.

In a prior study, the authors, in [5] and extended in [4], proposed a method (named Copycat CNN and referred here as Original Copycat Network) capable of creating a surrogate model with a similar performance of a black-box Oracle model by querying for predicted hard-labels using a dataset composed of NPD images. One highlight of their method is the use of public Internet images (i.e., random images representing natural and real human environments) not related to the problem domain to perform the attack. The authors also showed that the performance could be improved (therefore reducing the number of required queries) by incorporating Problem Domain (PD) images in the fine-tuning process. However, the fine-tuning process requires a good amount (not as much as training the original model, but still a good amount) of problem domain data. The issue is that PD images are often difficult to find on the Internet or expensive to produce.

In this study, we introduce a novel approach (named Few-Shot Copycat) to improve the Original Copycat Network. Instead of initially using only NPD images to attack the oracle and then fine-tuning the surrogate model with PD images, we build upon an assumption tested by [21]. Their model extraction method uses a large attack dataset composed of blended NPD images, which allowed the adversary to explore the Oracle's space better. However, we investigated how to more effectively explore this space using images with characteristics similar to those of Oracle's domain, i.e., PD images. We also added the constraint of using only a small number of PD images in the process. The procedure consists of generating images by blending just a few PD images into every image in the NPD dataset to compose a unique attack dataset that requires no additional fine-tuning with PD.

This work diverges from the original study [5] in the following aspects: (i) it requires only a few samples of the PD images (less than 0.3% of the images used

in [5]) for the attack and (ii) it reduces the number of queries to the Oracle by improving the performance of the attack using the same number of images.

The effectiveness of our approach was demonstrated through a comprehensive set of experiments conducted on five use case problems: Facial Expression Recognition (FER7), General Object Classification (GOC9), Street View House Number Classification (SHN10), Traffic Sign Classification (SIG30), and Fashion Image classification (FIC10). Results showed that the Few-Shot Copycat can reduce at least 6× the number of images required for extraction (i.e., reducing also the number of queries) and that only 1 image per class is already enough to improve the performance of the Original Copycat. Therefore, the Few-Shot Copycat could use fewer queries to generate a surrogate model with the same performance as the Original Copycat Network.

2 Related Works

This section presents some of the related works of the literature, grouping them by model extraction attacks with (i) general classifiers, (ii) CNN black-box models with soft-labels, and (iii) CNN black-box models with hard-labels.

Concerning general classifiers, [30] proposed an extraction in a black-box scenario using Logistic Regression, Support Vector Machine (SVM), Multilayer Perceptron (MLP), and Decision Trees. In [27], the authors explored a model extraction using Naive Bayes and SVM classifiers. The data used for extraction was from the problem domain. Different from the method proposed in this work, the works mentioned above assume a large amount of data from the problem domain in order to perform the copy.

Another less strict scenario relies on the soft-labels of the target model prediction (i.e., the probabilities of the images obtained from an Oracle) to label the fake dataset. Therefore, this type of attack is restricted to APIs or models that give access to the classification probabilities or, in some cases, access to the output layer before performing the softmax to compute the probabilities. Examples of this approach can be seen in [6,9,23].

Although effective, this type of attack imposes an additional constraint (the access to the Oracle's probabilities), which is alleviated in the method proposed in this work.

Using the hard labels to conduct attacks on black box models defines a scenario more aligned with our study. In this line, the pioneer methods [4,5] were already discussed in the introduction.

Following the Original Copycat idea, the authors in [20,21] proposed a similar approach but using mentor/student network training. During every epoch of the training process, NPD images are blended together to better map the feature space and avoid overfitting. The mentor network performs the Oracle role, whereas the student is the copied model. Diverging from our proposed method, these previous works either require a large set of NPD images (ImageNet images augmentation in [20,21]) to realize the attack or a decent amount of PD images [4,5], which consequently leads to many more queries to the Oracle.

Our blending approach enhances the performance of a pure NPD attack, thereby reducing the number of queries by adding just a few samples per class from the problem domain to better guide the mapping of the feature space.

3 Few-Shot Copycat

In the scenario addressed in this work, the Oracle model is the target model trained by a company and provided as a black-box service. The user only has access to the hard-label predictions of the Oracle (i.e., the predicted class but not their probabilities). In this context, an adversary (or attacker) can easily acquire thousands of NPD images from the Internet that do not belong to the problem domain. In addition, he can also easily search for a few public PD images (one to ten images per class) belonging to the problem domain and use them all to attack the Oracle.

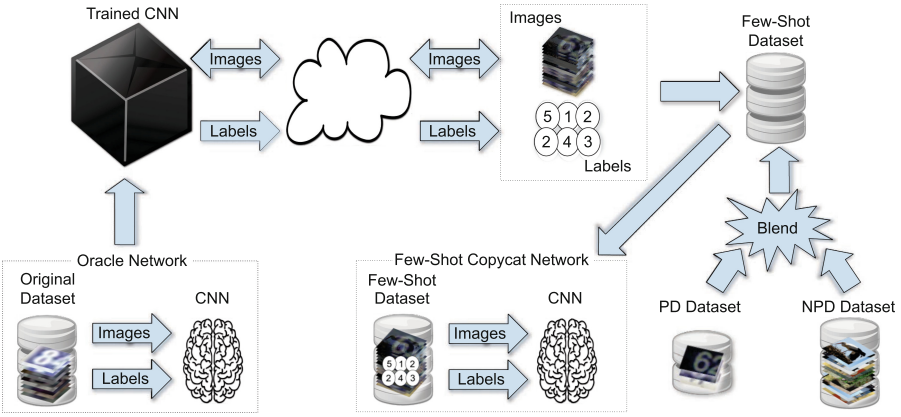


Fig. 1. The left part of the cloud illustrates the company API, which comprises an Oracle with original data. This API receives an image and returns the output. The right part of the cloud shows the adversary using the proposed Few-Shot Copycat. The images from the problem domain and the non-problem domain are blended to generate the images of the Few-Shot Dataset. Subsequently, these blended images are submitted to Oracle to acquire all corresponding labels, which are then employed to train the Few-Shot Copycat.

In summary, the proposed Few-Shot Copycat blends a few PD images representing each target model class with the NPD images (natural images collected from the Internet) to generate a larger dataset that more closely resembles the problem domain but with the random features from the Internet images. An overview is illustrated in Fig. 1. With the new blended dataset, the adversary can query the Oracle for hard-labels of those images, creating a fake dataset for training the Copycat model (called here Few-Shot Copycat since the fake

dataset uses a few images of the problem domain). Finally, the goal is to train the Few-Shot Copycat model with this fake dataset to achieve similar results (F1-Score) as the original Oracle model.

3.1 Few-Shot Fake Dataset Generation

The proposed method blends each PD image with every NPD image to generate a larger dataset containing features of the problem domain. The blended image I_B is the weighted average of a PD image I_{PD} and a NPD image I_{NPD}, as shown in the following equation:

$$I_B = (I_{PD} \times \alpha) + (I_{NPD} \times (1 - \alpha)) \tag{1}$$

where α represents a weight to vary the strength of each domain in the final blended image. Each NPD image is blended to one of the PD images using a predetermined weight and respecting the balance between the number of PD classes. Examples of the images generated through this process are illustrated in Fig. 2. The purpose of the blending process was to incorporate more information from the problem domain into the attack dataset, allowing better exploration of the Oracle's space while using only a few PD images. The weight α is a predefined decimal value that can range between 0 and 1. Therefore, the blended fake dataset comprises the same amount as the original NPD. Once the blended images are generated, the Oracle is used to label them, creating the blended fake dataset. Finally, this dataset is employed to train the Few-Shot Copycat model.

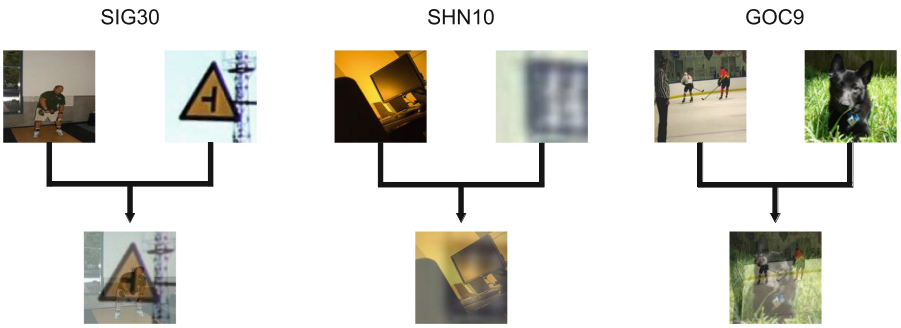


Fig. 2. Examples of images generated through the blending process with weight α set to 0.5 for the SIG30, SHN10 and GOC9 problems. For each problem, the image on the right, preceding the directional arrow, corresponds to the PD image, whereas the image on the left corresponds to the NPD image. The image subsequent to the directional arrow represents the outcome of the blending process with those images.

4 Experimental Methodology

This section outlines the experimental methodology employed to generate the Few-Shot Copycat. The proposed method is compared against two baselines: the Oracle and the Original Copycat Network (i.e., a surrogate model trained only with NPD images). The experiments are conducted across five problem domains: Facial Expression Recognition (FER7), Fashion Image Classification (FIC10), General Object Classification (GOC9), Traffic Sign Classification (SIG30), and Street View House Number Classification (SHN10). Except FIC10, the datasets used for the problems are the same as those employed in [4].

For each investigated problem, it is assumed that there is a private dataset composed of images from the problem domain, named Original Domain (OD). A part of the OD dataset is separated for test purposes (testing performance of the oracle and the copycats) and is referred to Test Data (TD). Therefore, the OD dataset represents the part used for training. In addition to the OD dataset, each of the investigated problems has an additional dataset comprising images from the same domain as the OD to assume the role of the PD dataset. The OD and PD configuration follows those defined in [4,5].

The remainder of this section details the baseline methods, the use case problems, the evaluation metrics, the performed experiments, and the general setup.

4.1 Baselines

Oracle Network. The Oracle Network is a model trained with the OD dataset that can be used as an API service. Therefore, when performing predictions with the Oracle, adversaries have access to the images' hard-labels only. This network is used as an upper bound baseline since the attack aims to have a model that performs as well as the Oracle.

Original Copycat Network. This is the Original Copycat model trained with NPD images only, as proposed in [4,5]. This network is used as a lower bound baseline since the proposed blend aims to improve the Original Copycat's performance, which does not rely on problem domain images.

4.2 Use Case Problems

The five problems referred at the beginning of this section are described here.

Facial Expression Recognition (FER7). This problem encompasses seven distinct classes: fear, sad, angry, disgust, surprise, happy, and neutral. The dataset was created using the Extended Cohn-Kanade Database (CK+) [16] as the Test Data (TD). The AR Face [19], Binghamton University 3D Facial Expression (BU3DFE) [33], The Japanese Female Facial Expression (JAFFE) [18], MMI [25] and Radboud Faces Database (RaFD) [13] as original domain

dataset, OD dataset, and Karolinska Directed Emotional Faces (KDEF) [17] as problem domain dataset, PD dataset. In contrast to the Original Copycat work, the NPD dataset retains its original color configuration, and data augmentation is not employed in the PD dataset.

Fashion Image Classification (FIC10). The goal of this problem is to classify fashion images using the Fashion MNIST (FMNIST) [31] with 10 distinct classes: t-shirt/top, trouser, pullover, dress, coat, sandal, shirt, sneaker, bag and ankle boot. The dataset was divided according to the suggested training and testing sets, which were used respectively as the OD dataset and Test Data (TD). Due to the absence of a similar dataset for the PD dataset, 10 random images of each class were extracted from the OD to compose the PD dataset and were not included in the training process.

General Object Classification (GOC9). This problem involves classifying different types of objects. The dataset is composed of CIFAR-10 [12] and STL-10 [3], with each dataset having 10 classes. However, each dataset includes one unique class ('monkey' in STL-10 and 'frog' in CIFAR-10). Consequently, these unique classes were excluded, resulting in datasets with 9 classes only. The CIFAR-10 dataset was partitioned into two segments based on the recommended training and testing sets, following [4]. The first subset was designated as the OD dataset, whereas the other served as the Test Data (TD). Meanwhile, the STL-10 dataset was defined as the PD dataset.

Street View House Number Classification (SHN10). The objective of this problem is to classify images of house numbers using the SVHN [22] as dataset, which includes 10 classes(numbers ranging from 0 to 9). The dataset was divided according to the suggested training, validation, and test sets, used respectively as OD dataset, Test Data (TD), and PD dataset (as also performed in [4]).

Traffic Sign Classification (SIG30). This problem concerns the classification of the traffic signs that are present in the images. This dataset comprises the TT100k [15] and the TSRD [10] datasets. Since the labels of these datasets are not synchronized, they were firstly preprocessed to remove specific categories (with less than 26 images in the TT100k) and to synchronize the labels in both datasets (as also performed in [4]) using the code provided by the authors[1]. This resulted in 30 retained categories. The suggested training and testing of the TT100k dataset were defined respectively as the OD dataset and Test Data (TD), whereas the TSRD was defined as the PD dataset.

[1] The dataset generation tool used for this purpose is available at: https://github.com/jeiks/Stealing_DL_Models/tree/master/Copycat_CNN-Expansion/data/SIG30/dataset_creation.

4.3 Metrics

Each model was evaluated based on F1-Score using the TD dataset for the given problem. The F1-Score was chosen as it provides a balanced measure between precision and recall, making it particularly useful in scenarios where there is an uneven class distribution or when the cost of false positives and false negatives is high. After training and evaluating each surrogate model (Original Copycat Network and Few-Shot Copycat) for each problem, performance metrics were determined by calculating the ratio of the surrogate model's F1-score to the F1-score of the upper bound Oracle Network. Given that each surrogate model experiment was performed three times (three runs), the mean performance along with the standard deviation was calculated for each set of experiments.

4.4 Experiment

The objective of this experiment is to show that the blending process with a few samples of the problem domain improves the performance of the lower bound baseline, i.e., the Original Copycat Network trained with NPD images only. Therefore, five scenarios are explored for each problem, in which the adversary applies Few-Shot Copycat with 1, 2, 3, 5, and 10 randomly selected PD images per class. The generated models are called Few-Shot$_\lambda$, where λ represents the number of images the adversary has for each class of that problem.

The experiment was performed considering an NPD dataset with $100k$ images obtained from ImageNet [26] and using the framework[2] provided in [5]. The whole setup was performed 3 times to see the effect of random factors. For each run, the NPD images and the network parameters were randomly changed. However, within the same run, efforts were made to mitigate the effect of randomness across the different models in terms of the impact of PD image selection and the impact of NPD image class association. This effect may interfere with the interpretation of the experiment results. After conducting several preliminary tests, we observed that high values of α (Eq. 1) result in images that are too similar due to the low number of PD images. In contrast, low values of α generate images that fail to extract as many desired features from the Oracle. Therefore, although the weight α can vary between 0 and 1, we set $\alpha = 0.5$ in all experiments. Moreover, a binary search was conducted on Original Copycat to find the number of NPD images needed to achieve performance equivalent to our proposed technique with $100k$ blended images. This search started with $300k$ and extended up to $1.2M$ images.

Regarding the impact of PD image selection, after randomly selecting λ images from the PD, the Few-Shot$_\lambda$ models with fewer images (i.e., smaller λ) are randomly chosen from this previously selected set of images. To clarify, let's consider that $\lambda = 10$ images are initially selected at random from the PD. Thus, Few-Shot$_{10}$ comprises the subset S_{10} (i.e., S_λ where $\lambda = 10$). Subsequently, each new subset of images is randomly selected from the larger subset, i.e., $S_1 \subset S_2 \subset S_3 \subset S_5 \subset S_{10}$.

[2] https://github.com/jeiks/copycat_framework.

After each run with each λ, the histogram of the image labels from the few-shot fake datasets and the NPD datasets was analyzed to check the label distribution over the classes. This analysis was conducted to determine whether the proposed technique can alter the label of the NPD image based on the label of the blended PD image, thus improving the balance of the attack dataset. Since the Original Copycat [4] truncates the histogram to balance the data prior to training[3], the analysis was performed in the truncated histogram. The truncate threshold value is the number of images equally distributed over the classes (total number of images divided by the number of classes of the respective problem). The analysis was performed by comparing the standard deviation calculated again over the ideal mean (total number of images divided by the number of classes of the respective problem) of the normalized truncated histogram, see an example in Fig. 3 for the FER problem with 10 images per class. It is expected that the better balanced method (with proposed blending) presents a smaller standard deviation of the expected mean than the less balanced method (with random NPD without blending).

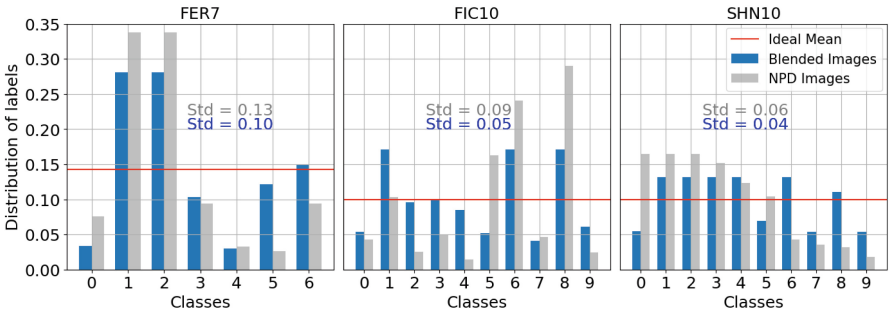


Fig. 3. An example of a normalized truncated histogram for the FER7, FIC10 and SHN10 problems with $\lambda = 10$, illustrating the distribution of labels across the classes relative to the ideal mean. Since the number of images is normalized, the standard deviation is calculated as $\sigma = \sqrt{\sum_{i=1}^{C} \left(\frac{c_i}{N} - \mu\right)^2}$, where C is the number of classes, c_i is the number of labels for the i-th class, N is the total number of images, and $\mu = \frac{1}{C}$ is the ideal mean.

Due to the potential impact of associating a domain class with an NPD image, it is undesirable for the same NPD image to be blended with different classes within the same set of experiments with varying values of λ. Therefore, to mitigate this effect, the NPD dataset was first shuffled and divided into subgroups with an equal number of images for each class of the target problem. Each

[3] In the Original Copycat approach, a threshold is determined based on the mean number of images per class. Images exceeding this threshold are randomly discarded, while those falling below it are randomly replicated.

subgroup was then associated with PD images of a respective class (Fig. 4). Furthermore, in the experiments, each subgroup was divided into 10 parts, which allows equal numbers of NPD images to be blended with each PD image when $\lambda = 10$. For smaller λ values, the segment of unused PD images is blended with those in use.

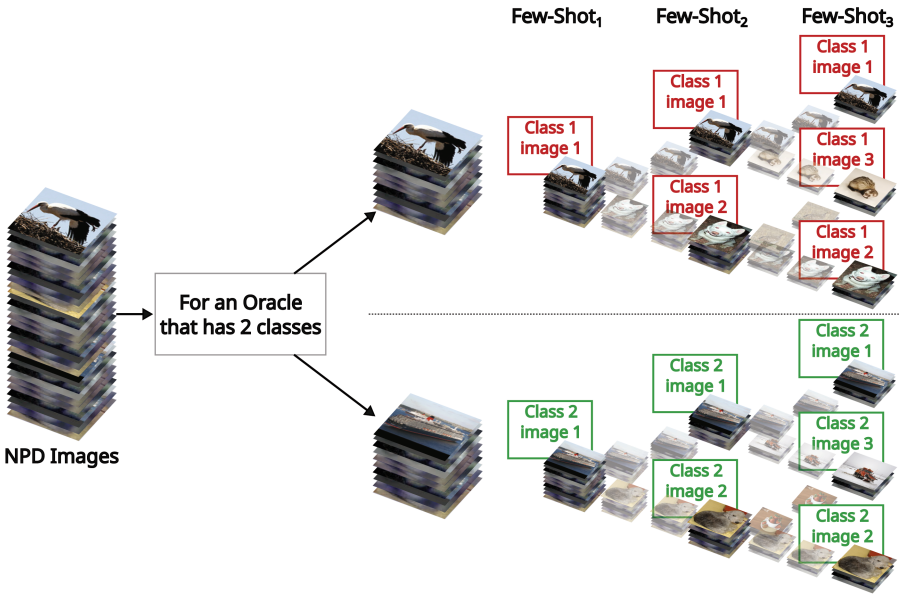


Fig. 4. Illustration of the process of distribution of the NPD images for blending with the PD images. The figure presents an illustrative example where NPD images are allocated for blending with PD images (1, 2, or 3 images per class, as shown in the illustration) within a problem involving 2 classes, each represented by a distinct color (red and green). Each PD image is represented by a square with a border colored according to its respective class. The process ensures a minimum variation with higher λ values being subset of the smaller values. For example, the first NPD is consistently paired with class 1 image 1 across all three λ values. (Color figure online)

4.5 General Setup

This work adopts the same architecture (Resnet-18 [8]) for both Oracle and the surrogate models. This choice aligns with the original research [4], which showed that the architecture does not substantially affect Copycat performance, as the method remains effective even with differing architectures for the Oracle and surrogate models. The training of all models used the Stochastic Gradient Descent (SGD) optimization algorithm with a Step-Down policy to adjust the learning rate. The epoch limit was determined through empirical experimentation, where

convergence in accuracy was consistently achieved within this specified range. The experimental setup included an Intel Core i7-10700 processor, 32 GB RAM, and an NVIDIA GeForce RTX 3060 with 12 GB memory, Ubuntu 22.04.3, CUDA 12.2, and cuDNN 8.9. The source code used was provided by the authors of the Original Copycat framework[4].

5 Experimental Results

The experimental results for each problem are shown in Table 1 and detailed in the following subsections, followed by a comprehensive discussion. For simplicity, the Few-Shot Copycat model will be referred to as the Few-Shot model throughout this section. The results are discussed in terms of average and standard deviation calculated over the three runs of each experiment combination.

The FER7 results are illustrated in Fig. 5. The Few-Shot models outperformed the Copycat baseline for all λ values. Even when using only 1 PD image per class ($\lambda = 1$), there was an improvement of 5.1%. The Copycat baseline achieved a performance of $71.5\% \pm 2.9$ over the Oracle, whereas the Few-Shot models exhibited performances of up to $87.0\% \pm 2.8$ over the Oracle. As presented in Fig. 6, much more than 600K NPD images (i.e., more than $6\times$ the number of queries) are required in the Original Copycat to achieve the Few-Shot Copycat performance.

The FIC10 results are illustrated in Fig. 5. Similarly to the FER7 results, the Few-Shot models outperformed the Copycat baseline for all λ values. There was an average improvement of 11.1% when using only 1 PD image per class ($\lambda = 1$). The Copycat baseline achieved performance of $79.2\% \pm 3.1$ over the Oracle, whereas the Few-Shot models exhibited performance of up to $92.6\% \pm 0.2$ over the Oracle. As presented in Fig. 6, around $1.2M$ NPD images (i.e., almost $12\times$ the number of queries) are required in the Original Copycat to achieve the Few-Shot Copycat performance.

The GOC9 results are illustrated in the Fig. 5. In contrast to the preceding results, the Few-Shot models did not improve the Copycat baseline. The Copycat baseline achieved a performance of $92.0\% \pm 0.4$ over the Oracle, whereas the best Few-Shot model achieved a performance of $90.3\% \pm 0.6$ with 10 images per class. One possible explanation is that the GOC9 is already close enough to the domain of ImageNet dataset used as NPD dataset, which would already result in good extraction performance with NPD.

The SHN10 results are illustrated in the Fig. 5. Similarly to the FER7 and FIC10 results, the Few-Shot models outperformed the Copycat baseline for all λ values. There was an average improvement of 2.4% when using only 1 PD image per class ($\lambda = 1$). The Copycat baseline achieved a performance of $93.4\% \pm 0.4$ over the Oracle, whereas the Few-Shot models exhibited performances of up to $97.3\% \pm 0.2$ over the Oracle. As presented in Fig. 6, around 600K NPD images (i.e., almost $6\times$ the number of queries) are required in the Original Copycat to achieve the Few-Shot Copycat performance.

[4] https://github.com/jeiks/Stealing_DL_Models.

The SIG30 results are illustrated in the Fig. 5. Similarly to the previous results (except for GOC9), the Few-Shot models outperformed the Copycat baseline for all λ values. There was an average improvement of 1.9% when using only 1 PD image per class ($\lambda = 1$). The Copycat baseline achieved a performance of $91.6\% \pm 2.1$ over the Oracle, whereas the Few-Shot models exhibited performances exceeding $98.1\% \pm 0.9$ over the Oracle. As presented in Fig. 6, around 1.2M NPD images (i.e., almost 12× the number of queries) are required in the Original Copycat to achieve the Few-Shot Copycat performance.

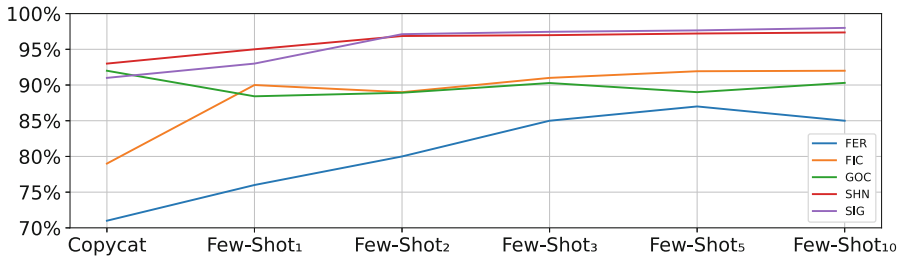


Fig. 5. Average extraction performance for the baseline and Few-Shot Copycat involved the utilization of 1, 2, 3, 5, and 10 images per class over the target network designed for the FER, FIC, GOC, SHN, and SIG problem.

5.1 Discussion

The general results are summarized in Table 1. There was an average improvement (considering all use case problems) over the baseline of 3.4 percentage point when adding 1 PD image per class only. This improvement can go up to 7.3 percentage point when adding 10 PD images per class. One possible reason for the improvement is that the blending process can alter the label of the NPD image, thus enhancing the balance of the dataset. As seen for all problems in Table 2, every standard deviation with the blending is smaller than its corresponding Original Copycat, showing a gain in the class balance. Moreover, there appears to be a pattern of improvement as more images from the problem domain are added to the extraction process, showing that the proposed Few-Shot Copycat is effective.

It is important to note that the Original Copycat already investigated using PD images to improve model extraction through fine-tuning. However, fine-tuning a CNN requires a decent number of images, much more than the few-shot proposed here. Preliminary results showed that fine-tuning with just a few images worsen the results. The results show that the proposed approach not only reduces the necessary number of images of the problem domain, but also avoids dealing with additional fine-tuning setup and training.

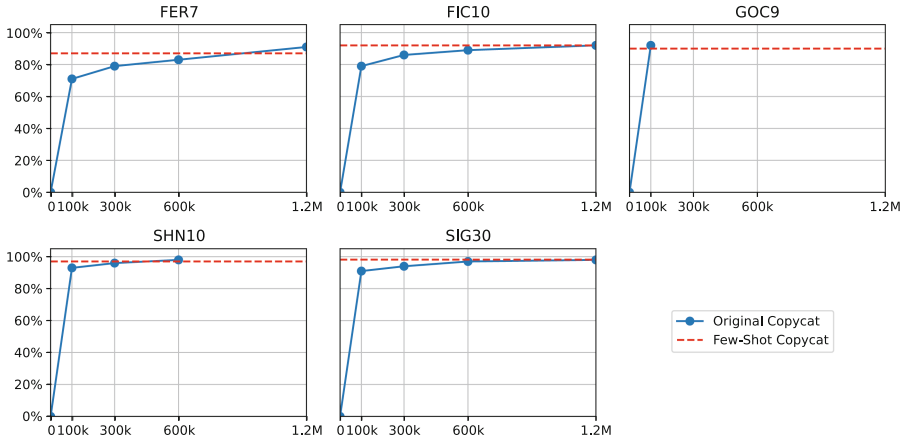


Fig. 6. Data curve performance of Original Copycat in a binary search starting with 300k images and going up to 1.2M to determine the number of NPD images required to achieve performance equivalent or better to the Few-Shot Copycat with 100k images. The vertical axis represents the network performance over the Oracle, and the horizontal axis represents the NPD dataset size.

Table 1. Summary of Experiment Averages and Standard Deviations.

Problem	Copycat	Few-Shot$_1$	Few-Shot$_2$	Few-Shot$_3$	Few-Shot$_5$	Few-Shot$_{10}$
FER7	$71.5\% \pm 2.9$	$76.6\% \pm 4.4$	$80.7\% \pm 2.6$	$85.1\% \pm 1.0$	$87.0\% \pm 2.8$	$85.5\% \pm 6.5$
FIC10	$79.2\% \pm 3.1$	$90.3\% \pm 0.2$	$89.7\% \pm 1.5$	$91.0\% \pm 0.3$	$91.9\% \pm 0.4$	$92.6\% \pm 0.2$
GOC9	$92.0\% \pm 0.4$	$88.4\% \pm 0.3$	$88.9\% \pm 0.6$	$90.2\% \pm 1.8$	$89.4\% \pm 0.9$	$90.3\% \pm 0.6$
SHN10	$93.4\% \pm 0.4$	$95.9\% \pm 0.3$	$96.8\% \pm 0.1$	$96.9\% \pm 0.2$	$97.2\% \pm 0.2$	$97.3\% \pm 0.2$
SIG30	$91.6\% \pm 2.1$	$93.5\% \pm 1.2$	$97.1\% \pm 0.6$	$97.4\% \pm 1.37$	$97.6\% \pm 0.4$	$98.1\% \pm 0.9$
Average	$85.5\% \pm 9.7$	$88.9\% \pm 7.4$	$90.6\% \pm 6.7$	$92.1\% \pm 5.1$	$92.6\% \pm 4.6$	$92.8\% \pm 5.2$

Table 2. Standard deviation calculated over the ideal mean considering the normalized truncated histogram of the attack dataset (NPD or few-shot dataset).

Problem	Copycat	Few-Shot$_1$	Few-Shot$_2$	Few-Shot$_3$	Few-Shot$_5$	Few-Shot$_{10}$
FER7	0.13	0.10	0.10	0.10	0.10	0.10
FIC10	0.09	0.05	0.05	0.05	0.05	0.05
GOC9	0.06	0.05	0.05	0.05	0.05	0.05
SHN10	0.06	0.03	0.02	0.03	0.03	0.04
SIG30	0.02	0.01	0.01	0.01	0.01	0.01

6 Conclusion

This paper introduced a novel Few-Shot Copycat approach to improve the Original Copycat Network by adding just a few samples of images of the problem domain in the model extraction process. The additional images are blended with NPD images so that the class space of the target problem is better sampled. With this, the proposed method can better copy blackbox models by only querying for hard-labels (i.e., the predicted class).

The results showed that our approach, Few-Shot Copycat, improved the average performance of the Copycat model from 85.5% to 92.8% with the addition of up to 10 problem domain images per class. Even with the addition of just 1 image per class, the performance jumps in average to 88.9%. Furthermore, it was demonstrated that the blending process can alter the class of the NPD, thereby improving class balance, as evidenced by the reduction in the standard deviation.

These numbers show the proposed approach is practical, reducing the number of queries needed to steal the knowledge of a model. For example, using this technique, an adversary performing $100K$ queries could copy (with 90% of performance) a model trained for facial expression recognition instead of performing around $600K$ queries with the Original Copycat Network. Considering this, companies offering their models as a service should prioritize robust measures to protect their models before making them available to users.

Future research should extend the applicability to different problem domains. The investigation should also bring insights about the necessary minimum number of images for performing the attack.

References

1. Chakraborty, A., Alam, M., Dey, V., Chattopadhyay, A., Mukhopadhyay, D.: A survey on adversarial attacks and defences. CAAI Trans. Intell. Technol. **6**(1), 25–45 (2021)
2. Cheng, M., Le, T., Chen, P.Y., Yi, J., Zhang, H., Hsieh, C.J.: Query-efficient hard-label black-box attack: an optimization-based approach. arXiv preprint arXiv:1807.04457 (2018)
3. Coates, A., Ng, A., Lee, H.: An analysis of single-layer networks in unsupervised feature learning. In: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, pp. 215–223. JMLR Workshop and Conference Proceedings (2011)
4. Correia-Silva, J.R., Berriel, R.F., Badue, C., De Souza, A.F., Oliveira-Santos, T.: Copycat CNN: are random non-labeled data enough to steal knowledge from black-box models? Pattern Recogn. **113**, 107830 (2021)
5. Correia-Silva, J.R., Berriel, R.F., Badue, C., de Souza, A.F., Oliveira-Santos, T.: Copycat CNN: stealing knowledge by persuading confession with random non-labeled data. In: 2018 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2018)
6. Gong, X., Chen, Y., Yang, W., Mei, G., Wang, Q.: Inversenet: augmenting model extraction attacks with training data inversion. In: IJCAI, pp. 2439–2447 (2021)

7. Han, G., Huang, S., Ma, J., He, Y., Chang, S.F.: Meta faster r-cnn: towards accurate few-shot object detection with attentive feature alignment. Proc. AAAI Conf. Artif. Intell. **36**, 780–789 (2022)
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
9. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
10. Huang, L.L., Yin, F.: Traffic sign recognition using perturbation method. In: Pattern Recognition: 6th Chinese Conference, CCPR 2014, Changsha, 17–19 November 2014. Proceedings, Part II, 6, pp. 518–527. Springer (2014)
11. Kesarwani, M., Mukhoty, B., Arya, V., Mehta, S.: Model extraction warning in MLAAS paradigm. In: Proceedings of the 34th Annual Computer Security Applications Conference, pp. 371–380 (2018)
12. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
13. Langner, O., Dotsch, R., Bijlstra, G., Wigboldus, D.H., Hawk, S.T., Van Knippenberg, A.: Presentation and validation of the radboud faces database. Cogn. Emot. **24**(8), 1377–1388 (2010)
14. Liu, T., Wang, K., Chang, B., Sui, Z.: A soft-label method for noise-tolerant distantly supervised relation extraction. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 1790–1795 (2017)
15. Lu, Y., Lu, J., Zhang, S., Hall, P.: Traffic signal detection and classification in street views using an attention model. Comput. Visual Media **4**, 253–266 (2018)
16. Lucey, P., Cohn, J.F., Kanade, T., Saragih, J., Ambadar, Z., Matthews, I.: The extended Cohn-Kanade dataset (CK+): a complete dataset for action unit and emotion-specified expression. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops, pp. 94–101. IEEE (2010)
17. Lundqvist, D., Flykt, A., Öhman, A.: Karolinska directed emotional faces. PsycTESTS Dataset **91**, 630 (1998)
18. Lyons, M., Akamatsu, S., Kamachi, M., Gyoba, J.: Coding facial expressions with Gabor wavelets. In: Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition, pp. 200–205. IEEE (1998)
19. Martinez, A., Benavente, R.: The AR face database: CVC technical report, 24 (1998)
20. Mosafi, I., David, E., Altshuler, Y., Netanyahu, N.S.: DNN intellectual property extraction using composite data. Entropy **24**(3), 349 (2022)
21. Mosafi, I., David, E.O., Netanyahu, N.S.: Stealing knowledge from protected deep neural networks using composite unlabeled data. In: 2019 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2019)
22. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y., et al.: Reading digits in natural images with unsupervised feature learning. In: NIPS Workshop on Deep Learning and Unsupervised Feature Learning, vol. 2011, p. 4. Granada (2011)
23. Orekondy, T., Schiele, B., Fritz, M.: Knockoff nets: stealing functionality of black-box models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4954–4963 (2019)
24. Pal, S., Gupta, Y., Shukla, A., Kanade, A., Shevade, S., Ganapathy, V.: Activethief: model extraction using active learning and unannotated public data. Proc. AAAI Conf. Artif. Intell. **34**, 865–872 (2020)

25. Pantic, M., Valstar, M., Rademaker, R., Maat, L.: Web-based database for facial expression analysis. In: 2005 IEEE International Conference on Multimedia and Expo, p. 5. IEEE (2005)
26. Russakovsky, O., et al.: Imagenet large scale visual recognition challenge. Int. J. Comput. Vision **115**, 211–252 (2015)
27. Shi, Y., Sagduyu, Y., Grushin, A.: How to steal a machine learning classifier with deep learning. In: 2017 IEEE International Symposium on Technologies for Homeland Security (HST), pp. 1–5. IEEE (2017)
28. Szegedy, C., et al.: Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 (2013)
29. Tang, Z., Wang, D., Zhang, Z.: Recurrent neural network training with dark knowledge transfer. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5900–5904. IEEE (2016)
30. Tramèr, F., Zhang, F., Juels, A., Reiter, M.K., Ristenpart, T.: Stealing machine learning models via prediction {APIs}. In: 25th USENIX Security Symposium (USENIX Security 16), pp. 601–618 (2016)
31. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747 (2017)
32. Yadav, S.S., Jadhav, S.M.: Deep convolutional neural network based medical image classification for disease diagnosis. J. Big Data **6**(1), 1–18 (2019)
33. Yin, L., Wei, X., Sun, Y., Wang, J., Rosato, M.J.: A 3D facial expression database for facial behavior research. In: 7th International Conference on Automatic Face and Gesture Recognition (FGR06), pp. 211–216. IEEE (2006)
34. Zhao, J., Chen, Y., Zhang, W.: Differential privacy preservation in deep learning: challenges, opportunities and solutions. IEEE Access **7**, 48901–48911 (2019)

Learning Using Generated Privileged Information by Text-to-Image Diffusion Models

Rafael-Edy Menadil, Mariana-Iuliana Georgescu, and Radu Tudor Ionescu(✉)

Department of Computer Science, University of Bucharest, Bucharest, Romania
raducu.ionescu@gmail.com

Abstract. Learning Using Privileged Information is a particular type of knowledge distillation where the teacher model benefits from an additional data representation during training, called privileged information, improving the student model, which does not see the extra representation. However, privileged information is rarely available in practice. To this end, we propose a text classification framework that harnesses text-to-image diffusion models to generate artificial privileged information. The generated images and the original text samples are further used to train multimodal teacher models based on state-of-the-art transformer-based architectures. Finally, the knowledge from multimodal teachers is distilled into a text-based (unimodal) student. Hence, by employing a generative model to produce synthetic data as privileged information, we guide the training of the student model. Our framework, called Learning Using Generated Privileged Information (LUGPI), yields noticeable performance gains on four text classification data sets, demonstrating its potential in text classification without any additional cost during inference.

Keywords: Learning Using Privileged Information · Knowledge Distillation · Text Classification · Diffusion Models · Data Augmentation

1 Introduction

In the quest of developing effective and efficient machine learning models, researchers developed the knowledge distillation framework [5,21], in which the outputs of one [5,31] or more [21,46,47] typically heavy models, called *teachers*, are used as target for a typically lightweight model, called *student*. This framework is primarily used to compress very deep models into shallower, yet effective models [12,27,31,44,46,47]. A secondary use of knowledge distillation is to leverage additional data representations, available only at training time, to improve the performance of a model which does not have access to the extra representation. This latter framework, called Learning Using Privileged Information (LUPI) [42], was introduced well before the era of deep learning, but it was later shown [27] that it represents a particular kind of knowledge distillation.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 423–438, 2025.
https://doi.org/10.1007/978-3-031-78183-4_27

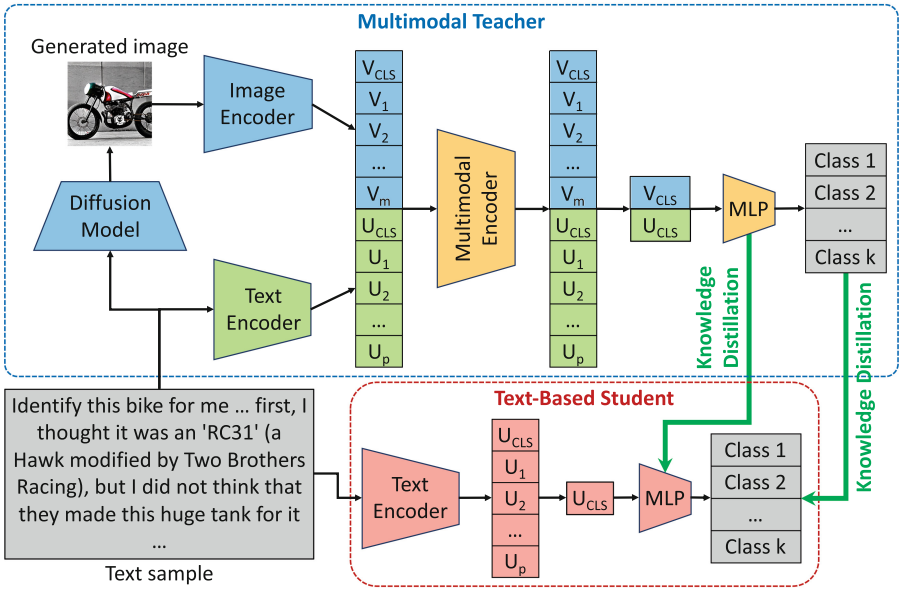


Fig. 1. An illustration of our Learning Using Generated Privileged Information (LUGPI) framework. For each text sample, a diffusion model generates an image. The original text sample and the generated image are used to train a multimodal teacher model. Then, a text-based student model is trained via knowledge distillation from the teacher. The distillation is carried out at two levels.

Although LUPI is an interesting and useful framework, it has rarely been applied in solving mainstream machine learning problems [1,16,17,23,42,50], since finding additional modalities to represent the training data is not an easy task. With the advent of diffusion models [6,22,40,41], which demonstrated impressive capabilities in generating realistic and diverse images based on text prompts [3,20,34,35], we can now automatically generate image representations of text samples without much effort. To this end, we propose a novel framework called Learning Using Generated Privileged Information (LUGPI), which harnesses a state-of-the-art text-to-image diffusion model to generate the privileged information, namely Stable Diffusion v2 [34]. Our framework is applied on text classification tasks, where the original modality is represented by text samples and the additional modality is represented by images. Next, we train multimodal teacher models based on combining state-of-the-art transformer-based architectures, such as Distilled Bidirectional Encoder Representations from Transformers (DistilBERT) [37], Vision Transformer (ViT) [11] and Contrastive Language-Image Pre-Training (CLIP) [33]. Remarkably, we find that our multimodal teachers outperform the standalone text-based (unimodal) model. However, employing the multimodal teachers during inference would inherently imply the use of the diffusion model to generate the images. This greatly impacts the inference time of the whole framework, since diffusion models are notoriously known

for being computationally expensive [6]. For instance, Stable Diffusion v2 [34] comprises about 865 million learnable parameters, requiring about 17 seconds to generate a single image on an NVIDIA GeForce RTX 3090 24GB GPU. To address this limitation, we distill the knowledge from a multimodal teacher into a text-based student model, as shown in Figure 1. This completely eliminates the need to generate images during inference. Thus, LUGPI does not increase the computational cost at test time.

We carry out experiments on four text classification data sets to evaluate the proposed framework and compare it with the conventional training approach based on pre-training and fine-tuning, while preserving the underlying DistilBERT architecture [37]. Our empirical results indicate that LUGPI brings significant performance gains on all four data sets.

In summary, our contribution is threefold:

- We propose to harness diffusion models in order to artificially generate an extra data modality in the form of images, complementing the text modality, which enables us to train more powerful multimodal neural models.
- We introduce the novel Learning Using Generated Privileged Information framework to distill knowledge from our multimodal teachers into text-based (unimodal) students.
- We conduct experiments on four benchmarks, showing that the proposed framework improves the accuracy rates of text-based models by noticeable margins, without any extra cost during inference.

2 Related Work

Learning Using Privileged Information. There are two types of knowledge distillation frameworks, which were independently introduced in literature, namely model compression [5,21] and learning using privileged information [42]. In 2016, Lopez-Paz et al. [27] unified the model compression and learning under privilege information paradigms into the knowledge distillation framework.

The model compression technique [5,21] is mainly aimed at training a shallow and efficient student architecture using one or more deeper and powerful teachers. In this way, a shallow student could benefit from the knowledge gained by a deep teacher, while having less parameters, and consequently, a lower running time during inference.

The learning using privileged information paradigm [42] was introduced to transfer the knowledge from a teacher model, which is trained with privileged information, to a student model, which does not have access to the privileged data. In this scenario, the teacher and the student can share the same architecture, the main difference being the data used to train the two models. Many recent works [1,14–17,25,26,48] applied the LUPI framework to improve the performance of the student without using additional information at test time. For example, Yuan et al. [48] trained a student to estimate the 3D hand pose using only the RGB image at test time. The knowledge about the depth channel was transferred from the teacher during the knowledge distillation process.

Alehdaghi et al. [1] decreased the gap between RGB and infrared images used in the person re-identification task by applying the LUPI framework. They proposed to create an intermediate virtual domain that acts as a bridge between the two image modalities. The intermediate virtual domain was used as privileged information for the student model during training. Georgescu et al. [17] applied LUPI for facial expression recognition under strong occlusion, where the teacher learns from completely visible faces, but the student can only use occluded faces as input. They later extended their approach to age estimation and gender prediction from faces [16].

Similar to the aforementioned works [1,14–17,25,26,48], we use extra data as privileged information during training. Different from the related studies on LUPI, our method does not require the existence of additional representations, since it generates the privileged data using a generative diffusion model. Hence, our framework broadens the applicability of LUPI to text-based corpora that do not have additional representations of the data samples.

Data augmentation. Our approach can also be seen as a rather unconventional data augmentation technique. However, data augmentation is usually employed to improve the robustness to data variation [9,49], while in our case, we employ it to obtain privileged information. In general, data augmentation plays an important role in increasing the performance of deep learning architectures [9,49], especially when the available training data is limited. The most common data augmentation methods used in computer vision are methods based on rotating, cropping and flipping the images [7]. Although techniques like these can offer better performance than just training on the original data, they lack the capability of creating a completely different data point, instead relying on the existing data and manipulating it just enough to have a variety within the training data.

In recent years, we have seen generative models, such as Generative Adversarial Networks (GANs) [19] and diffusion models, that have been used to successfully augment data and improve the accuracy of various models [2,4,32,36]. Generative models can create new data points that closely resemble the training data distribution, often being mistaken with natural data points. Therefore, classification models can leverage this new data variety to offer high performance without having to gather any new data points. Furthermore, there are some examples that successfully use generative models when conventional techniques fall short [39,43]. Yang et al. [43] proposed to use diffusion models to generate images illustrating human-object interactions, conditioned by prompts explaining the interactions. Shivashankar et al. [39] trained a GAN model to generate images along with their segmentation label for medical and face segmentation data sets. In these cases, conventional data augmentation methods provide suboptimal results when compared with generative models. This is because the latter models can generate new data points that resemble the training data distribution, aside from being able to generate variations of existing data points conditioned by some specific features that need to be present in the generated output.

Unlike other data augmentation techniques, we propose to generate image-based representations from text samples, essentially obtaining a new modality. Thus, our technique requires employing multimodal models to benefit from the extra data representation. To return to using a unimodal input while keeping the benefits of the multimodal data, we employ knowledge distillation.

3 Method

Overview and motivation. Learning Using Privileged Information [42] is suitable for machine learning tasks where the training data is represented by multiple modalities. However, the majority of machine learning problems only involve a single modality, rendering LUPI inapplicable. To overcome this challenge in the area of natural language processing and text classification, we propose to utilize a text-to-image diffusion model to generate privileged information in the form of images, in order to solve text classification problems where privileged information is not typically available.

We believe that our proposal is grounded in how the human mind works. For instance, humans use their imagination to mentally visualize objects, colors, textures or other visual aspects evoked in a text. This process helps humans in reaching a better and deeper text comprehension [13]. In a similar way, we conjecture that imaginary pictures can boost the performance of neural models such as BERT [8] or DistilBERT [37], provided that the visualizations are sufficiently representative. To increase the chances of successfully implementing our proposal, we make use of diffusion models, which are considered by many researchers as state-of-the-art text-to-image generators [10], surpassing previous models based on GANs.

To harness the generated images, a straightforward approach is to employ models on both text and image modalities in order to improve text classification performance. However, this approach is suboptimal in terms of speed, requiring additional time to generate and process images during inference. Our framework addresses this issue through knowledge distillation, i.e. the knowledge learned by the multimodal model, called teacher, is distilled into a text-based model, called student. At test time, we employ the student model to make predictions, thus eliminating the need to generate and process images. Our training framework is formally introduced in Algorithm 1. We first introduce the notations, then continue by presenting the three stages of our algorithm, namely image generation, teacher model training and knowledge distillation.

Notations. Let $\mathcal{D} = (X, Y) = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$ represent a training set of text samples, where n is the number of samples in the data set, and y_i is the ground-truth label associated with text sample x_i. Let T and θ_T represent the multimodal teacher model and its weights, respectively. Similarly, let S and θ_S represent the text-based student model and its weights. The weights of the teacher and student models are updated using the learning rates η_T and η_S, respectively. Let $X' = \{x'_1, x'_2, ..., x'_n\}$ represent the set of images generated by a diffusion model G with the weights θ_G. Let $\mathcal{N}(\mu, \sigma^2)$ represent the normal

Algorithm 1: Learning Using Generated Privileged Information

Input: $\mathcal{D}$ - the training set of labeled text samples, G - the text-conditional diffusion model, θ_G - the weights of the diffusion model, T - the multimodal teacher model, S - the student model, θ_T^* - (optional) pre-trained weights for the teacher, θ_S^* - (optional) pre-trained weights for the student, η_T - the teacher's learning rate, η_S - the student's learning rate, α - the importance of the cross-entropy between the teacher and the student, β - the importance of the mean squared error between the teacher and student embeddings.

Output: θ_S - the trained weights of the student model.

1 $n \leftarrow |\mathcal{D}|$; ◁ get the number of training samples
2 $X' \leftarrow \emptyset$; ◁ initialize the set of generated images
3 **foreach** $i \in \{1, 2, ..., n\}$ **do**
4 $\quad x_i' \leftarrow G(x_i, \theta_G)$; ◁ generate an image for the text sample x_i
5 $\quad X' \leftarrow X' \cup \{x_i'\}$; ◁ add the generated image to the set X'
6 **if** $\theta_T^* \neq \emptyset$ **then**
7 $\quad \theta_T \leftarrow \theta_T^*$; ◁ initialize weights of teacher using pre-trained weights
8 **else**
9 $\quad \theta_T \sim \mathcal{N}\left(0, \frac{2}{d_{in}+d_{out}}\right)$; ◁ initialize weights of teacher using Xavier init [18]
10 **repeat**
11 $\quad$ **foreach** $i \in \{1, 2, ..., n\}$ **do**
12 $\quad\quad t_i \leftarrow T(x_i, x_i', \theta_T)$; ◁ get class probabilities predicted by the teacher
13 $\quad\quad \theta_T \leftarrow \theta_T - \eta_T \cdot \nabla \mathcal{L}_{\text{CE}}(y_i, t_i)$; ◁ train the teacher using cross-entropy
14 **until** *convergence*;
15 **if** $\theta_S^* \neq \emptyset$ **then**
16 $\quad \theta_S \leftarrow \theta_S^*$; ◁ initialize weights of student using pre-trained weights
17 **else**
18 $\quad \theta_S \sim \mathcal{N}\left(0, \frac{2}{d_{in}+d_{out}}\right)$; ◁ initialize weights of student using Xavier init [18]
19 **repeat**
20 $\quad$ **foreach** $i \in \{1, 2, ..., n\}$ **do**
21 $\quad\quad t_i, e_i^T \leftarrow T(x_i, x_i', \theta_T)$; ◁ get probabilities and embedding from teacher
22 $\quad\quad s_i, e_i^S \leftarrow S(x_i, \theta_S)$; ◁ get probabilities and embedding from student
23 $\quad\quad \mathcal{L}_{\text{KD}} \leftarrow \mathcal{L}_{\text{CE}}(y_i, s_i) + \alpha \cdot \mathcal{L}_{\text{CE}}^T(t_i, s_i) + \beta \cdot \mathcal{L}_{l_2}^T(e_i^T, e_i^S)$; ◁ apply Eq. (2)
24 $\quad\quad \theta_S \leftarrow \theta_S - \eta_S \cdot \nabla \mathcal{L}_{\text{KD}}$; ◁ train the student using the joint loss
25 **until** *convergence*;

distribution of mean μ and standard deviation σ. Let e_i^T and e_i^S denote the embedding vectors produced by the teacher and the student for the i-th data sample, respectively. The embedding vectors are taken just before the classification layer of each model.

Image generation. In steps 2-5 of Algorithm 1, we utilize a pre-trained text-to-image diffusion model to generate privileged information in the form of images. In step 4, the generator G generates an image denoted by x_i' conditioned on the

text sample x_i. In step 5, the generated image is added to the set X'. Steps 4 and 5 are repeated until all training examples are passed through G.

We choose the Stable Diffusion v2 [34] model trained on the LAION-5B [38] data set as our generator G. The use of this model is chosen in favor of another open-source diffusion model, namely GLIDE [30]. To decide on which generator to use, we visually inspected their outputs on a subset of 100 prompts from the chosen data sets. We observed that Stable Diffusion v2 is usually better aligned with the provided text prompts than GLIDE. This influenced our decision towards using the former model.

Teacher training. The second stage of our pipeline is dedicated to training the teacher model. This stage corresponds to steps 6-14 of Algorithm 1. The teacher model is a multimodal architecture comprising three transformer-based encoders: a text encoder, an image encoder, and a multimodal encoder. As illustrated in Figure 1, the tokens produced by the text encoder are concatenated with the tokens given by the image encoder. The concatenated set of tokens is further passed through the multimodal encoder, which comprises a vanilla transformer block based on multi-head attention, having 8 attention heads. The multimodal encoder learns to perform cross-modal attention, strengthening relations across the text and image modalities. From the resulting set of multimodal tokens, we keep the classification token U_{CLS} from the text modality and the classification token V_{CLS} from the image modality, discarding the other tokens. This is a conventional procedure when transformers are applied to downstream classification tasks [8,11]. Next, the classification tokens are concatenated and given as input to a multi-layer perceptron (MLP) with two layers, where the first layer comprises 786 neurons and the second one comprises k neurons, where k is the number of classes. A softmax function computes the output probabilities.

In order to make the prediction t_i, the teacher model T takes the text sample x_i and the generated image x_i' as input, according to step 12 of Algorithm 1. In step 13, the weights of the teacher θ_T are updated using gradient descent, where the gradient is computed with respect to the cross-entropy loss. For the vector of predicted class probabilities t_i and the one-hot label encoding y_i, the cross-entropy loss is given by:

$$\mathcal{L}_{\text{CE}}(y_i, t_i) = -\sum_{j=1}^{k} y_{ij} \cdot \log(t_{ij}), \forall i \in \{1, 2, ..., n\}, \tag{1}$$

where k is the number of classes.

In our implementation, we choose to use pre-trained architectures for the text and image encoders. For a fair and representative evaluation, we use the same text encoder as the baseline and the student models, namely DistilBERT [37]. This is to ensure that the observed performance gains are not due to the use of a more powerful text encoder for the teacher model, but rather due to the extra image modality. For the image encoder, we consider two alternative architectures, namely ViT [11] and CLIP Image [33].

Knowledge distillation. After training the teacher, we apply the knowledge distillation procedure to transfer the knowledge from the multimodal teacher to the student. This stage corresponds to steps 15-25 of Algorithm 1. According to steps 15-18, the student can optionally be pre-trained in a standard fashion, prior to the knowledge distillation procedure. We utilize this option to ensure a fair comparison with the baseline model. More precisely, both the baseline DistilBERT and our student DistilBERT are pre-trained. In general, when there are no pre-trained weights for the student, we can simply initialize the model using a conventional approach (step 18), such as Xavier initialization [18].

The student model is jointly optimizing three objectives. On the one hand, the student has to minimize the cross-entropy loss with respect to the ground-truth (hard) labels, to ensure that its predictions are correct. On the other hand, the student has to optimize the cross-entropy with respect to the probabilities (soft labels) predicted by the teacher, as well as minimize the mean squared error between the corresponding embeddings produced by the teacher and the student, which enables the student to learn knowledge from the teacher model. Formally, for the i-th data sample, the joint objective is computed as follows:

$$\begin{aligned}\mathcal{L}_{KD} &= \mathcal{L}_{\text{CE}}(y_i, s_i) + \alpha \cdot \mathcal{L}^T_{\text{CE}}(t_i, s_i) + \beta \cdot \mathcal{L}^T_{l_2}(e^T_i, e^S_i) \\ &= -\sum_{j=1}^{k} y_{ij} \cdot \log(s_{ij}) - \alpha \cdot \sum_{j=1}^{k} t_{ij} \cdot \log(s_{ij}) + \beta \cdot \|e^T_i - e^S_i\|^2_2, \forall i \in \{1, ..., n\},\end{aligned} \tag{2}$$

where $\alpha, \beta \geq 0$ are two hyperparameters that control the importance of the knowledge distillation objectives. Note that the distillation is carried out at two levels, namely with respect to the embedding space and the output space. Our ablation study shows the importance of distilling knowledge at both levels.

4 Experiments

We conduct experiments on four data sets covering three tasks: opinion mining, text categorization by topic, and complex word identification. The data sets are chosen to provide a comprehensive evaluation of image generation and privileged information in different target tasks.

4.1 Data Sets

IMDB Large Movie Review. The IMDB Large Movie Review data set [29] is a well-known benchmark for polarity classification, which is composed of 50,000 movie reviews separated into 25,000 for training and 25,000 for testing. We keep 10% of the training set for validation purposes. The scope of this data set is to predict the polarity of the sentiment (positive or negative).

20 Newsgroups. The 20 Newsgroups data set [24] is a popular benchmark for text categorization by topic. It comprises 18,828 documents that are assigned to one of 20 different categories, ranging from technology to sports and religion. In

our experiments, we divide the data set into 11,353 training documents, 1,261 validation documents and 6,214 test documents.

English News. The English News corpus [45] comprises 17,861 sentences with marked words or multi-word phrases that are annotated with complexity levels by native and non-native English speakers. The task is to determine if the target words or multi-word phrases are complex or not. The corpus is divided into 14,002 training sentences, 1,764 validation sentences and 2,095 test sentences.

English WikiNews. Another corpus for complex word identification introduced by Yimam et al. [45] is English WikiNews. It has a similar format to English News. The English WikiNews data set is divided into 7,746 training sentences, 870 validation sentences and 1,287 test sentences.

Table 1. Accuracy rates on IMDB Large Movie Review [29], 20 Newsgroups [24], English News [45] and English WikiNews [45] data sets. Our teacher and student models are compared with the fine-tuned vanilla DistilBERT [37]. For reference, we report results with the independent image encoders, namely ViT [11] and CLIP [33]. The best accuracy on each corpus is highlighted in bold. Significantly better results (at a p-value of 0.001) based on McNemar / Cochran Q testing are marked with ‡.

Model	Modality		IMDB	20 News	English	English
	Text	Image	Reviews	groups	News	WikiNews
DistilBERT [37]	3		0.919	0.918	0.861	0.842
ViT [11]		3	0.559	0.137	0.832	0.754
CLIP Image [33]		3	0.549	0.523	0.822	0.746
DistilBERT+ViT (Teacher 1)	3	3	0.920	0.919	0.867‡	0.843
DistilBERT+CLIP (Teacher 2)	3	3	0.931‡	0.926‡	0.868‡	0.846
DistilBERT (Student 1)	3		0.930‡	0.928‡	0.869‡	0.843
DistilBERT (Student 2)	3		**0.931**‡	**0.929**‡	**0.871**‡	**0.848**‡

4.2 Experimental Setup

Baselines and backbones. As baseline, we choose the DistilBERT model [37], a variant of BERT [8] that exhibits good performance with a reasonable number of learnable parameters. For a fair comparison with the baseline, we employ the DistilBERT architecture for our students as well. Moreover, the text encoder inside the multimodal teachers is also based on DistilBERT. To encode the generated images, we alternatively employ the pre-trained image encoder of the CLIP architecture [33], or the pre-trained ViT [11] model. We thus obtain a teacher based on DistilBERT+ViT (Teacher 1), and a teacher based on DistilBERT+CLIP (Teacher 2). We distill the knowledge from Teacher 1 into a student based on DistilBERT (Student 1), and the knowledge from Teacher 2

into a different student (Student 2), which is also based on DistilBERT. We underline that the two students have the same architecture, but they differ in terms of the source providing the privileged information.

Hyperparameters. We train the models with the AdamW [28] optimizer using a learning rate of $5{\cdot}10^{-5}$ with linear decay, which converges to good optima across all our experiments. The baseline DistilBERT, the teachers and the students are each trained for 100 epochs on an Nvidia GeForce GTX 1080Ti GPU with 11 GB of VRAM. In all the experiments, we use a mini-batch size of 14 samples. Following previous works on knowledge distillation [5,27], we soften the output of the teacher using the temperature τ. We validate this hyperparameter in the range 1-10, achieving optimal results with $\tau = 8$. The hyperparameters α and β from Eq. (2) are validated in the range from 0.1 to 5. The optimal values are $\alpha = 3$ and $\beta = 1$.

Data preprocessing. Before generating images with Stable Diffusion v2 [34], we perform some preprocessing steps to clean up the text samples. For the IMDB data set, we remove the HTML tags that are sometimes present in movie reviews. For the 20 Newsgroups data set, we discard email addresses and subjects, using the remaining content as text prompt. For the English News and English WikiNews data sets, we provide the target word or multi-word phrase in each sentence as input for the text-conditional diffusion model. This is because the task is to identify the complexity of the target words, not of the whole sentences.

To process the examples from the English News and English WikiNews corpora with DistilBERT, we modify each sentence by marking the target words or multi-word phrases with the *[SEP]* token. No further preprocessing is required for the other data sets.

4.3 Results

We present the results obtained on the IMDB, 20 Newsgroups, English News and English WikiNews data sets in Table 1.

IMDB. The baseline DistilBERT model [37], which is trained using only text data, reaches an accuracy of 91.9%, while the image encoders barely surpass the random chance baseline. The best multimodal teacher employing the CLIP image encoder reaches an accuracy of 93.1%. Our first student outperforms its teacher by 1%, while our second student is on par with its teacher. Notably, both students surpass the baseline model by more than 1.1%.

20 Newsgroups. The baseline DistilBERT [37] obtains a performance of 91.8%, while the individual image encoders lag far behind. Since ViT is much worse than CLIP, the corresponding teacher (DistilBERT+ViT) barely surpasses the baseline model, while DistilBERT+CLIP (Teacher 2) reaches an accuracy of 92.6%. Meanwhile, our students based on privileged information surpass their teachers, showing considerable performance gains over the baseline DistilBERT.

English News. On the English News corpus, the baseline DistilBERT obtains an accuracy of 86.1%. The ViT and CLIP image encoders obtain competitive results, being less than 4% behind DistilBERT [37]. Both multimodal teachers outperform the baseline DistilBERT. Moreover, our student models surpass their teachers. The best student outperforms the baseline DistilBERT by 1%, reaching an accuracy of 87.1% in complex word identification.

English WikiNews. The results on the English WikiNews corpus are consistent with those on the English News corpus. Indeed, the independent image encoders obtain fairly good results, given that they only take generated images as input. The multimodal teachers outperform the baseline DistilBERT, while the students yield even better results.

Fig. 2. Text samples and generated images that are correctly classified by the multimodal teacher based on DistilBERT+CLIP. The target label is displayed on top of each sample. The examples on top belong to the 20 Newsgroups [24] data set, while the examples below are taken from English News [45] and English WikiNews [45].

Overall. We notice that the text modality leads to better results than the image modality, regardless of the data set. This is a natural consequence of the fact that the images are generated by a diffusion model, which can produce images that do not reflect the label. Another generic observation is that the multimodal teacher based on the CLIP image encoder (Teacher 2) is generally better than the other teacher. This leads to a better DistilBERT student (Student 2). Furthermore, we observe that the students generally surpass their teachers. We explain this observation through the fact that the multimodal teachers assign equal importance to the text and image modalities, although the image modality is naturally inferior. In contrast, the students focus on the original text modality, obtaining information about the image modality only through knowledge distillation.

Since both students surpass the baseline DistilBERT in each and every case, we conclude that our LUGPI framework is beneficial in various text classification tasks, such as polarity classification, text categorization by topic, and complex word identification.

Qualitative results. In Figure 2, we illustrate some examples which are incorrectly classified by the baseline DistilBERT, but are correctly classified by our second teacher model (DistilBERT+CLIP). Remarkably, we observe that the images generated by Stable Diffusion v2 contain important clues. For instance, a car is generated when the prompt is about cars, even though the word "car" is never mentioned inside the prompt. For the complex word identification task, we observe that the images generated for simple (non-complex) words tend to be less abstract, while those generated for complex words tend to be more abstract. In summary, the illustrated examples show that the generated images can complement the corresponding text samples. Although our students do not see these images at test time, our quantitative results presented in Table 1 show that the students clearly benefit from the privileged information transferred from the multimodal teachers.

Ablation study. Our LUGPI framework performs the distillation at two network levels, via two distinct loss terms. To demonstrate the utility of both terms, we perform an ablation study of the knowledge distillation loss terms $\mathcal{L}^T_{\text{CE}}$ and $\mathcal{L}^T_{l_2}$ from Eq. (2). We present the corresponding results in Table 2. Distilling knowledge at the output level via $\mathcal{L}^T_{\text{CE}}$ is not beneficial for the first student. In contrast, distilling knowledge at the embedding level via $\mathcal{L}^T_{l_2}$ helps both students on three data sets (except IMDB). In summary, the ablation study shows that both distillation losses are required to obtain consistent improvements.

Training and inference time. The inference time of our final model is identical to that of the vanilla DistilBERT. However, the training time of our pipeline is between 2.3× and 2.8× higher (depending on the dataset and the vision model) than that of the student. This includes the time for generating the images with the pre-trained Stable Diffusion model. Note that Stable Diffusion is kept frozen in our pipeline.

Table 2. Accuracy rates on IMDB Large Movie Review [29], 20 Newsgroups [24], English News [45] and English WikiNews [45] data sets, while ablating the knowledge distillation components of our loss defined in Eq. (2). The best accuracy on each corpus is highlighted in bold.

Model	Loss Terms		IMDB	20 News	English	English
	$\mathcal{L}_{\text{CE}}^{T}$	$\mathcal{L}_{l_2}^{T}$	Reviews	groups	News	WikiNews
DistilBERT (Student 1)			0.919	0.918	0.861	0.842
DistilBERT (Student 2)			0.919	0.918	0.861	0.842
DistilBERT (Student 1)	3		0.913	0.922	0.765	0.842
DistilBERT (Student 2)	3		0.923	0.926	0.870	0.844
DistilBERT (Student 1)		3	0.911	0.926	0.869	0.840
DistilBERT (Student 2)		3	0.919	0.925	0.865	0.843
DistilBERT (Student 1)	3	3	0.930	0.928	0.869	0.843
DistilBERT (Student 2)	3	3	**0.931**	**0.929**	**0.871**	**0.848**

5 Conclusion

In this work, we proposed the Learning Using Generated Privileged Information framework, which employs a diffusion model to generate privileged images, which were further used to train a multimodal teacher taking both text and image data as input. A unimodal student was subsequently trained by distilling privileged information from the multimodal teacher. We performed experiments on four text classification data sets, namely IMDB Movie Reviews, 20 Newsgroups, English News and English WikiNews. We alternatively employed two different image encoders to extract image features, demonstrating accuracy gains in both cases. All our distilled students outperformed the baseline model and even the multimodal teachers, without any extra cost during inference. In future work, we aim to extend our framework to more NLP tasks.

References

1. Alehdaghi, M., Josi, A., Cruz, R.M.O., Granger, E.: Visible-Infrared Person Re-Identification Using Privileged Intermediate Information. In: Proceedings of ECCVW. pp. 720–737 (2022)
2. Antoniou, A., Storkey, A., Edwards, H.: Augmenting image classifiers using data augmentation generative adversarial networks. In: Proceedings of ICANN. pp. 594–603 (2018)
3. Avrahami, O., Lischinski, D., Fried, O.: Blended diffusion for text-driven editing of natural images. In: Proceedings of CVPR. pp. 18208–18218 (2022)
4. Azizi, S., Kornblith, S., Saharia, C., Norouzi, M., Fleet, D.J.: Synthetic Data from Diffusion Models Improves ImageNet Classification. arXiv preprint arXiv:2304.08466 (2023)
5. Ba, J., Caruana, R.: Do deep nets really need to be deep? In: Proceedings of NIPS. pp. 2654–2662 (2014)

6. Croitoru, F.A., Hondru, V., Ionescu, R.T., Shah, M.: Diffusion models in vision: A survey. IEEE Trans. Pattern Anal. Mach. Intell. **45**(9), 10850–10869 (2023)
7. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.: RandAugment: Practical Automated Data Augmentation with a Reduced Search Space. In: Proceedings of NeurIPS. vol. 33, pp. 18613–18624 (2020)
8. Devlin, J., Chang, M.W., Lee, K., Toutanova, L.K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In: Proceedings of NAACL-HLT. pp. 4171–4186 (2019)
9. DeVries, T., Taylor, G.W.: Improved Regularization of Convolutional Neural Networks with Cutout. arXiv preprint arXiv:1708.04552 (2017)
10. Dhariwal, P., Nichol, A.: Diffusion models beat GANs on image synthesis. In: Proceedings of NeurIPS. vol. 34, pp. 8780–8794 (2021)
11. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In: Proceedings of ICLR (2021)
12. Feng, Y., Wang, H., Hu, R., Yi, D.T.: Triplet distillation for deep face recognition. In: Proceedings of ICIP. pp. 808–812 (2020)
13. Gambrell, L.B., Jawitz, P.B.: Mental Imagery, Text Illustrations, and Children's Story Comprehension and Recall. Read. Res. Q. **28**, 264–276 (1993)
14. Gao, Z., Wu, S., Liu, Z., Luo, J., Zhang, H., Gong, M., Li, S.: Learning the implicit strain reconstruction in ultrasound elastography using privileged information. Med. Image Anal. **58**, 101534 (2019)
15. Garcia, N.C., Morerio, P., Murino, V.: Learning with privileged information via adversarial discriminative modality distillation. IEEE Trans. Pattern Anal. Mach. Intell. **42**(10), 2581–2593 (2019)
16. Georgescu, M.I., Duţă, G.E., Ionescu, R.T.: Teacher-student training and triplet loss to reduce the effect of drastic face occlusion: Application to emotion recognition, gender identification and age estimation. Mach. Vis. Appl. **33**(1), 12 (2022)
17. Georgescu, M.I., Ionescu, R.T.: Teacher-student training and triplet loss for facial expression recognition under occlusion. In: Proceedings of ICPR. pp. 2288–2295 (2021)
18. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of AISTATS. pp. 249–256 (2010)
19. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Proceedings of NIPS. vol. 27, pp. 2672–2680 (2014)
20. Gu, S., Chen, D., Bao, J., Wen, F., Zhang, B., Chen, D., Yuan, L., Guo, B.: Vector quantized diffusion model for text-to-image synthesis. In: Proceedings of CVPR. pp. 10696–10706 (2022)
21. Hinton, G., Vinyals, O., Dean, J.: Distilling the Knowledge in a Neural Network. In: Proceedings of NIPS Deep Learning and Representation Learning Workshop (2014)
22. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: Proceedings of NeurIPS. vol. 33, pp. 6840–6851 (2020)
23. Jung, B., Johansson, F.D.: Efficient learning of nonlinear prediction models with time-series privileged information. In: Proceedings of NeurIPS. vol. 35, pp. 19048–19060 (2022)
24. Lang, K.: NewsWeeder: Learning to Filter Netnews. In: Proceedings of ICML. pp. 331–339 (1995)

25. Lee, W., Lee, J., Kim, D., Ham, B.: Learning with privileged information for efficient image super-resolution. In: Proceedings of ECCV. pp. 465–482 (2020)
26. Liu, Z., Wei, J., Li, R., Zhou, J.: Learning multi-modal brain tumor segmentation from privileged semi-paired MRI images with curriculum disentanglement learning. Comput. Biol. Med. **159**, 106927 (2023)
27. Lopez-Paz, D., Bottou, L., Schölkopf, B., Vapnik, V.: Unifying distillation and privileged information. In: Proceedings of ICLR (2016)
28. Loshchilov, I., Hutter, F.: Decoupled Weight Decay Regularization. In: Proceedings of ICLR (2019)
29. Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C.: Learning Word Vectors for Sentiment Analysis. In: Proceedings of ACL. pp. 142–150 (2011)
30. Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., Chen, M.: GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models. In: Proceedings of ICML. pp. 16784–16804 (2021)
31. Park, W., Kim, D., Lu, Y., Cho, M.: Relational Knowledge Distillation. In: Proceedings of CVPR. pp. 3962–3971 (2019)
32. Qian, Y., Hu, H., Tan, T.: Data augmentation using generative adversarial networks for robust speech recognition. Speech Commun. **114**, 1–9 (2019)
33. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning Transferable Visual Models From Natural Language Supervision. In: Proceedings of ICML. pp. 8748–8763 (2021)
34. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-Resolution Image Synthesis with Latent Diffusion Models. In: Proceedings of CVPR. pp. 10684–10695 (2022)
35. Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S.K.S., Ayan, B.K., Mahdavi, S.S., Lopes, R.G., et al.: Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. In: Proceedings of NeurIPS. vol. 35, pp. 36479–36494 (2022)
36. Sandfort, V., Yan, K., Pickhardt, P.J., Summers, R.M.: Data augmentation using generative adversarial networks (CycleGAN) to improve generalizability in CT segmentation tasks. Sci. Rep. **9**(1), 16884 (2019)
37. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In: Proceedings of EMC^2 (2019)
38. Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., et al.: LAION-5B: An open large-scale dataset for training next generation image-text models. In: Proceedings of NeurIPS. vol. 35, pp. 25278–25294 (2022)
39. Shivashankar, C., Miller, S.: Semantic Data Augmentation with Generative Models. In: Proceedings of CVPRW. pp. 863–873 (2023)
40. Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using non-equilibrium thermodynamics. In: Proceedings of ICML. pp. 2256–2265 (2015)
41. Song, Y., Ermon, S.: Generative modeling by estimating gradients of the data distribution. In: Proceedings of NeurIPS. vol. 32, pp. 11918–11930 (2019)
42. Vapnik, V., Vashist, A.: A new learning paradigm: Learning using privileged information. Neural Netw. **22**(5–6), 544–557 (2009)
43. Yang, J., Li, B., Yang, F., Zeng, A., Zhang, L., Zhang, R.: Boosting human-object interaction detection with text-to-image diffusion model. arXiv preprint arXiv:2305.12252 (2023)

44. Yim, J., Joo, D., Bae, J., Kim, J.: A Gift from Knowledge Distillation: Fast Optimization, Network Minimization and Transfer Learning. In: Proceedings of CVPR. pp. 7130–7138 (2017)
45. Yimam, S.M., Štajner, S., Riedl, M., Biemann, C.: Multilingual and Cross-Lingual Complex Word Identification. In: Proceedings of RANLP. pp. 813–822 (2017)
46. You, S., Xu, C., Xu, C., Tao, D.: Learning from multiple teacher networks. In: Proceedings of KDD. pp. 1285–1294 (2017)
47. Yu, L., Yazici, V.O., Liu, X., van de Weijer, J., Cheng, Y., Ramisa, A.: Learning Metrics from Teachers: Compact Networks for Image Embedding. In: Proceedings of CVPR. pp. 2907–2916 (2019)
48. Yuan, S., Stenger, B., Kim, T.K.: RGB-based 3D hand pose estimation via privileged learning with depth images. arXiv preprint arXiv:1811.07376 (2018)
49. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond Empirical Risk Minimization. In: Proceedings of ICLR (2018)
50. Zhao, P., Xie, L., Wang, J., Zhang, Y., Tian, Q.: Progressive privileged knowledge distillation for online action detection. Pattern Recogn. **129**, 108741 (2022)

Deep Hardware Modality Fusion for Image Segmentation

Yi Liu[1(✉)], Sudhakar Kumawat[2], and Hajime Nagahara[2]

[1] Graduate School of Information Science and Technology, Osaka University, Osaka, Japan
liuyi@is.ids.osaka-u.ac.jp
[2] Institute for Datability Science, Osaka University, Osaka, Japan
{sudhakar,nagahara}@ids.osaka-u.ac.jp

Abstract. Multimodal image segmentation utilizes a variety of modality images with RGB, infrared, polarization, etc. Unfortunately, the mainstream focus on digital modality fusion leads to the cost of computing abundant information and increased model size. To selectively capture meaningful information and compress model size, we introduce a novel framework for hardware level modality fusion (Our code is available at https://github.com/eki-1996/Deep_Hardware_Modality_Fusion. Our hardware modality fusion module simulates a captured image with fused modality on hardware in two strategies: frame coding and pixel coding. It also enables a single encoder model structure and significantly reduces model size. This modality fused image makes the encoder eliminate redundant information to extract the necessary information for downstream tasks. We evaluated our methods on two multimodal segmentation datasets that contain different modalities and got comparable performance on both datasets compared to baseline methods, which explore different digital modality fusion strategies. Our proposed method outperforms baseline methods on evaluated datasets and tasks with compressed information passed to the segmentation model.

Keywords: Image segmentation · Multimodal image segmentation · Hardware modality fusion

1 Introduction

Semantic segmentation of digital images is classifying regions of pixels that belong to semantically meaningful areas or objects [14]. It is an important preprocessing step for various computer vision applications such as autonomous driving [3], robotics [16], and medical diagnosis [13,25]. In recent years, with the popularity of deep learning and the availability of good computing power, many state-of-the-art segmentation methods have been proposed with promising results. Interestingly, an important feature common among most of these methods is their reliance on the object's shape [1,18,20] for learning segmentation maps, which can be done efficiently using RGB images. However, object shapes

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 439–454, 2025.
https://doi.org/10.1007/978-3-031-78183-4_28

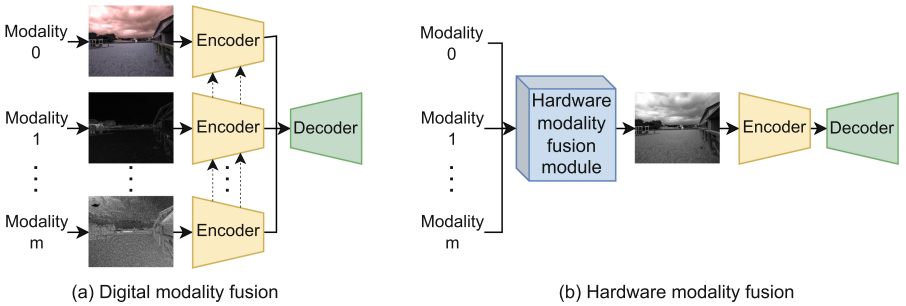


Fig. 1. Digital modality fusion and hardware modality fusion. (a) Digital modality fusion. It extracts features from different encoders and fuses them before the decoder. Normally, model size increases M times compared to a single encoder structure. (b) Hardware modality fusion. It fuses all modalities at hardware level and leads to a smaller model size, which enables the possibility of implementation on limited computational power devices.

can be ambiguous in the real world, and RGB images may not be enough to discriminate between objects with different materials. For example, a real apple and a plastic apple have similar shapes, and a shape-based segmentation model may not be able to discriminate between them. Another example is a ceramic cup and a glass cup.

To tackle this problem in an image segmentation task, various works propose using multi-modal inputs instead of RGB alone. For example, Zhang *et al.* [24] used polarization and RGB images to segment road scene images and showed that polarization can significantly enhance model ability for segmentation. Similarly, Sun *et al.* [23] used RGB and thermal images and proposed RTFNet for the semantic segmentation of urban day and night scenes. Recently, Liang *et al.* [10] used three modalities - RGB, Near Infrared (NIR), and polarization to propose MCubeSNet for segmenting road scene images into real-world material labels such as asphalt, concrete, metal, and water. Fig. 1 (a) provides a general illustration of the above methods. Note that a common characteristic of these methods is that they fuse information from different modalities in the model's feature (digital) space. However, such an approach has various disadvantages. First, all input modalities must be captured, which may lead to high data redundancy and storage costs. Second, each modality requires its encoder, which leads to a high cost of feature extraction in terms of computation and memory.

One way to solve the above issues associated with fusing multi-modal information in digital space is to fuse this information in the camera/hardware space during acquisition. This approach not only helps reduce the redundancy in captured data but also develops lightweight encoders for feature extraction. Many works have studied fusing modalities at the hardware level for specific tasks. For example, Nie *et al.* [17] optimized to design the three-spectrum filter combination for reconstructing hyper-spectrum images. Monno *et al.* [15] proposed an RGB-NIR mosaic filter for embedding four color channels into a single image.

Kurita *et al.* [9] proposed an RGB-polarization mosaic filter to reconstruct the full resolution of RGB-polarized images. Note that most of the above works limit their focus on fusing modalities for improving the quality of the reconstructed multimodal images and do not explore complex downstream computer vision tasks such as image segmentation.

This work introduces a framework for jointly training a hardware modality fusion and image segmentation model. Hardware modality fusion differs from early fusion as it fuses modality on the sensor, while early fusion fuses modality after the sensor. After the training, we can get an optimal filter design and an optimal model for the target segmentation task. The hardware modality fusion module simulates filter design for fusing multiple optical modalities into a single image, including spectrum and polarization. The single image generated by the hardware modality fusion module enables a single encoder network design, significantly reducing model size. Furthermore, the optimized hardware modality fusion module focuses on extracting the necessary information for the following image segmentation model. It makes the computation more efficient. We introduce two different strategies for the hardware modality fusion module to simulate hardware modality fusion as frame coding and pixel coding, respectively. Frame coding has the advantage of full spatial resolution but needs more time to capture enough information for a segmentation task. On the other hand, pixel coding eliminates the time-consuming limitation of frame coding with the cost of lower spatial resolution. We conducted frame coding and pixel coding experiments on two datasets, MCubeS and RGB-Thermal datasets, that target different segmentation tasks with different modalities. We compared the results with baseline methods, MCubeSNet and RTFNet, on these two datasets. The results show that the optimized segmentation model achieves comparable accuracy to the competitive methods, while it has compressed image data as input and a smaller model with lower computation compared to baseline methods.

Briefly, our contributions are:

- Proposing a novel framework for hardware modality fusion and training with subsequent segmentation model in an end-to-end manner.
- Verified that our methods get comparable results on two different multimodal segmentation datasets with their baseline methods.

2 Related Work

RGB Image Segmentation. In the field of computer vision, different from an image classification task, image segmentation is a more complicated task since it does pixel-wise classification for the whole image. Thanks to the introduced end-to-end trainable deep convolutional neural network (CNN) structure [11,22], the performance of deep learning models for semantic segmentation has been significantly improved. Mask R-CNN [6] adds a branch to Faster R-CNN for object mask prediction parallel with bounding box recognition. DeepLab [3] uses atrous convolution and atrous spatial pyramid pooling (ASPP) to enlarge the field of view and extract feature maps at multiple scales. Based on Deeplab,

Deeplab v3 [4] and Deeplab v3+ [5] achieved state-of-the-art performance in PASCAL VOC 2012 and Cityscapes datasets benefit from atrous convolution with multiple atrous rates, augment ASPP with image-level features encoding global context and implement a decoder to refine segmentation results.

Multimodal Image Segmentation. Multimodal image segmentation uses a variety of modalities to get more accurate and robust performance. In the case of RGB, it only captures visible light and drops all other information, each Red, Green, and Blue filter covers different light wavelength ranges in visible light and these covered ranges overlap each other in a small amount. To complement RGB, infrared sensor and polarization sensor can be used; infrared sensor is sensitive to light in the infrared wavelength range 780 nm to 50 μm, and polarization sensor captures light polarization information. Instead of trying to capture different light, the other direction is to narrow the light wavelength range captured by the sensor even though this wavelength range is still inside the RGB wavelength range, called hyperspectral images. Liang et al. [10] introduced a multimodal material segmentation dataset, the MCubeS dataset. Their proposed MCubeSNet uses four encoders for extracting RGB, Polarization, and NIR features and fuses these features after the encoder, as well as a region-guided filter selection layer to select the best combination of these modalities for material segmentation tasks. Sun et al. [23] evaluated their RTFNet method on a RGB-Thermal dataset. RTFNet uses two encoders, one for RGB and one for thermal, fusing extracted features from encoders after each skip connection layer block.

Multimodal Image Acquisition and Fusion. The camera has optical filters to obtain different modalities since an image sensor does not have color or polarization sensitivities. Multi-spectrum images are classically captured by a grayscale CMOS sensor with mechanically rotating filter wheels, liquid crystal tunable filter (LCTF), and acousto-optic tunable filters. The systems synchronize the filter responses and the sensor shutter to capture the multi-spectrum images frame-by-frame. Jeon et al. [8] combine to use an LCTF and polarization filter to get multi-spectrum and polarization modalities at the same time. However, these capturing systems have long scanning times since the number of modalities is the same as the number of images, and one modality is captured by a frame at once.

The current color sensor has filters on the sensor pixels and embeds RGB colors into a single image. Bayer filter array is the most popular color array on a sensor, and it embeds RGB channels of the modality into the single sensor of the image. A polarization sensor was also developed using a polarized filter on the sensor pixels. However, these filter assignments are not optimized for the downstream tasks.

Chakrabarti et al. [2] optimized RGB color filter arrangement instead of Bayer pattern and reconstructed the better-demosaiced color image. Nie et al. [17] proposed optimizing spectrum filters for reconstructing hyper-spectrum reconstruction from three filtered image channels as input. Monno et al. [15] developed a high-quality system design with a different filter array with conventional RGB-

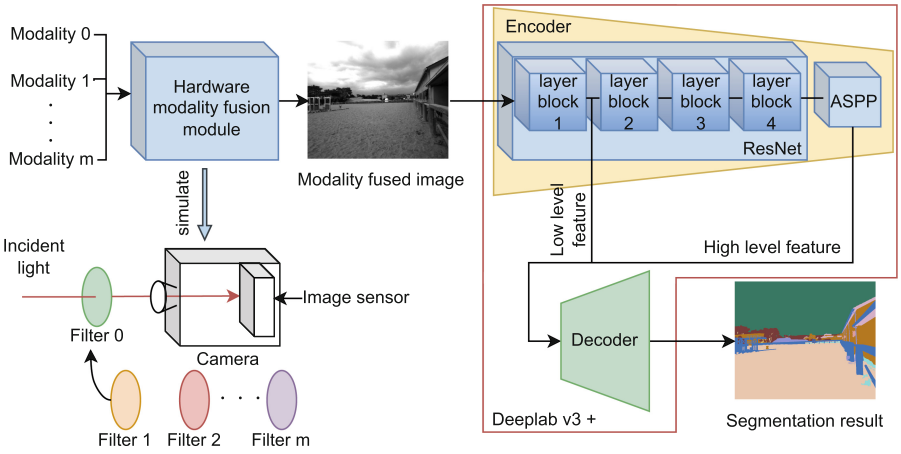


Fig. 2. Structure of proposed framework. It mainly contains a trainable hardware modality fusion module and a segmentation model, which consists of an encoder and a decoder. The encoder takes the modality fused image from the hardware modality fusion module as input to extract essential features. The decoder takes these features to predict segmentation results.

NIR filter array for single-sensor RGB-NIR imaging by considering sensor characteristics and imaging pipeline. Kurita et al. [9] proposed a sparse polarization sensor for capturing high-quality RGB images and polarization simultaneously. However, there is still no research on hardware modality fusion and optimizing the fusion with subsequent deep neural networks for various image segmentation tasks.

3 Method

We propose a novel framework that leverages a hardware modality fusion module for fusing modalities at hardware level and achieves comparable results with baseline methods in multimodal image segmentation. Fig. 2 shows the overall structure. The segmentation model is inspired by Deeplab v3+ [5] and MCubeSNet [10]. Our framework exploits a trainable hardware modality fusion module to integrate modalities at hardware level and then pass the modality fused image to a single branch of a normal segmentation network. The encoder loads an ImageNet [19] pretrained ResNet-101 [7] model, which takes a three-channel input. To fully utilize this pretrained ResNet-101, our hardware modality fusion module fuses modalities into a single three-channel modality fused image and feeds them to the segmentation model.

3.1 Hardware Modality Fusion

The structure of a modality fusion camera is shown in Fig. 3. The camera has a tunable filter, lens, and programmable exposure sensor. The tunable filter is

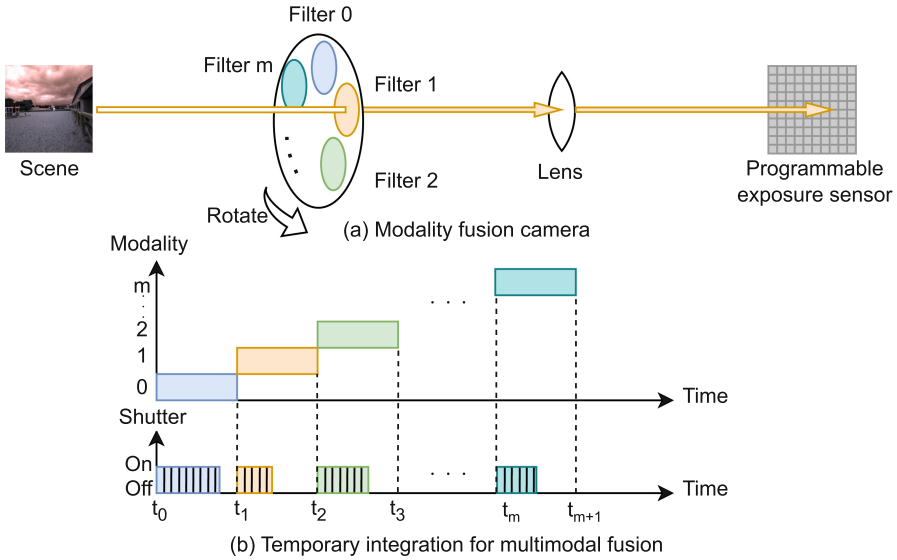


Fig. 3. How to modulate the modalities by modality fusion camera. (a) Structure of modality fusion camera. It consists of a tunable filter (rotating filter wheel in this example), lens, and programmable exposure sensor. (b) The modalities are fused into a frame by synchronizing the filters and shutter. The filter 0 is aligned to the sensor during t_0 to t_1, and the shutter opens and closes within the duration to control the exposure time and weight of the modality.

temporally changing the modality. It can be implemented by a mechanically rotating filter as shown in Fig. 3 (a), or Liquid crystal tunable filter (LCTF), acousto-optic tunable filters, etc. The programmable exposure sensor [12,21] can control that the shutter temporally opens or closes during the single frame acquisition. We temporally synchronize the filter and shutter timings to modulate the multiple modalities into a single image and obtain the modality fused image by the modality fusion camera as shown in Fig. 3 (b). The figure shows the timing charts of the modality selection by the filter and exposure patterns of an arbitrary pixel. The filter passes through the modality m from the time t_m to t_{m+1}. Binary exposure function $\delta(t) \in \{0, 1\}$ can be controlled by the programmable exposure sensor. Hence, a weight k_m for each modality can be expressed by:

$$k_m = \sum_{t_m}^{t_{m+1}} \delta(t), \tag{1}$$

Although the shutter patterns are binary, we can get the continuous weight for the exposure duration control. We also introduce two modulation schemes, frame coding and pixel coding, for modulating the exposure timing in frame-wise and pixel-wise manners. We explain the details below and illustrate in Fig. 4 (a) and (b), respectively.

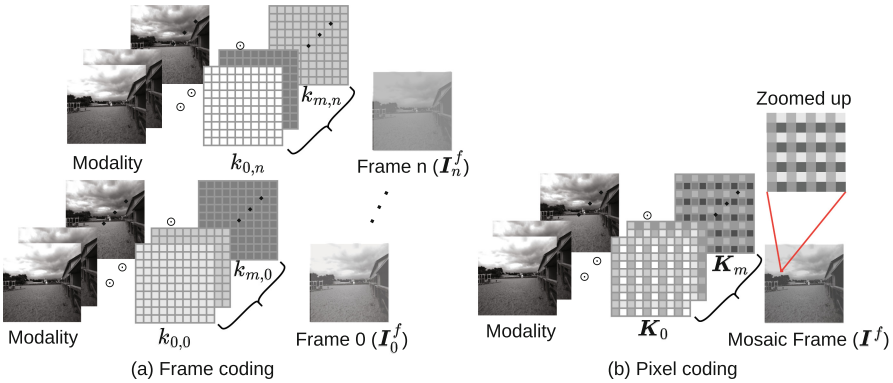


Fig. 4. (a) Frame coding. All pixels in a frame have the same weights of modalities. The different frames n have different weights. (b) Pixel coding. Each pixel has different weights for different modalities.

Frame Coding. Figure 4 (a) shows the frame coding, which modulates the modality fusion in a frame-by-frame manner. The modality weights are the same for the whole of the pixels in the image by broadcasting $k_{m,n}$ to all pixels for frame n, and the modality weights are different for the consecutive frame $n+1$. Hence, the modality fusion by frame coding is expressed as:

$$\boldsymbol{I}_n^f = \frac{\sum_{m=0}^{m} k_{m,n} \cdot \boldsymbol{I}_m}{m+1}, \tag{2}$$

where $\boldsymbol{I}_m$ is the m-th modality of multimodal input, $k_{m,n}$ is the weight of m-th modality in n-th frame for all pixels and $\boldsymbol{I}_n^f$ is the fused n-th frame.

Frame coding has the advantage of full spatial resolution but needs more time to capture information on different weighted modalities.

Pixel Coding. Figure 4 (b) shows pixel coding scheme that embeds different modality fusion weights in a pixel-by-pixel manner similar to the Bayer color filter. The pixel coding can embed the different fusion modalities into a single image while the spatial resolution would be sacrificed. We assume that the coding block is a $p \times p$ pixels block, and the m-th modality fusion weight is described as $\boldsymbol{k}_{m,p,p}$. The modality weights differ with neighboring pixels according to the pixel position (p, p). Hence, the pixel block coding pattern for m-th modality is expressed as:

$$\boldsymbol{B}_m = \begin{bmatrix} \boldsymbol{k}_{m,1,1} & \boldsymbol{k}_{m,1,2} & \cdots & \boldsymbol{k}_{m,1,p} \\ \boldsymbol{k}_{m,2,1} & \boldsymbol{k}_{m,2,2} & \cdots & \boldsymbol{k}_{m,2,p} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{k}_{m,p,1} & \boldsymbol{k}_{m,p,2} & \cdots & \boldsymbol{k}_{m,p,p} \end{bmatrix}. \tag{3}$$

Like the regular Bayer-like filter, the block pattern recursively repeats to the whole image with $w \times h$ pixels. The mosaic-like coding pattern for the whole of

the image is expressed as:

$$\boldsymbol{K}_m = \underbrace{\left.\begin{bmatrix} \boldsymbol{B}_m & \boldsymbol{B}_m & \cdots & \boldsymbol{B}_m \\ \boldsymbol{B}_m & \boldsymbol{B}_m & \cdots & \boldsymbol{B}_m \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{B}_m & \boldsymbol{B}_m & \cdots & \boldsymbol{B}_m \end{bmatrix}\right\} h/p \text{ rows.}}_{w/p \text{ columns}} \tag{4}$$

The modality fused image $\boldsymbol{I}^f$ by pixel coding is expressed by:

$$\boldsymbol{I}^f = \frac{\sum_{m=0}^{m} \boldsymbol{K}_m \odot \boldsymbol{I}_m}{m+1}. \tag{5}$$

Pixel coding eliminates the time-consuming limitation of frame coding with the cost of lower spatial resolution, as different pixels contain different contributions for each modality.

3.2 Segmentation Model

Our segmentation model takes the same structure with DeepLab v3+ [5]. The segmentation model contains an encoder and a decoder. The encoder consists of an ImageNet pretrained ResNet-101 and an ASPP module. The encoder extracts low-level features from the first layer block of ResNet-101 and high-level features from ResNet-101 and ASPP module. The ASPP module captures contextual information with different scales via different atrous convolution rates. Since the input image for the encoder will be in the value range of [0, 1], we need to constrain the weight of the hardware modality fusion module in [0, 1] to guarantee that the pixel value of the modality fused image falls in the same range. Because of this reason, the sigmoid function is applied to the weight of the hardware modality fusion module. The decoder leverages these low-level features and high-level features to predict segmentation results.

3.3 Training and Inference

The hardware modality fusion module and the segmentation model are trained in an end-to-end manner. During training, frame coding $k_{m,n}$ or pixel coding $\boldsymbol{K}_m$ and the following segmentation model are optimized through the backpropagation algorithm. The frame coding $k_{m,n}$ or pixel coding $\boldsymbol{K}_m$ will be optimized to generate a modality fused image by taking the optimal weight of different modalities. The encoder and decoder will be guided to estimate corresponding segmentation results by leveraging the modality fused image. After training, the optimized hardware modality fusion module will be implemented into the modality fusion camera. For inference, we can directly capture the modality fused images via this camera and pass these modality fused images to the optimized segmentation model to make segmentation predictions. This means only the segmentation model will be used during inference.

4 Experiments

We carried out the experiment results by using MCubeS dataset and RGB-Thermal dataset. We compared our proposed framework with MCubeSNet and RTFNet to show the advantage of realizing similar segmentation accuracy with a modality fused image as the input. We compared some cases: frame coding with single and three frames and pixel coding with a block size of 2×2, 4×4, and 8×8 pixels.

4.1 Datasets

MCubeS dataset [10] consists of RGB, polarization, and NIR images of road scenes using a pair of RGB polarization cameras and an NIR camera with a LiDAR to assist annotation. They convert RGB polarization images to RGB images, Degree of Linear Polarization (Dolp), and Angle of Linear Polarization (Aolp) numpy files with a resolution of 1224x1024 pixels. MCubeS dataset is annotated pixel-wise in 20 material classes, such as asphalt, concrete, and metal, and contains 500 annotated images, which are divided into 302, 96, and 102 images for training, validation, and testing, respectively.

Our modality fusion camera assumes switching the optical filters to obtain multimodal fused image intensity. The dataset [10] captures polarization images by linear polarization filters and then calculates Dolp and Aolp images. The calculated Dolp and Aolp are not image intensity, but the polarization images are image intensity. The dataset does not release the captured polarization images to the public. Hence, we reconstructed the four captured polarization images: $\boldsymbol{I}_0$, $\boldsymbol{I}_{\pi/4}$, $\boldsymbol{I}_{\pi/2}$ and $\boldsymbol{I}_{3\pi/4}$ from the RGB, Dolp and Aolp images.

The conversion from RGB, Dolp, and Aolp to the polarization images $\boldsymbol{I}_\phi$ is formulated by:

$$\boldsymbol{I}_\phi = \boldsymbol{I}_{gray}(1 + Dolp \times \cos 2(\phi - Aolp)), \tag{6}$$

where $\boldsymbol{I}_{gray}$ is the grayscale image calculated by RGB image $\boldsymbol{I}_R$, $\boldsymbol{I}_G$ and $\boldsymbol{I}_B$.

We used eight modalities, RGB images $\boldsymbol{I}_R$, $\boldsymbol{I}_G$, $\boldsymbol{I}_B$, NIR image $\boldsymbol{I}_{NIR}$, and four linear polarization images $\boldsymbol{I}_0$, $\boldsymbol{I}_{\pi/4}$, $\boldsymbol{I}_{\pi/2}$ and $\boldsymbol{I}_{3\pi/4}$ instead of Aolp and Dolp images for generating fusion images in the experiment on MCubeS dataset.

RGB-Thermal dataset recorded RGB and thermal images of urban scenes using an InfRec R500 RGB IR camera. RGB-Thermal dataset combines thermal images with RGB images along channel dimensions to form a four-channel image dataset. RGB-Thermal dataset is annotated pixel-wise in 8 obstacle classes common in driving, such as car, person, and color cone. RGB-Thermal dataset contains 820 daytime images and 749 nighttime images, totaling 1569 images with a resolution of 480x640. RGB-Thermal dataset is separated into three parts: 50% of daytime images and nighttime images for the training set, 25% of daytime images and nighttime images for the validation set, and 25% of daytime images and nighttime images for the test set. We used four modalities, RGB images $\boldsymbol{I}_R$, $\boldsymbol{I}_G$, $\boldsymbol{I}_B$, NIR image $\boldsymbol{I}_{NIR}$ for generating fusion images in the experiment on RGB-Thermal dataset.

4.2 Training Details

Our experiments are conducted on a single NVIDIA RTX 6000 Ada Generation GPU. We keep almost the same training setting with the baseline methods.

For MCubeS dataset, we flip training images and labels horizontally with a 50% chance. We randomly crop training sets. Furthermore, Gaussian blur is added to training sets with 50% chance, and normalization by mean=(0.485, 0.456, 0.406) and standard deviation=(0.229, 0.224, 0.225) calculated in the ImageNet dataset is applied. As our hardware modality fusion module fuses all modalities to generate a modality fused image, we place data normalization just after the hardware modality fusion module. For RGB-Thermal dataset, the training dataset gets flipped horizontally with a 50% chance and is cropped with a 0.1 crop ratio. Cross entropy loss and stochastic gradient descent optimizer are used to guide training. Referring to Deeplab v3+ [5], we set the initial learning rate as 0.007 with polynomial decay from 0.05 with 0.9 power during 500 epochs. We set the initial learning rate as the learning rate of pretrained ResNet-101. The learning rate of the other layers, such as the hardware modality fusion module, ASPP module, and decoder, is set to 10 times the initial learning rate as these layers are trained from scratch. Class mean Intersection over Union (mIoU) is used as the evaluation matrix for model performance.

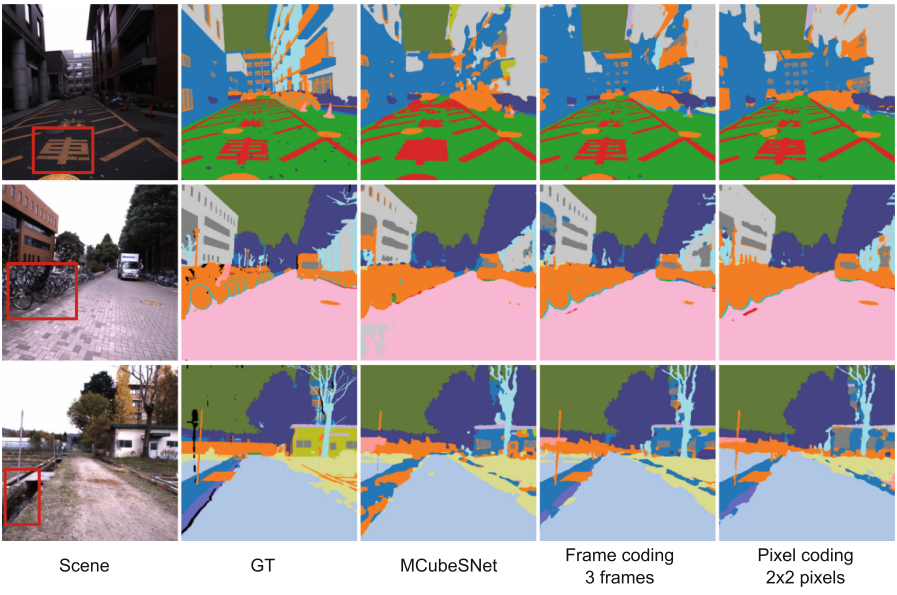


Fig. 5. Results on MCubeS dataset. Each row shows a different sample from MCubeS dataset. Here, we visualize the results of our frame coding with 3 frames and pixel coding with a 2×2 pixel block along with the baseline method, MCubeSNet.

4.3 Comparison with Multimodal Methods in Digital Domain

We implement the proposed methods to two datasets and compare our methods with their baseline methods. The results show our methods achieve comparable results with baseline methods. We take the mean and standard deviation of our methods for three independent runs.

Comparison with MCubeSNet on MCubeS Dataset. Table 1 shows a detailed quantitative comparison of model size and cost. Since the introduced strategies, frame coding and pixel coding, have similar properties on these compared aspects, we combine them as "Ours". Our methods get a better score on all measured matrices according to the quantitative results. Regarding model size, our methods have 59.3 million parameters and occupy 226.8MB of memory space, which is less than a quarter of MCubeSNet. Our methods take 88.9G flops, while MCubeSNet needs 401.0G flops, which is over 4.5 times of ours in terms of computational cost. Furthermore, our methods have a 2.1-second faster inference time than MCubeSNet concerning the whole test set. We believe this inference time difference will become obvious when executed on computational power-limited devices, such as autonomous vehicles and IoT devices.

Table 2 shows the results of our methods with MCubeSNet. Our methods get comparable performance with the baseline method. Frame coding with three frames achieves 43.8 mIoU, surpassing the baseline method with a margin of 0.9 mIoU. Overall, pixel coding performs poorly compared to the baseline method, but it has acceptable performance when the pixel block is 2×2 pixels. Our methods improve performance in some classes but decrease performance in other classes compared to the baseline method. For example, our methods significantly increase the performance of road marking, plastic, rubber, and water classes while decreasing the performance of plaster, sand, gravel, and cobbles.

Figure 5 shows segmentation results of our methods and baseline method, MCubeSNet. In the first row, our methods perform better in road marking than MCubeSNet. In the second row, our methods make a better prediction about bricks and perform better on rubber but make some wrong predictions about the left center glass. In the third row, our methods' and MCubeSNet's performance are similar, but our methods do better in the left water region.

Table 1. Model quantitative comparison. Inference time is measured on the whole test set, a total of 102 images.

Model	Parameters (M)	Model size (MB)	flops (G)	Inference time (whole test set) (s)
MCubeSNet	240.0	917.9	401.0	10.7
Ours	59.3	226.8	88.9	8.6

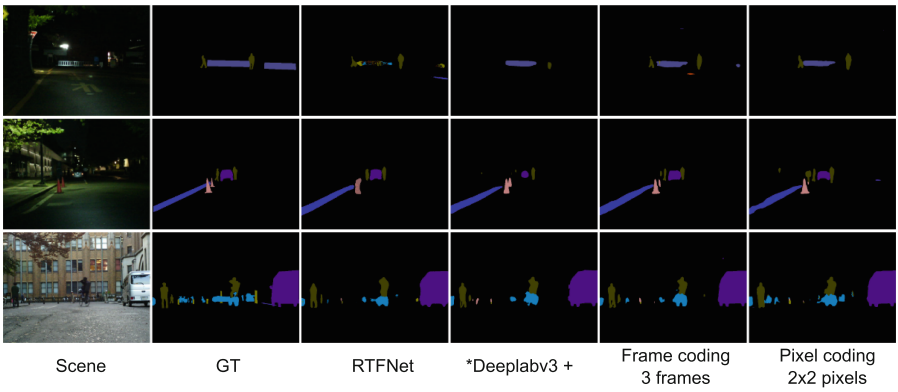


Fig. 6. Results on RGB-Thermal dataset. Each row shows a different sample from RGB-Thermal dataset. Here, we visualize the results of our frame coding with 3 frames and pixel coding with a 2×2 pixel block along with baseline methods, RTFNet, and *Deeplab v3+ that has two encoders for RGB and thermal data.

Comparison with RTFNet on RGB-Thermal Dataset. We also conduct experiments on another RGB-Thermal semantic segmentation dataset. Table 3 shows the results of our methods with two baseline methods, RTFNet, and another method as *Deeplab v3+ that is similar to Deeplab v3+ while has two encoders, one for RGB and one for thermal.

The results show our methods can still get comparable results with baseline methods on RGB-Thermal dataset. Similar to MCubeS dataset, frame coding with three frames achieves 53.8 mIoU, surpassing the baseline methods with a margin of 0.6 mIoU and 1.5 mIoU, respectively. The pixel coding with 2×2 pixels also has acceptable performance, but other pixel block sizes perform relatively poorly. Our methods get a similar class-wise performance on RGB-Thermal dataset with MCubeS dataset. Our methods significantly increase the performance of guardrail and color cone with the cost of decreasing bump classes' performance compared to RTFNet. Our methods are able to increase the performance of curve compared to *Deeplab v3+.

Figure 6 shows segmentation results of our methods and baseline method, RTFNet, and *Deeplav v3+. In the first row, our methods perform better in guardrail and person than RTFNet and Deeplab v3+. In the second row, our methods perform better for color cone than RTFNet and Deeplab v3+. In the third row, the performance between our methods and baseline methods is similar, but our methods do a better job on bike.

4.4 Discussion on Frame Coding and Pixel Coding

Regarding overall performance, mIoU, in the case of frame coding, increasing frame number increases overall performance, while increasing pixel blocking size decreases overall performance in the case of pixel coding, according to the results

Table 2. Performance comparison with our methods and MCubeSNet on MCubeS dataset.

Class	Method					
	Frame coding		Pixel coding (pixels)			Baseline
	1 frame	3 frames	2 × 2	4 × 4	8 × 8	MCubeSNet [10]
asphalt	81.5±2.0	84.3±1.9	85.4±0.9	82.7±1.1	84.0±1.1	85.7
concrete	41.4±1.4	44.8±1.3	44.3±2.0	43.7±0.7	41.8±1.1	42.6
metal	43.9±2.0	47.8±0.3	45.3±1.7	45.0±0.4	44.2±0.9	47.0
road marking	64.8±0.6	67.5±1.0	66.5±0.9	64.9±0.4	65.7±1.3	59.2
fabric	11.4±0.4	16.3±1.4	11.8±1.3	15.9±3.2	12.3±3.0	12.5
glass	42.1±1.8	44.9±0.3	42.9±0.8	44.3±1.5	42.3±2.3	44.3
plaster	2.3±2.0	0.8±0.3	1.1±0.5	1.0±0.9	0.7±0.4	3.0
plastic	18.5±3.5	23.6±2.2	22.1±2.2	19.2±0.9	17.6±4.3	10.6
rubber	17.6±1.3	17.9±1.0	17.9±1.0	18.4±0.5	17.4±1.8	12.7
sand	60.5±2.9	58.0±0.8	59.7±0.6	59.4±2.1	58.0±2.0	66.8
gravel	29.0±13.3	36.6±11.5	46.3±10.1	25.0±11.9	39.5±4.9	67.1
ceramic	22.6±0.4	23.8±0.4	21.6±0.2	22.4±0.2	21.9±2.8	27.8
cobbles	61.2±2.5	58.3±2.5	58.9±6.2	55.8±1.0	60.1±2.5	65.8
brick	30.7±1.6	37.0±0.2	34.4±2.4	35.3±0.9	30.8±2.4	36.8
grass	46.5±2.3	59.7±0.1	56.5±0.5	54.7±0.8	49.3±0.6	54.8
wood	36.7±0.7	40.5±0.8	39.7±0.7	38.2±0.9	36.0±1.8	39.4
leaf	69.7±0.9	74.2±0.5	73.1±0.6	72.0±0.1	70.7±0.1	73.0
water	27.5±6.4	44.5±2.0	34.2±1.2	37.4±1.2	38.8±1.5	13.3
sky	95.3±0.1	95.5±0.0	95.4±0.0	95.4±0.1	95.4±0.0	94.8
mIoU	40.2±0.7	**43.8**±0.8	42.9±0.5	41.5±0.6	41.3±0.7	42.9

of Tables. 2 and 3. Respecting class-wise performance, in the case of frame coding, increasing frame number increases most classes' performance except plaster, sand, cobbles, and grass. However, the impact of increasing pixel coding block size varies for different classes.

Although the 3-frame case of frame coding always performs better than pixel coding, it needs more time to capture 3 frames. Conversely, pixel coding performs worse than frame coding with 3 frames but still gets comparable results on specific block sizes with the advantage of shorter capturing time. The 2 × 2 pixel coding performs better than single-channel frame coding in both datasets, verifying that the 2 × 2 pixel coding indeed contains more information than single-channel frame coding. However, the results that 2×2 pixel coding does not achieve the performance of frame coding with three frames mean the lower spatial resolution and mosaicked input image harm the performance of the segmentation model since pixel coding sacrifices spatial resolution and generates a mosaicked image.

Table 3. Performance comparison with RTFNet on RGB-Thermal dataset.

Class	Method						
	Frame coding		Pixel coding (pixels)			Baseline	
	1 frame	3 frames	2 × 2	4 × 4	8 × 8	*Deeplab v3+	RTFNet [23]
car	84.5±0.1	86.3±0.5	85.0±0.4	84.7±0.2	84.0±0.4	85.3±0.5	87.4
person	63.7±0.5	70.0±1.3	68.7±0.6	66.6±1.2	66.0±0.4	65.2±0.6	70.3
bike	60.3±0.8	60.7±1.1	61.0±0.1	60.7±0.3	59.5±0.6	61.3±0.5	62.7
curve	40.4±0.6	44.6±0.8	42.2±0.2	39.5±2.3	39.1±0.8	38.7±1.3	45.3
car stop	25.1±2.5	27.7±1.1	26.3±1.8	29.3±0.3	21.8±3.3	24.8±2.3	29.8
guardrail	5.2±1.3	7.9±0.5	6.3±0.5	4.7±1.4	6.9±1.3	6.0±0.8	0.0
color cone	43.0±1.0	44.1±2.2	44.3±0.6	41.3±1.7	41.6±1.2	45.3±2.5	29.1
bump	44.3±1.1	45.3±1.2	42.3±3.3	41.2±2.6	42.2±1.2	46.4±2.0	55.7
mIoU	51.6±0.4	**53.8**±0.2	52.7±0.7	51.8±0.3	51.0±0.7	52.3±0.9	53.2

Table 4. Ablation study on active function of frame coding with three frames hardware modality fusion module on MCubeS dataset.

	None	Sigmoid	Softmax	Relu
mIoU	41.6	**44.4**	43.2	42.9

4.5 Ablation Study

We also investigate the influence of different active functions of the hardware modality fusion module on MCubeS dataset. We conducted experiments on four conditions for frame coding with three frames: None active function, Sigmoid function, Softmax function, and Relu function. Table 4 shows the results. Results show that the sigmoid function achieves the best performance, and the performance of none active function is worse than cases with an active function. However, with active function, different active functions have different effects on performance with small mIoU differences.

5 Conclusion

In this work, we target hardware modality fusion for multimodal segmentation tasks. We propose a novel framework for hardware modality fusion with the advantage of reduced model size and maintained performance. We evaluate our methods on two different segmentation datasets, MCubeS dataset and RGB-Thermal dataset, and get comparable results with baseline methods. We believe our work could provide some new hints on task-oriented hardware design and deep neural network model design.

As our methods fuse modalities at the hardware level, which means accumulating pixel intensity of different modalities, it might hinder our methods from deploying in dynamic scenes. Our future work is to alleviate this problem and implement our methods into a prototype camera for real-world experiments.

Acknowledgements. This paper is partially supported by JSPS KAKENHI 23H05490 and JST CREST JPMJCR22C1.

References

1. Badrinarayanan, V., Kendall, A., Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **39**(12), 2481–2495 (2017)
2. Chakrabarti, A.: Learning sensor multiplexing design through back-propagation. Advances in Neural Information Processing Systems **29** (2016)
3. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE Trans. Pattern Anal. Mach. Intell. **40**(4), 834–848 (2017)
4. Chen, L.C., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587 (2017)
5. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Proceedings of the European conference on computer vision (ECCV). pp. 801–818 (2018)
6. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 2961–2969 (2017)
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
8. Jeon, Y., Choi, E., Kim, Y., Moon, Y., Omer, K., Heide, F., Baek, S.H.: Spectral and polarization vision: Spectro-polarimetric real-world dataset. arXive (2311.1796) (2023)
9. Kurita, T., Kondo, Y., Sun, L., Moriuchi, Y.: Simultaneous acquisition of high quality rgb image and polarization information using a sparse polarization sensor. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 178–188 (2023)
10. Liang, Y., Wakaki, R., Nobuhara, S., Nishino, K.: Multimodal material segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 19800–19808 (2022)
11. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3431–3440 (2015)
12. Luo, Y., Ho, D., Mirabbasi, S.: Exposure-programmable cmos pixel with selective charge storage and code memory for computational imaging. IEEE Trans. Circuits Syst. I Regul. Pap. **65**(5), 1555–1566 (2017)
13. Milletari, F., Navab, N., Ahmadi, S.A.: V-net: Fully convolutional neural networks for volumetric medical image segmentation. In: 2016 fourth international conference on 3D vision (3DV). pp. 565–571. Ieee (2016)
14. Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., Terzopoulos, D.: Image segmentation using deep learning: A survey. IEEE Trans. Pattern Anal. Mach. Intell. **44**(7), 3523–3542 (2021)
15. Monno, Y., Teranaka, H., Yoshizaki, K., Tanaka, M., Okutomi, M.: Single-sensor rgb-nir imaging: High-quality system design and prototype implementation. IEEE Sens. J. **19**(2), 497–507 (2018)

16. Mousavian, A., Toshev, A., Fišer, M., Košecká, J., Wahid, A., Davidson, J.: Visual representations for semantic target driven navigation. In: 2019 International Conference on Robotics and Automation (ICRA). pp. 8846–8852. IEEE (2019)
17. Nie, S., Gu, L., Zheng, Y., Lam, A., Ono, N., Sato, I.: Deeply learned filter response functions for hyperspectral reconstruction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4767–4776 (2018)
18. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18. pp. 234–241. Springer (2015)
19. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. Int. J. Comput. Vision **115**, 211–252 (2015)
20. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4510–4520 (2018)
21. Sarhangnejad, N., Katic, N., Xia, Z., Wei, M., Gusev, N., Dutta, G., Gulve, R., Zhi, P., Li, X., Feng Ke, H., Haim, H., Moreno-García, M., Stoppa, D., Kutulakos, K.N., Genov, R.: Dual-tap computational photography image sensor with per-pixel pipelined digital memory for intra-frame coded multi-exposure. IEEE Journal of Solid-state Circuit **54**(11), 3191–3202 (2019)
22. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
23. Sun, Y., Zuo, W., Liu, M.: Rtfnet: Rgb-thermal fusion network for semantic segmentation of urban scenes. IEEE Robotics and Automation Letters **4**(3), 2576–2583 (2019)
24. Zhang, Y., Morel, O., Blanchon, M., Seulin, R., Rastgoo, M., Sidibé, D.: Exploration of deep learning-based multimodal fusion for semantic road scene segmentation. In: VISAPP 2019 14Th international conference on computer vision theory and applications (2019)
25. Zhou, Z., Rahman Siddiquee, M.M., Tajbakhsh, N., Liang, J.: Unet++: A nested u-net architecture for medical image segmentation. In: Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings 4. pp. 3–11. Springer (2018)

Temporal Insight Enhancement: Mitigating Temporal Hallucination in Video Understanding by Multimodal Large Language Models

Li Sun[1(✉)], Liuan Wang[1], Jun Sun[1], and Takayuki Okatani[2]

[1] Fujitsu Research and Development Center, Beijing, China
sun.li@fujitsu.com
[2] Graduate School of Information Sciences, Tohoku University, Sendai, Japan
okatani@tohoku.ac.jp

Abstract. Recent advancements in Multimodal Large Language Models (MLLMs) have enabled to process diverse input modalities, leading to significantly better understanding of multimedia contents. However, understanding videos is still difficult and even the latest models often create hallucinations. This study introduces a novel method to address event-level hallucinations in MLLMs with a special focus on inferring temporal information of events occurring in an input video. It targets event-related information from both the text query and the video content to enhance MLLMs' response. Specifically, our method first decomposes these event queries into iconic actions, and then identifies the timestamps of these actions by utilizing external multi-modal models such as CLIP and BLIP2. Experiments using the Charades-STA dataset show that the method decreases the number of hallucinations and improves the MLLM's responses. We also introduce a quantifiable approach to access these models' performance in understanding and responding to time-related queries. We designed two question-and-answer tasks to measure response hallucinations in terms of detailed timestamps and the order of time events, respectively. After using our method, the error rates of MLLM's responses in these two tasks decreased by 39.7% and 36.1%, respectively.

Keywords: Multimodal language model · Temporal hallucination · Correction · Iconic Action

1 Introduction

The advent of large-scale foundational models [4,13,14,18,19] is enabling machines to understand multimodal data at an unprecedented level. The emer-

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78183-4_29.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 455–473, 2025.
https://doi.org/10.1007/978-3-031-78183-4_29

gence of open-source LLMs [7,23,24] have facilitated the development of Multimodal Large Language Models (MLLMs), which can take images in addition to text as inputs. MLLMs [5,16,17,28,32] enable interactions with users in natural language about the input images etc. in the format of Q&A. Subsequently, MLLMs capable of handling not only still images but also videos have emerged.

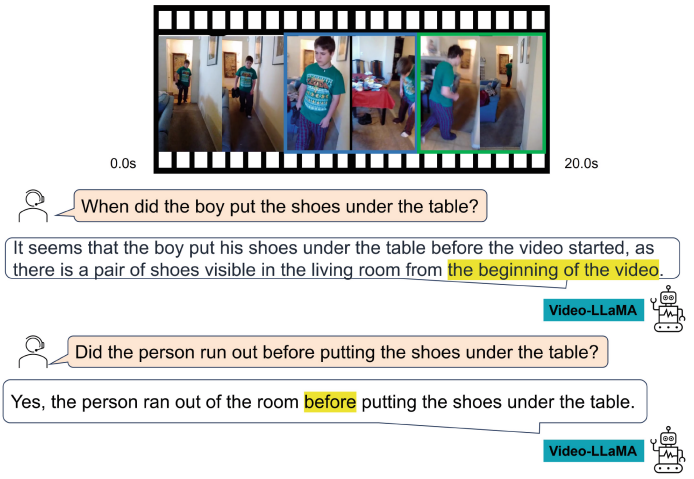


Fig. 1. Examples illustrating hallucination generated by MLLMs in predicting event occurrence timestamps and sequencing.

These MLLMs are adept at crafting engaging descriptions in their text outputs, yet they often struggle with visual inference, resulting in the creation of hallucinations — incorrect perceptions or interpretations of visual contents, as noted in recent surveys and studies [21,27,29]. When dealing only with still images, the primary type of hallucinations is object-level, i.e., incorrect recognition of objects [15,27]. However, when dealing with videos, there is an additional type of hallucinations, i.e., event-level hallucinations, which refers to the incorrect recognition of events occurring within the video. The precise understanding of various video events is the cornerstone for MLLMs to become real multimodal world models. Currently, MLLMs still exhibit many hallucinations at the event level, which severely hinders their applications. On the other hand, obtaining high-quality video annotations is difficult, and improving the performance of MLLMs at the video event level through iterative training is costly and progresses slowly. Therefore, this paper proposes a novel and efficient method using external tools and low-cost language models to correct event hallucinations in video MLLMs.

The most common queries from users about videos involve seeking temporal information on specific events within the video, such as when a particular event happens or the order in which multiple events occur. Users specify an event or multiple events using natural language text. For MLLMs to correctly answer

this type of queries with a diverse set of events, they must perform thorough analysis and understanding of the input video contents to ensure not missing the occurrence/ending of a specified event. However, the inherent constraint of context size in the original LLMs poses a barrier. Just like text and images, video inputs also need to be converted into a sequence of tokens. If each frame constituting a video is tokenized in the same manner as still images, the number of tokens would exponentially exceed that of still images.

Figure 1 demonstrates how Video-LLaMA [28] struggles with accurately predicting the timing of events and their sequence in videos, a problem highlighted by the presence of temporal hallucinations when using raw video inputs. Due to the limitation of the context window size and the need to manage training costs, Video-LLaMA adopts a strategy of uniformly sampling videos at a fixed frequency. This strategy inevitably results in loss of information.

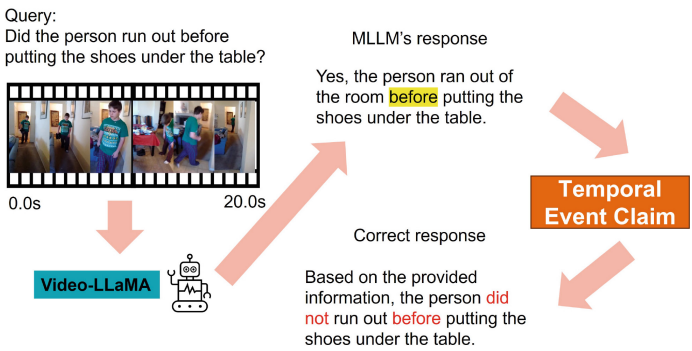


Fig. 2. Framework overview of our temporal hallucination mitigating method.

To tackle the challenge of information loss that the user desires to query in videos causing event-level hallucinations, we pose the following questions related to MLLM's event-level hallucinations: What types of hallucinations do MLLLMs output? How can we correct them?

Regarding the first question, we focus on MLLMs' limitations in predicting the temporal location of events, designing two tasks for quantitative evaluation. One involves predicting the precise timestamp of an event occurring in an input video, and the other task involves predicting the order of multiple events. Then, our empirical investigation highlights that MLLMs are prone to temporal hallucinations. Specifically, MLLMs exhibit subpar performance in accurately predicting the temporal location of queried events, consequently limiting their ability to predict the temporal order of multiple queried events.

To mitigate temporal hallucinations, we propose a novel method to produce accurate claims about the timing of events; see Fig. 2. The claim, which our method produces, is a structured assertion or conclusion that incorporates or is based on the temporal and event-related information extracted from video

content. We employ external tools, specifically CLIP [19] and BLIP2 [13], to extract precise temporal information from video frames corresponding to the events of interest. By aligning the temporal information from the frames accurately matched with the event queries, our method reduces the creation of hallucinations, particularly in responses related to the event timing and order predictions.

The study makes following contributions:

- We introduce a novel framework to alleviate temporal hallucinations in MLLMs when they process queries related to the timing of events. This framework leverages event-specific information to enhance the accuracy of MLLMs' responses to queries regarding temporal information about events. It does not need additional training, providing an accessible and straightforward solution for improving MLLMs' handling of temporal information, without the need for extensive resources or complex modifications.
- We develop a quantitative evaluation method that allows for a more precise assessment of MLLMs' performance in addressing questions related to the timing and order of events.

2 Research Background

2.1 MLLMs

MLLMs are advanced AI models that integrate and process information from multiple modalities, such as text, images, and videos, to perform a variety of tasks. Unlike language models which primarily focus on text-based inputs and outputs, MLLMs are designed to understand and generate content that involves both language and other forms of data, such as visual or auditory inputs [26]. Currently, most MLLMs that support visual modality only accept image inputs [5,16,17,32]. We choose Video-LLaMA [28] for our baseline model as it can supports images, videos and audios as its inputs.

2.2 Hallucination in MLLMs

With the increasing uses of LLMs/MLLMs, the issues with hallucinations has gradually gained attention. For MLLMs and LLMs, hallucination refers to the model erroneously perceiving its output as correct [21,27,29]. In the context of MLLMs, hallucination can be categorized, based on its content, into object-level and event-level, and, based on the reasons for hallucination, into knowledge-deficiency and inductive-bias types.

Object-level hallucination. Object-level hallucination refers to a phenomenon in which machine learning models, particularly MLLMs and LLMs, generate incorrect or distorted outputs related to object recognition. In this context, hallucination occurs when the model mistakenly perceives or includes objects in its generated outputs that do not exist in the input data or misinterprets their characteristics.

Here are some common object-level hallucinations:

Attribute hallucinations. The model thinks an object is pink when it is actually red in the input image or video.

Number hallucinations: The model reports four cars when there are only three in the input image or video.

Relative position hallucinations: The model thinks a dog is on the right side of a person when it is actually on the left side in the input image or video. Liu et al. introduced an evaluation and correction method to address object-level hallucination [15]. Additionally, Yin et al. proposed a train-free method specifically designed for object-level hallucination, as detailed in their work [27].

Event-level hallucination. Currently, there is a gap in research regarding event-level hallucination. To the best of our knowledge, this paper is the first to specifically address event-level hallucination in MLLMs. Fig. 1 illustrates typical examples of the event-level hallucination. There are two events mentioned in this figure. One is *the boy put the shoes under the table* and the other one is *the boy ran out of the room*. The MLLM believes that the event *the boy put the shoes under the table* occurred at the beginning of the video. This is not true; it is a hallucination. In fact, as shown by the blue ground-truth bounding box in the image, this event happens in the middle of the video. On the other hand, the MLLM thinks the event *the boy ran out of the room* occurred before the event *the boy put the shoes under the table*. This is also a hallucination. The green GT-bbox indicates when the event *the boy ran out of the room* occurs, and it is clear that it happens after the event represented by the blue bounding box.

Our work concentrates on events within videos, specifically examining hallucinations that arise when posing temporal-related queries. We have defined two tasks–event occurrence time and the order of occurrences for multiple events–to evaluate event-level temporal hallucination.

2.3 Hallucination Correction

Some studies indicate that erroneous outputs and misinformation from LLMs negatively impact the development of artificial intelligence, particularly the user experience of AI products [1,2]. Therefore, hallucination correction is crucial for LLMs/MLLMs. Dhuliawala et al. [8] divides hallucination correction methods into three categories: training-time correction, generation-time correction, and correction based on external tools.

Typical methods for addressing hallucinations with external tools include retrieval augmented generation (RAG) [22], fact tool (FacTool) [6], chain-of-thought verification [30] and correction during decoding procedure [11,12].

RAG is an advanced technique in the field of NLP that combines information retrieval with text generation [20]. The development of LLMs has made RAG a crucial technique for addressing knowledge deficiencies of LLMs. Beyond LLMs, there is also research on vision-language models with RAG. Yang et al. [25] introduces Re-ViLM, a Retrieval-augmented Visual Language Model that extends Flamingo. It incorporates an external database for knowledge retrieval, result-

ing in enhanced image-to-text generation performance, particularly in zero-shot and few-shot scenarios. MLLMs are also well-suited for RAG scenarios.

Methods that use external tools [6] assume the availability of tools that are expected to be consistently correct in all possible cases. While it is challenging to find a perfect tool, we can rely on the results from a tool whose accuracy is higher than that of the model to be corrected, i.e. MLLMs in this study.

Recently, some methods try to correct the hallucination during decoding procedure. OPERA [11] , points out that hallucinations in LLMs are related to summary tokens such as periods, which lead to partial over-trust phenomena, meaning that hallucinations are more likely to occur after summary tokens. Therefore, the article corrects hallucinations by detecting and eliminating the probability of summary tokens during the decoding phase. OPERA mainly addresses hallucinations in image LLMs and aims to solve hallucination problems caused by the biases inherent in LLMs, which is different from the focus of this paper. This paper aims to solve hallucinations caused by the loss of temporal information due to insufficient video sampling. In the responses within the two tasks set in this paper, summary tokens are rarely produced, so OPERA's effectiveness in correcting temporal hallucinations is very limited. Leng et. al. [12] propose a method using visual contrast (comparing normal visual input with distorted visual input) to obtain the uncertainty of target predictions under different visual conditions. Then, it corrects object-level hallucinations by retaining target category outputs with more consistent predictions and eliminating those with higher uncertainty. However, this article does not specifically address temporal hallucinations, so it cannot effectively solve hallucination problems in temporal-related responses in video Q&A.

Our approach is similar to RAG and FacTool, involving the extraction of queried event information through external tools to improve the performance of MLLMs in addressing temporal-related questions about video events. CLIP and BLIP2 are used as external tools for understanding the various events in videos. They both have good generalization capabilities, allowing them to understand a substantial portion of real-world concepts and recognize open-world video events effectively. However, their limitations and potential drawbacks are quite apparent. Despite having a very extensive training dataset, there are still some things that are not included in their training data, leading to poor recognition performance for these unseen items.

3 Method

We employ Video-LLaMA [28] for our base MLLM, which extends a standard LLM, LLaMA [23,24], to deal with multimodal inputs, such as image, video, and audio. To comprehend both visual and auditory elements in videos, Video-LLaMA has two branches in its architecture: the Vision-Language branch for dealing with input video frames, and the Audio-Language branch for handling input audio signals. Furthermore, it employs position embedding techniques to incorporate temporal information into these inputs, thereby enabling the model to accurately recognize/identify the timing of events in videos.

The rest of this section is organized as follows. Section 3.1 first introduces two tasks to quantitatively evaluate temporal hallucinations created by MLLMs. In Sec. 3.2, we explain a general method for event temporal hallucination correction. Finally, a method for response correction will be described in Sec. 3.3.

3.1 Two Tasks for Temporal Hallucination Evaluation

We introduce two tasks here to understand MLLMs' vulnerability to temporal hallucination. The tasks are defined in the form of Video Question Answering (VQA). The first task, which we will refer to as Task 1, involves predicting the timestamp of event occurrences, with typical query questions such as *When does/did the event ... occur?* The second task, called Task 2, involves predicting the order of occurrences for two events, with typical questions like *Did event A occur before/after event B?* Details of how such questions are created and evaluation methods will be explained below.

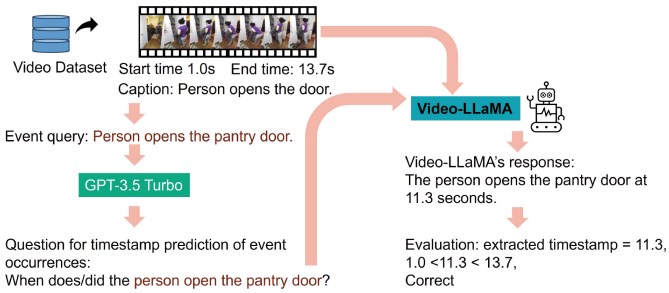


Fig. 3. Illustration of Task 1. The Q&A pairs are created from existing video datasets.

Timestamp Prediction of Event Occurrences. Task 1 requests an MLLM to correctly answer a question as to when a specific event occurs in an input video, e.g., *When does the person open the pantry door?* as shown in Fig. 3.

To create such a question and its corresponding answer, we utilize existing video datasets that provide captions of events occurring in the videos, as shown in Fig. 3. Specifically, for a video from the datasets, we first select a specific event in the video and identify its temporal duration, i.e., the timestamps for the start and end of the event. To convert the caption of the selected event into the form of a question, we use GPT-3.5-turbo with a prompt, detailed in Sec. 1.1 of Supplementary.

At the subsequent evaluation phase, we input each of the generated questions into the MLLM and perform a thorough evaluation of its answers. It is noteworthy that MLLMs answer the questions in diverse forms. They may output one or more timestamps, a time duration, or even vague answers like *at the beginning* or *the end.*; see Table 1. To handle this diversity without manual judgment, we

convert the diverse responses about time into a numerical time list or interval. We use GPT-3.5-turbo to parse these descriptive responses into a time list or interval for evaluation; see prompt in Sec. 1.2 in Supplementary. Some examples from the MLLM's responses and the the corresponding parsed time lists or intervals are shown in Table 1.

Table 1. Examples of response transformation for the evaluation of Task 1.

Response example	Parsed time set
Person opens the door at 3.2 second, 4.5 second.	$t \in [3.2, 4.5]$
Person opens the door from 3.2 second to 4.5 second.	$3.2 \leq t \leq 4.5$
Person opens the door in the beginning of the video	$0.0 \leq t \leq \frac{1}{3}L$
Person opens the door in the middle of the video	$\frac{1}{3}L \leq t \leq \frac{2}{3}L$
Person opens the door in the end of the video	$\frac{2}{3}L \leq t \leq L$
Person opens the door throughout the video	$0.0 \leq t \leq L$
No information mentioned.	$t \in \emptyset$

Order Prediction of Event Occurrences. Task 2 requests MLLMs to predict orders of two event A and B specified in an input question; see Fig. 4. As in Task 1, we utilize existing video datasets to create the task. Specifically, we randomly select two temporally different events with captions from an video, each with its own set of start and end time annotations. We shuffle their order, and utilize GPT-3.5-turbo to create a question such as *Did event A occur before/after event B?*. The prompt used for generating the event order questions is provided in Sec. 1.3 in Supplementary.

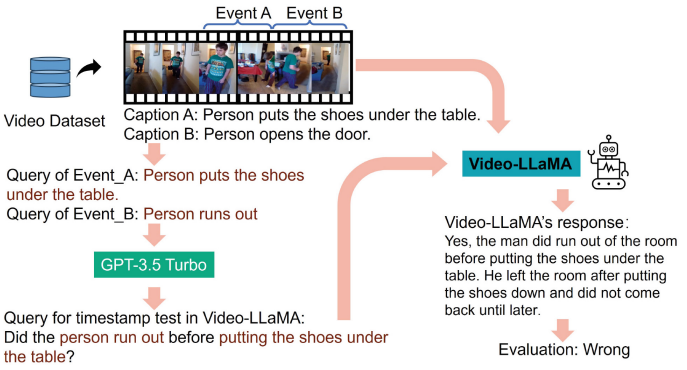


Fig. 4. Illustration of Task 2.

At the evaluation phase, we input each of the generated questions into an MLLM and assess its answers based on the ground-truth annotations of the two

events. Using in-context learning, we allow MLLMs to generate three types of answers: **Yes**, **No**, and **No relevant information**. The correct answers for the event orders can only be either **Yes** or **No**.

3.2 Event Temporal Hallucination Correction

To address the event-level temporal hallucinations of MLLMs demonstrated in Section 3.1, we propose to use external tools. We utilize them to infer temporal information about events more accurately and reliably, thereby correcting the time-related hallucination in MLLM's responses. Such corrected, accurate temporal statements or assertions are referred to as "claims" in the previous studies of hallucination correction and fact verification on LLMs. The process of utilizing claims for this purpose generally encompasses three phases:

- Claim Activation, which involves determining whether time-related hallucination correction is necessary;
- Claim Generation, i.e., the creation of structured assertions that incorporate or are based on the precise temporal and event-related information extracted from video content;
- Claim Integration, i.e., the incorporation of these claims to amend MLLM's output.

For each MLLM's response, we conduct these three phases, aiming to mitigate potential temporal hallucination.

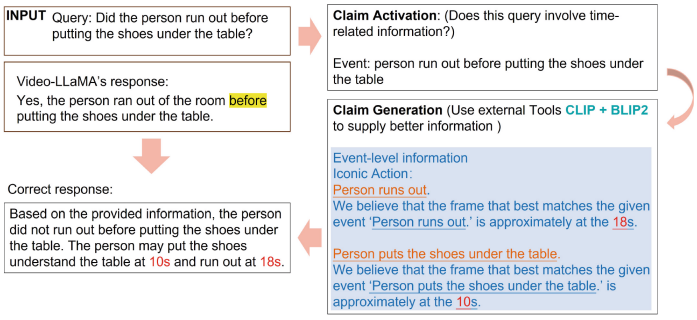


Fig. 5. Illustration of event temporal hallucination correction.

Figure 5 illustrates the first two correction phases. In the initial correction step, the Claim Activation phase takes the user's query as input and utilizes GPT-3.5-turbo to determine if the query requires temporal information support. Additionally, it detects the events in the query, and the identified event text serves as input for the Claim Generation module.

In second phase, the Claim Generation module generates a claim for correcting temporal hallucinations based on the inputted events. We have designed an

external tool using CLIP and BLIP2 to obtain specific event temporal information. After filling in the template with this information, the Claim is generated. This phase ensures that the correction process is informed by accurate temporal details that have been queried, mitigating the temporal hallucination in the MLLM's responses.

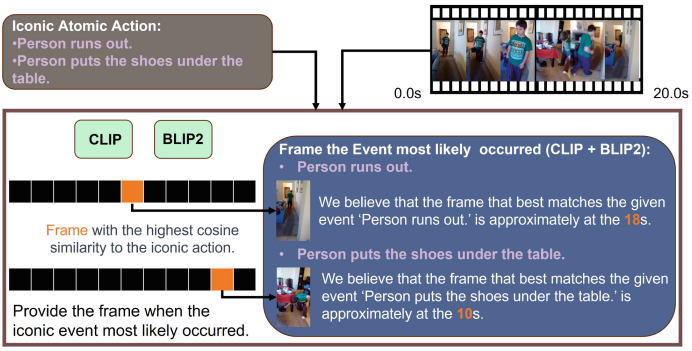


Fig. 6. Illustration of timestamp identification for iconic actions.

In Fig. 5, the text enclosed by dotted lines represents the claim template, and the portions with underlines indicate the corresponding event information to be filled in based on external tools.

The whole Claim Generation module can be organized as two steps:

1. Decompose the given event description to several iconic actions. This step is to improve the event prediction precision via CLIP-like external tools. We thus decompose the original event query to multiple "Iconic Actions", which refer to actions with visual representations that are easily recognized by image-based vision language models such as CLIP.
2. Provide the frame when the iconic event most likely occurred. In this step we find the frame when the iconic event most likely occurred leverage CLIP and BLIP2. Then we can predict the timestamp of the frame as the specific occurrence time of the give event. The timestamps we predict will be utilized as factual evidence to populate the claim template.

Decompose Event to Iconic Actions. To improve the event prediction precision through CLIP-like external tools, our method decomposes the provided event description into distinct "Iconic Actions". "Iconic Action" refers to distinctive, easily recognizable actions or events within a video that are emblematic of the content or narrative, aiding in quick comprehension and contextual understanding for viewers.

For example, in a video of a football match, iconic actions might be captioned as follows:

The player kicks the ball, it sails past the goalkeeper, and lands in the net, followed by the crowd's loud cheers.
This description involves four iconic actions:

1. *The player kicks the ball.*
2. *The ball sails past the goalkeeper.*
3. *The ball lands in the net.*
4. *The crowd's loud cheers.*

With human common sense, we can easily envision stereotypical images of these four iconic actions. It is also easy to visually separate each action by observing dynamic changes such as the position of the player's foot, the location of the ball, and the state of the audience.

We employ GPT-3.5-turbo to obtain such iconic actions from the input event descriptions using a prompt given in Sec. 2.1 of Supplementary. The resulting iconic actions lay the groundwork for a more nuanced understanding of the event.

Timestamp Identification for Iconic Actions. To identify the timestamps of iconic actions, our method finds the frame when the iconic action most likely occurred using external tools, i.e., CLIP and BLIP2, as illustrated in Fig. 6. CLIP and BLIP2 calculate the matching score between each frame and the text of each iconic event, providing the frames with the highest scores. Let N be the number of frames, I_k be the k^{th} frame, and Q_j be the text of the j^{th} iconic action. Denoting the CLIP's image and text matching score by $\cos_{\text{CLIP}}$ and BLIP2's by $\cos_{\text{BLIP2}}$, we select the most representative timestamp T_τ for the j^{th} iconic action as

$$T_\tau = \underset{k=1,\ldots,N}{\operatorname{argmax}} (\cos_{\text{CLIP}}(I_k, Q_j) + \cos_{\text{BLIP2}}(I_k, Q_j)). \tag{1}$$

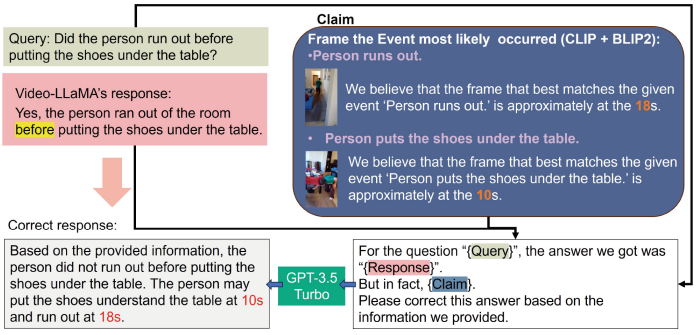


Fig. 7. Illustration of Claim Integration for response correction.

To further improve frame matching performance, we employ the test-time distribution normalization method [31] to enhance CLIP's matching performance.

Specifically, we normalize the text and image feature of the current Q_j and all frames $I_k, k = 1, 2, ..., N$. The new matching score is computed as:

$$S_{DN} = \cos_{\text{CLIP}}(I_k - \lambda\mu_I, Q_j - \lambda\mu_{Q_j}), \tag{2}$$

where μ is the mean value; λ is set to 0.25, as in [31]. The timestamp T_{τ^*} is predicted by the updated matching score as

$$T_{\tau^*} = \underset{k=1,...,N}{\text{argmax}}(\cos_{\text{CLIP}}(I_k, Q_j) + \cos_{\text{BLIP2}}(I_k, Q_j) + S_{DN}). \tag{3}$$

Both Q_j and T_{τ^*} will be filled into the claim template.

3.3 Claim Integration for Response Correction.

The user's query, MLLM's response, and the generated claim are integrated to derive the new corrected response in the claim integration phase. Figure 7 illustrates the procedure of correcting the MLLM's response using the generated claim. A corrective prompt (see Fig. 7), comprising the user's query, MLLM's response, the generated claim, and GPT-3.5-turbo, is utilized to integrate the correct response.

4 Experiment

4.1 Experimental Setting

Dataset. We employ Charades-STA [10], which is a widely used temporal activity dataset in the field of moment retrieval and temporal sentence grounding. In Sec. 3.1, we explain how we use videos with captions and temporal annotations to create our specific evaluation tasks. For Task 1 (i.e., timestamp prediction), we evaluate using all sentences in the test set (3,720 sentences from 1,334 videos) of Charades-STA. For Task 2 (i.e., order prediction), we initially filter 637 videos and then randomly generate 1272 questions containing *before* or *after*. The temporal overlap between each pair of events in these questions is less than 0.5.

Baseline model. Video-LLaMA [28] with Llama-2-7B-Chat as language decoder is used as the baseline MLLM for it supports video and image input. GPT-3.5-turbo is used as the LLM. CLIP ViT-L/14 336px [19] and BLIP2 [14] are used as external tools to get relevant frames' timestamps.

Implementation details. Video-LLaMA's number of beam search is set as 1. The temperature is set as 0.1 and 1.0 for different experiments. The beam search number in experiments is set as 1.0. We use 1 FPS for extracting frames from video.

Evaluation metrics. In Task 1, a relaxed metric is used to evaluate MLLM's various responses. If the extracted timestamps set from the MLLM's response is represented as $\{t\}$, and the annotated start and end timestamps of the event are denoted as T_s and T_e respectively, the response is deemed correct if the condition $\exists t \in \{t\} : T_s \leq t \leq T_e$ is satisfied; otherwise, it is considered incorrect. We use

R@1 and **R@5** as the evaluation metric of random experiment and corrected responses. **R@1** means only the frame timestamp with largest score in Equation 3 will be used for evaluation while **R@5** means the top 5 frame timestamps will be used for evaluation. In Task 2, the MLLM's response is considered correct only if its categorized results match the ground truth class (**Yes** / **No**).

4.2 Temporal Hallucination Evaluation and Correction Result

We evaluate and correct the temporal hallucination on Task 1 and 2.

Results on timestamp prediction of event occurrences. The results for timestamp prediction of event occurrences are presented in Table 2. Results marked with an asterisk in the Video-LLaMA column were obtained without restricting the output format of Video-LLaMA.

Even with the relaxed evaluation criteria for Video-LLaMA, it can be observed that Video-LLaMA only marginally outperforms random temporal predictions. In contrast, our method significantly outperforms both random predictions and the original Video-LLaMA. The result shows that our temporal hallucination method can reduce 39.7% event timestamps prediction errors. An intuition example that can the temporal hallucination is corrected is shown in Fig. 8(a). This video is about a car crash. If we only use the original video MLLM, such as video-LLaMA, the car crash event may be lost because it occurs in a very short duration for the key car crash frame might not be sampled by the video MLLM. However, by using frame-based external tools like CLIP and BLIP, these tools can find the precise timestamp of the event at the frame level. By obtaining accurate timestamps with CLIP and BLIP, we successfully corrected the output of the original MLLM.

In this experiment, we also comparing the influence of different MLLM's parameter. The performance of MLLM under different temperatures is very close.

Table 2. Results on timestamp prediction

Method	R@1 Acc	R@5 Acc
Random	25.59	52.63
Video-LLaMA [28] (temp = 0.1)	29.57	
Video-LLaMA [28] (temp = 1.0)	29.81	
Hallucination-reduced MLLM (ours)	57.66	85.29

Results on order prediction of event occurrences. The results are shown in Table 3. In terms of answer range, predicting the sequence of events is relatively easier. As shown in the table, the results indicate a clear improvement with our method compared to both random predictions and Video-LLaMA.

The results show that our temporal hallucination method can reduce 36.1% event occurrence order prediction errors. An intuition example that can the

temporal hallucination is corrected is shown in Fig. 8(b). This video is about a woman playing with dogs. There are several fine-grained action events in the video. The two events shown in the figure are *The woman is patting the dogs* and *The woman walks towards the dogs*. Due to the same sampling problem, some temporal frames will be lost, making it difficult to recognize these detailed action events or determine their sequence accurately. However, our frame-based external tools, CLIP and BLIP, can identify these fine-grained actions, thereby improving the accuracy of action sequence determination.

Table 3. Results on order prediction

Method	Acc
Random	24.20
Original Video-LLaMA [28]	49.21
Hallucination-reduced MLLM (ours)	67.53

4.3 Ablation Experiment for External Tools

We compared the performances of different tools–CLIP, BLIP2, and CLIP with subtracted mean values for both image and text–in determining timestamps (see Table 4). The experimental results indicate that ensembling these models can effectively enhance timestamp prediction performance.

The performance improvement with different embedding methods for retrieval is not significant. This is because the two foundation models used, CLIP and BLIP2, are highly homogeneous. In other words, both models perform well on certain events and poorly on others. This behavior is related to the similar nature of the training data used for both models, leading to homogeneous training-induced biases in their performance.

Table 4. Ensemble Tool Results Comparison

Model	R@1 Acc	R@5 Acc
Original CLIP	56.59	84.89
BLIP2	54.67	83.49
CLIPwithDN	56.56	85.00
CLIP + BLIP2	57.31	84.62
CLIP + CLIPwithDN	56.53	84.92
CLIP + BLIP2 + CLIPwithDN	57.66	85.29

4.4 Different Prompt Instruction for Response

In Task 1, we attempt to include prompts in the final query that explicitly guide the MLLM to predict specific time points, facilitating a more precise response in terms of timestamps rather than broad temporal information. We compare the results of MLLM under two conditions: with added timestamp prompts and without prompts. The results in Table 5 show that questions asking for timestamp responses obtain better performance.

Table 5. Results of different prompt instructions

Question	Acc
When the event occur?	27.84
When the event occur? Please answer the timestamp when you think the event is most likely to occur. Only return the timestamp.	29.81

5 Discussion

This section aims to discuss the strengths and weaknesses of our solution compared to the fine-tuning approach for MLLMs. It also explores whether the performance of the large pre-trained model and the external tools we rely on would impact the mitigation of temporal hallucination.

5.1 Advantages Over Fine-Tuning MLLM Solution

The concept scope of user-queried events is extensive. Enhancing the MLLMs' comprehension of time-related event information through fine-tuning necessitates a substantial amount of annotation, resulting in high training costs. The method proposed in this paper utilizes a widely-used zero-shot vision concept identifier model. This approach not only accommodates various real-time event queries from users but also eliminates the need for training, providing significant cost advantages.

5.2 Performance Impact of LLMs and External Tools

Our method relies on external tools such as CLIP and BLIP2 to provide temporal event information, while also utilizing large language models to process textual event information. Contrasting the results of Video-LLaMA in Table 2 and Table 3 with those of external tools in Table 4, it's evident that external tools significantly outperform Video-LLaMA in temporal event localization. Therefore, employing CLIP and BLIP2 to correct temporal hallucinations is

reasonable. They excel in generalizing to many real-world concepts and effectively recognizing open-world video events. However, they struggle to recognize items not covered in their training data, resulting in poor performance for these unseen elements. While it is challenging to find an external tool that consistently achieves 100% accuracy in temporal event localization, we can easily find zero-cost, zero-shot external tools like CLIP and BLIP2, which outperform them and can help improve the performance of MLLM in certain aspects.

In this paper, we employed the GPT-3.5-turbo, a moderately intelligent LLM. Experimental results demonstrate its capability in handling text-related tasks proficiently. Currently, there are many open-source LLMs that rival the performance of GPT-3.5-turbo. By utilizing such LLMs, the cost of our method is further reduced without compromising performance.

5.3 Choice of Evaluation Metrics

According to introduction in Section 4.1, **R@1** and **R@5** are used as the evaluation metrics for random experiments and corrected responses. These metrics are widely used in retrieval tasks, such as moment retrieval tasks. In moment retrieval tasks, top-1 or top-5 predicted moment candidates are selected to evaluate moment localization accuracy. The difference between our method and the metric for moment retrieval tasks lies in how we determine whether a prediction is correct.

In moment retrieval tasks, a predicted moment is considered correct if its overlap with the ground-truth moment exceeds a set threshold (e.g., 0.3, 0.5, 0.7). Otherwise, it is considered incorrect. However, current video MLLMs cannot predict precise moments but rather generate free text containing temporal information. Therefore, our method first parses the temporal information from the free text into a numerical set. We then relax the correctness condition to consider a prediction correct if the parsed numerical set has any intersection with the ground-truth event duration.

5.4 Scalability and Generalization

The temporal hallucination mitigating method we propose has excellent scalability and generalization. This method utilizes external tools to supplement the temporal event information lost due to sampling limitations in video MLLMs, and then uses the RAG mechanism to correct temporal hallucinations in MLLM responses. The RAG mechanism we employ is highly scalable and generalizable, not being constrained by the architecture of video MLLMs, and can provide temporal hallucination correction for different video MLLMs.

Regarding the type of input video, the scalability and generalization of our method strongly depend on the external tools used. In our experiments, we use CLIP and BLIP2, vision-language (VL) foundation models trained on millions of image-text pairs. These models have considerable cognitive abilities for various open-world concepts and generalize well to concepts involved in various

video events. To apply this method to other specific domains, specialized VL foundation models and domain-specific models can be used as external tools.

In Figure 8, we illustrate two typical examples from two domains that may cause temporal hallucinations due to sampling issues. The first is instantaneous events, such as car crashes, which occur in a very short duration. These events can easily lose information due to sampling limitations, as they happen almost instantaneously. The second type is fine-grained action events, where sparse sampling can cause temporal action frames to change. This alteration in the temporal sequence can lead to failures in the video MLLM's ability to accurately recognize and interpret such action sequences. Our proposed framework offers potential improvements by reducing temporal hallucinations when querying instantaneous events and fine-grained action events in long videos. Furthermore, when combined with specialized and powerful external tools, it can further enhance accuracy in addressing these issues.

Fig. 8. Events that may be affected by sampling need temporal hallucination correction. (a) is from [3]. (b) is from [9].

6 Conclusion

In this study, we address a significant challenge in the field of Multi-Modal Language Models (MLLMs): event-level hallucinations during video input processing. We tackle this issue by breaking down input event queries into their constituent iconic actions. Utilizing models such as CLIP and BLIP2 for frame identification with the decomposed iconic actions, our method significantly enhances the accuracy of pinpointing specific events and understanding their temporal sequences. Our evaluation using the Charades-STA dataset has demonstrated a substantial reduction in temporal hallucinations, boosting the accuracy and reliability of MLLMs in processing video content. These confirm the effectiveness of our method. We hope the qualitative advancements will lay a solid foundation for further research in this area.

Acknowledgements. This work was partly supported by JSPS KAKENHI Grant Numbers 20H05952 and 23H00482. The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Amaro, I., Barra, P., Della Greca, A., Francese, R., Tucci, C.: Believe in artificial intelligence? a user study on the chatgpt's fake information impact. IEEE Transactions on Computational Social Systems pp. 1–10 (2023). https://doi.org/10.1109/TCSS.2023.3291539
2. Amaro, I., Della Greca, A., Francese, R., Tortora, G., Tucci, C.: Ai unreliable answers: A case study on chatgpt. In: International Conference on Human-Computer Interaction. pp. 23–40. Springer (2023)
3. Bao, W., Yu, Q., Kong, Y.: Uncertainty-based traffic accident anticipation with spatio-temporal relational learning. In: ACM Multimedia Conference (May 2020)
4. Brown, T., Mann, B., Ryder, N., et al.: Language models are few-shot learners. In: Annual Conference on Neural Information Processing Systems, NeurIPS. pp. 1877–1901 (2020)
5. Chen, J., Zhu, D., Shen, X., et al.: Minigpt-v2: Large language model as a unified interface for vision-language multi-task learning. arXiv preprint arXiv:2310.09478 (2023)
6. Chern, I., Chern, S., Chen, S., et al.: Factool: Factuality detection in generative ai–a tool augmented framework for multi-task and multi-domain scenarios. arXiv preprint arXiv:2307.13528 (2023)
7. Chiang, W.L., Li, Z., Lin, Z., et al.: Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. See https://vicuna. lmsys. org (accessed 14 April 2023) (2023), accessed 14 April 2023
8. Dhuliawala, S., Komeili, M., Xu, J., et al.: Chain-of-verification reduces hallucination in large language models. arXiv preprint arXiv:2309.11495 (2023)
9. Fabian Caba Heilbron, Victor Escorcia, B.G., Niebles, J.C.: Activitynet: A large-scale video benchmark for human activity understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 961–970 (2015)
10. Gao, J., Sun, C., Yang, Z., et al.: Tall: Temporal activity localization via language query. In: Proceedings of the IEEE international conference on computer vision, ICCV. pp. 5267–5275 (2017)
11. Huang, Q., Dong, X., Zhang, P., Wang, B., He, C., Wang, J., Lin, D., Zhang, W., Yu, N.: Opera: Alleviating hallucination in multi-modal large language models via over-trust penalty and retrospection-allocation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13418–13427 (2024)
12. Leng, S., Zhang, H., Chen, G., Li, X., Lu, S., Miao, C., Bing, L.: Mitigating object hallucinations in large vision-language models through visual contrastive decoding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13872–13882 (2024)
13. Li, J., Li, D., Savarese, S., et al.: BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In: Proceedings of the International Conference on Machine Learning. pp. 19730–19742 (2023)
14. Li, J., Li, D., Xiong, C., et al.: BLIP: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In: Proceedings of the International Conference on Machine Learning. pp. 12888–12900. PMLR (2022)
15. Li, Y., Du, Y., Zhou, K., et al.: Evaluating object hallucination in large vision-language models. arXiv preprint arXiv:2305.10355 (2023)
16. Liu, H., Li, C., Li, Y., et al.: Improved baselines with visual instruction tuning. arXiv preprint arXiv:2310.03744 (2023)

17. Liu, H., Li, C., Wu, Q., et al.: Visual instruction tuning. arXiv preprint arXiv:2304.08485 (2023)
18. Ouyang, L., Wu, J., Jiang, X., et al.: Training language models to follow instructions with human feedback. In: Annual Conference on Neural Information Processing Systems, NeurIPS. pp. 27730–27744 (2022)
19. Radford, A., Kim, J.W., Hallacy, C., et al.: Learning transferable visual models from natural language supervision. In: Proceedings of the International Conference on Machine Learning. pp. 8748–8763 (2021)
20. Ram, O., Levine, Y., Dalmedigos, I., et al.: In-context retrieval-augmented language models. arXiv preprint arXiv:2302.00083 (2023)
21. Rawte, V., Sheth, A., Das, A.: A survey of hallucination in large foundation models. arXiv preprint arXiv:2309.05922 (2023)
22. Shuster, K., Poff, S., Chen, M., Kiela, D., Weston, J.: Retrieval augmentation reduces hallucination in conversation. In: Findings of the Association for Computational Linguistics: EMNLP. pp. 3784–3803 (2021). https://doi.org/10.18653/V1/2021.FINDINGS-EMNLP.320
23. Touvron, H., Lavril, T., Izacard, G., et al.: Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971 (2023)
24. Touvron, H., Martin, L., Stone, K., et al.: Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288 (2023)
25. Yang, Z., Ping, W., Liu, Z., et al.: Re-vilm: Retrieval-augmented visual language model for zero and few-shot image captioning. arXiv preprint arXiv:2302.04858 (2023)
26. Yin, S., Fu, C., Zhao, S., et al.: A survey on multimodal large language models. arXiv preprint arXiv:2306.13549 (2023)
27. Yin, S., Fu, C., Zhao, S., et al.: Woodpecker: Hallucination correction for multimodal large language models. arXiv preprint arXiv:2310.16045 (2023)
28. Zhang, H., Li, X., Bing, L.: Video-llama: An instruction-tuned audio-visual language model for video understanding. arXiv preprint arXiv:2306.02858 (2023)
29. Zhang, Y., Li, Y., Cui, L., et al.: Siren's song in the ai ocean: A survey on hallucination in large language models. arXiv preprint arXiv:2309.01219 (2023)
30. Zhao, R., Li, X., Joty, S., et al.: Verify-and-edit: A knowledge-enhanced chain-of-thought framework. arXiv preprint arXiv:2305.03268 (2023)
31. Zhou, Y., Ren, J., Li, F., et al.: Test-time distribution normalization for contrastively learned visual-language models. In: Annual Conference on Neural Information Processing Systems, NeurIPS (2023)
32. Zhu, D., Chen, J., Shen, X., et al.: Minigpt-4: Enhancing vision-language understanding with advanced large language models. arXiv preprint arXiv:2304.10592 (2023)

MC-DBN: A Deep Belief Network-Based Model for Modality Completion

Zihong Luo[1(✉)], Chengzhi Liu[1], Zheng Tao[1], Kexin He[2], and Yitao Xu[1]

[1] Xi'an Jiaotong-Liverpool University, Xi'an, China
zihong.luo22@student.xjtlu.edu.cn
[2] Kean University, Union, USA

Abstract. Recent advancements in multi-modal artificial intelligence (AI) have revolutionized the fields of stock market forecasting and heart rate monitoring. Utilizing diverse data sources can substantially improve prediction accuracy. Nonetheless, additional data may not always align with the original dataset. Interpolation methods are commonly utilized for handling missing values in modal data, though they may exhibit limitations in the context of sparse information. Addressing this challenge, we propose a Modality Completion Deep Belief Network-Based Model (MC-DBN). This approach utilizes implicit features of complete data to compensate for gaps between itself and additional incomplete data. It ensures that the enhanced multi-modal data closely aligns with the dynamic nature of the real world to enhance the effectiveness of the model. We conduct evaluations of the MC-DBN model in two datasets from the stock market forecasting and heart rate monitoring domains. Comprehensive experiments showcase the model's capacity to bridge the semantic divide present in multi-modal data, subsequently enhancing its performance. The source code is available at: https://github.com/logan-0623/DBN-generate/.

Keywords: Multi-Modal · DBN Network · Stock Market Forecasting · Heart Rate Monitoring

1 Introduction

In the field of stock market predictive analysis and heart rate monitoring, recent advancements in artificial intelligence (AI) and multimodal methodologies have become indispensable for informed decision-making. These innovative approaches amalgamate various forms of data, including numerical, textual, and visual inputs, providing a holistic view of both market dynamics [1] and physiological health indicators. For stock market forecasting, this includes integrating intermittent and non-sequential multimodal data like erratic financial news and policy updates. In heart rate monitoring, it involves synthesizing diverse physiological and environmental data to predict health outcomes accurately. Traditional methods, including linear interpolation techniques[2], have been used to

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 474–488, 2025.
https://doi.org/10.1007/978-3-031-78183-4_30

address the discontinuities and intermittencies inherent in such multimodal data. However, these conventional methods often fail to capture the true trends and variability present in real-world data, leading to incomplete or imprecise models for both stock market trends and heart rate patterns.

The significant advancements in artificial intelligence and multi-modal technologies have profoundly impacted the fields of stock market prediction and heart rate monitoring. These advanced methodologies integrate various data types[1], including numerical, textual, and visual inputs, to provide comprehensive insights into market trends and physiological health indicators. In stock market prediction, the inclusion of intermittent and non-sequential data, such as financial news and policy updates, is crucial. However, these data types are likely to face the risk of incomplete modalities due to missing data. Similarly, heart rate monitoring necessitates the combination of different physiological and environmental data for accurate health predictions, where missing modalities also present challenges. Traditional modal completion techniques, such as linear interpolation methods[2], often fail to capture the true trends and variances in such multi-modal data, leading to suboptimal modelling outcomes. Thus, both fields confront the common issue of potential modality incompleteness in their predictive analyses.

To overcome these limitations, our research introduces the MC-DBN model, an innovative solution designed to intelligently impute missing data, capturing the inherent temporal volatility and patterns in both stock market and heart rate data. This approach not only fills gaps in multi-modal datasets but effectively addresses gaps in multi-modal datasets in alignment with the temporal dynamics of the data, enhancing the robustness and reliability of predictive models. Our work significantly contributes to the fields of stock market forecasting and heart rate monitoring in the following three aspects:

- **Innovative Multimodal Data Integration Framework**
  In constructing our framework, we carefully consider the unique characteristics of different data types and strategically employ Transformer architectures to process sequential data. This approach is combined with the LSTM model to capture extended dependencies in news or physiological data. Furthermore, in the design of the feature fusion module, we meticulously implement attention mechanisms to ensure the precise capture and integration of data.
- **Enhanced Multi-modal Data Completion Model:**
  This study proposes an innovative encoder model that combines Deep Belief Networks (DBN) with attention mechanisms, specifically designed for processing multimodal data. The model capitalizes on the hierarchical structure of DBN and the global perception capabilities of attention mechanisms to effectively extract complex data features, surpassing traditional linear interpolation methods. The unique architecture of DBN facilitates the reconstruction or completion of partially lost visible layer data after training, by activating hidden layers, thus efficiently addressing the modal completion challenge. The model demonstrates exceptional performance in applications such as heart rate detection and stock market prediction.

– **Innovative Modality Completion Loss Functions:**
During the training process of our model, we introduce three cooperative loss functions. Two of these functions utilize Mean Squared Error (MSE) loss to compare the data generated by modal completion with the original information, thereby optimizing the quality of the DBN modal-completed data. By combining specific loss for downstream tasks, we construct a global loss function for the overall network framework, thereby optimizing the overall network performance.

2 Related Work

2.1 Multi-modal Data Integration

Multimodal data refers to data that comes from different sources or modalities, such as numerical, categorical, and textual data[3]. In finance, multimodal data can be used for risk classification, accident detection, and stock price prediction, among others[4,5]. For instance, a study used a Multimodal Transformer for risk classification and investigated the use of data augmentation for risk data through automated retrieval of news articles[3]. Similarly, heart rate monitoring, involves combining physiological data with environmental and behavioral factors for comprehensive health analysis. The study found that the inclusion of numerical and categorical data improves the performance of the model, particularly for risks that are difficult to classify based on textual data alone[3]. The inclusion of diverse data types enhances model performance in both domains, especially for scenarios where single-modality data is insufficient. Another study proposed a hybrid information mixing module using BERT and BiLSTM for finance [5]. This methodology is equally applicable in heart rate monitoring, where it can facilitate the extraction of multimodal interactions between physiological time series and contextual health information.

2.2 Stock Market Forecasting and Heart Rate Monitoring

The convergence of stock market forecasting and heart rate monitoring in the field of multi-modal data analysis has opened new avenues in both the financial and healthcare sectors.

In stock market forecasting, AI and machine learning algorithms have been increasingly employed to predict market trends and movements. A notable example is the use of Deep Learning models, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), to analyze historical price data and market sentiment extracted from news articles and social media. The work of Mogharet al. (2020) demonstrates the effectiveness of using LSTM networks for predicting stock prices based on historical data [6]. Additionally, the integration of sentiment analysis from financial news, as explored by Agarwal et al. (2020), provides a more holistic approach to forecasting [7].

Concurrently, heart rate monitoring has seen significant advancements with the application of AI. Heart rate data, collected through wearable devices, is

analyzed using sophisticated algorithms to detect anomalies and predict potential health risks. A study by Bertsimas et al. (2021) highlights the use of machine learning techniques for real-time heart rate monitoring and anomaly detection [8]. Moreover, the combination of heart rate data with other physiological parameters for comprehensive health assessment is explored in the research by Hussain and colleagues (2020) [9].

2.3 Methods for Handling Missing Values

Handling missing values is essential in financial analysis and heart rate monitoring. Missing data, which can result from various causes like incomplete data entry or equipment malfunctions, leads to biases and reduced statistical power, affecting the validity of conclusions [10,11]. The three primary types of missing data are missing completely at random (MCAR), missing at random (MAR), and missing not at random (MNAR) [11].

Addressing missing data involves methods such as imputation or data removal [12]. Traditional imputation techniques include the Last Observation Carried Forward (LOCF) and the Next Observation Carried Backward (NOCB), which may introduce biases [13,14]. Advanced techniques like rolling statistics and interpolation methods offer more sophistication but can struggle with irregular patterns in financial and heart rate data [15,16].

PCA-based methods like bPCA and llsPCA have been effective in financial time series and can be adapted for heart rate data [17,18]. Deep learning-based imputation, including Multi-Layer Perceptrons (MLP), is emerging for long-term missing value estimation in both fields [19].

3 Methodology

3.1 RBM-Based Latent Representation Learning

To enhance and elaborate on our methodology described, it's essential to give a comprehensible explanation of how Restricted Boltzmann Machines (RBMs) work and then discuss how they are applied in the given context.

RBMs play a crucial role in unsupervised learning by identifying the probabilistic characteristics of input data. They form a bipartite graph consisting of visible units (representing input data) and hidden units (capturing latent features). The 'restricted' nature of RBMs means that there are no connections within either the visible or hidden layers, only between them.

The learning process in RBMs involves adjusting the weights and biases to minimize the reconstruction error of the input data. This is typically achieved using training algorithms like Contrastive Divergence, which approximates the gradient of the log-likelihood.

$$p(\mathbf{h}|\mathbf{v}) = \prod_j p(h_j|\mathbf{v}), \quad p(\mathbf{v}|\mathbf{h}) = \prod_i p(v_i|\mathbf{h}) \tag{1}$$

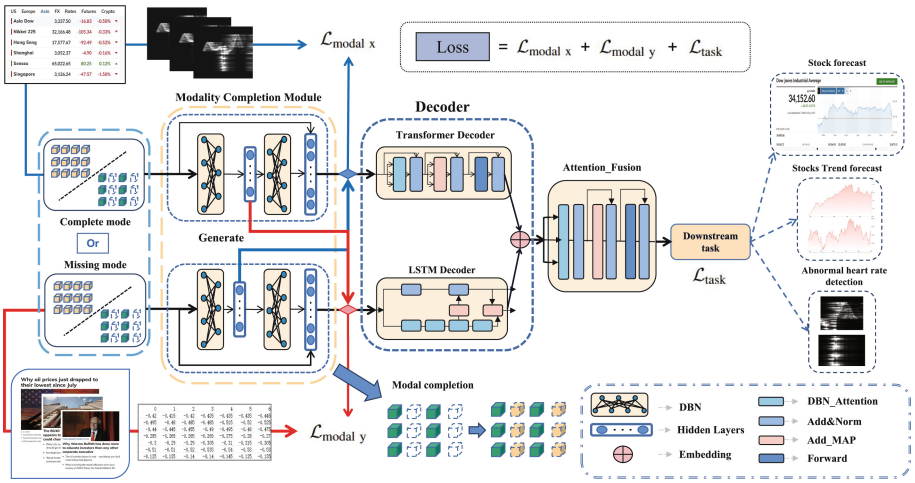


Fig. 1. The network comprises three main components. Initially, the modality completion module, leveraging a Deep Belief Network (DBN), identifies and fills absent modal data. Secondly, the modality feature extraction module employs Transformer[20] and LSTM architectures[21] for proficient feature extraction. Lastly, a fusion module with an attention mechanism integrates these features. Subsequently, the downstream network generates tailored predictions for multi-modal sequence data. This framework adeptly manages scenarios involving incomplete multi-modal data, which is a common situation in datasets like stock or heart rate data.

In this context, $\mathbf{h}$ and $\mathbf{v}$ represent the hidden and visible units, respectively.

In our proposed method, we start by inputting data into the RBM, a type of unsupervised learning algorithm adept at discovering probabilistic features in input data. An RBM is a bipartite graph consisting of visible units (corresponding to input data) and hidden units (representing latent features). The 'restricted' aspect in RBM comes from the fact that there are no intra-layer connections within visible or hidden units, only inter-layer connections.

The process can be mathematically described as follows:

$$\mathbf{h} = \sigma(\mathbf{W} \cdot \mathbf{v} + \mathbf{b}_h) \tag{2}$$

Here, $\mathbf{v}$ represents the visible units (input data), $\mathbf{h}$ the hidden units, $\mathbf{W}$ the weights of the inter-layer connections, and $\mathbf{b}_h$ the bias of the hidden units. σ is the sigmoid activation function.

The integration of RBM in this manner allows the model to capture and utilize deep non-linear relationships within the data, enhancing feature representation and supporting the generation of new synthetic data samples consistent with the learned distribution.

Deep Belief Network (DBN) is a sophisticated deep learning architecture featuring an input layer, numerous stacked layers housing random variables, and an output layer. The optimization process initiates with a meticulous layer-by-layer training approach for each Restricted Boltzmann Machine (RBM) network,

advancing systematically from the lowermost layer to the uppermost. Following this, the entire network undergoes refinement through fine-tuning, a process facilitated by the backpropagation algorithm. The fundamental structure of a Restricted Boltzmann Machine (RBM) is depicted in Fig. 2.

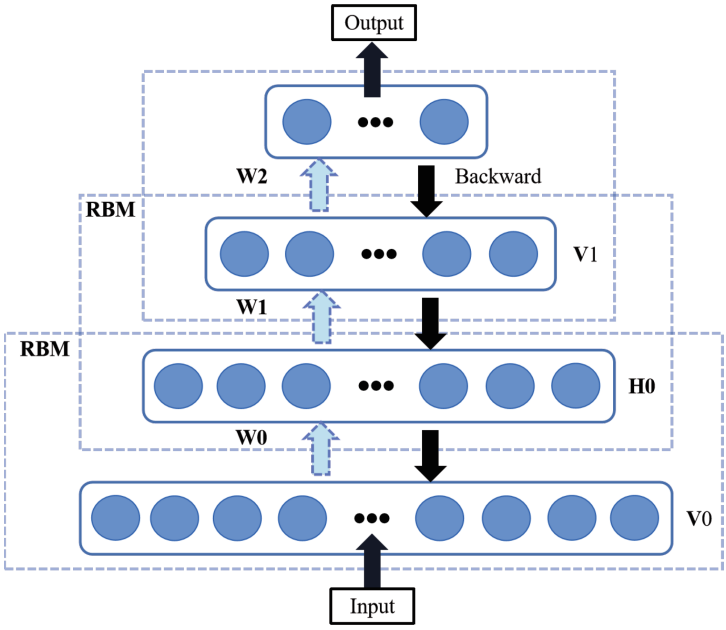


Fig. 2. Deep Belief Network (DBN) is a sophisticated deep learning architecture featuring an input layer, numerous stacked layers housing random variables, and an output layer. The optimization process initiates with a meticulous layer-by-layer training approach for each Restricted Boltzmann Machine (RBM) network, advancing systematically from the lowermost layer to the uppermost. Following this, the entire network undergoes refinement through fine-tuning, a process facilitated by the backpropagation algorithm.

3.2 Modal Completion Encoder-Decoder Framework

Tackling the challenge of incomplete data across various modalities, our approach leverages hidden features from an available modality to infer and regenerate missing data from another. This process, known as reverse sampling, backtracks from known to unknown modality data. It ensures comprehensive restoration of bimodal information, crucial in complex datasets where complete information is vital for accurate analysis.

We utilize Deep Belief Networks (DBNs) to construct the modality generation model. DBNs are generative models built by stacking multiple Restricted Boltzmann Machines (RBMs), allowing them to generate new data samples based

on learned representations. Their multi-layer architecture captures high-level abstract features, providing strong representation capabilities for complex data structures. We propose a modal completion framework based on an encoder-decoder structure designed to effectively address missing modality information (shown in Fig. 1). Employing a reverse sampling approach, we leverage the hidden features of a known modality to infer and reconstruct missing data in another modality. This innovative method not only fills gaps in the dataset but also significantly enhances the model's multimodal processing capabilities. Furthermore, we introduce an attention mechanism to perceptually complete modality information from a global perspective. The detailed schematic diagram of the modal completion structure is illustrated in Fig. 3.2. The workflow of the modal completion framework is outlined in Algorithm 3.2.

Algorithm 1. Modal Completion Encoder-Decoder Module

Require: Data of different modalities I_{x}, I_{y}
  Modal Completion Encoder:
  $W_{\mathrm{attn}} = \text{Self-Attention}(I_{\mathrm{x,y}})$
  $Attn_{\mathrm{x,y}} = \text{Softmax}\left(I_{\mathrm{x,y}} \odot W_{\mathrm{attn}}\right)$
  $\hat{H}_{\mathrm{x,y}} = \text{MC-RBM}_{\mathrm{hidden}}(I_{\mathrm{x,y}}, Attn_{\mathrm{x,\,y}})$
  Modal Completion Decoder:
  if I_x is incomplete **then**
    $G_{\mathrm{x}} = \text{MC-RBM}_{\mathrm{x\text{-}complete}}(\hat{H}_y) \rightarrow \mathcal{L}_{\mathrm{modal\ x}}(I_x, G_x)$
  else if I_y is incomplete **then**
    $G_{\mathrm{y}} = \text{MC-RBM}_{\mathrm{y\text{-}complete}}(\hat{H}_x) \rightarrow \mathcal{L}_{\mathrm{modal\ y}}(I_y, G_y)$
  end if
  $MC_{\mathrm{x}} = \text{Decoder\_Transformer}(G_x)$
  $MC_{\mathrm{y}} = \text{Decoder\_LSTM}(G_y)$
  return MC_{x}, MC_{y}

In the modal completion encoder sub-module, we employ an attention mechanism to process two modal data, denoted as I_x and I_y, obtaining corresponding attention weights W_{attn}. By appropriately balancing and weighting these attention-weighted representations with the original modal inputs, we obtain a novel attentional input $Attn_{x,y}$. This ensures a global awareness of the underlying modal information. By transforming the attention input into the form of a Modal Completion Restricted Boltzmann Machine (MC-RBM), we obtain the initial hidden state h. This transformation is achieved using a weight matrix $\mathbf{W}$ and a bias vector $\mathbf{b}_h$, with the Sigmoid function σ serving as the non-linear transformation.

The hidden state h undergoes a further Bernoulli sampling process, yielding a set of sampled binary hidden states h_{sampled}. This step introduces randomness, simulating the probabilistic nature of the hidden layer in the MC-RBM. The state of each hidden unit is determined by the corresponding unit's probability distribution, and it is barbarized through Bernoulli sampling, forming the final

hidden state.

$$\mathbf{h}_{sampled} = \text{Bernoulli}\Big(\sigma(\mathbf{W} \odot Attn_{x,y} + \mathbf{b}_h)\Big) \tag{3}$$

Finally, through the process of MC-RBM on the hidden layer information, a remodelling of the latent representation of multi-modal data is achieved to capture latent patterns and correlations within the data. In this process, a supplementary operation of cross-modal data mapping is introduced to ensure effective information completion in case of missing data in any modality. Through this transformation, the model can effectively fill in missing modality information, enhancing the overall data integrity and information representation capability.

In selecting the decoder for our framework, we opted for a combination of Long Short-Term Memory (LSTM)[21] and Transformer networks[20], recognizing their distinct strengths in handling sequential data and long-term dependencies. LSTM excels at capturing temporal relationships in data, while the Transformer, with its self-attention mechanism, performs exceptionally well in scenarios where input data has complex interdependencies. The combination of these two network structures is particularly suitable for applications like stock prediction and heart rate monitoring, which require handling complex data with long-term dependencies.

3.3 Attention Fusion Module

The outputs from independently processed modalities are integrated into the fusion module, which transforms raw data from each modality into a comprehensive and nuanced representation. This fusion module is composed of attention layers, mapping layers, and normalization mechanisms. The key component, $Attention_{dbn}$, globally assesses attention weights across both complete and missing data modalities.

The multi-head attention mechanism in this module selectively concentrates on information from various representation subspaces. It utilizes query (Q) and key (K) matrices for complete modality information, and a value (V) matrix for missing modality data. These matrices are represented as $Q, K \in \mathbb{R}^{d_q \times k}$ and $V \in \mathbb{R}^{d_v \times k}$, respectively. The embedding matrices $W_i^{Q,K,V}$ correspond to K, Q, V, enhancing the fusion module's capacity to utilize both complete and missing modality information effectively.

The attention mechanism is mathematically represented as:

$$Attention_{dbn} = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \times \text{Norm}(V) \tag{4}$$

For each head i in the multi-head attention mechanism, the process is:

$$\text{Multi\_attn}_i = \text{Concat}\big[Attention_{dbn}(Q_i, K_i, V_i) \times W_i^{Q,K,V}\big] \tag{5}$$

The fusion of feature information is further refined through mapping and normalization layers, forming the $Fusion$ output. Notably, a decoder supplements this process with a stochastic mechanism, integrating both complete and

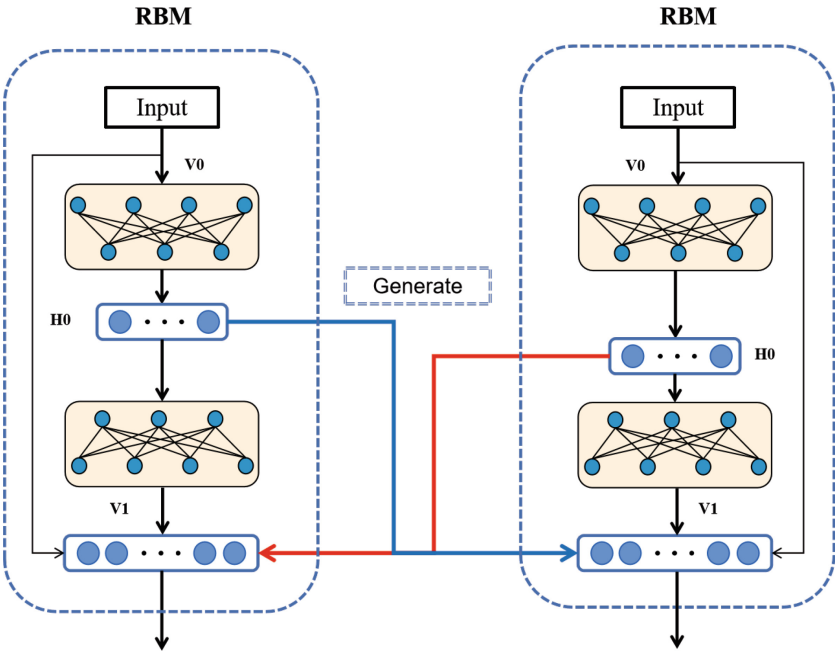


Fig. 3. For our modality completion mechanism, the algorithm employs MC-RBMs as probabilistic generative models to capture the latent representations of input data. The quality of the completed modality is optimized using Mean Squared Error (MSE) loss concerning the original modality features, enhancing the effectiveness of modal generation. The features sampled post-completion are then subject to a residual connection with the original features undergoing convolution operations, facilitating further processing and analysis.

missing information. This integration significantly enriches the fusion feature's informational depth and relevance.

$$Fusion = \text{Map}\Big(\text{Norm}(\text{Multi\_attn})\Big) \oplus \text{decoder}(MC_{x,y}) \tag{6}$$

3.4 Multiple Loss Function Design

Our model employs two Mean Squared Error-based loss functions to optimize and enhance performance. One is a modality completion loss function specifically designed for MC-DBN to accurately complete multi-modal data. The other is tailored for different downstream tasks. These loss functions address different aspects and components of the model, ensuring accuracy and efficiency in handling multi-modal data.

Loss Function for Modality Completion. The first loss function is dedicated to optimizing the generated modality of a specific data type, referred to as

Modality A. This function computes the MSE between the generated data of Modality A and the original input data of the same modality.

$$\mathcal{L}_{\text{modal x,y}} = \frac{1}{n}\sum_{i=1}^{n}\left(G^{(i)}_{\text{modal x,y}} - I^{(i)}_{\text{modal x,y}}\right)^2 \tag{7}$$

where $G^{(i)}_{\text{modal x,y}}$ represents the i-th complete data point for different modalities, and $I^{(i)}_{\text{model x,y}}$ is the corresponding original data point.

Loss Function for Specific Downstream Task. The second loss function is tailored for specific downstream tasks, such as classification, regression, or other predictive modelling tasks. This function is designed to optimize the model's performance in these tasks and is chosen based on the specific requirements of the application.

$$\mathcal{L}_{\text{task}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(p_{ij}) \quad \text{or} \quad \frac{1}{N}\sum_{i=1}^{N}\left(Y_{\text{actual}} - Y_{\text{predict}}\right)^2 \tag{8}$$

The first part corresponds to the loss function for a classification task, while the second part represents to the loss function for a regression task. y_{ij} represents the true label for class j of sample i. p_{ij} is the model's predicted probability for class j of sample i.

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{modal x}} + \mathcal{L}_{\text{modal y}} + \mathcal{L}_{\text{task}} \tag{9}$$

The overall loss function of the model combines these three losses, aiding in simultaneously optimizing the accuracy of completed modalities and the performance in the specific downstream task.

3.5 Model Training and Evaluation

We trained an LSTM-based network on datasets completed using different methods, including traditional interpolation, mean imputation, and our proposed MC-DBN. The performance was evaluated using RMSE, F1, and MAPE scores to assess the accuracy and predictive capability of each method.

4 Experiment

In this section, we elaborate on our experimental setup, comparing our MC-DBN methodology with other established approaches and conducting ablation experiments.

4.1 Data Preparation

Our study encompasses two primary datasets: a financial market dataset and the esteemed MIT-BIH Arrhythmia Database. The financial market dataset comprises a collection of stock opening prices and associated discrete news events, covering the period from January 1, 2020, to January 1, 2023. This dataset has been meticulously preprocessed to meet the input criteria of our analytical models. A visualization of this stock data is provided in Fig. 4.

In the realm of cardiac research, the MIT-BIH Arrhythmia Database holds a distinguished position as a comprehensive repository for electrocardiogram (ECG) signals, instrumental in the development and validation of ECG heartbeat classification algorithms [22]. Curated jointly by the Massachusetts Institute of Technology and Beth Israel Hospital (now known as Beth Israel Deaconess Medical Center) in Boston, this dataset has been employed extensively for investigating cardiac arrhythmia detection and diagnostic methodologies. A visualization of the arrhythmia dataset is presented in Fig. 5.

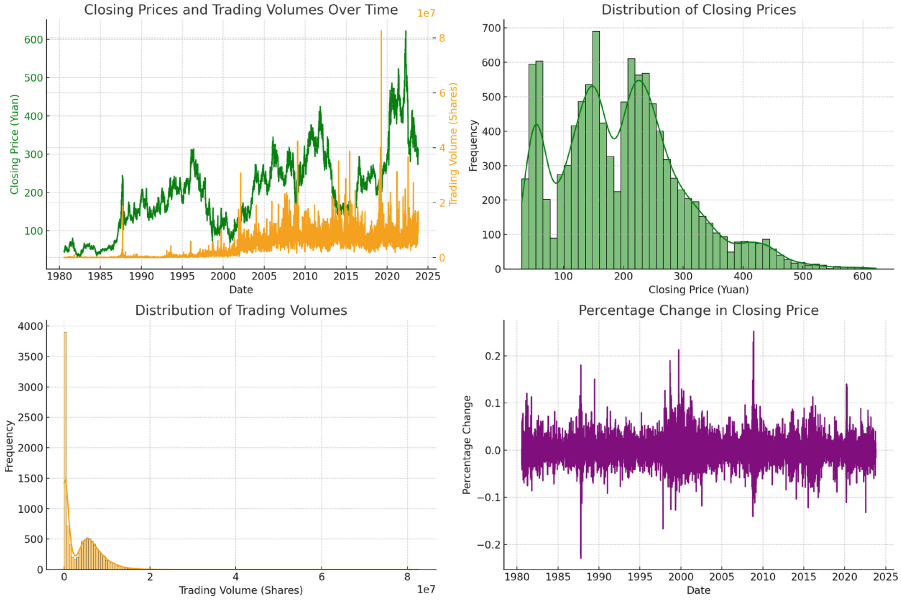


Fig. 4. Combined Visualizations of NEWMONT (NEM.N) Stock Data

4.2 Comparative Experiment

In our research, we integrated two diverse datasets: our unique stock data and the publicly available MIT-BIH arrhythmia dataset. This integration was essential to test the effectiveness and general applicability of our Deep Belief Network-based multimodal data completion method (MC-DBN).

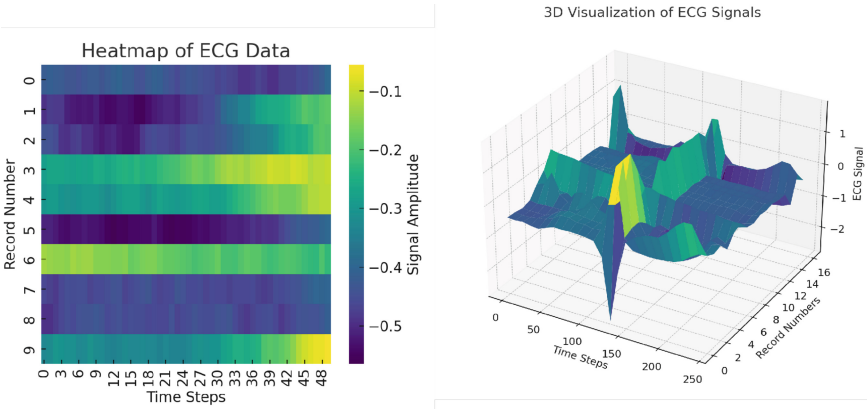


Fig. 5. Combined Visualizations of MIT-BIH ECG Data

This phase of our experiment focused on using various methods to complete a multimodal dataset. The dataset comprised two types of data: the opening prices of stocks in the medical sector and discontinuous news data, both covering the period from January 1, 2020, to January 1, 2023. The discontinuous news data were the primary target for completion using different methods. Subsequently, these datasets were utilized to train an LSTM-based network for predicting stock opening prices. We evaluated the network's performance using RMSE, MAPE, F1, and Accuracy Scores. RMSE and MAPE measured the model's error magnitude, whereas the F1 score assessed the model's accuracy in predicting price movements. The network's performance provided insights into the effectiveness of the multimodal data completion methods. To minimize error, we averaged the results from tests on ten different stocks.

As Table 1 illustrates, the single-modal data, being dependent on one data type for prediction, showed the highest RMSE and MAPE values, indicating larger prediction errors. In contrast, the use of multimodal data reduced these errors and improved price movement prediction accuracy. Notably, the MC-DBN processed multimodal data exhibited superior performance across all metrics, attributed to its ability to learn nonlinear features and ensure that the supplemented news data reflected actual news trends and variability.

The second Table 2 presents a similar comparison but focuses on the MIT-BIH dataset. As well, the MC-DBN method outperforms other approaches, reinforcing its efficacy in dealing with multimodal data for accurate predictions (Table 3).

4.3 Ablation Experiment

In our ablation experiments, focusing on stock opening price prediction, we analyzed the roles of the decoder and two loss functions within the MC-DBN. Table 4 demonstrates that $\mathcal{L}_{\text{modal x}}$ and $\mathcal{L}_{\text{modal y}}$ complement each other, thereby

Table 1. Comparison of other data completion methods on Stock dataset

Method	RMSE ↓	MAPE ↓	F1 ↑	Accuracy ↑
Single modal data	0.341	0.391	0.824	0.845
Multimodal data with Zero Filling	0.295	0.387	0.792	0.798
Multimodal data with Forward Fill	0.286	0.365	0.832	0.852
Multimodal data with Mean Imputation	0.279	0.361	0.856	0.870
MC-DBN (ours)	**0.268**	**0.346**	**0.874**	**0.913**

Table 2. Comparison of other data completion methods on MIT-BIH dataset

Method	F1 ↑	Accuracy ↑
Single modal data	0.852	0.874
Multimodal data with Zero Filling	0.867	0.886
Multimodal data with Forward Fill	0.876	0.892
Multimodal data with Mean Imputation	0.893	0.914
MC-DBN(ours)	**0.964**	**0.982**

Table 3. Comparison with previous state-of-the-art methods on MIT-BIH Arrhythmia test dataset.

Method	Accuracy ↑	F1 ↑
SR-based[23]	0.947	0.786
1D-CNN[24]	0.959	0.864
GANs[25]	0.987	0.929
MC-DBN(our)	**0.982**	**0.964**

enhancing the completion capability of the MC-DBN. Furthermore, $\mathcal{L}_{\text{modal x}}$ is crucial in guiding new data completion during model training, significantly improving model accuracy and consistency. Table 5 underscores the importance of employing LSTM for decoding news data and Transformers for sequential data decoding, highlighting their superior performance over traditional linear decoding methods.

Table 4. Ablation Experiment about Loss

	RMSE ↓	MAPE ↓	F1 ↑	Accuracy ↑
$\mathcal{L}_{\text{modal y}}$	0.282	0.358	0.853	0.869
$\mathcal{L}_{\text{modal x}}$	0.271	0.349	0.872	0.873
$\mathcal{L}_{\text{modal x}} + \mathcal{L}_{\text{modal y}}$	**0.268**	**0.346**	**0.874**	**0.913**

Table 5. Ablation Experiment about core components of decoder

	RMSE ↓	MAPE ↓	F1 ↑	Accuracy ↑
Both Linear	0.276	0.358	0.854	0.862
Only Transformer	0.274	0.352	0.859	0.875
Only LSTM	0.272	0.349	0.867	0.874
LSTM + Transformer	**0.268**	**0.346**	**0.874**	**0.913**

5 Conclusion

This study marks a significant leap in multi-modal data analysis, especially in stock market forecasting. By integrating Deep Belief Networks (DBNs) and Restricted Boltzmann Machines (RBMs), our methodology offers a novel approach to handle the inherent complexities and missing modalities in financial datasets. The use of RBMs for latent representation learning enables the extraction of deep, non-linear relationships within the data, thus enhancing the feature representation and reliability of the predictive models. Furthermore, our approach addresses the critical challenge of incomplete data by innovatively employing reverse sampling techniques. This not only ensures comprehensive bimodal information restoration but also significantly bolsters the resilience and versatility of our model in complex data environments. Empirical tests show our model's superior accuracy and predictive performance, notably in synthesizing data that closely mirrors real-world financial markets. This research not only advances AI's theoretical framework in multi-modal data analysis but also offers practical tools for financial forecasting. As we look to the future, the potential applications of this methodology extend beyond stock market forecasting, promising advancements in other domains where multi-modal data plays a crucial role.

References

1. Wang, H., Lu, S., Zhao, J.: Aggregating multiple types of complex data in stock market prediction: A model-independent framework. Knowl.-Based Syst. **164**, 193–204 (2019)
2. Wang, Q., Ward, R.K.: A new orientation-adaptive interpolation method. IEEE Trans. Image Process. **16**(4), 889–900 (2007)
3. N. Holtz, J. M. Gómez, https://api.semanticscholar.org/CorpusID:259255231 Multimodal transformer for risk classification: Analyzing the impact of different data modalities, Natural Language Processing and Machine Learning (2023). URL https://api.semanticscholar.org/CorpusID:259255231
4. H. Hozhabr Pour, F. Li, L. Wegmeth, C. Trense, R. Doniec, M. Grzegorzek, R. Wismüller, A machine learning framework for automated accident detection based on multimodal sensors in cars, Sensors 22 (10) (2022) 3634
5. A. Buche, M. B. Chandak, https://api.semanticscholar.org/CorpusID:261654967 Enhancing predictive modeling for indian banking stock trends: A fusion of bert and attention-based bilstm approach, Journal of Intelligent & Fuzzy Systems (2023). URL https://api.semanticscholar.org/CorpusID:261654967

6. Moghar, A., Hamiche, M.: Stock market prediction using lstm recurrent neural network. Procedia Computer Science **170**, 1168–1173 (2020)
7. A. Agarwal, Sentiment analysis of financial news, in: 2020 12th International Conference on Computational Intelligence and Communication Networks (CICN), IEEE, 2020, pp. 312–315
8. Bertsimas, D., Mingardi, L., Stellato, B.: Machine learning for real-time heart disease prediction. IEEE J. Biomed. Health Inform. **25**(9), 3627–3637 (2021)
9. L. Hussain, I. A. Awan, W. Aziz, S. Saeed, A. Ali, F. Zeeshan, K. S. Kwak, et al., Detecting congestive heart failure by extracting multimodal features and employing machine learning techniques, BioMed research international 2020 (2020)
10. Kang, H.: The prevention and handling of the missing data. Korean J. Anesthesiol. **64**(5), 402–406 (2013)
11. Donders, A.R.T., Van Der Heijden, G.J., Stijnen, T., Moons, K.G.: A gentle introduction to imputation of missing values. J. Clin. Epidemiol. **59**(10), 1087–1091 (2006)
12. Bennett, D.A.: How can i deal with missing data in my study? Aust. N. Z. J. Public Health **25**(5), 464–469 (2001)
13. Lachin, J.M.: Fallacies of last observation carried forward analyses. Clin. Trials **13**(2), 161–168 (2016)
14. Engels, J.M., Diehr, P.: Imputation of missing longitudinal data: a comparison of methods. J. Clin. Epidemiol. **56**(10), 968–976 (2003)
15. Broadstock, D.C., Li, R., Wang, L.: Integration reforms in the european natural gas market: A rolling-window spillover analysis. Energy Economics **92**, 104939 (2020)
16. Arun, P.V.: A comparative analysis of different dem interpolation methods. The Egyptian Journal of Remote Sensing and Space Science **16**(2), 133–139 (2013)
17. Schneider, M.P., Hilf, M., Vogt, U.F., Schmidt, M.W.: The benzene polycarboxylic acid (bpca) pattern of wood pyrolyzed between 200 c and 1000 c. Org. Geochem. **41**(10), 1082–1088 (2010)
18. C. John, E. J. Ekpenyong, C. C. Nworu, Imputation of missing values in economic and financial time series data using five principal component analysis approaches, CBN Journal of Applied Statistics (JAS) 10 (1) (2019) 3
19. R. Kruse, S. Mostaghim, C. Borgelt, C. Braune, M. Steinbrecher, Multi-layer perceptrons, in: Computational intelligence: a methodological introduction, Springer, 2022, pp. 53–124
20. T. Lin, Y. Wang, X. Liu, X. Qiu, A survey of transformers, AI Open (2022)
21. Yu, Y., Si, X., Hu, C., Zhang, J.: A review of recurrent neural networks: Lstm cells and network architectures. Neural Comput. **31**(7), 1235–1270 (2019)
22. Moody, G., Mark, R.: The impact of the mit-bih arrhythmia database. IEEE Eng. Med. Biol. Mag. **20**(3), 45–50 (2001). https://doi.org/10.1109/51.932724
23. M. Yamaç, M. Duman, İ. Adalıoğlu, S. Kiranyaz, M. Gabbouj, A personalized zero-shot ecg arrhythmia monitoring system: From sparse representation based domain adaption to energy efficient abnormal beat detection for practical ecg surveillance, arXiv preprint arXiv:2207.07089 (2022)
24. Kiranyaz, S., Ince, T., Gabbouj, M.: Real-time patient-specific ecg classification by 1-d convolutional neural networks. IEEE Trans. Biomed. Eng. **63**(3), 664–675 (2015)
25. Shaker, A.M., Tantawi, M., Shedeed, H.A., Tolba, M.F.: Generalization of convolutional neural networks for ecg classification using generative adversarial networks. IEEE Access **8**, 35592–35605 (2020)

Author Index

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15307, pp. 489–490, 2024.
https://doi.org/10.1007/978-3-031-78183-4