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

LNCS 15324

Pattern Recognition

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

24 Part XXIV

ICPR 2024 INDIA

IAPR

Springer

Lecture Notes in Computer Science 15324

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 XXIV

Springer

Editors

Apostolos Antonacopoulos
University of Salford
Salford, UK

Subhasis Chaudhuri
Indian Institute of Technology Bombay
Mumbai, India

Rama Chellappa
Johns Hopkins University
Baltimore, MD, USA

Cheng-Lin Liu
Chinese Academy of Sciences
Beijing, China

Saumik Bhattacharya
IIT Kharagpur
Kharagpur, India

Umapada Pal
Indian Statistical Institute Kolkata
Kolkata, India

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-031-78382-1 ISBN 978-3-031-78383-8 (eBook)
https://doi.org/10.1007/978-3-031-78383-8

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

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

If disposing of this product, please recycle the paper.

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

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 XXIV

Judgment Analysis in a Streaming Setting Incurring Logarithmic Space of the Annotators

Jyoti Patel, Soumen Mandal, and Malay Bhattacharyya(✉)

Indian Statistical Institute, Kolkata, India
malaybhattacharyya@isical.ac.in

Abstract. Judgment analysis in a crowdsourcing environment refers to seeking opinions from a diverse and large set of online annotators for various applications and making a consensus out of them. The existing approaches to judgment analysis work only in the static scenario where all the opinions are available beforehand. We aim to develop a generalized approach of judgment analysis in a streaming setting, where the questions are available but opinions received from the annotators are streaming in. The current paper provides the very first algorithm that can perform judgment analysis on crowdsourced opinions received in streams. We demonstrate the performance of the proposed approach on two datasets achieving an accuracy closer to the majority voting, although the space requirement of our approach is significantly better. The required space is bounded by a logarithmic factor of the number of annotators.

Keywords: Judgment analysis · Streaming algorithms · Patterns in streaming data · Crowdsourcing

1 Introduction

Crowdsourcing has made it possible to solve various kinds of decision making tasks by deploying human intelligence at scale. The power of the crowd has emerged as a revolutionary force promoting collaboration, innovation, and problem-solving in the current world of interconnectedness [1,11]. The idea of crowdsourcing has completely revamped the way we approach tasks, initiatives, and difficulties by leveraging the pooled intelligence and variety of abilities of a group of people [4,13]. Organizations and individuals can access a scalable pool of resources, ideas, and viewpoints that were previously inaccessible by utilizing the collective wisdom and skill of a crowd [12,22]. Crowdsourcing embraces the democratization of information and participation, moving beyond conventional hierarchical frameworks. It gives people the freedom to work together to achieve a similar objective while coming from various backgrounds, locations, and skill sets.

The potential of crowdsourcing to produce original and unique ideas is one of its main advantages. People can build on one other's ideas, refute preconceived

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 1–15, 2025.
https://doi.org/10.1007/978-3-031-78383-8_1

notions, and find ground-breaking solutions by bringing together a variety of perspectives. This collective intelligence frequently inspires unconventional thinking and game-changing inventions that might not have been possible through conventional methods. Crowdsourcing additionally enables effective resource allocation. Organizations can select individuals from a much broader talent pool (to assign tasks), rather than depending upon limited experts. This expedites the flow of work and guarantees diversity of talents and experiences.

Crowd-powered systems have existed for hundreds of years, despite the fact that they have only recently gained popularity [23]. Although these models are dispersed, they are noteworthy due to their combined potential to handle a variety of problems [11]. As an official system powered by the population, Amazon Mechanical Turk (MTurk)[16] made its debut in 2005 (Wikipedia). This is a new chance to acquire pertinent opinions for developing Information Retrieval (IR) test collections in a scalable and affordable manner. Thereafter, numerous platforms have been developed that include the biotechnology company 23andMe, crowdsourcing learning platform WikiProjects, crowdfunding website Kickstarter, Crowdfynd, etc. These crowdsourced systems can be broadly categorized as either collaborative or competitive depending on the talents of the crowd workers they employ [20]. Competitive crowdsourcing is a situation in which a sizable number of people work independently on a certain issue while harboring competing interests [20]. When a group of people collaborates to solve an issue, this is referred to as collaborative crowdsourcing [5]. For the first time, Ipeirotis [10] investigated collaborative crowdsourcing's behavior in 2010, and Boudreau et al. [2] investigated competitive crowdsourcing in 2011.

Different crowdsourcing topologies [3] call for various approaches to matching tasks to agents [8]. Open invitations to participate are sent by contest-based platforms like Top-Coder and InnoCentive, and the best submissions receive awards [14]. On the other hand, small jobs are delegated to available crowd agents on microtask platforms like Amazon Mechanical Turk on a first-come, first-served basis. However, it becomes essential to use effective allocation algorithms when working with platforms that involve skilled crowds and specialized labor, such as oDesk (now Upwork) [3], IBM's Application Assembly Optimization platform [21], and to some extent Samasource's SamaHub platform [9]. Based on the delivery technique they use, crowdsourcing platforms may be divided into two different models: a distributed micro-task model and a centralized model. They either use specialists or non-experts for group projects. In the past, tasks like the Netflix Prize and DARPA's Red Balloon competition [20] highlighted the engagement of expert crowds. The real-world behaviors seen on collaborative crowd-powered platforms are starting to be better understood by researchers [10]. These initiatives draw attention to a number of flaws and suggest that numerous systems powered by the masses need to be updated.

With crowd-based systems, opinion-based judgment analysis have been quite successful [18,19]. It is possible to combine multiple crowdsourced opinions to estimate the "gold" judgment (ground truth) for a given question [6]. Due to the involvement of non-experts as crowd workers, these collaborative models are often not effective in attaining the appropriate judgment. According to the studies by Sorokin and Forsyth, inaccuracies obtained in final judgments are caused by erroneous annotations [19]. In order to enable trustworthy crowdsourcing applications, it is essential to develop algorithms that can automatically separate the truth from possibly contradicting and noisy assertions made by multiple information sources. Truth-finding algorithms that are desirable must be efficient in order to handle crowdsourcing applications involving streaming data.

Streams of data from numerous sources, such as social networks, online stores, military monitoring, and sensors, to mention a few, are creating an enormous amount of data in the contemporary interconnected digital world. This makes it difficult to find a mechanism for classifying data streams that is both dependable and effective (in terms of time and space). Contrary to traditional models, big data models operate on the premise that the full dataset cannot be kept in its entirety and must instead be analyzed in real-time or on the fly [15]. As a result, we must evaluate the data in several passes—or maybe just one—due to memory constraints. Once more, as a fundamental requirement, it is expected that the storage space needed for data streams and sublinear growth will be seen in the processing time per item [17]. This new computing challenge, which goes beyond approximation and randomization, calls for the creation of innovative kinds of algorithms for many common problems that are faced in streaming environments.

A streaming scenario has never been considered for modeling the problem of judgment analysis. In this paper, we suggest a ground-breaking method for doing judgment analysis on streams of data. The main contributions made in this paper are listed below.

- We provide the first formalization of the judgment analysis problem in a streaming setting.
- We also propose an algorithm that can perform judgment analysis on crowdsourced opinions received in streams.
- We demonstrate the effectiveness of the proposed approach on real-life datasets.

The rest of the paper is organized as follows. Section 2 describes the basic motivation behind the current study. The basic terminologies of judgment analysis are introduced in section 3. Section 4 formulates the problem. Section 5 presents the state-of-the-art. Section 6 institutes the proposed methodology. The empirical analyses are provided in section 7. Finally, section 8 concludes the paper.

2 Motivation

The problem of judgment analysis for crowdsourced opinions has always been addressed in a static scenario [6]. Streaming data analysis is capable for handling the influx of continuous data streams. The analysis of such streaming data is necessary for many real-world scenarios in order to get insightful knowledge and make quick judgments. Therefore, it is a demanding issue to pose the problem judgment analysis in a streaming setting. Billions of online users participate in various activities at any given moment. Therefore, it is impossible to simultaneously solicit the opinions of multiple users. The crowd workers typically offer their comments at various periods. Additionally, it is not feasible to store every response provided by the crowd-workers in order to make judgments. We provide the first-ever algorithm to analyze streaming opinions for judgment analysis, while restricting the space requirements within a logarithmic bound of the number of annotators. Though it might appear that, given the current space availability in high-end machines, this problem is not challenging, however for resource-constrained scenarios, it is always a practical problem. Moreover, the current approaches are not scalable, which we claim as our main theoretical motivation behind this work.

3 Basic Terminologies

Some basic domain-specific terminologies, which are fundamental to judgment analysis and used throughout the paper, are discussed below. The standard terminologies have their usual meaning unless specified otherwise.

- **Question**: It is formulated as a decision making query.
- **Annotator**: It is a crowd worker who provides an answer to a specific question that helps in decision making. An annotator may be good at some aspects of a task but may be bad at others.
- **Opinion**: It is an answer given by an annotator to a decision-making question.
- **Annotation**: It is the process of collecting opinions from the annotators.
- **Domain of opinions**: It denotes the possible set of opinions. Note that there is a finite set of possible opinions for a given question.
- **Aggregation**: It is the process of finding an ensemble of opinions after collecting the opinions from the annotators. In this way, by aggregating these individual opinions, a judgment can be obtained for each question.
- **Gold judgment**: It is the ground truth opinion for each question.
- **Question difficulty**: It is the level of hardness of a question.
- **Annotator accuracy**: It means how reliable the annotator is. The superiority of annotators is determined by their capability of providing accurate judgments.

We use the term *response matrix* to refer to the matrix of opinions collected from a set of annotators (demarcated across the rows) on a set of questions (demarcated across the columns). A *response column vector* refers to a single column of the response matrix, thereby denoting all the annotators' responses (though many of the entries might be empty) for a particular question. We consider that the response column vectors are received as streams. Hence, we aim to analyze multiple data streams of opinions (response column vectors) corresponding to the questions for deriving the judgment.

4 Problem Formulation

In this section, we formalize the judgment analysis problem for streaming data [15]. We take into account a set of decision making questions, $Q = (q_1, q_2, q_{|Q|})$, a set of annotators $A = \{A_1, A_2, \ldots, A_{|A|}\}$ at a particular timestamp. In a streaming environment, the annotation process symbolizes a mapping function $Q \times A \longrightarrow O$ at a particular timestamp. We consider this happening at a particular timestamp of receiving a window of data.

After any timestamp, we have to obtain the final judgment for all the questions in Q with the constraints mentioned above. For each timestamp, we can consider receiving a response matrix R as a matrix of dimension $|A| \times |Q|$, whose elements R_{ij} denote the opinion provided by the i^{th} annotator of that timestamp for the j^{th} question such that $R_{ij} \in O$ for all i, j. We presume that the data stream $T =< T_1, T_2, \ldots, T_l >$ consists of the opinions. Since the input data stream comes progressively in a streaming environment, it is not possible to store all the opinions at once, and therefore we process them using a window-based model where a window of data is received at a time. We aim to perform judgment analysis on the stream of opinions (response column vector) for each question such that (i) the processing time for each question is O(N), and (ii) the storage requirement is O(polylog(N)), at any point in the data stream where N denotes the number of annotators.

It is considered that the annotator chooses only one option for a particular question from that set of possible opinions. Moreover, all the annotators do not provide their opinions at the same time. So, the responses of the annotators come in a streaming way. Depending on the mass opinions in that context we have to find the best option for each question. In reality, most of the annotators respond to limited questions, and hence R is a sparse matrix. Our approach is inspired from the DGIM (Datar-Gionis-Indyk-Motwani) algorithm [7].

5 Related Work

There are numerous approaches in the literature to perform judgment analysis on crowdsourced opinions [6]. However, most of these approaches are incapable of working on streams of opinions and provide judgments incurring space linear to the number of annotators. One of the limited approaches that seem applicable in a streaming setting is the majority voting algorithm. It is a simple yet effective approach used in judgment analysis to aggregate the opinions for identifying the majority opinion of multiple individuals to be returned as the final judgment. Majority voting has successfully been applied to obtain the ensemble of opinions. The advantages of the majority voting approach are simplicity, robustness, and low computational cost. On the other side, its disadvantages are giving equal weightage to the annotators, incapability to deal with correlated opinions, lack of consensus resolution, and space complexity. This can be further extended to assign different weightages to the opinions of different annotators based on their accuracy. The weighted majority voting considers the weights of annotators while identifying the majority opinion of multiple individuals. However, both these approaches have linear space requirements in terms of the number of annotators. We aim to reduce the space requirements of judgment analysis by considering the problem in a streaming setting.

6 Proposed Methodology

In this section, we first introduce a naive method for the streaming judgment analysis and then discuss a more practical and efficient approach.

6.1 A Naive Method

In this method, we use the mean value of the bitstream (mean of 0's and 1's) of opinions to derive the final judgment. We start with the very first bit as the initial mean and keep on revising the mean with the stream of bits inflowing. Let m be the old mean at a time point, i be the current bit, and n be the total number of bits processed so far, then the new mean value m' is calculated as $m' = \frac{m*(n-1)}{n}$. This formula ensures that the mean is updated incrementally as each new bit is processed in the stream. It is a rolling average that incorporates the information from the previous mean and the new bit to provide an updated mean with each iteration. At any arbitrary time point, from the current mean value, we can derive the final judgment. If the mean is more than 0.5, it is 1; if less than 0.5, it is 0; otherwise, it is a tie.

Time complexity: The time complexity of this approach is $O(N)$, where N is the number of bits in the stream for each question. In each iteration, the algorithm performs constant-time operations, updating the mean based on the new bit. Since it processes each bit once, the overall time complexity is linear with respect to the number of bits.

Space complexity: The space complexity is $O(|A|)$, where $|A|$ denotes the number of annotators. The algorithm only requires a few variables to store the current mean, the count of bits processed, and the loop variable. Regardless of the size of the input stream, the space used by the algorithm remains constant, making it efficient in terms of space.

6.2 An Efficient Method

In this method, we use the counts of 0's and 1's in the bitstream of opinions to derive the final judgment. We store the count of 0's and 1's in the stream using buckets having a variable number of elements but having exponentially increasing sizes in the reverse order inspired by the DGIM approach [7]. The counting of 0's and 1's are to be run in parallel. A bucket is a segment of the window having the following properties: (i) The size of a bucket (number of 0's or 1's in it) is in the form of 2^i, (ii) Each bucket contains the timestamp of its end bit (requires $O(\log |A|)$ bits) and its size (requires $O(\log \log |A|)$ bits).

Note that each bit in the stream has a timestamp, which is defined with a (mod $|A|$) function (to map everything within the window). The bitstream is represented with a collection of buckets following the conditions: (i) There can be either one or two buckets having the same size, (ii) Buckets are sorted by size, (iii) Buckets do not overlap, and (iv) Buckets are removed as and when their end-time is more than N time units in the past.

The buckets are updated when a new bit comes in. On the arrival of a new bit, the oldest bucket is dropped if its end-time is prior to N time units before the current time. No changes are required if the new bit is 0. Otherwise, do the following: (i) Create a new bucket of size 1 containing the new bit, (ii) Define the timestamp of the new bucket with the current time, and (iii) Starting from $i = 0$, recursively check whether there are now three buckets of size 2^i, and if so combine the oldest two to create a new bucket of size 2^{i+1}.

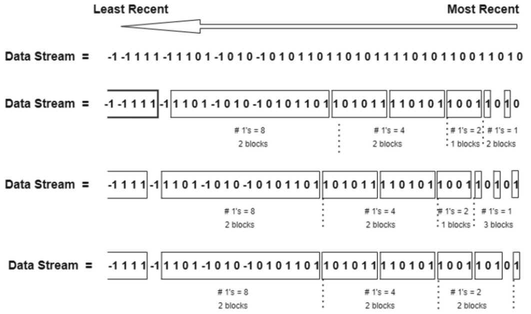

Least Recent
Most Recent
Data Stream = -1 -1 1 1 1 -1 1 1 0 1 -1 0 1 0 -1 0 1 0 1 1 0 1 1 0 1 0 1 1 1 1 0 1 0 1 1 0 0 1 1 0 1 0
Data Stream = -1 -1 1 1 1 -1 1 1 0 1 -1 0 1 0 -1 0 1 0 1 1 0 1 1 0 1 0 1 1 1 1 0 1 0 1 1 0 0 1 1 0 1 0
# 1's = 8
2 blocks
# 1's = 4
2 blocks
# 1's = 2
1 blocks
# 1's = 1
2 blocks
Data Stream = -1 1 1 1 -1 1 1 0 1 -1 0 1 0 -1 0 1 0 1 1 0 1 1 0 1 0 1 1 1 1 0 1 0 1 1 0 0 1 1 0 1 0 1
# 1's = 8
2 blocks
# 1's = 4
2 blocks
# 1's = 2
1 blocks
# 1's = 1
3 blocks
Data Stream = -1 1 1 1 -1 1 1 0 1 -1 0 1 0 -1 0 1 0 1 1 0 1 1 0 1 0 1 1 1 1 0 1 0 1 1 0 0 1 1 0 1 0 1
# 1's = 8
2 blocks
# 1's = 4
2 blocks
# 1's = 2
2 blocks

Fig. 1. The approach of counting the number of 1's in a data stream of opinions consisting of the elements 0/1/-1. Similarly, one can also count the number of 0's.

Note that while combining two buckets into a new one, the timestamp of the newest bucket becomes the timestamp of the new bucket. To estimate the number of 0's or 1's in the most recent $k \leq N$ bits: (i) Consider only those buckets whose timestamp is at most k bits in the past, (ii) Sum the sizes of all these buckets except the oldest one, and (iii) Add half the size of the oldest bucket. Some instances of the aforementioned process are highlighted in Fig. 1 and the main algorithm is formally presented as Algorithm 1.

In Algorithm 1, we take a window of response matrix $R_{|A| \times |Q|}$ and return the judgment vector $Judgement$ consisting of an ensemble of opinions corresponding to each question in Q. For each question (steps 1-13), we process a column vector denoting the set of opinions provided by the annotators and construct the buckets accordingly to count the number of 0's (steps 2-6) and 1's (steps 7-11). Based on the majority of opinions (step 12), we derive the final judgment and return the same (step 14).

The $INITIAL - BUCKETING()$ procedure is used to initialize a set of buckets for a given parameter a It involves creating a series of empty buckets represented by $BUCKETS$ The number of buckets created is determined by

Algorithm 1. Main algorithm for opinion ensemble on streaming data

Input: A window of response matrix $R_{|A|\times|Q|}$, where $|A|$ denotes the number of annotators in the current window and $|Q|$ denotes the number of questions.
Output: The judgment vector *Judgement* consists of ensemble of opinions corresponding to each question in Q.
Algorithmic Steps:

```
for $i := 1$ to $|Q|$ do
    $target \leftarrow 0$
    $TS \leftarrow 0$
    $BUCKETS \leftarrow$ INITIAL-BUCKETING($|A|$)
    RUN($R[*][i]$, $|A|$, $TS$, $BUCKETS$, $target$)
    $count0 \leftarrow$ COUNT($|A|$, $TS$, $BUCKETS$)          ▷ Count of 0's
    $target \leftarrow 1$
    $TS \leftarrow 0$
    $BUCKETS \leftarrow$ INITIAL-BUCKETING($|A|$)
    RUN($R[*][i]$, $|A|$, $TS$, $BUCKETS$, $target$)
    $count1 \leftarrow$ COUNT($|A|$, $TS$, $BUCKETS$)          ▷ Count of 1's
    $Judgment[i] \leftarrow \max(count0, count1)$
end for
return $Judgment$
```

the logarithm base 2 of a with each bucket initially empty. Finally, it returns the set of initialized buckets, setting the stage for further data organization and processing within these buckets based on the specified parameter a.

```
INITIAL-BUCKETING($a$) {
for $i := 1$ to $\lfloor \log_2(a) \rfloor$ do
    $BUCKETS[i] \leftarrow \emptyset$
end for
return $BUCKETS$
}
```

The OLDBUCKET() procedure identifies the oldest bucket within a specified time window, determined by a and TS in a structure called $BUCKETS$. It initializes *obIndex* and *obTimestamp* to 0, then iterates through each bucket and its elements. If an element's timestamp is within the time window $(TS - a)$, it updates *obIndex* and *obTimestamp*. If no such element is found in a bucket, it returns the current *obIndex* and *obTimestamp*, effectively locating the oldest relevant bucket in the data structure.

```
OLDBUCKET($a$, $TS$, $BUCKETS$){
$obIndex \leftarrow 0$                                  ▷ Initialize old bucket size
$obTS \leftarrow 0$                                     ▷ Initialize old bucket timestamp
for $i := 1$ to $size(BUCKETS)$ do                      ▷ Each bucket in $BUCKETS$
    for $ets \in BUCKETS[i]$ do                         ▷ Each element in $BUCKETS[i]$
        if $ets \geq (TS - a)$ then
```

```
            $obIndex \leftarrow i$
            $obTS \leftarrow endTS$
        **else**
            **return** $obIndex, obTS$
        **end if**
    **end for**
    **return** $obIndex, obTS$
**end for**
}
```

In the UPDATE() method, we update the buckets based on some constraints. If we have more than 2 buckets of the same size, merge them. The *delete*(X, i) function deletes the element at index i from X and returns the respective element. The *insert*(X, y, i) function inserts the element y at index i of X and shifts the elements at indices $i + 1$ and onwards to the next index.

```
UPDATE($BUCKETS$){
for $i := 1$ to $size(BUCKETS)$ do                    ▷ Each bucket in $BUCKETS$
    $l \leftarrow length(BUCKETS[i])$
    if $l \geq 2$ then
        $delete(BUCKETS[i], l - 1)$                    ▷ Remove the last element
        $temp \leftarrow delete(BUCKETS[i], l - 2)$    ▷ Remove the second last element
        if $i \neq size(BUCKETS) - 1$ then
            $insert(BUCKETS[i], temp, 0)$              ▷ Insert at the beginning
        end if
    end if
end for
return $BUCKETS$
}
```

The $COUNT()$ procedure calculates a count based on parameters a, TS, and a structure $BUCKETS$ It begins by initializing a count variable, *count* to 0. Then, it uses the $OLDBUCKET$ function to find the oldest bucket within the specified time window. Next, it iterates through the buckets in $BUCKETS$ If the current bucket index is greater than or equal to the oldest bucket's index, it returns the count plus 1. Otherwise, it iterates through the elements in each bucket, incrementing the count based on conditions related to timestamps. Finally, it returns the calculated count plus 1. This procedure effectively determines a count associated with the given parameters and data structure.

```
COUNT($a$, $TS$, $BUCKETS$){
$count \leftarrow 0$
$obIndex, obTS \leftarrow$ OLDBUCKET($a$, $TS$, $BUCKETS$)
for $i := 1$ to $size(BUCKETS)$ do
    if $i \geq obTS$ then
        return $count + 1$
    end if
    for $endTS \in BUCKETS[i]$ do
        if $endTS \geq obTS$ then
            $count \leftarrow count + 2^i$
        else if $endTS == obTS$ then
            $count \leftarrow count + 0.5 * 2^i$
        end if
    end for
end for
return $count + 1$
}
```

The $RUN()$ procedure processes a data stream based on parameters C a TS, $BUCKETS$, and *target* It iterates through each data point in the stream. For each data point, it increments the timestamp TS and finds the oldest bucket using $OLDBUCKET()$If the oldest bucket's timestamp is not 0 and matches $TS - a$ it checks if the current data point matches the *target* If it does, it updates the corresponding bucket in $BUCKETS$ by subtracting the old timestamp. Otherwise, if the data point is equal to the*target*, it adds the current timestamp to the first bucket and updates $BUCKETS$ using the $UPDATE()$ function. The procedure takes no action if the data point is -1. This process is repeated for each data point in the stream, effectively updating the buckets and their contents based on specified conditions.

```
RUN($C$, $a$, $TS$, $BUCKETS$, $target$){
for each $c \in C$ do                          ▷ Pick each data $(0/1/-1)$ from the stream
    $TS \leftarrow TS + 1$
    $obIndex, obTS \leftarrow$ OLDBUCKET($a$, $TS$, $BUCKETS$)
    if $obTS \neq 0$ and $obTS == TS - a$ then
        if $obTS \in BUCKETS[obIndex]$ then
            $BUCKETS[obIndex] \leftarrow BUCKETS[obIndex] - obTS$
        end if
    else if $c == target$ then                      ▷ Current data is $target$ (0/1)
        $BUCKETS[0] \leftarrow \{TS\} \cup BUCKETS[0]$
        $BUCKETS \leftarrow$ UPDATE($BUCKETS$)
    end if                                     ▷ Take no action if the data is $-1$
end for
}
```

Time complexity: Algorithm 1 iterates (steps 1-13) through each question in the input data, thereby running a loop $|Q|$ times, where $|Q|$ denotes the count of questions. For each such case, some initializations happen (steps 2-3) in constant time. Thereafter, the call to the INITIAL-BUCKETING() function (step 4) executes a loop that iterates $\lfloor \log 2_(|A|)\rfloor$ times, where $|A|$ denotes the count of annotators, which incurs a time complexity of $O(\log |A|)$. Then the call to the RUN() function (step 5) incurs a time complexity of $O(|A|)$. After this, the call to the COUNT() function (step 6) iterates through all the buckets, thereby incurring a time complexity of $O(\log|A|)$. This entire process (steps 2-6) of counting the number of 0's repeats once again (steps 7-11) for counting the number of 1's. Finally the judgment is obtained in a constant time operation as it involves comparing two values and taking the maximum. Hence, the total time complexity becomes as follows:

$$|Q| * \left(\overline{O(\log|A|) + O(|A|) + O(\log|A|)}^{\text{steps 2-6}} + \overline{O(\log|A|) + O(|A|) + O(\log|A|)}^{\text{steps 7-11}} \right)$$

$$\simeq O(|Q||A|).$$

Space complexity: Algorithm 1 iterates (steps 1-13) through each question in the input data, thereby running a loop $|Q|$ times, where $|Q|$ denotes the count of questions. For each such case, the INITIAL-BUCKETING() function (step 4) initializes $\log|A|$ number of buckets each having constant lengths, say k, where $|A|$ denotes the count of annotators. Hence, the total space requirement for creating and managing the $BUCKETS$ is $k * \log|A|$. The same space is getting utilized in the other function calls to RUN() (step 5), COUNT() (step 6), and so on. Space requirements of the rest of the variables are constants, say v. This entire process (steps 2-6) of counting the number of 0's repeats once again (steps 7-11) for counting the number of 1's. Finally, the judgment calculation also incurs a constant space, say j. Hence, the total time complexity becomes:

$$|Q| * \left(\overline{k * \log|A| + v}^{\text{steps 2-6}} + \overline{k * \log|A| + v}^{\text{steps 7-11}} + \overline{j}^{\text{step 12}} \right) \simeq O(|Q|\log|A|).$$

The overall space complexity of the algorithm is the sum of the space complexity for BUCKETS and the space complexity for other variables: Overall Space Complexity = Space complexity for BUCKET S + Space complexity for other variables Overall Space Complexity = $O(\log|A|)$ * C + D.

Error Factor Since there is at least one bucket of each of the sizes less than 2^i, and at least one from the oldest bucket, the true sum is no less than 2^i. Thus, the error is at most 50%.

7 Empirical Analysis

In order to measure and analyze the performances of the proposed approach in comparison with the other existing methods, we have used two Datasets, namely Fact Evaluation, opinion dataset and Sentiment Analysis Dataset. Experiments have been performed with Python 3.0 and the running environment is an 11th Gen Intel(R) Core(TM) i5-1135G7 with 2.4 GHz Processor, having 16 GB installed (15.8 GB usable) RAM, and 64-bit operating system.

7.1 Results on Fact Evaluation Dataset

The Fact Evaluation dataset contains the judgments of relations from people about public figures on Wikipedia. It contains 42,623 examples of “attended or graduated from an institution” example. Each of these was judged by a minimum of 5 annotators, resulting in a total of 216,725 judgments from the annotators who were already trained for this task, and as a whole, 57 annotators provide their opinions. The full version of the dataset contains relations in the form of a triplet: the relation in question, called a predicate; the subject of the relation; and the object of the relation. Subjects and objects are represented by their Freebase MIDs, and the relation is defined as a Freebase property. The evidence for the relations is also included in the form of a URL and an excerpt from the web page that our raters judged. The format of the files included in this dataset is in JSON.

Each line of the dataset consists of the following fields: predicate, subject, object, an array of evidence, the web page from which this evidence was obtained, a short piece of text supporting the triple, an array of judgments from human annotators, hash code of the identity of the annotator, and judgment of the annotator (0/1/2 denoting no/yes/skip respectively). In the annotation process, where 0, 1, and 2 are utilized to respectively represent “No”, “Yes”, and “Skip”, a modification has been made for improved interpretation. The original annotation “2” denoting “Skip“ has been replaced with “-1”. Consequently, the revised scale now consists of “0” for “No”, “1” for “Yes”, and “-1” indicating instances where the annotator opted to abstain from judgment or skip evaluation. This adjustment facilitates a more consistent and easily interpretable labeling system, simplifying subsequent analyses. Each line of annotation can now be understood in the context of this refined scale, where positive judgments are denoted by “1”, negative judgments by “0”, and skipped instances by “-1”. This approach enhances the clarity and uniformity of the annotation data, contributing to a more straightforward interpretation of the annotator’s judgments across the dataset.

Answers to 576 facts are available as the gold data. The basic version of the dataset contains two columns: one is question ID and the other is metadata. This metadata contains a JSON-encoded dictionary containing the judgments of all the raters’ for this question and other relevant data described below in the description section. The dataset is provided under the Creative Common license.

Table 1. The performance in terms of accuracy obtained for the Fact evaluation dataset and space requirement.

Algorithm	Accuracy(%)	Data Setting	Space requirement
Majority Voting	94.36	Non-Streaming	$\|A\|$
Weighted Majority Voting	94.36	Non-Streaming	$\|A\|$
Proposed Efficient Method	80.87	Streaming	$\log \|A\|$

The algorithm is compared with the well-known majority voting algorithm along with weighted majority voting applied on the Fact Evaluation dataset. Table 1 shows the comparative accuracy values obtained for the proposed approach on the large-scale Fact Evaluation dataset. Since our method is based on a streaming setting, we have obtained a much lesser accuracy value given the data is truly big. However, our space requirements are bounded by a logarithmic factor of the number of annotators unlike the other existing approaches, which are not applicable in a streaming setting.

7.2 Results on Sentiment Analysis Dataset

We used a sentiment analysis dataset from CrowdFlower, a crowd-powered company. The dataset consists of 98,979 tweets about the weather, each evaluated by at least five annotators, resulting in around 569,375 evaluations. The dataset includes detailed information such as: **Question ID**: The ID of the tweet being evaluated, **Rater ID**: The ID of the annotator, **Judgment**: The annotator's answer, which can be: `0`: Negative, `1`: Neutral (just sharing information), `2`: Positive, `3`: Not related to the weather, `4`: Cannot determine the sentiment, **Tweet Text**: The content of the tweet, **Country, Region, City**: The location details of the annotator, **Started At**: When the annotator started the evaluation, **Created At**: When the annotator finished the evaluation. For our analysis, we simplified the dataset to three columns: **Question ID**: The ID of the tweet, **Rater ID**: The ID of the annotator, **Judgment**: The annotator's answer (0 to 4). We converted the judgments as follows: `0` (Negative) remains `0`, `2` (Positive) becomes `1`, `4` (Cannot determine) becomes `-1`, We ignored the other responses (`1` and `3`) in our analysis (Table 2).

Table 2. The performance in terms of accuracy obtained for the Sentiment Analysis dataset and space requirement.

Algorithm	**Accuracy(%)**	**Data Setting**	**Space requirement**
Majority Voting	90.67	Non-Streaming	$\|A\|$
Weighted Majority Voting	97.12	Non-Streaming	$\|A\|$
Proposed Efficient Method	84.69	Streaming	$\log \|A\|$

8 Conclusion

We have proposed a space-efficient approach to judgment analysis in a streaming setting. The experimental results show promising directions of addressing the problem of judgment analysis in streaming settings. As the proposed methods of counting 0's and 1's are independent of each other, we can run them in parallel. Moreover, for each question, the algorithm can run in parallel. Hence, with a parallel computer, we can significantly reduce the time taken by our algorithm. The error factor in the proposed approach can be further reduced but with the cost of incurring more space. The proposed approach provides the count of 0's and 1's from a stream of opinions. This can be used to compute the entropy of the opinions and this might help us to quantify the difficulty of a question. From a practical perspective, if the entropy is more, there is more variability in the opinions, thereby suggesting that the question is more difficult. Using this one can plan for designing a weighted version of the proposed approach in a streaming scenario.

Data and Code Availability. Both the datasets, namely Fact Evaluation and Sentiment Analysis, analyzed in this paper along with the source codes of the algorithms applied on them are freely accessible from the GitHub link: https://github.com/malaybhattacharyya/Judgment_Streaming_Logarithmic.

References

1. Amer-Yahia, S., Roy, S.B.: Human factors in crowdsourcing. Proceedings of the VLDB Endowment **9**(13), 1615–1618 (2016)
2. Boudreau, K.J., Lacetera, N., Lakhani, K.R.: Incentives and problem uncertainty in innovation contests: An empirical analysis. Manage. Sci. **57**(5), 843–863 (2011)
3. Boudreau, K.J., Lakhani, K.R.: Using the crowd as an innovation partner. Harv. Bus. Rev. **91**(4), 60–9 (2013)
4. Buecheler, T., Sieg, J.H., Füchslin, R.M., Pfeifer, R.: Crowdsourcing, open innovation and collective intelligence in the scientific method: a research agenda and operational framework. In: The 12th International Conference on the Synthesis and Simulation of Living Systems, Odense, Denmark, 19-23 August 2010. pp. 679–686. MIT Press (2010)
5. Chatterjee, S., Bhattacharyya, M.: Judgment analysis of crowdsourced opinions using biclustering. Inf. Sci. **375**, 138–154 (2017)
6. Chatterjee, S., Mukhopadhyay, A., Bhattacharyya, M.: A review of judgment analysis algorithms for crowdsourced opinions. IEEE Trans. Knowl. Data Eng. **32**(7), 1234–1248 (2019)
7. Datar, M., Gionis, A., Indyk, P., Motwani, R.: Maintaining stream statistics over sliding windows. SIAM J. Comput. **31**(6), 1794–1813 (2002)
8. Dustdar, S., Gaedke, M.: The social routing principle. IEEE Internet Comput. **15**(4), 80–83 (2011)
9. Gino, F., Staats, B.R.: The microwork solution. Harvard Business Review **90**(12), 92–+ (2012)
10. Ipeirotis, P.G.: Analyzing the amazon mechanical turk marketplace. XRDS: Crossroads, The ACM magazine for students **17**(2), 16–21 (2010)
11. Kittur, A., Nickerson, J.V., Bernstein, M., Gerber, E., Shaw, A., Zimmerman, J., Lease, M., Horton, J.: The future of crowd work. In: Proceedings of the 2013 Conference on Computer Supported Cooperative Work. pp. 1301–1318 (2013)
12. Leimeister, J.M., Huber, M., Bretschneider, U., Krcmar, H.: Leveraging crowdsourcing: activation-supporting components for it-based ideas competition. J. Manag. Inf. Syst. **26**(1), 197–224 (2009)

13. Li, B., Cheng, Y., Yuan, Y., Yang, Y., Jin, Q., Wang, G.: Acta: Autonomy and coordination task assignment in spatial crowdsourcing platforms. Proceedings of the VLDB Endowment **16**(5), 1073–1085 (2023)
14. Liu, T.X., Yang, J., Adamic, L.A., Chen, Y.: Crowdsourcing with all-pay auctions: A field experiment on taskcn. Manage. Sci. **60**(8), 2020–2037 (2014)
15. Muthukrishnan, S., et al.: Data streams: Algorithms and applications. Foundations and Trends® in Theoretical Computer Science **1**(2), 117–236 (2005)
16. Paolacci, G., Chandler, J., Ipeirotis, P.G.: Running experiments on amazon mechanical turk. Judgm. Decis. Mak. **5**(5), 411–419 (2010)
17. Rubinfeld, R., Shapira, A.: Sublinear time algorithms. SIAM J. Discret. Math. **25**(4), 1562–1588 (2011)
18. Snow, R., O'connor, B., Jurafsky, D., Ng, A.Y.: Cheap and fast–but is it good? evaluating non-expert annotations for natural language tasks. In: Proceedings of the 2008 conference on empirical methods in natural language processing. pp. 254–263 (2008)
19. Sorokin, A., Forsyth, D.: Utility data annotation with amazon mechanical turk. In: 2008 IEEE computer society conference on computer vision and pattern recognition workshops. pp. 1–8. IEEE (2008)
20. Tang, J.C., Cebrian, M., Giacobe, N.A., Kim, H.W., Kim, T., Wickert, D.B.: Reflecting on the darpa red balloon challenge. Commun. ACM **54**(4), 78–85 (2011)
21. Varshney, L.R., Agarwal, S., Chee, Y.M., Sindhgatta, R.R., Oppenheim, D.V., Lee, J., Ratakonda, K.: Cognitive coordination of global service delivery. arXiv preprint arXiv:1406.0215 (2014)
22. Yang, Y., Cheng, Y., Yuan, Y., Wang, G., Chen, L., Sun, Y.: Privacy-preserving cooperative online matching over spatial crowdsourcing platforms. Proceedings of the VLDB Endowment **16**(1), 51–63 (2022)
23. Yu, L.: Daren c. brabham: Crowdsourcing: The mit press, 2013, 176 pp, 12.95, *isbn*9780262518475 (2014)

Active Self-paced Knowledge Distillation and Diffusion-Based Generation for Few-Call Model Stealing

Vlad Hondru and Radu-Tudor Ionescu(✉)

Department of Computer Science, University of Bucharest, 010014 Bucharest, Romania
raducu.ionescu@gmail.com

Abstract. The recent AI advancements, most notably Foundation Models, were naturally followed by an increased demand of AI-based solutions, which became widely available. However, the entities who offer these models are exposed to the risk of their models being stolen. Knowledge distillation demonstrated great benefits in reducing the inference time of deep neural networks, hence it has been an area of great interest. Model stealing represents a sub-category of knowledge distillation, which has a malign purpose: extracting the capability of a black-box model (teacher) into a local model (student). In this context, we propose a framework for stealing a model (which is accessible through an API) under very strict constraints (*e.g.* no access to the original training data, the architecture or the weights of the model), with a focus on making as few calls as possible to the teacher model. We first generate synthetic data using diffusion models, a powerful class of generative models showcasing strong capabilities in image synthesis. Although they have been extensively applied to many tasks in computer vision, we propose to explore a new use case in our work, namely generating an artificial data set (called proxy data set) for knowledge distillation. Next, we obtain the labels for synthetic samples by passing them through the black-box model. More precisely, we only collect the predictions (either soft or hard) for a number image samples, as we have to comply with a fixed number of allowed API calls. In our framework, we introduce a novel active self-paced learning mechanism to thoroughly utilize the proxy data during distillation to its complete potential. The final step consists of distilling the knowledge of the black-box teacher (attacked model) into a student model (copy of the attacked model). Besides the labeled collected data, we also utilize the remaining unlabeled data generated by the diffusion model. Our empirical results on two data sets with different characteristics confirm the superiority of our framework over two state-of-the-art methods in the few-call model extraction scenario.

Keywords: Diffusion models · Knowledge distillation · Model stealing

1 Introduction

In the last couple of years, Generative AI has gained a lot of popularity, starting with the release of ChatGPT [37]. This has sparked an increased interest in AI technologies. This surge in demand has persuaded enterprises of all sizes to deploy deep

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 16–30, 2025.
https://doi.org/10.1007/978-3-031-78383-8_2

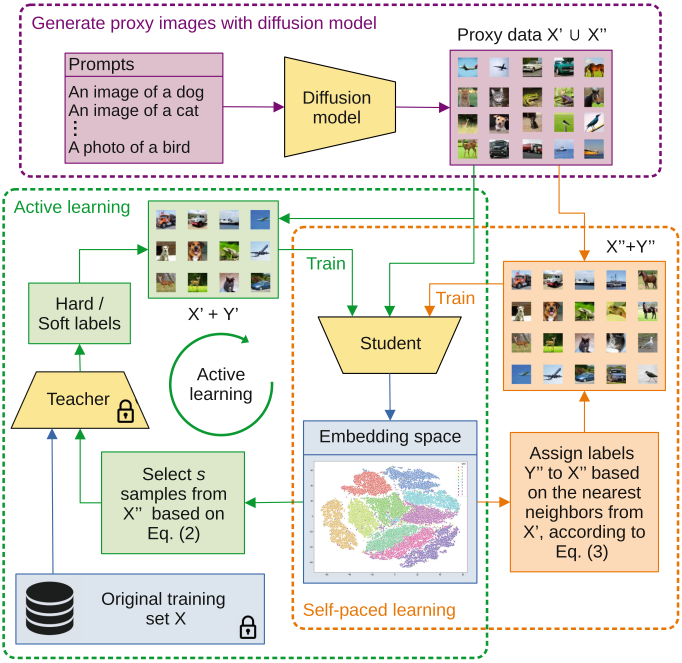


Fig. 1. The pipeline of our framework for model stealing starts by generating proxy images using a diffusion model. Then, the labels for the the synthetic images are gradually collected from the black-box teacher model. The annotated sample pairs are used to train the student model via an active learning scheme. At the same time, the remaining proxy images are pseudo-labeled via a nearest neighbor scheme that operates in the latent space of the student. The pseudo-labeled images are also used to optimize the student via a self-paced learning scheme. Best viewed in color.

learning models for commercial purposes. Such products are usually available through Machine Learning as a Service (MLaaS) platforms, which provide users with scalable AI tools via cloud-based APIs. They are accessible via various payment schemes, such as subscription-based or pay-as-you-go (for example, most Large Language Model APIs have a flat fee per an exact number of tokens). Although these models act in black-box regimes, the intellectual property is vulnerable to being stolen by carefully designed attacks [6,9,35,38,40,50,56,58], compromising the competitive advantage of such proprietary AI systems. The research of various attacking methods can only expose potential risks, which will lead to better prevention mechanisms (such as watermarking, limiting the query rate, or utilizing adversarial defenses).

To this end, we propose a novel pipeline to replicate the functionality of a black-box classification model (referred to as the *teacher*) by creating a locally trained surrogate model (referred to as the *student*). This is achieved through knowledge distillation [1,6,29,32,63] using artificially created data, also known as *proxy data*. Our method comprises two novel components that aim to increase the performance, while complying with a fixed query budget. Furthermore, we impose other constraints as well, *e.g.* hiding any information about the training procedure and preventing access to gradients, in order to holistically recreate a real-world scenario.

In Figure 1, we present the pipeline of our method. As depicted by the encircled regions, the pipeline is composed of three stages. The first stage is represented by the image generation using a diffusion model [11]. The synthetic data acts as proxy data, which is used to train the student. We employ diffusion models due to their capability of generating realistic, qualitative, and diverse images, demonstrating better synthesizing performance than generative adversarial networks (GANs) [12], which represents the previous state-of-the-art approach. Diffusion models have been widely applied to a diverse spectrum of tasks, ranging from unconditional image generation [18,34,53,54], inpainting [27,33] and text-to-image generation [3,45,48] to image segmentation [2,4] and medical imaging [59]. Nevertheless, to the best of our knowledge, we are the first to employ diffusion models to generate proxy data for model stealing attacks.

In the subsequent phase of our pipeline, the labels for a subset of the synthetic images are obtained. As in any practical setting, the size of the subset is restricted by the number of permissible API calls (imposed either by a financial budget or for security reasons). Furthermore, we consider both soft (probability distributions over classes) and hard (discrete classes) labels being returned by the teacher model. Then, with our active learning strategy, we aim to select the more relevant samples to be passed through the teacher. This is achieved by adopting a clustering-based scheme: the latent representations (of the student) are obtained for all data points and clustered. Then, the sampling probability of a generated image is computed according to the distance to its corresponding cluster centroid (one centroid for each object class). This process is repeated for multiple iterations. With the collected (sample, label) pairs, the student model is trained in a typical supervised setup. Finally, with the introduction of our self-paced learning method, we exploit the unlabeled samples (left after reaching the query limit) as well. A pseudo-label, determined by a nearest neighbor algorithm, is assigned to each remaining sample. The student model is further trained on the joint data set containing data samples with labels from the teacher, as well as pseudo-labeled examples. To the best of our knowledge, we are the first to study model stealing under the constraint of limited API queries, a setting we refer to as *few-call model stealing*.

We conduct experiments on two image data sets (CIFAR-10 [24] and Food-101 [5]), using multiple well-known convolutional-based architectures for the teacher and student models. Furthermore, we evaluate the performance with various query (API call) budgets, attesting the efficiency of our method. Finally, we confirm the applicability of our method in real scenarios, as we vary the type of output given by the black-box model without negatively affecting the performance. We compare our pipeline with two prominent methods [6,38]. The results of our experiments uphold the superiority of our method.

In summary, our contributions on replicating black-box classification models are:

– We leverage diffusion models to create synthetic proxy data sets consisting of relevant samples for the original data set and carry out attacks on the black-box model.
– We propose a novel strategy on how to actively choose the samples for which to collect labels from the attacked model, obtaining improved results in the few-call model stealing scenario.

– We introduce a novel strategy that assigns pseudo-labels to the remaining samples and uses them to further boost the performance of the student via self-paced learning.

2 Related Work

With the recent trend of increasing the size of neural networks, knowledge distillation has become an important area in machine learning, aiming to mitigate the hardware requirements to run such models. Knowledge distillation [14,17,46,57] represents a technique that transfers the knowledge of a large, complex model (named teacher) to a smaller, simpler model (named student). After this process, the student model should have a similar performance with the teacher, in some situations even surpassing it. Initially, the method was applied to classification tasks [8,30,43,65,66], where the student used the output of the teacher model as target labels. Currently, distillation is also employed for more complex models, such as diffusion models [19,28,49,55] and large language models [10,15,41,60], being a typical step of deploying such models in production.

A sub-branch of knowledge distillation is model stealing, or model extraction, which represents a malign process. While following the same technical principles, it applies some extra constraints on the level of access to the teacher model. The objective is to distill the knowledge of a confidential proprietary teacher model without permission, where the targeted model is usually being run as a service.

The model stealing domain is vast due to the different scenarios in which it is applied. Firstly, there are two categories in which the model stealing studies can be split: attacking and defensive. While the former category comprises methods describing how to copy black-box models [6,9,38,40,50], the latter presents defensive mechanisms against such attacks [20,22,26,62,64]. An important aspect when stealing a model is represented by the number of inferences (calls) to the teacher model. As a results, some works focus on reducing the number of calls to the teacher [7,42,52,56], while others assume no limit on this number, and rather aim for an increased performance [6,9,51]. Another significant element to consider is the data used during the stealing process. A relaxed scenario has been adopted by some methods [9,39,40], in which real data sets have been used. However, some works apply a tight constraint and assume no training data is available, and thus resort to generating artificial data and using it as proxy data sets [6,21,31,38,50,61]. The comprehensive survey paper on model stealing by Oliynyk *et al.* [36] goes into a greater detail, and thus, we encourage readers to study it for a holistic view of the available research.

Orekandy *et al.* [38] introduced Knockoff Nets, an approach for model stealing that focused on the balance between accuracy and efficiency (w.r.t. the number of teacher calls). Using a reinforcement learning strategy, they trained a model able to determine which samples from a large-scale proxy data set (they employed ImageNet [47] in their experiments) are more relevant, and thus, they carefully crafted the queries sent to the teacher model. These examples were selected and used for training the surrogate student model.

Bărbălău *et al.* [6] presented a complex framework for stealing black-box models, named Black-Box Ripper. While they applied all constraints in a typical model stealing scenario, they assumed an unlimited number of calls to the teacher model. Given that the original training data was not available, the first part of their framework was to generate synthetic samples with a GAN. These images were then optimized using an evolutionary algorithm. In the second part, the student model was trained using the best candidate samples from each iteration.

Different from the aforementioned related works, we do not require any additional data to obtain the proxy data samples. Moreover, we take a step further in regards to the number of permitted API calls, and not only try to minimize them, but rather have a fixed low number of queries. To the best of our knowledge, we are the first to propose a few-call model stealing framework that is applicable in all respects to a real model theft scenario.

3 Method

In this section, we will begin by presenting the studied task, and then continue by introducing our method for replicating black-box models, called active self-paced knowledge distillation (ASPKD). We will describe our novel components and how are they integrated in the proposed framework.

Problem statement. As stated in previous works [38], the model stealing task is derived from knowledge distillation, *i.e.* in both cases, the task is to infuse the functionality of a teacher model into a student model. However, the two methods are carried out with different goals: while knowledge distillation aims to produce a compressed model with comparable accuracy, model stealing is executed with the only harmful intention of extracting the teacher's capability. In the context of model stealing, the assumption of having no knowledge about the training data, the architecture and the weights of the teacher is natural. Another difference between the two is that the student architecture is not required to be less complex in model stealing. Nevertheless, in a similar manner, we refer to the black-box model as the teacher, and the copy model as the student.

Black-box models are usually available as MLaaS. In a real scenario, service providers, whose main purposes are commercial, do not disclose any information about the model. The training data, the architecture of the model, its weights, gradients or hyperparameters, and other related details are unknown to the MLaaS users. Furthermore, for each query, the providers only supply the output of the model, either as soft labels (class probabilities) or hard labels. We consider an even more strict scenario where the number of queries is limited due to the following consideration: the model stealing attack might get detected due to the high number of API calls. Moreover, even if the attack remains undetected, the costs might rise to unjustifiable levels after a certain number of API calls.

Formally, we can formulate the problem statement using the following objective:

$$\min_{\theta_S} \| T(X, \theta_T) - S(X' \cup X'', \theta_S) \|, \text{ subject to } |X'| \leq n, \tag{1}$$

where T is the black-box teacher model, S is the student model in which we distill the knowledge, θ_T and θ_S are their corresponding weights, X is the original data set, while X' and X'' represent the two parts for the synthetic (proxy) data set, namely the part labeled by T and the part with pseudo-labels. The aim is to optimize the student weights such that the difference between the outputs of the two models is negligible, subject to making at most n passes through the teacher T, *i.e.* n represents the number of API calls. Following previous work [1,6,38], instead of using models accessible via APIs, we train the teacher ourselves, prior to launching the attack. During the attack, we use the teacher in a black-box regime, thus preserving all the constraints mentioned above. We hereby attest that no information about the teacher is leaked while training the student.

Overview. Our framework comprises three stages, as illustrated in Figure 1. The first step in our pipeline is to generate the required synthetic data. While there is no upper bound on the quantity of images, the lower limit is represented by the budget cap. Although we demonstrate the capability of our method for as few as one sample per class, it is more effective to fully take advantage of the query budget in order to achieve the highest performance. After the generation process, only a part of the data is given to the teacher. In order to select which images to collect the labels for, we employ an active learning technique. The resulting chosen samples, together with their associated labels predicted by the teacher model, are used to train the student. In the third stage, we employ a self-paced learning scheme to compute pseudo-labels for the samples that were left out by active learning. Once these are obtained, the student model is trained for a final round with all the data. We next describe the individual stages.

Data generation. The first challenge in addressing black-box model stealing is acquiring suitable training data. To this end, an effective approach is to create synthetic data that is similar to the original training data. While previous studies [6,50] utilized GANs [13] to generate proxy samples, we adopt diffusion models for this purpose. More specifically, we opt for text-conditional diffusion models, which can generate images of object classes specified via a prompt. This ensures that the framework can be applied on diverse image classification tasks. To thoroughly validate our method, we carry out experiments with two different open-source diffusion models: GLIDE [33] and Stable Diffusion [45]. Stable Diffusion is based on a latent diffusion model, where the diffusion process occurs in the latent space of an auto-encoder. Furthermore, the U-Net model in Stable Diffusion, which is in charge of gradually estimating the noise to be removed, integrates a cross-attention mechanism to condition the image synthesis on text. In contrast, GLIDE is a diffusion model that supports two guidance methods, alternating between a classifier-free method and a CLIP-based method [44]. In our approach, we prefer the former option. We use the publicly released GLIDE model, which is trained on a rigorously filtered data set. Regarding the design of prompts, we employ two distinct templates for each class, randomly varying between the two: “An image of a *{class}*” and “A photo of a *{class}*”. Here, the placeholder *{class}* is replaced with the actual class name, *e.g.* *dog*, *car*, etc. We generate nearly half of the images of each class using the first prompt and the other half using the second prompt. This prompt variation is meant to enhance the variability in the generative process, resulting in more diverse images.

Active learning. The limit on the number of API calls results in only a part of the synthetic data being annotated by the black-box model. As a result, achieving a high accuracy heavily depends on which samples are selected, as they should not only be representative, but also diverse, to enhance the student model's generalization ability. To this extent, we propose an active learning strategy to perform an informative selection of the more relevant samples to be inferred by the teacher.

At any iteration of the active learning procedure, we compute the latent vectors of all samples from the proxy subset X'', given by the student. These samples are grouped into clusters representing the classes predicted by the student. The centroid of each cluster is then computed. Afterwards, we implement a sampling strategy that advocates for selecting the examples that are closer to the centroids. The rationale behind our strategy is to minimize the selection of outliers, as these have a greater chance to be incorrectly labeled by the teacher. In order to compute the sampling probability of each image, we apply a Radial Basis Function to the distance of the corresponding latent vector to its nearest centroid, as follows:

$$p_i = \exp\left(-\frac{\mathit{\Delta}\left(\bar{S}(x_i''), \mu_c\right)}{2 \cdot \sigma^2}\right), \tag{2}$$

where μ_c is the closest centroid to $\bar{S}(x_i'')$, and σ is a hyperparameter that controls the importance of the proximity to the nearest centroid. We select a uniformly distributed number of samples from each cluster to ensure the diversity of samples. The selected samples are passed as input to the teacher, which returns either a soft or hard label that is stored in Y'. We use the subset labeled by the teacher, denoted as X', to train the student until convergence.

Self-paced learning. As previously stated, we can only obtain the class labels from the black-box model for a part of our proxy data set. As a results, depending on the limit of API calls, a substantial subset X'' of data samples remains unlabeled by the teacher model. In order to utilize the remaining data as well to improve the student, we propose a self-paced knowledge distillation procedure. This involves gradually assigning labels (which we call pseudo-labels) by applying a nearest neighbor algorithm within the latent embedding space learned by the student. More specifically, we operate in the latent space of the layer just before the flattening operation or the global average pooling layer, depending on the backbone architecture of our student. Using $\bar{S}$ to denote the latent space encoder, the first step in our self-paced learning method is to pass each example from the annotated proxy subset X' through the student model, storing the latent vectors and the corresponding labels assigned by the student. Then, for each unlabeled image $x_i'' \in X''$, we extract its corresponding latent representation and search for the closest k samples from the labeled training set. We employ two different metrics, namely the Euclidean distance and the cosine distance, to compute the distance between two samples $x_i'' \in X''$ and $x_j' \in X'$ within the latent space. Afterwards, the psuedo-label assigned to each sample x_i'' is determined by the labels of its k nearest neighbors. Depending on the type of output provided by the teacher, we propose two different label assignment strategies. If the teacher returns soft labels, we compute the class distribution for x_i'' as a weighted average of the soft labels, where the weight of a sample x_j' is inversely proportional to its distance from x_i''. When the black-box model returns hard labels, we use a majority voting scheme, with the distances between x_i'' and its neighbors serving as tiebreakers. Following label assignment, the student model is trained on the entire proxy data $X' \cup X''$.

The active self-paced learning procedure is executed for a number of $r = n/s$ steps, continuing until the API limit n is reached. It is important to note that the student's latent space is continuously evolving. Thus, the latent vectors are recomputed at every step of the active learning procedure to ensure that the sample selection procedure aligns with the current state of the student. In our experiments, we set $r = 3$, except for the one-shot and two-shot experiments, where r is constrained to $r = 1$ and $r = 2$, respectively.

4 Experiments

4.1 Experimental Setup

Data sets. We conduct experiments on two image data sets, namely CIFAR-10 [24] and Food-101 [5]. CIFAR-10 is a data set of 50,000 training images and 10,000 test images, representing objects of 10 categories. Each image has a resolution of 32×32 pixels. Food-101 [5] is a data set containing images of 101 food categories. Each image has a resolution of 224×224 pixels. The original split contains 75,750 training images and 25,250 test images. These data set choices are aimed at testing the model stealing frameworks in distinct settings, comprising both low-resolution and high-resolution images, as well as a small and a large number of classes. The training sets are only used to train the black-box teachers. In contrast, the copy models are trained on generated proxy data.

Diffusion models. We generate the proxy data set for CIFAR-10 with GLIDE [33]. The generated images are resized to match the input size of 32×32 pixels, as required by the black-box teacher. We generate 5,000 images per class. For Food-101, we generate a proxy data set with 1000 images per class, using Stable Diffusion v2 [45]. Although we can easily generate many more proxy images, we choose to limit the number of generated images in each proxy data set to the number of samples available in the original CIFAR-10 and Food-101 data sets. For each proxy data set, we keep 15% of the generated images for validation purposes.

Teacher and student models. In our experiments, we employ well-known model architectures that are fundamental to research, as this facilitates comparison with other baselines. For the black-box models, we use two architectures: AlexNet [25] and ResNet-50 [16]. Following previous research on model stealing [1,6], we consider lighter student architectures. For the AlexNet teacher, the student is Half-AlexNet, an architecture where the number of convolutional filters and the number of neurons in fully-connected layers are reduced by 50%. For the ResNet-50 teacher, the corresponding student is ResNet-18 [16].

Baselines. We compare our approach with two state-of-the-art model stealing methods [6,38]. The first competitor is Black-Box Ripper [6], a framework that employs a generative model in order to create proxy data, which is rather focused on achieving a high accuracy, irrespective of the number of API calls. Knockoff Nets [38] represent our second baseline. Aside from their relevance in the model stealing research, Knockoff Nets have a similar focus to our own, namely to optimize the number of teacher (or victim) passes. Following Orekondy *et al.* [38], we use CIFAR-100 as proxy data for Knockoff Nets, when the evaluation is performed on CIFAR-10. Similarly, we use ImageNet-200 [47] (a subset of 200 classes from ImageNet) as proxy data for Food-101. For a fair comparison, we impose the same limit on the number of API calls for all methods. Moreover, we use the same teacher and student architectures for all frameworks. Therefore, the reported accuracy rates reflect the performance levels of the training frameworks, namely Black-Box Ripper [6], Knockoff Nets [38], and ASPKD (ours).

Table 1. Optimal hyperparameters of the student models for both data sets.

Student	Diffusion model	Learning rate	Step size	γ
Half-AlexNet	GLIDE	$9 \cdot 10^{-4}$	20	0.95
ResNet-18	GLIDE	$9 \cdot 10^{-4}$	20	0.95
Half-AlexNet	Stable Diffusion	$7 \cdot 10^{-5}$	10	0.95
ResNet-18	Stable Diffusion	10^{-4}	30	0.95

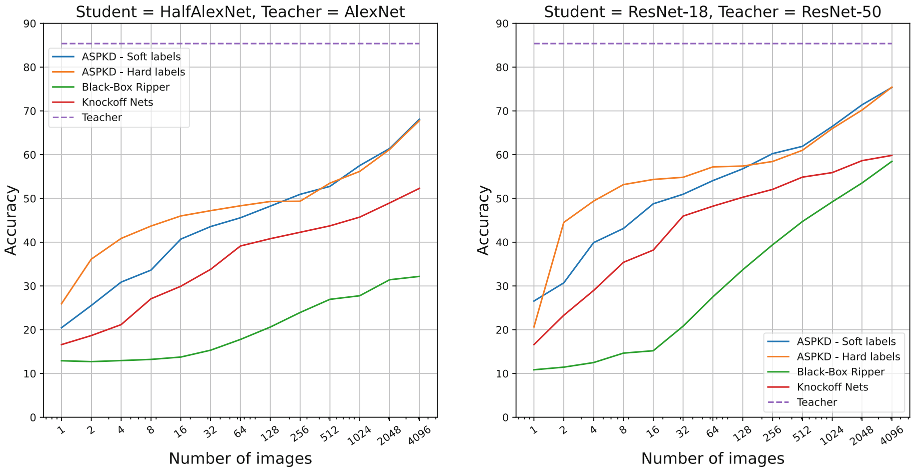


Fig. 2. Empirical results for the experiments conducted on the CIFAR-10 data set, where we vary the maximum number of black-box calls within the set $\{1, 2, 4, ..., 4096\}$. The left plot showcases the performance of the student model based on the Half-AlexNet architecture, while the right plot features results using a ResNet-18 student model. We compare the performance of ASPKD using both soft and hard labels against two state-of-the-art frameworks: Black-Box Ripper [6] and Knockoff Nets [38]. For context, the accuracy of the corresponding teacher model is also indicated in each plot. Each experiment reports the average accuracy over five independent runs with each student model. Best viewed in color.

Hyperparameters. Throughout the experiments, we employ the Adam optimizer [23] with a decaying learning rate scheduler. The hyperparameters for the teachers are tuned independently of the students, thus preserving the black-box nature of the teachers. In the case of CIFAR-10, the teachers are trained for 100 epochs with early stopping and a

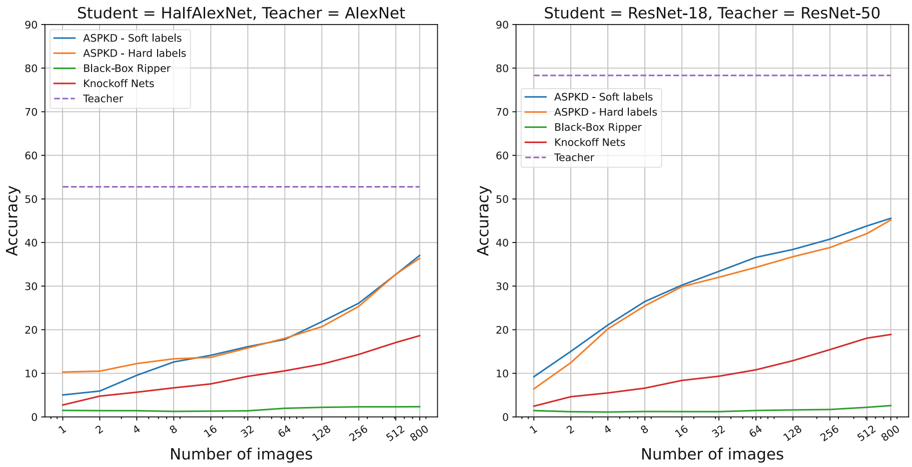


Fig. 3. Empirical results for the experiments conducted on the Food-101 data set, where we vary the maximum number of black-box calls within the set $\{1, 2, 4, ..., 512, 800\}$. The left plot showcases the performance of the student model based on the Half-AlexNet architecture, while the right plot features results using a ResNet-18 student model. We compare the performance of ASPKD using both soft and hard labels against two state-of-the-art frameworks: Black-Box Ripper [6] and Knockoff Nets [38]. For context, the accuracy of the corresponding teacher model is also indicated in each plot. Each experiment reports the average accuracy over five independent runs with each student model. Best viewed in color.

learning rate of $5 \cdot 10^{-4}$ on mini-batches of 64 samples, while the scheduler has a step size of 5 with $\gamma = 0.95$. For the experiments on Food-101, the teachers are trained for 100 epochs with a learning rate of 10^{-3} and a mini-batch size of 64. For the learning rate scheduler, the step size is 20 with $\gamma = 0.95$.

The student models are validated on 15% of the proxy data. The students are trained for 100 epochs with early stopping on mini-batches of 64 samples. As far as the active learning strategy is concerned, we set the value of σ in Eq. (2) to 17. The nearest neighbors algorithm in the self-paced learning method uses $k = 5$ neighbors. The optimal values for the other hyperparameters of the students on all data sets are reported in Table 1. We present results for two variations of our framework: one that learns from hard teacher labels, and one that learns from soft teacher labels. In the former version of ASPKD, the nearest neighbors during self-paced learning phase are identified using the Euclidean distance. On the other hand, in the latter version, we employ cosine distance to determine.

Evaluation. All models are evaluated using the official test splits of CIFAR-10 and Food-101. For the teacher models, we report the classification accuracy based on the ground-truth labels. Since the objective of the student models is to reproduce the predictions of the teachers, we assess each student by computing the classification accuracy against the labels predicted by the corresponding teacher. To ensure the robustness of our results, we report the average performance across 5 separate runs for each experiment.

4.2 Results

We have two evaluation scenarios for each data set, since there are two teacher-student pairs. In total, we compare our framework (ASPKD) with Black-Box Ripper [6] and Knockoff Nets [38] in four scenarios. The maximum number of API calls per class takes values in the set $\{1, 2, 4, ..., 4096\}$ for the two CIFAR-10 scenarios, and the set $\{1, 2, 4, ..., 512, 800\}$ for the two Food-101 scenarios. The corresponding results are presented in Figures 2 and 3. In both plots, the first column contains the teacher-student pair represented by AlexNet→Half-AlexNet. For the second column, the teacher-student pair is ResNet-50→ResNet-18. In each plot, we present results with two versions for ASPKD, corresponding to the type of labels returned by the teacher, soft or hard.

When compared to Black-Box Ripper [6], our framework (ASPKD) obtains significantly better results in all evaluation scenarios, regardless of the maximum number of API calls. In all evaluation scenarios, both ASPKD versions outperform Knockoff Nets [38] by considerable margins. In the CIFAR-10 experiments, ASPKD based on hard labels yields better results, especially in the more challenging few-call settings, namely when the maximum number of API calls per class is below 16, as illustrated in Figure 2. As the number of API calls per class increases, the two ASPKD versions register faster performance gains than the baselines. Overall, we conclude that ASPKD leads to gen-

Table 2. Accuracy rates of the Half-AlexNet student with different training procedures. The vanilla procedure consists of a conventional training strategy on proxy data with teacher labels. Next, we present the impact of separately and jointly introducing our active learning and self-paced learning (based on cosine distance and soft labels) components, respectively. All experiments are based on 5 trials, the mean accuracy and the standard deviation being reported in each case. The data set used for the evaluation is the original CIFAR-10 test split, while the proxy training data is generated by GLIDE [33].

#Samples per class	Vanilla	+ Active learning	+ Self-paced learning	+ Active & self-paced learning
1	19.4±2.1	20.7±3.6	25.0±2.7	25.9±4.0
2	25.9±3.7	26.5±2.0	25.6±1.1	36.1±2.5
4	27.7±2.6	29.9±1.2	28.5±0.5	40.9±2.7
8	29.4±2.0	33.0±2.0	35.3±1.7	43.7±2.7
16	36.3±3.5	41.8±3.0	40.8±1.1	46.0±1.6
32	39.2±2.1	43.9±1.3	46.4±1.7	47.2±1.8
64	43.6±1.4	46.0±2.7	45.8±1.2	48.3±1.7
128	46.6±1.8	48.8±1.5	48.0±1.6	49.3±1.6
256	47.7±3.2	50.3±1.9	50.0±1.5	49.4±3.2
512	52.5±2.1	53.6±1.9	53.6±0.8	53.5±1,6
1024	55.5±2.5	56.0±1.5	56.6±2.2	56.2±1.4
2048	60.3±1.7	61.5±2.1	63.1±0.9	61.2±1.3
4096	66.3±1.8	68.0±0.7	67.7±1.3	67.8±1.7

erally better results, surpassing both Black-Box Ripper [6] and Knockoff Nets [38] in all evaluation scenarios.

Ablation study. Table 2 illustrates the performance impact of introducing the active learning and self-paced learning strategies, both independently and jointly. When each strategy is applied separately, we observe significant improvements in most cases. Notably, when the number of API calls per class is 128 or fewer, combining both strategies leads to even greater performance gains. These results demonstrate the benefits of each strategy individually, as well as the effectiveness when used together.

5 Conclusion

In this study, we aimed to introduce a model stealing method that addresses the practical challenges encountered in real-world scenarios, where several constraints are present, *i.e.* no access to the training set, no information about the architecture of the victim model, or about its training process. The main objective was to respect a strict limitation on the number of allowed teacher model calls, even for as low as a single sample from each class. Our first contribution was to employ a text-to-image diffusion model to create synthetic training data, enabling us to produce entities from every output class, as well as having diverse samples to encompass all the class properties. We presented an active learning strategy designed to optimize the selection of proxy data to be labeled by the teacher model, ensuring that the most informative samples are prioritized. Additionally, given the imposed limit on the number of API calls, we introduced a self-paced learning method for annotating the generated images that never get passed through the black-box teacher model by assigning them pseudo-labels. We carried out extensive experiments focusing on a reduced the number of API calls, reporting results on various testing scenarios: multiple combinations of teacher-student architectures, black-box model returning varying output types, and distinct data sets obtained with different diffusion models. With out research, we aim to raise awareness of the model-stealing vulnerabilities of the publicly exposed methods, especially nowadays, given the current rise of artificial intelligence solutions.

In future work, we aim to address defensive methods as well. We will look into preventing few-call model stealing attacks through robust protection mechanisms in the early stages of the AI model deployment.

References

1. Addepalli, S., Nayak, G.K., Chakraborty, A., Radhakrishnan, V.B.: DeGAN: Data-Enriching GAN for Retrieving Representative Samples from a Trained Classifier. In: AAAI. vol. 34, pp. 3130–3137 (2020)
2. Amit, T., Nachmani, E., Shaharbany, T., Wolf, L.: SegDiff: Image Segmentation with Diffusion Probabilistic Models. arXiv preprint arXiv:2112.00390 (2021)
3. Avrahami, O., Lischinski, D., Fried, O.: Blended diffusion for text-driven editing of natural images. In: CVPR. pp. 18208–18218 (2022)
4. Baranchuk, D., Rubachev, I., Voynov, A., Khrulkov, V., Babenko, A.: Label-Efficient Semantic Segmentation with Diffusion Models. In: ICLR (2022)

5. Bossard, L., Guillaumin, M., Van Gool, L.: Food-101 – Mining Discriminative Components with Random Forests. In: ECCV. pp. 446–461 (2014)
6. Bărbălău, A., Cosma, A., Ionescu, R.T., Popescu, M.: Black-Box Ripper: Copying black-box models using generative evolutionary algorithms. In: NeurIPS. vol. 33, pp. 20120–20129 (2020)
7. Chandrasekaran, V., Chaudhuri, K., Giacomelli, I., Jha, S., et al.: Exploring connections between active learning and model extraction. In: USENIX. pp. 1309–1326 (2020)
8. Cho, J.H., Hariharan, B.: On the Efficacy of Knowledge Distillation. In: ICCV. pp. 4794–4802 (2019)
9. Correia-Silva, J.R., Berriel, R.F., Badue, C., de Souza, A.F., et al.: Copycat CNN: Stealing Knowledge by Persuading Confession with Random Non-Labeled Data. In: IJCNN. pp. 1–8 (2018)
10. Costa-jussà, M.R., Cross, J., Çelebi, O., Elbayad, M., Heafield, K., Heffernan, K., Kalbassi, E., Lam, J., Licht, D., Maillard, J., et al.: No Language Left Behind: Scaling Human-Centered Machine Translation. arXiv preprint arXiv:2207.04672 (2022)
11. 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)
12. Dhariwal, P., Nichol, A.: Diffusion models beat GANs on image synthesis. In: NeurIPS. vol. 34, pp. 8780–8794 (2021)
13. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., et al.: Generative adversarial nets. In: NeurIPS. **27**, 2672–2680 (2014)
14. Gou, J., Yu, B., Maybank, S.J., Tao, D.: Knowledge distillation: A survey. Int. J. Comput. Vision **129**(6), 1789–1819 (2021)
15. Gu, Y., Dong, L., Wei, F., Huang, M.: MiniLLM: Knowledge Distillation of Large Language Models. In: ICLR (2024)
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016)
17. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
18. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: NeurIPS. **33**, 6840–6851 (2020)
19. Huang, T., Zhang, Y., Zheng, M., You, S., Wang, F., Qian, C., Xu, C.: Knowledge Diffusion for Distillation. In: NeurIPS. vol. 36 (2024)
20. Juuti, M., Szyller, S., Marchal, S., Asokan, N.: PRADA: protecting against DNN model stealing attacks. In: EuroS&P. pp. 512–527 (2019)
21. Kariyappa, S., Prakash, A., Qureshi, M.K.: MAZE: Data-Free Model Stealing Attack Using Zeroth-Order Gradient Estimation. In: CVPR. pp. 13814–13823 (2021)
22. Kesarwani, M., Mukhoty, B., Arya, V., Mehta, S.: Model Extraction Warning in MLaaS Paradigm. In: ACSAC. pp. 371–380 (2018)
23. Kingma, D.P., Ba, J.L.: Adam: A method for stochastic gradient descent. In: ICLR (2015)
24. Krizhevsky, A.: Learning multiple layers of features from tiny images. University of Toronto, Tech. rep. (2009)
25. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet Classification with Deep Convolutional Neural Networks. In: NeurIPS. vol. 25 (2012)
26. Liu, X., Ma, Z., Liu, Y., Qin, Z., et al.: SeInspect: Defending Model Stealing via Heterogeneous Semantic Inspection. In: ESORICS. pp. 610–630 (2022)
27. Lugmayr, A., Danelljan, M., Romero, A., Yu, F., et al.: RePaint: Inpainting using Denoising Diffusion Probabilistic Models. In: CVPR. pp. 11461–11471 (2022)
28. Meng, C., Rombach, R., Gao, R., Kingma, D., Ermon, S., Ho, J., Salimans, T.: On Distillation of Guided Diffusion Models. In: CVPR. pp. 14297–14306 (2023)

29. Micaelli, P., Storkey, A.J.: Zero-shot knowledge transfer via adversarial belief matching. In: NeurIPS. vol. 32, pp. 9551–9561 (2019)
30. Mirzadeh, S.I., Farajtabar, M., Li, A., Levine, N., Matsukawa, A., Ghasemzadeh, H.: Improved Knowledge Distillation via Teacher Assistant. In: AAAI. vol. 34, pp. 5191–5198 (2020)
31. Mosafi, I., David, E.O., Netanyahu, N.S.: Stealing knowledge from protected deep neural networks using composite unlabeled data. In: IJCNN. pp. 1–8 (2019)
32. Nayak, G.K., Mopuri, K.R., Shaj, V., Radhakrishnan, V.B., et al.: Zero-shot knowledge distillation in deep networks. In: ICML. pp. 4743–4751 (2019)
33. Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., et al.: GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models. In: ICML. pp. 16784–16804 (2021)
34. Nichol, A.Q., Dhariwal, P.: Improved denoising diffusion probabilistic models. In: ICML. pp. 8162–8171 (2021)
35. Oh, S.J., Schiele, B., Fritz, M.: Towards reverse-engineering black-box neural networks. Explainable AI: Interpreting, Explaining and Visualizing Deep Learning pp. 121–144 (2019)
36. Oliynyk, D., Mayer, R., Rauber, A.: I know what you trained last summer: A survey on stealing machine learning models and defences. ACM Computing Surveys **55**(14s) (2023)
37. OpenAI: ChatGPT: A Conversational Language Model. https://openai.com/research/chatgpt (2022)
38. Orekondy, T., Schiele, B., Fritz, M.: Knockoff Nets: Stealing functionality of black-box models. In: CVPR. pp. 4954–4963 (2019)
39. Pal, S., Gupta, Y., Shukla, A., Kanade, A., et al.: A framework for the extraction of deep neural networks by leveraging public data. arXiv preprint arXiv:1905.09165 (2019)
40. Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., et al.: Practical black-box attacks against machine learning. In: ASIACCS. pp. 506–519 (2017)
41. Peng, B., Li, C., He, P., Galley, M., Gao, J.: Instruction Tuning with GPT-4. arXiv preprint arXiv:2304.03277 (2023)
42. Pengcheng, L., Yi, J., Zhang, L.: Query-efficient black-box attack by active learning. In: ICDM. pp. 1200–1205 (2018)
43. Phuong, M., Lampert, C.: Towards Understanding Knowledge Distillation. In: ICML. pp. 5142–5151 (2019)
44. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., et al.: Learning transferable visual models from natural language supervision. In: ICML. pp. 8748–8763 (2021)
45. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., et al.: High-Resolution Image Synthesis with Latent Diffusion Models. In: CVPR. pp. 10684–10695 (2022)
46. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: FitNets: Hints for Thin Deep Nets. In: ICLR (2014)
47. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. Int. J. Comput. Vision **115**(3), 211–252 (2015). https://doi.org/10.1007/s11263-015-0816-y
48. Saharia, C., Chan, W., Saxena, S., Li, L., et al.: Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. In: NeurIPS. vol. 35, pp. 36479–36494 (2022)
49. Salimans, T., Ho, J.: Progressive Distillation for Fast Sampling of Diffusion Models. In: ICLR (2022)
50. Sanyal, S., Addepalli, S., Babu, R.V.: Towards data-free model stealing in a hard label setting. In: CVPR. pp. 15284–15293 (2022)
51. Shi, Y., Sagduyu, Y., Grushin, A.: How to steal a machine learning classifier with deep learning. In: HST. pp. 1–5 (2017)

52. Shi, Y., Sagduyu, Y.E., Davaslioglu, K., Li, J.H.: Active deep learning attacks under strict rate limitations for online API calls. In: HST. pp. 1–6 (2018)
53. Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. In: ICML. pp. 2256–2265 (2015)
54. Song, J., Meng, C., Ermon, S.: Denoising Diffusion Implicit Models. In: ICLR (2021)
55. Song, Y., Dhariwal, P., Chen, M., Sutskever, I.: Consistency models. In: ICML. pp. 32211–32252 (2023)
56. Tramèr, F., Zhang, F., Juels, A., Reiter, M.K., et al.: Stealing machine learning models via prediction APIs. In: USENIX. pp. 601–618 (2016)
57. Urban, G., Geras, K.J., Kahou, S.E., Aslan, O., Wang, S., Mohamed, A., Philipose, M., Richardson, M., Caruana, R.: Do Deep Convolutional Nets Really Need to be Deep and Convolutional? In: ICLR (2022)
58. Wang, B., Gong, N.Z.: Stealing hyperparameters in machine learning. In: SP. pp. 36–52 (2018)
59. Wolleb, J., Bieder, F., Sandkühler, R., Cattin, P.C.: Diffusion Models for Medical Anomaly Detection. In: MICCAI. pp. 35–45 (2022)
60. Wu, M., Waheed, A., Zhang, C., Abdul-Mageed, M., Aji, A.F.: LaMini-LM: A diverse herd of distilled models from large-scale instructions. In: ACL. pp. 944–964 (2024)
61. Xie, Y., Huang, M., Zhang, X., Dong, C., et al.: Game: Generative-based adaptive model extraction attack. In: ESORICS. pp. 570–588 (2022)
62. Yan, H., Li, X., Li, H., Li, J., et al.: Monitoring-based differential privacy mechanism against query flooding-based model extraction attack. IEEE Trans. Dependable Secure Comput. **19**(4), 2680–2694 (2022)
63. Yin, H., Molchanov, P., Alvarez, J.M., Li, Z., et al.: Dreaming to Distill: Data-free Knowledge Transfer via DeepInversion. In: CVPR. pp. 8715–8724 (2020)
64. Zhang, Z., Chen, Y., Wagner, D.: SEAT: Similarity Encoder by Adversarial Training for Detecting Model Extraction Attack Queries. In: AISec. pp. 37–48 (2021)
65. Zhao, B., Cui, Q., Song, R., Qiu, Y., Liang, J.: Decoupled Knowledge Distillation. In: CVPR. pp. 11953–11962 (2022)
66. Zhu, X., Gong, S., et al.: Knowledge Distillation by On-the-Fly Native Ensemble. In: NeurIPS. vol. 31 (2018)

A Novel Ensemble Aggregation Method Based on Deep Learning Representation

Truong Thanh Nguyen[1](✉), Eyad Elyan[1], Truong Dang[2], Tien Thanh Nguyen[2], and Martin Longmuir[3]

[1] School of Computing, Robert Gordon University, Aberdeen, UK
t.nguyen3@rgu.ac.uk

[2] National Subsea Centre, Robert Gordon University, Aberdeen, UK

[3] AquaTerra Group Ltd., Aberdeen, UK

Abstract. We propose a novel ensemble aggregation method by using a deep learning-based representation approach. Specifically, we applied the Cross-Validation procedure on training data with a number of learning algorithms to obtain the predictions for training data called meta-data. A neural network model is trained on this meta-data to generate representations associated with class labels. In our method, the neural network model functions as an encoder, learning the relationship between base classifiers' outputs and mapping meta-data to a representation space. The vectors in the mapped space provide a more accurate representation than traditional methods by reducing the distance of vectors in the same class and increasing the distance in different classes. Our method was compared with four well-known ensemble methods: Decision Template, an ensemble with a MultiLayer Perceptron (MLP)-based combiner, gcForest, and XgBoost. Experiments conducted on 20 UCI datasets demonstrate the outstanding performance of our ensemble aggregation method. The results show that our method achieves better delegation of class label representations, enhancing the final results of classification tasks.

Keywords: Ensemble learning · Ensemble combining · Ensemble aggregation · Multilayer perceptron · Ensemble method

1 Introduction

In recent years, the advancement of Machine Learning (ML) has significantly contributed to solving problems (e.g. classification) in many areas. The performance success of ML models can be attributed to their ability to approximate complex unknown relationships between training observations and their associated labels via various approaches. Despite these strengths and effectiveness, each model expresses specific advantages or faces inherent performance limits when learning on a dataset. The ensemble technique is a potential approach to obtain a better and more robust performance by combining multiple advantages

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 31–46, 2025.
https://doi.org/10.1007/978-3-031-78383-8_3

of individual models. The foundation ideas of ensemble learning were formed in the 1980s in [1] which built the groundwork for ensemble learning. Nowadays, ensemble learning has become an attractive topic, resulting in various ensemble methods being proposed.

Two main stages are considered when designing an ensemble: ensemble generation and ensemble aggregation. Ensemble generation is the process of generating multiple models to form an ensemble. An ensemble can be generated by either training a ML algorithm on different training data schemes or training multiple different ML algorithms on a training dataset [2]. Both approaches introduce ideas to breaking through the bounding performance of individual learning models. Ensemble aggregation meanwhile is a critical process to combine the outputs of multiple models in the ensemble. In the past, there were several ensemble aggregation methods proposed such as Decision Template [3], Fixed Rule [4], Fuzzy Rule-based Combining [5], and Multi-Linear Regression [6].

On the other hand, recent years have seen the meteoric rise of deep learning, with successful applications in multiple areas. The reasons for the successes of deep learning are due to layer-by-layer processing, in-model feature transformation, and sufficient model complexity [7]. In this paper, we seek to develop a novel ensemble aggregation method by employing these deep learning advances. By using a deep learning model as an encoder to learn each feature and classifier's contribution to the final result, and combining this with Lifted structured loss functions [8], we enhance the accuracy of representations for class labels. The function maps original vectors to other spaces, with the expectation that the distributions of the training dataset in these spaces will meet the conditions where the similarity distances between vectors of the same class achieve minimum values, and the distances between vectors of different classes are maximized. By increasing class distances, the mean presentation vectors of each class become more distinct from those of other classes, thereby reducing errors in edge vectors.

Our contributions are as follows: (i) We introduce a novel ensemble aggregation method by learning representations associated with class labels. (ii) The representations are obtained by training a neural network (e.g. Multilayer Perceptron (MLP)) with Lifted structured loss function on the predictions of classifiers for training data. (iii) The experimental results show that our proposed method is better than 4 well-known benchmark algorithms on 20 experimental datasets.

The paper includes 5 sections. Section 2 introduces ensemble learning and related background. Sections 3 and 4 introduce our proposed ensemble aggregation method and experimental studies. Finally, the conclusion is provided in Sect. 5.

2 Literature Review

2.1 Ensemble Learning

Ensemble learning refers to the process of combining the outputs of multiple models (i.e. classifiers) to obtain better results than using single models. Ensemble learning can be divided into the following types: Homogeneous and heterogeneous. Random Forest [9] and XgBoost [10] are two well-known examples of successful homogeneous methods, constructing an ensemble on different data schemes generated from the training data to help reduce variance and enhance model stability and generalizability. Dudzik et al. [11] proposed an ensemble of cascades of Support Vector Machine (SVM), trained from different subsets of the training set. The selection of training subsets, as well as the SVM hyperparameters, were optimized by using an evolutionary algorithm. In [12], the authors proposed an AdaBoost-based ensemble of one-class support vector machines (OCSVMs) with a bounded exponential loss function to mitigate outlier performance. Dang et al. [13] constructed a homogeneous ensemble by using random projection in which multiple data schemes are generated by projecting training data into a subspace with different projection matrices.

On the other hand, heterogeneous ensembles enhance the diversity of models' predictions by employing a variety of different base models. Dang et al. [14] proposed a heterogeneous ensemble of medical image segmentation models in which a classifier will be added to the ensemble based on the confidence threshold computed from its predictions. Hossein et al. [15] investigated the use of bagging and boosting-based heterogeneous ensembles for the imbalanced class problem. Their experiments on 66 datasets showed that simple, bagging- and boosting-based heterogeneous ensembles achieve much better results than several homogeneous counterparts. In [16], the authors proposed a heterogeneous ensemble to handle changes in the data stream. The proposed method dynamically selects an appropriate subset of base classifiers to predict data under the stream settings by using classifier accuracy and confidence score.

In recent years, deep neural networks have achieved great success in multiple areas. One of the main reasons for the achievements of deep neural networks is due to the layer-by-layer processing nature, in which the first layers extract concrete features while more abstract features are extracted in the deeper layers [7]. Based on this observation, there has been many works on multi-layer ensembles in recent years. One of the earliest works on multi-layer ensembles is gcForest [7], in which the authors used decision trees and completely random trees as base classifiers in each layer. The number of layers of gcForest is determined based on the performance of the method on a validation set in which if the accuracy does not improve the the layer construction procedure stops and the final ensemble is returned. Nguyen et al. proposed MULES [2], which used an evolutionary algorithm to select the optimal subset of classifiers and features to learn at each layer. The fitness measure used by the evolutionary algorithm is composed of not only classification accuracy but also ensemble diversity as well. Dang et al. [17] proposed a two-layer ensemble of deep learning models for medical image segmentation. The segmentation output for each pixel by each classifier in the first layer is used as augmented data to improve the segmentation results of the second layer. A weight-based scheme is used to combine the predictions in the second layer. In [18], the authors noted that the input features of gcForest do not allow for better performance in deeper layers. Based on this observation, the authors proposed DEFEG, which is a multi-layer ensemble in which the predictions of each classifier at each layer is weighted together either by summation or concatenation. Variable-length PSO is used to find the optimal weights for the ensemble.

2.2 Ensemble Aggregation

Ensemble aggregation is the process of combining the predictions of the base models to obtain the final predictions. It is considered to be crucial to any effective ensemble system. Ensemble aggregation techniques are often classified as fixed methods and trainable methods. Fixed methods do not exploit the label of training data for final predictive labels, instead, they employ the classifiers' results on each test sample to calculate final decisions. Kittler et al. [4] investigated six fixed combining rules: Product, Sum, Median, Minimum, Maximum, and Majority Vote. The results indicated that the Sum rule provides better performance compared to the other combining rules. In [19], the authors investigated several voting methods for the bagging algorithm and found that single transferable vote can provide better results compared to plurality voting, despite some computational overhead. In [20], Weighted Majority Voting Ensemble methods use additional weights to determine the effect of each classifier on the final result, based on the assumption each classifier has different contributions based on their results.

In the trainable approach, the label information associated with the classifier's outputs on training data (also called meta-data of the training set) is exploited to construct a meta-classifier that combines the predictions of the base classifiers. There are two types of trainable combining, namely weight-based and representation-based combining. Weight-based combining aims to set weights for classifiers in the ensemble to reflect their different contributions to the final combining result. Dang et al. [21] proposed a weight-based ensemble of deep learning models for medical image segmentation. The predictions on each pixel by each model are weighted to get the combined results, and Comprehensive Learning Particle Swarm Optimisation is used to find the weights for each base model. On the other hand, in representation-based combining methods, different representations associated with class labels computed from the meta-data are generated. Kuncheva et al [3] propose a Decision Template method that utilizes the mean of the predictions for each class label to construct the representation vectors. Dang et al. [22] used a weight-based approach to find the optimal decision template for an ensemble of deep neural networks for medical image segmentation. The decision templates for each class is created using a weight-based approach, and Particle Swarm Optimisation is used to optimize the weights of each segmentation model. Nguyen et al. [23] proposed a Bayesian-based representation combining method in which the multivariate Gaussian distribution of the predictions of each classifier for each class is estimated via variational inference.

3 Proposed Method

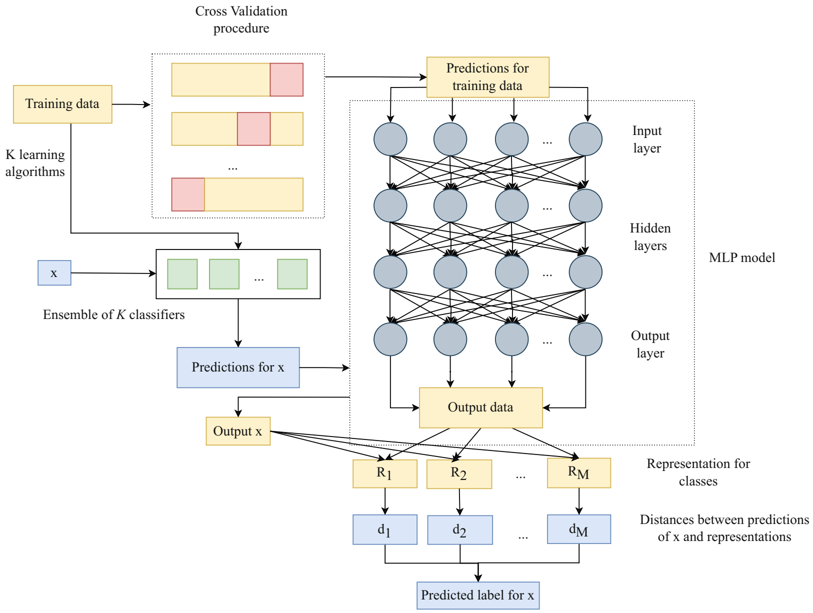


Fig. 1. Overall architecture of our proposed method

Assume we have N training observations, denote as $\mathcal{D}$, where each observation is a pair feature vector x_n and true label $\hat{y_n}$. First, we describe the process to obtain predictions for training data (meta-data) using a cross-validation procedure as in Fig. 1. We apply T-fold cross-validation procedure in which the training data $\mathcal{D}$ is divided into T disjoint equally-sized folds $\{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_T\}$. The model is trained on $T-1$ folds and validated on the remaining fold. This process repeats T times with each fold serving as the validation set once. Finally, the probabilities of each validation are collected in the form of meta-data used for later steps. The meta-data $\mathbf{L}$ is an $N \times MK$ matrix, where M denotes the number of classes and K denotes the number of classifiers.

$$\mathbf{L} = \begin{bmatrix} P_1(y_1|x_1) & \cdots & P_1(y_M|x_1) & \cdots & P_K(y_1|x_1) & \cdots & P_K(y_M|x_1) \\ P_1(y_1|x_2) & \cdots & P_1(y_M|x_2) & \cdots & P_K(y_1|x_2) & \cdots & P_K(y_M|x_2) \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ P_1(y_1|x_N) & \cdots & P_1(y_M|x_N) & \cdots & P_K(y_1|x_N) & \cdots & P_K(y_M|x_N) \end{bmatrix} \tag{1}$$

Next, we discuss how to create a representation vector for each class which can be used to classify test samples. Representation-based combining methods usually classify test instances based on the distance of their meta-data to that of the representation vectors for each class. The representation vectors can be chosen in various ways, such as the mean of the predictions in [3], or by optimization as in [22]. Here we propose to train a neural network that will map the meta-data of each class into a vector in the representation space, such that the vectors in the same class will be close to each other and vectors in different classes will be far from each other. In other words, the new vectors in the representation space will be:

$$\mathbf{R} = NeuralNetwork(L) = \begin{bmatrix} r_{1,1} & \cdots & r_{1,S} \\ r_{2,1} & \cdots & r_{2,S} \\ \cdots & \cdots & \cdots \\ r_{N,1} & \cdots & r_{N,S} \end{bmatrix} \tag{2}$$

where S is the dimension of the representation space. In 2, the i^{th} row of $\mathbf{R}$: $\left[r_{i,1} \cdots r_{i,S}\right]$ represents for the predictions for observation x_i in $\mathbf{L}$ $\left[P_1(y_1|x_i) \cdots P_1(y_M|x_i) \cdots P_K(y_1|x_i) \cdots P_K(y_M|x_i)\right]$. After that, the representation vector for each class will be calculated as follows:

$$R_m = \left[\sum_{n=1}^{N} \frac{\mathbb{I}(y_n=m)*r_{n,1}}{\mathbb{I}(y_n=m)}, \sum_{n=1}^{N} \frac{\mathbb{I}(y_n=m)*r_{n,2}}{\mathbb{I}(y_n=m)}, \cdots \sum_{n=1}^{N} \frac{\mathbb{I}(y_n=m)*r_{n,S}}{\mathbb{I}(y_n=m)}\right] \tag{3}$$

in which $\mathbb{I}$ is the indicator function. The Formula 3 calculates the representation vector for each class label by averaging values $r_{i,j}$ in each row of $\mathbf{R}$ that is associated with this label.

Next, we discuss the choice of the loss function used to train the neural network on $\mathbf{L}$. Several loss functions are often utilized in deep learning to enhance the contrast between vectors of different classes, such as contrastive and triplet embedding [24,25]. Both methods find embedding vectors using mini-batches in which the number of pairs chosen for calculation is n from n input vectors in the batch. The lifted structured loss function uses all $O(N^2)$ pairs, instead of $O(N)$ to enhance performance in fixed mini-batches of size 8. Figure 2 shows the process mining for a pair positive in the batch, there are 6 observations in the batch and x_3, x_4 in the same class and all others belong to different classes. The red edges are present for pairs in the same class, and the blue edges are present for pairs in different classes. Each node in positive pairs is independently compared against all negative edges, which allows the model to learn from the negatives from both the left and right of a pair.

Lifted structured loss functions optimize the model's output by increasing the similarity of vectors within the same class and enhancing the dissimilarity between vectors of different classes. This approach aims to improve the representation vectors and robustness of class label predictions. Let $D_{ij} = \|f(\mathbf{x}_i) - f(\mathbf{x}_j)\|_2$, formula of loss function [8] is defined as:

$$\mathcal{L}_{\text{struct}}^{(ij)} = D_{ij} + \mathcal{L}_{\text{struct}}^{(ij)} = D_{ij} + \log\left(\sum_{(i,k)\in\mathcal{N}} \exp(\alpha - D_{ik}) + \sum_{(j,l)\in\mathcal{N}} \exp(\alpha - D_{jl})\right) \tag{4}$$

in which $\mathcal{P}$ is the set of positive pairs and $\mathcal{N}$ is the set of negative pairs, the label $y_{i,j} \in \{0, 1\}$, value 0 determine the same class and contrast. The formula 4 has

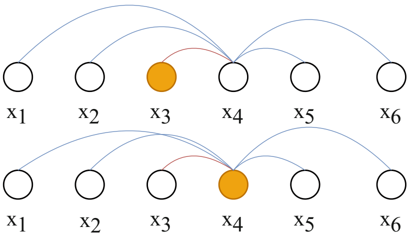


Fig. 2. An illustration of lifted structured loss.

2 main parts: D_{ij} is used to calculate the loss of positive pairs in mini-batches and to optimize the distance between output vectors that are positive pairs, and $\log\left(\sum_{(i,k)\in\mathcal{N}} \exp(\alpha - D_{ik}) + \sum_{(j,l)\in\mathcal{N}} \exp(\alpha - D_{jl})\right)$ is utilized for optimizing for negative pairs. After loss of a positive pair is calculated, the final loss is combined by 5,

$$\mathcal{L}_{\text{struct}} = \frac{1}{2|\mathcal{P}|} \sum_{(i,j)\in\mathcal{P}} \max(0, \mathcal{L}^{(ij)}_{\text{struct}})^2 \tag{5}$$

It can be seen that lifted structure loss is more flexible and effective than the triplet structure method in [26] where only the predefined determined points are calculated. Therefore, in this paper, the lifted structure loss is used as the loss function when training the neural network on **L** to obtain **R**.

Given a test instance x, the algorithm starts by using K learning classifiers to create the probability vector for this sample.

$$\mathbf{L}(x) = \left[P_1(y_1|x) \cdots P_1(y_M|x) \cdots P_K(y_1|x) \cdots P_K(y_M|x)\right] \tag{6}$$

Then, the neural network will map the probability vector into the representation space

$$\mathbf{R}(x) = NeuralNetwork(\mathbf{L}(x)) = \left[r^x_{1,1} \cdots r^x_{1,S}\right] \tag{7}$$

The predicted class for x will then be chosen as follows:

$$y_{pred} = \text{argmin}_{m=1,\ldots,M} ||R_m - R(x)|| = \text{argmin}_{m=1,\ldots,M}\, d_m \tag{8}$$

where $d_m = ||R_m - R(x)||$ is the distance between the representation of the test instance x and the representation of class m, and $||.||$ is a distance function. In this paper, Euclidean distance is chosen as the distance function.

Algorithm 1 describes the training process. The algorithm receives as inputs the training set $\mathcal{D}$, the learning algorithms $\mathcal{K} = \{K_i\}_{i=1,\ldots,M}$, the size of the representation vector S and the number of epochs N_{epoch}. From line 1 to line 3, the learning algorithms are trained on the training set to create the base classifiers $\{BC_i\}_{j=1,\ldots,K}$.

Algorithm 1 Training process

Input: $\mathcal{D}$: training set, $\mathcal{K} = \{K_i\}_{i=1,\ldots,M}$: learning algorithms, S: size of representation vector, N_{epoch}: maximum number of epochs

Output: Ensemble of classifiers: $\{BC_i\}_{j=1,\ldots,K}$, Representation: $\{R_i\}_{i=1,\ldots,M}$, and Neural Network model

```
 1: for $i \leftarrow 1$ to $K$ do
 2:     Base classifier $BC_i$ = Learn($K_i, \mathcal{D}$)
 3: end for
 4: $\mathbf{L} = \varnothing$, $\{\mathcal{D}_1, \ldots, \mathcal{D}_T\}$ = $T$-Partition($\mathcal{D}$)
 5: for each $\mathcal{D}_i$ do
 6:     $\tilde{\mathcal{D}}_i = \mathcal{D} - \mathcal{D}_i$
 7:     for each $\mathcal{K}_j$ do
 8:         Classifier $BC_j^{\sim i}$ = Learn($\mathcal{K}_j$, $\tilde{\mathcal{D}}_i$)
 9:         $\mathbf{L} = \mathbf{L} \cup$ Classify($BC_j^{\sim i}$, $\mathcal{D}_i$)
10:     end for
11: end for
12: Neural Network Model= Initialize parameters()
13: for epoch $\leftarrow 1$ to $N_{epoch}$ do
14:     for each mini-batch (X_batch, Y_batch) in ($\mathbf{L}$, $\mathbf{Y}$) do
15:         Q_pred = Predict(X_batch)
16:         loss = Lifted structured loss (Q_pred, Y_batch)
17:         Perform backward propagation and update model parameters
18:         if loss doesn't decrease in 10 epochs then
19:             Early stop
20:         end if
21:     end for
22: end for
23: $\mathbf{Q} = \varnothing$
24: for each mini-batch (X_batch, Y_batch) in ($\mathbf{L}$, $\mathbf{Y}$) do
25:     Q_pred = Predict(X_batch)
26:     $\mathbf{Q} = \mathbf{Q} \cup$ (Q_pred, Y_batch)
27: end for
28: Initialize $\mathbf{V_i} = \varnothing$, where $i = 1, \ldots, M$
29: for V, y_label in $\mathbf{Q}$ do
30:     $\mathbf{V_{y\_label}} = \mathbf{V_{y\_label}}$ + V
31: end for
32: $\mathbf{R} = \varnothing$
33: for $m \leftarrow 1$ to $M$ do
34:     $R_m$ = mean($\mathbf{V_m}$)
35:     $\mathbf{R} = \mathbf{R} \cup R_m$
36: end for
37: return $\{BC_i\}$ $i = 1, \ldots, M$, Neural Network Model, $\mathbf{R}$
```

In lines 4-11, cross-validation is performed to create the meta-data **L**. From lines 13-22, the neural network is trained on the meta-data using the lifted structure loss for a maximum of N_{epoch} epochs, and during the training process, if the loss doesn't decrease after a number of epochs, then the training of the neural network is stopped. In lines 23-27, the trained neural network is used to map the meta-data to the representation space. In lines 29-36, the representation vector for each class is calculated using Eq. 3. Finally, the base classifiers, the trained neural network model, and the representation vectors are returned in line 37.

Algorithm 2 Testing process

Input: x: unlabeled sample, $\mathbf{R}$: representation vector set, Neural Network model, $\{BC_i\}$ $i = 1, \ldots, M$: ensemble of classifiers
Output: Predicted class label for x

```
1: for $i \leftarrow 1$ to $K$ do
2:     $L(x) = \text{Classify}(BC_i, x)$
3: end for
4: $R(x) = \text{Neural Network model } (L(x))$
5: for $m \leftarrow 1$ to $M$ do
6:     $d_m = d(R_m, R(x))$
7: end for
8: return $x \in y_t$ if $t = \arg\min_{m=1,\ldots,M} d_m(x)$
```

Algorithm 2 describes the testing process. The inputs of the algorithm are the unlabeled observation x, the representation vectors for each class, the neural network models, and the base classifiers. In lines 1-3, the base classifiers predict x to create the meta-data. In line 4, the neural network is used to create the representation vector of x using its meta-data. In lines 5-8, the distance between the representation vector of x and the representation vector of each class is calculated. Finally, in line 9, the class label whose representation vector has the smallest distance to that of x is returned as the production result for x.

4 Experiments

4.1 Experimental Settings

The experiments were conducted on 20 different datasets from the UCI Machine Learning Repository (http://archive.ics.uci.edu/ml/datasets.html). We detailed the description of these datasets in Table 1. Each dataset was split into training and testing with a train-to-test ratio of 7:3. Five classifiers were used to create meta-data which are Random Forest (using 200 estimators), Logistic Regression (Softmax regression), K nearest neighbors (the number of nearest neighbors was set to 5), XgBoost (using 200 estimators), and Naive Bayes classifiers (using Gaussian distribution). These classifiers were selected due to their widespread use and demonstrated effectiveness in previous studies, as referenced in [27]. We chose 5-fold cross-validation to train and create meta-data.

The fully connected MLP including 4 layers was used on the meta-data L to train $\mathbf{R}$. Each layer applies an affine transformation, followed by a non-linear activation function, enhancing the model's ability to capture patterns in the data. Purposes of the input layer typically increase the dimension of the input, enabling the extraction of higher-level features: $\mathbf{h}_1 = \sigma(\mathbf{W}_1\mathbf{x}+\mathbf{b}_1)$. Where $\mathbf{W}_1 \in \mathbb{R}^{S\times 64}$ is the weight matrix, $\mathbf{b}_1 \in \mathbb{R}^{64}$ is the bias vector, 64 is input size of the first hidden layer. σ is a non-linear activation function ReLU ($\sigma(z) = \max(0, z)$), $\mathbf{h}_1 \in \mathbb{R}^{64}$ is the output of the first layer. Hidden layers have size (64, 64) which are used to finetune these features, and facilitate the learning of complex relationships within the data: $\mathbf{h}_2 = \sigma(\mathbf{W}_2\mathbf{h}_1 + \mathbf{b}_2)$, $\mathbf{h}_3 = \sigma(\mathbf{W}_3\mathbf{h}_2 + \mathbf{b}_3)$ where $\mathbf{W}_2, \mathbf{W}_3 \in \mathbb{R}^{64\times 64}$

are the weight matrices, $\mathbf{b}_2, \mathbf{b}_3 \in \mathbb{R}^{64}$ are the bias vectors, and $\mathbf{h}_2, \mathbf{h}_3 \in \mathbb{R}^{64}$ are the outputs of the first and second hidden layers, respectively. The output layer reduces the dimensionality to match the number of output targets, providing the necessary structure for making representation vectors: $\mathbf{y} = \mathbf{W}_4\mathbf{h}_3 + \mathbf{b}_4$. Where $\mathbf{W}_4 \in \mathbb{R}^{c\times 64}$ is the weight matrix, $\mathbf{b}_4 \in \mathbb{R}^c$ is the bias vector, $\mathbf{y} \in \mathbb{R}^c$ is the output vector,where c is the number of output dimensions. In the study, c was chosen in the range {2, 4, 8, 16, 32, 64, 128, 256}. This architecture's depth enables it to capture relationships within the data and the effectiveness of each classifier, while the linear layers ensure the model's computational efficiency and facilitate gradient-based optimization.

Table 1. The description of experimental datasets

Dataset	# of features	# of observations	# of classes
Australian	14	690	2
Biodeg	41	1055	2
Breast-cancer	9	683	2
Bupa	6	345	2
Colon	2000	62	2
Electricity	8	45312	2
Embryonal	7129	60	2
Heart	13	270	2
Hill_valley	10	2424	2
Madelon	50	2000	2
Magic	10	19020	2
Mammographic	5	830	2
Musk2	16	6598	2
Newthyroid	5	215	3
Ring	20	7400	2
Satimage	36	6435	6
Sonar	60	208	2
Vertebral	6	310	3
Wdbc	30	569	2
Wine	13	178	3

4.2 Results and Discussions

Influence of the Dimension of Representation Vectors: In Fig. 3, we evaluate our method using representation vectors of varying dimensions: 2, 4, 8, 16, 32, 64, 128, and 256, to analyze the influences of the dimension of representation

Table 2. Accuracy of the benchmark algorithms and our proposed method

Name	Decision Template	MLP Combiner	gcForest	XgBoost	Proposed Method
Australian	**0.8841**	0.8599	0.8792	0.8744	**0.8841**
Biodeg	0.858	0.877	0.8707	0.8549	**0.8833**
Breast-cancer	0.9707	**0.9854**	0.9707	0.9561	0.9805
Bupa	0.7212	0.7308	0.7212	0.7019	**0.7404**
Colon	**0.8421**	0.7368	0.7368	0.7895	**0.8421**
Electricity	0.9063	0.9155	0.7998	0.8529	**0.9172**
Embryonal	0.4444	0.5556	0.5	0.5	**0.6111**
Heart	0.8272	0.7778	0.8272	0.7531	**0.8519**
Hill_valley	0.6484	0.6525	0.5824	0.6126	**0.6593**
Madelon	0.7533	0.785	0.635	0.7	**0.7967**
Magic	0.8819	0.8829	0.8375	0.8735	**0.8845**
Mammographic	0.8032	0.7952	**0.8394**	0.8233	0.8353
Musk2	0.9859	0.9894	0.951	0.9768	**0.9949**
Newthyroid	**0.9846**	0.9538	0.9692	0.9692	**0.9846**
Ring	0.9779	**0.9784**	0.9635	0.9707	**0.9784**
Satimage	0.9156	0.9151	0.8695	0.912	**0.9208**
Sonar	**0.8889**	0.8095	0.8254	0.8413	0.873
Vertebral	0.8172	0.828	0.7957	0.828	**0.8387**
Wdbc	**0.9766**	0.9532	0.9649	**0.9766**	**0.9766**
Wine	**1**	0.9815	**1**	0.9444	**1**

vectors on the method's performance. Overall, the performance of the proposed method on experimental datasets varies with the changes in the representation vector dimensions. Especially, this variation is particularly in datasets with large feature dimensions, such as Embryonal and Colon (see Table 1), which have 7129 and 2000 input features, respectively. The proposed method achieved the best result with a dimension of 256 on Wine and Newthyroid datasets. On a number of datasets, such as Musk2 and Ring, the accuracy and F1 score are stable when the representation dimension changes. On the other hand, the dimension of the representation vector has a strong effect on the performance on some datasets. For example, on the Wine dataset, when the dimension is 2, the accuracy is just around 0.89 but when the dimension is 4, the algorithm obtains an accuracy of 0.98. When the dimension increases again

Comparison to the Benchmark Algorithms: We conducted a comparative analysis using several benchmark algorithms: Decision Template, an ensemble with MLP-based combiner (called MLP combiner), gcForest, and XgBoost to validate our approach at 128-dimension. It is noted that the MLP models in

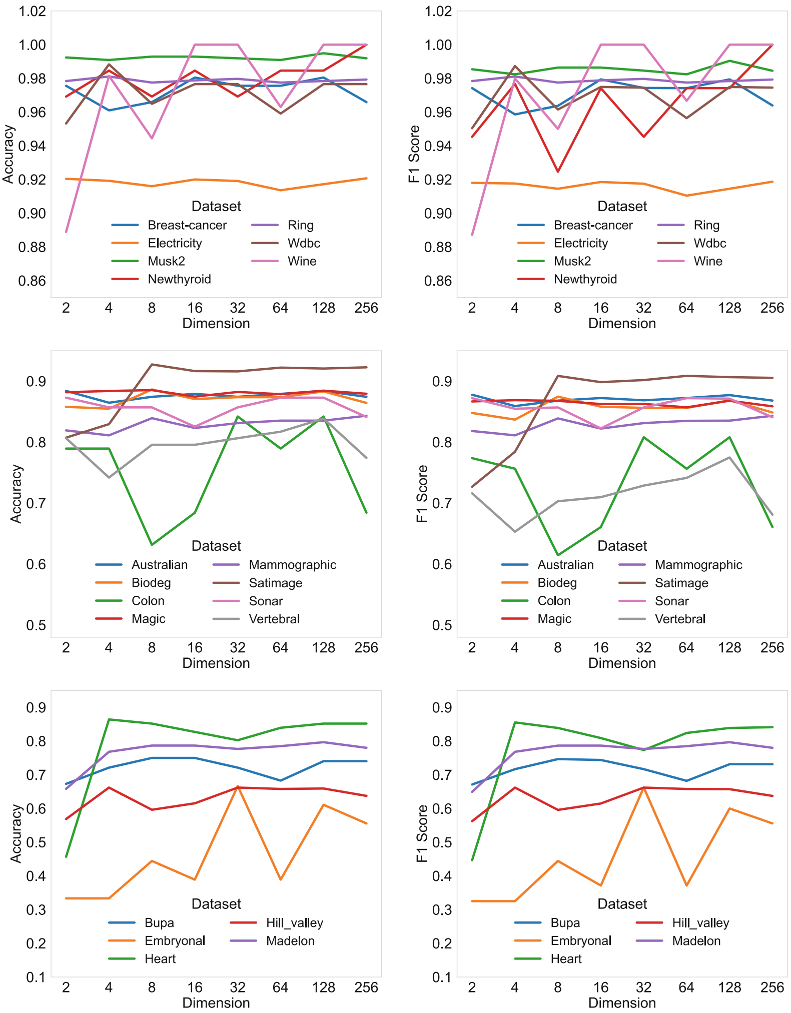


Fig. 3. Comparison of different dimensions of representation to 8, the accuracy is reduced to slightly above 0.94, but when the number of dimensions goes to 16 and 32, then the accuracy goes up to almost 1.0. This shows that an important research direction is to determine the right dimension size of the representation vector. In this paper, by visual inspection of the results, we propose to use the dimension size of 128 for the following sections.

our study function as encoders, optimizing representation vectors through the application of lifted structured loss as in Fig. 1. Meanwhile, in the ensemble with MLP-based combiner, MLP works directly on the meta-data as a combiner. Here we want to demonstrate the effectiveness of the proposed representation vectors through the MLP-based transformation in our study compared to exploiting the meta-data without any transformations. Table 2 illustrates the accuracy of

Table 3. F1 score of the benchmark algorithms and our proposed method

Name	Decision Template	MLP Combiner	gcForest	XgBoost	Proposed Method
Australian	**0.8772**	0.8519	0.8704	0.8669	**0.8772**
Biodeg	0.8473	0.8601	0.8604	0.8378	**0.8686**
Breast-cancer	0.969	**0.9846**	0.9691	0.9528	0.9794
Bupa	0.7152	0.7063	0.6899	0.6937	**0.7314**
Colon	**0.8081**	0.6801	0.636	0.7286	**0.8081**
Electricity	0.9038	0.9135	0.7961	0.8483	**0.9145**
Embryonal	0.4444	0.4462	0.4109	0.4582	**0.6**
Heart	0.809	0.75	0.8056	0.7271	**0.8389**
Hill_valley	0.6482	0.6524	0.5803	0.6122	**0.6573**
Madelon	0.7533	0.785	0.6248	0.6996	**0.7967**
Magic	0.867	**0.868**	0.8177	0.8551	**0.868**
Mammographic	0.8032	0.7941	**0.839**	0.8231	0.8353
Musk2	0.9723	0.9795	0.8968	0.9538	**0.9904**
Newthyroid	**0.9742**	0.9126	0.9453	0.9453	**0.9742**
Ring	0.9779	**0.9784**	0.9635	0.9707	**0.9784**
Satimage	0.9001	0.8945	0.8322	0.8932	**0.9067**
Sonar	**0.8879**	0.8083	0.8209	0.8393	0.8714
Vertebral	0.7253	0.7609	0.6809	0.7395	**0.7749**
Wdbc	0.9745	0.9501	0.962	**0.9749**	**0.9749**
Wine	**1**	0.9833	**1**	0.9466	**1**

the proposed method compared to benchmark algorithms. At first glance, our method achieves the highest scores among the benchmark algorithms across most datasets. We have the largest margin, approximately 7% higher than the second-best method, MLP combiner. XgBoost meanwhile shares the same highest score with our approach only in the Wdbc dataset, while our method outperforms all the other datasets. On the Heart and Madelon datasets, the differences peak at around 10%. For gcForest, there is a significant gap between the proposed method and gcForest, achieving over 5% in the results of 7 datasets, namely Colon, Electricity, Embryonal, Hill_valley, Madelon, Satimage, and Vertebral.

The highest gap is nearly 16% in the Madelon dataset, and the second-highest difference is in the Electricity dataset, in which the gcForest score is around 80% compared to 91.72% in our approach. The Decision Template method obtains the highest score on 6 datasets, 5 of which (Australian, Colon, Newthyroid, Wdbc, and Wine datasets) share the same performance with our results. On the Sonar dataset, the accuracy of the Decision Template is the highest and greater than our score by just 1.5%.

We analyzed the F1 score performance of our proposed method in comparison with several benchmark algorithms, as shown in Table 3. Once again, a similar pattern like accuracy was observed. The F1-score results demonstrate that our proposed method achieves the best performance when evaluated across most datasets. Compared to XgBoost, our method achieves a higher F1 score in all datasets except the Wdbc dataset where we share the same result with Xgboost.

On the Heart dataset, we achieved an F1 score greater than Xgboost around 11% and the number is 9% on Madelon. The proposed method also outperforms gcForest on 18 datasets and shares the same result on Wine dataset. Additionally, on Mammographic dataset, the top two highest scores are 83.9% and 83.5% respectively gcForest, our propose. MLP combiner gains 3 highest F1 scores in datasets namely Beast-cancer, Magic, and Ring, 2 of which share the same score as our method. Only on Breast-cancer, the performance of MLP combiner is highest and greater than our proposed just a round 0.5%. The Decision Template method achieves the highest score in 5 datasets, 4 of them Australian, Colon, Newthyroid, and Wine share the same performance as our proposed method. In the Sonar dataset, the accuracy of the Decision Template is the highest and greater than our score by just 0.52%.

5 Conclusion

In the paper, we introduced a novel combining method based on deep learning representation. Our approach utilizes the neural network model to generate representation vectors from meta-data collected from base learning models' outputs. The loss function lift structure combines with the model to optimize the distribution of output vectors, by decreasing the distance of vectors in the same class and increasing the distance of vectors in different classes. Then using the mean to create class representation vectors from output representation vectors of the neural network, new representation vectors are trained on the meta-data on delegation for classes. We used a 4-layer MLP training on the meta-data in the experiment to obtain the representation vectors. Experimental results on twenty UCI datasets demonstrated the benefit of our approach compared with four other well-known algorithms: Decision Template, an ensemble with MLP-based combiner, gcForest, and XgBoost. We believe that the MLP model in our architecture can be altered with other deep-learning models to exploit and optimize more features from meta-data. This will be our next immediate research focus. We believe that the MLP model in our architecture can be altered with other deep-learning models such as Long short-term memory (LSTM) or recurrent neural network (RNN) as multiple-layer combination architectures to exploit and optimize more features from meta-data. This will be our next immediate research focus.

References

1. Schapire, R.E.: The strength of weak learnability. In: 30th Annual Symposium on Foundations of Computer Science, pp. 28–33 (1989). https://doi.org/10.1109/SFCS.1989.63451
2. Nguyen, T.T., Pham, N.V., Dang, M.T., et al.: Multi-Layer Heterogeneous Ensemble with Classifier and Feature Selection. In: Proceedings of GECCO, pp. 725–733 (2020)
3. Kuncheva, L.I., Bezdek, J.C., Duin, R.P.W.: Decision templates for multiple classifier fusion: an experimental comparison. Pattern Recogn. **34**(2), 299–314 (2001)

4. Kittler, J., et al.: On combining classifiers. IEEE Trans. Pattern Anal. Mach. Intell. **20**(3), 226–239 (1998)
5. Nguyen, T.T., et al.: Heterogeneous Classifier Ensemble with Fuzzy Rule-based Meta Learner. Information Sciences **422**, (2017)
6. Ting, K.M., Witten, I.H.: Issues in stacked generalization. Journal of Artificial Intelligence Research **10**, 271–289 (1999)
7. Zhou, Z.-H., Feng, J.: Deep Forest: Towards an Alternative to Deep Neural Networks. In: Proceedings of IJCAI, pp. 3553–3559 (2017)
8. Song, H.O., et al.: Deep metric learning via lifted structured feature embedding. In: Proceedings of CVPR, pp. 4004–4012 (2016)
9. Fernández-Delgado, M., et al.: Do we need hundreds of classifiers to solve real world classification problems? JMLR, pp. 3133–3181 (2014)
10. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of SIGKDD, pp. 785–794 (2016)
11. Dudzik, W., Nalepa, J., Kawulok, M.: Ensembles of evolutionarily constructed support vector machine cascades. Knowl.-Based Syst. **288**, 111490 (2024)
12. Xing, H.-J., Liu, W.-T., Wang, X.-Z.: Bounded exponential loss function based AdaBoost ensemble of OCSVMs. Pattern Recogn. **148**, 110191 (2024)
13. Dang, M.T., et al.: An Ensemble System with Random Projection and Dynamic Ensemble Selection. In: Intelligent Information and Database Systems, pp. 576–586 (2018)
14. Dang, T., et al.: Ensemble Learning based on Classifier Prediction Confidence and Comprehensive Learning Particle Swarm Optimisation for Medical Image Segmentation. In: 2022 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 269–276 (2022)
15. Ghaderi Zefrehi, H., Altınçay, H.: Imbalance learning using heterogeneous ensembles. Expert Syst. Appl. **142**, 113005 (2020)
16. Luong, A.V., et al.: Heterogeneous ensemble selection for evolving data streams. Pattern Recognition **112** (2021)
17. Dang, T., et al.: Two-layer Ensemble of Deep Learning Models for Medical Image Segmentation. Cogn. Comput. **16**, 1–20 (2024)
18. Luong, A.V., et al.: DEFEG: Deep Ensemble with Weighted Feature Generation. Knowl.-Based Syst. **275**, 110691 (2023)
19. Leon, F., Floria, S.-A., Bădică, C.: Evaluating the effect of voting methods on ensemble-based classification. In: 2017 IEEE International Conference on INnovations in Intelligent SysTems and Applications (INISTA), pp. 1–6 (2017)
20. Dogan, A., Birant, D.: A weighted majority voting ensemble approach for classification. In: 2019 4th International Conference on Computer Science and Engineering (UBMK), IEEE, pp. 1–6 (2019)
21. Dang, T., et al.: Weighted Ensemble of Deep Learning Models based on Comprehensive Learning Particle Swarm Optimization for Medical Image Segmentation. In: Proceedings of IEEE CEC, pp. 744–751 (2021)
22. Dang, T., et al.: Ensemble of deep learning models with surrogate-based optimization for medical image segmentation. In: 2022 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8 (2022)
23. Nguyen, T.T., et al.: A novel combining classifier method based on Variational Inference. Pattern Recogn. **49**, 198–212 (2016)
24. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: Proceedings of CVPR, vol. 2, IEEE, pp. 1735–1742 (2006)

25. Weinberger, K.Q., Blitzer, J., Saul, L.: Distance metric learning for large margin nearest neighbor classification. In: Advances in Neural Information Processing Systems **18** (2005)
26. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: Proceedings of CVPR, pp. 815–823 (2015)
27. Han, K., Nguyen, T. T., Vu, V. A., Liew, A. W.-C., Dang, T., & Nguyen, T. T. (2024). VISTA: A variable length genetic algorithm and LSTM-based surrogate assisted ensemble selection algorithm in multiple layers ensemble system. In: 2024 IEEE Congress on Evolutionary Computation (CEC) (pp. 1–9). IEEE

GUARDEV: Guided Unified Adaptive Response for Defending Electric Vehicles

Koustav Kumar Mondal[1](✉) and Debasis Das[2]

[1] Inter-Disciplinary Research Division - IoT and Applications, Indian Institute of Technology Jodhpur, Jodhpur, Rajasthan, India
mondal.4@iitj.ac.in

[2] Department of Computer Science Engineering, Indian Institute of Technology Jodhpur, Jodhpur, Rajasthan, India
debasis@iitj.ac.in

Abstract. In the rapidly evolving landscape of electric vehicles (EVs), integrating the Internet of Vehicles (IoV) has opened up new dimensions in automotive technology. While enhancing vehicle performance, safety, and security, this advancement also exposes EVs to heightened cyber threats. To address this critical concern, we introduce GUARDEV: Guided Unified Adaptive Response for Defending Electric Vehicles, a novel Machine Learning (ML)-enhanced conflict resolution-based ensemble Intrusion Detection System (IDS) architecture. GUARDEV leverages the power of adaptive ML techniques to provide a unified and guided response in safeguarding EVs against sophisticated cyber-attacks. Our framework evaluates the effectiveness of three prominent ML models - XGBoost, Random Forest, and Extra Trees - in detecting specific categories of cyber threats. By harnessing the prediction confidence scores from these models, GUARDEV offers a nuanced and intelligent approach to recognizing and thwarting diverse cyber-attacks. Our research, grounded in rigorous experimentation using public IoV security datasets, demonstrates GUARDEV's exceptional accuracy of 99.47% in intrusion detection. This remarkable achievement not only underscores the potential of ML in enhancing vehicular cybersecurity but also establishes GUARDEV as a powerful tool for creating safer, more secure EVs. GUARDEV paves the way for robust protection of internet-connected electric vehicles in today's digital age by providing a guided, unified, and adaptive response to cyber threats.

Keywords: Intrusion Detection System · Electric Vehicle · Internet of Vehicle · Conflict Resolution · Ensemble Learning

1 Introduction

The fusion of the Internet of Things (IoT) with Electric Vehicles (EVs) has led to the emergence of the Internet of Vehicles (IoV) [22], revolutionizing the transportation sector and paving the way for a smarter, more integrated future on

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 47–62, 2025.
https://doi.org/10.1007/978-3-031-78383-8_4

wheels [5]. At the core of this transformation lies the Controller Area Network (CAN) bus within EVs, serving as a vital communication hub linking Electronic Control Units (ECUs) to coordinate various vehicular functions. Additionally, the burgeoning Vehicle-To-Everything (V2X) technology expands this communication network beyond the vehicle, connecting EVs with a diverse array of IoT devices, from roadside infrastructure to smart gadgets [22].

1.1 Motivation

In the rapidly evolving landscape of automotive technology, the convergence of IoT with EVs heralds a new era of connectivity, promising numerous advancements in functionality and convenience. However, this wave of innovation brings with it a critical challenge: the need for robust cybersecurity measures. The integration of IoT into EVs amplifies connectivity but also exposes vulnerabilities within the Internal Vehicular Network (IVN), particularly due to the inherent simplicity of CAN packets lacking encryption or authentication. The potential threats are diverse and significant, ranging from DDoS and Brute Force attacks to Spoofing, DoS, Recon, Web-based intrusions, and the notorious Mirai botnet. As EVs increasingly interface with external networks, the scope of vulnerability expands, necessitating an intelligent, adaptive, and unified approach to fortify cyber defenses and mitigate risks.

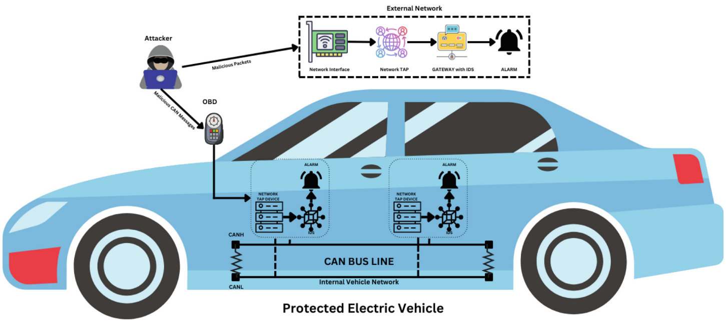


Fig. 1. State of The Art Intrusion Detection System for Electric Vehicle

To address these challenges, we introduce GUARDEV: Guided Unified Adaptive Response for Defending Electric Vehicles. As illustrated in Fig. 1, GUARDEV represents a significant advancement in automotive cybersecurity, offering a comprehensive framework for protecting EVs against cyber threats. This innovative system embodies a proactive approach, leveraging machine learning techniques to provide real-time monitoring, adaptive threat detection, and guided response capabilities. GUARDEV's unified architecture integrates various components of vehicle security, ensuring a cohesive defense strategy. Its adaptive nature allows it to evolve with emerging threats, continuously learning and improving its detection and response mechanisms. The guided aspect of GUARDEV ensures that responses to detected threats are optimized and contextually appropriate, minimizing false positives and enhancing overall system efficiency. By implementing GUARDEV, we aim to create a robust, intelligent defense system that not only protects against known cyber threats but also adapts to new, unforeseen challenges in the ever-evolving landscape of EV cybersecurity. This approach is crucial in ensuring the resilience and safety of IoT-integrated EVs, paving the way for a secure and connected automotive future.

1.2 Problem Statement

In the context of rapidly evolving cyber threats to Electric Vehicles (EVs), recent advancements in Machine Learning (ML) have sparked considerable interest in enhancing cybersecurity measures, particularly through Intrusion Detection Systems (IDSs). While ML techniques offer promising avenues for bolstering automotive cybersecurity by analyzing network traffic data, their effectiveness varies significantly across different types of cyber-attacks. This variability poses a critical challenge to ensuring the reliability and robustness of IDSs in real-world automotive environments. Moreover, issues such as data scarcity, model interpretability, and vulnerability to adversarial attacks complicate the deployment of ML-driven IDSs in automotive systems. Addressing these complexities is crucial for enhancing the resilience and effectiveness of cybersecurity measures in connected EVs, ultimately ensuring the safety and security of future automotive ecosystems. To tackle these challenges, we introduce GUARDEV: Guided Unified Adaptive Response for Defending Electric Vehicles. This novel framework harnesses the power of advanced ML algorithms, including Extreme Gradient Boosting (XGBoost) [17], Random Forest Classifier [9], and Extra Trees Classifier [18]. GUARDEV aims to achieve optimal performance with reduced execution time across various cyber-attack types by providing a guided, unified, and adaptive response to threats.

1.3 Contributions

This study contributes significantly to the field of EV cybersecurity in three key ways:

1. **Development of GUARDEV Architecture:** We introduce GUARDEV, a guided unified adaptive response framework for defending electric vehicles. This architecture employs a unique combination of balanced weight class and confidence decision techniques, specifically designed for efficient intrusion detection in EVs. By integrating multiple ML models into a unified framework, GUARDEV enhances the robustness and reliability of intrusion detection systems in automotive environments while providing guided and adaptive responses to threats.
2. **Rigorous Testing on Cutting-Edge IoT Datasets:** GUARDEV undergoes rigorous testing on the CICIoT2023 [16] Dataset, which represents a wide spectrum of both internal and external network scenarios. By evaluating the

performance of our framework on this dataset, we ensure its applicability and effectiveness across real-world cyber-attack scenarios encountered in automotive systems, demonstrating the adaptive nature of GUARDEV.
3. **Benchmarking Against State-of-the-Art Methodologies:** We benchmark GUARDEV's performance against other state-of-the-art methodologies, showcasing its superiority in EV cybersecurity. Through comparative analysis and comprehensive evaluation, we demonstrate the effectiveness and efficacy of our guided and unified approach in mitigating cyber threats and enhancing the security posture of electric vehicles.

2 Brief Literature Review

This section explores the current landscape of cybersecurity for Electric Vehicles (EVs) and the Internet of Vehicles (IoV), reviewing recent advancements and identifying gaps in Intrusion Detection Systems (IDSs). While various innovative approaches using machine learning and deep learning techniques have emerged, the challenge of developing a guided, unified, and adaptive response system for IoV-enabled EVs remains.

2.1 Deep Learning Approaches in IDS

Song et al. [20] and Agrawal et al. [1] focused on deep learning-based IDSs, with Song et al. developing a convolutional neural network model for car-hacking detection and Agrawal et al. proposing a novel system for various attack types. While effective, these approaches lack the unified and adaptive response mechanism that GUARDEV aims to provide. Awajan [7] presented a Deep Learning-based IDS for IoT devices, achieving high precision, recall, and F1-scores, but did not specifically address the guided response needs of IoV scenarios.

2.2 Ensemble and Hybrid Models

Elmasry et al. [13] and Chen et al. [11] introduced ensemble models for network intrusion detection, using combinations of various neural network architectures. While innovative, these studies did not focus on providing a unified, adaptive response for EV protection. Aldhyani et al. [3] and Chaganti et al. [8] explored deep learning models for securing various IoT contexts, offering insights that could be adapted for a more comprehensive EV defense system.

2.3 Decision Tree-Based and Other Novel Solutions

Yang et al. [23] developed a Decision Tree-based IDS for autonomous and networked vehicles, targeting both internal and external vehicle network intrusions. This approach, while valuable, lacks the guided and adaptive elements that GUARDEV proposes. Alamleh et al. [2] proposed a framework for standardizing ML-based IDSs in complex network settings, emphasizing the need for efficient evaluation - a principle that aligns with GUARDEV's unified approach. Verma et al. [21] investigated specific attack vectors, underscoring the importance of a comprehensive defense strategy like GUARDEV.

3 GUARDEV: Proposed Framework

Our research introduces GUARDEV (Guided Unified Adaptive Response for Defending Electric Vehicles), a novel Intrusion Detection System (IDS) framework designed to secure Electric Vehicles (EVs) against various network threats. GUARDEV employs an ensemble stacking approach based on conflict resolution, integrating machine learning capabilities on the Internet of Vehicles (IoV) server. This framework leverages three advanced ML algorithms: Extreme Gradient Boosting (XGBoost) [17], Random Forest Classifier [9], and Extra Trees Classifier [18]. These algorithms work in concert to analyze IoV-based EV traffic data, creating a balanced, accurate, and adaptive intrusion detection model. The system architecture of GUARDEV is depicted in Fig. 2.

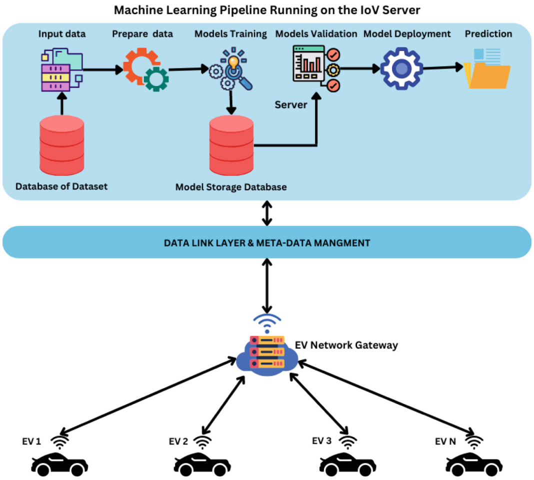


Fig. 2. GUARDEV System Architecture

GUARDEV's architecture is designed to provide a guided response to detected threats, unifying various security measures into a cohesive defence strategy. The adaptive nature of the system allows it to evolve with emerging threats, continuously improving its detection and response capabilities.

3.1 Dataset Description

To train and evaluate GUARDEV, we utilized the CICIoT2023 [16] dataset. This comprehensive dataset is crucial for developing machine learning models capable of effectively detecting and responding to cyberattacks in IoV environments. The CICIoT2023 [16] dataset was meticulously curated by its authors, who collected data from 33 distinct cyberattacks conducted on a network comprising 105 IoT devices. These attacks span seven categories, encompassing threats highly relevant to EV security: DDoS, DoS, Recon, Web-based, Brute Force, Spoofing, and Mirai. The dataset is publicly available, facilitating its use in training GUARDEV's machine learning algorithms to detect, classify, and guide responses to cyberattacks. By leveraging this rich dataset, GUARDEV can develop a nuanced understanding of various attack patterns, enabling it to provide guided, unified, and adaptive responses to protect EVs in real-world scenarios. The diversity of attack types represented in the dataset ensures that GUARDEV can offer comprehensive protection against a wide range of cyber threats in the IoV ecosystem.

3.2 Foundation ML Models for GUARDEV

GUARDEV incorporates three robust ML algorithms, each contributing to its guided, unified, and adaptive response capabilities:

- **Extreme Gradient Boosting (XGBoost):** XGBoost [17] forms a critical component of GUARDEV's adaptive response mechanism. It minimizes a regularized objective function:

$$\mathrm{Obj}(\Theta) = \sum_i L(y_i, \hat{y}i^{(t)}) + \sum k\Omega(f_k), \tag{1}$$

  where $\hat{y}_i^{(t)}$ is the prediction at the t-th iteration, and f_k represents the k-th tree [10].
  Loss function: XGBoost typically uses logistic loss for binary classification:

$$L(y_i, \hat{y}_i) = y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i) \tag{2}$$

- **Random Forest (RF):** RF [9] contributes to GUARDEV's unified approach by creating an ensemble of decision trees:

$$RF(x) = \frac{1}{K}\sum_{k=1}^{K} h(x, \Theta_k), \tag{3}$$

  where K is the number of trees and Θ_k are the random parameters for the k-th tree [9].
  Loss function: RF typically uses Gini impurity for classification:

$$G = \sum_{i=1}^{c} p_i(1 - p_i) \tag{4}$$

  where c is the number of classes and p_i is the probability of an item being classified to class i.

- **Extra Trees (ET):** ET enhances GUARDEV's guided response by introducing controlled randomness:

$$ET(x) = \frac{1}{K}\sum_{k=1}^{K} h(x, \Theta_k, \theta_{best}), \tag{5}$$

  where $\theta_{best} = \arg\max_\theta Q(D_v, \theta)$ [18].
  Loss function: ET also typically uses Gini impurity, similar to Random Forest.

These algorithms were chosen for their complementary strengths in analytics, automated feature selection, low computational cost, and support for parallelization and GPU execution. Their integration in GUARDEV creates a versatile and efficient stacked-based ensemble model capable of providing a guided, unified, and adaptive response to cyber threats in Electric Vehicles. XGBoost's adaptive nature allows GUARDEV to continuously refine its response strategies. Random Forest contributes to the unified approach by aggregating multiple decision trees, providing robust predictions across various attack scenarios. Extra Trees introduce an element of controlled randomness, enhancing GUARDEV's ability to guide responses in novel threat situations. The combination of these algorithms enables GUARDEV to:

- Guide responses based on the strengths of each algorithm.
- Unify diverse prediction strategies for comprehensive threat detection.
- Adapt to evolving cyber threats in the Internet-connected EV ecosystem

The proposed model algorithm for GUARDEV is outlined in Algorithm 1, which details how these base models are integrated to create a cohesive, adaptive defense system for Electric Vehicles.

3.3 GUARDEV: A Stacked Ensemble Model for Guided, Unified, and Adaptive EV Cyber Defense

GUARDEV employs a sophisticated stacked ensemble comprising XGBoost [17], Random Forest [9], and Extra Trees [18] to enhance accuracy in identifying and responding to network attacks. Our approach integrates these diverse machine learning algorithms, leveraging each model's strengths to capture intricate patterns in network data and provide a guided, unified, and adaptive response. GUARDEV's stacking methodology aggregates predictions from multiple base models via a meta-learner, enabling adaptability to dynamic threat landscapes. The ensemble model demonstrates superior resilience, accuracy, and adaptability, making GUARDEV well-suited for real-world deployment in safeguarding critical EV infrastructure. This is supported by a detailed algorithmic overview, elucidating steps from data preprocessing to final decision-making and response generation.

GUARDEV Algorithm Overview: The core of GUARDEV is encapsulated in Algorithm 1, which outlines the comprehensive steps for providing a guided, unified, and adaptive response to cyber threats in electric vehicles, underpinned by intricate mathematical operations.

Mathematical Formulations of GUARDEV: GUARDEV's algorithm is underpinned by rigorous mathematical formulations that enable its guided, unified, and adaptive response to cyber threats in electric vehicles. Here, we present these formulations along with their relevance to GUARDEV's key features. *Data Transformation and Normalization:* The preprocessing phase employs the SMOTE algorithm for class balancing, creating synthetic samples (D_{balanced}) through interpolation:

$$D_{\text{balanced}}(x) = x + \lambda \cdot (x_{\text{nn}} - x), \quad \lambda \sim \text{Uniform}(0,1), \tag{6}$$

where x_{nn} is a nearest neighbor of (x). This is followed by PCA for dimensionality reduction:

$$D_{\text{reduced}} = W^T \cdot D_{\text{std}}, \quad W = \arg\max \text{Var}(W^T \cdot D_{\text{std}}). \tag{7}$$

These transformations ensure that GUARDEV's base models work with balanced, normalized data, enhancing the system's adaptive capabilities.
Ensemble Learning and Model Evaluation: The ensemble learning process involves training each base learner on the transformed dataset. The accuracy (A_m) for each model is calculated using k-fold cross-validation. The ensemble's decision-making employs a weighted voting mechanism:

$$P_{\text{ensemble},x} = \arg\max_c \sum_{m \in M} w_m \cdot \mathbb{I}[P_{m,x} = c], \tag{8}$$

where w_m is the weight assigned to model m, and $\mathbb{I}[\cdot]$ is an indicator function. This unified approach ensures all models contribute to the defence strategy.
Guided Model Selection and Confidence Scoring: For each threat type (t), GUARDEV selects the most appropriate model:

$$M_t = \arg\max_{m \in M} A_m(t) \tag{9}$$

The confidence score for each prediction is calculated as:

$$C_{m,x} = \frac{\exp(f_m(x)c)}{\sum i \exp(f_m(x)_i)} \tag{10}$$

where $f_m(x)_c$ is the model's output for class (). This guided selection and confidence scoring enable targeted responses to specific threat types.
Adaptive Conflict Resolution and Decision Making: When model predictions differ, GUARDEV employs an adaptive conflict resolution algorithm. The final prediction and response are determined based on Bayesian updating of confidence scores:

$$P_{\text{final},x}, R_x = \text{GuideAdaptiveResponse}(P_{m,x}, C_{m,x} m \in M, M_t) \tag{11}$$

Algorithm 1 GUARDEV: Guided Unified Adaptive Response for Defending Electric Vehicles

```
Require: Dataset $D$ with EV network traffic data, Set of ML models $M =$ XGBoost, Random Forest, Extra Trees
Ensure: Guided, unified, and adaptive intrusion detection and response for each data sample
Data Preprocessing and Adaptation:
Apply SMOTE: $D_{\text{balanced}} \leftarrow \text{SMOTE}(D)$
Standardize features: $D_{\text{std}} \leftarrow \text{StandardScaler}(D_{\text{balanced}})$
Apply PCA: $D_{\text{reduced}} \leftarrow \text{PCA}(D_{\text{std}})$
Training Unified Base Learners:
for each model $m \in M$ do
    Train $m$ on $D_{\text{reduced}}$
    Perform k-fold cross-validation
    $A_m \leftarrow \text{ComputeAccuracy}(m, D_{\text{reduced}})$
Guided Model Selection for Each Threat Type:
for each threat type $t$ in $D$ do
    $M_t \leftarrow \arg\max_{m \in M} A_m(t)$
    if tie in $M_t$ then
        Choose model with minimum response time
Adaptive Prediction Phase:
for each sample $x$ in $D_{\text{test}}$ do
    for each model $m \in M$ do
        $P_{m,x} \leftarrow \text{Predict}(m, x)$
        $C_{m,x} \leftarrow \text{Confidence}(m, x)$
Unified Consensus and Guided Conflict Resolution:
for each sample $x$ in $D_{\text{test}}$ do
    if all $P_{m,x}$ are equal then
        Accept prediction $P_{m,x}$
    else
        $P_x, R_x \leftarrow \text{GuideAdaptiveResponse}(P_{m,x}, C_{m,x}, M_t)$
Final Guided Decision Making:
$D_{\text{responses}} \leftarrow \bigcup_{x \in D_{\text{test}}} (P_x, R_x)$
Adaptive Complexity Analysis:
$O(DFC)$, with $D$ as dataset volume, $F$ as number of features, and $C$ as complexity of adaptive models
Continuous Learning and Adaptation:
Update model weights based on performance: $W_m \leftarrow \text{UpdateWeights}(M, D_{\text{responses}})$
```

where

$$P_{\text{final},x} = \arg\max_{c} \prod_{m \in M} P(c|P_{m,x}, C_{m,x}), \tag{12}$$

and $P(c|P_{m,x}, C_{m,x})$ is the posterior probability of class c given the model's prediction and confidence score. The response R_x is generated based on this final prediction and the specific threat type.

Table 1. Performance Comparison of GUARDEV with State-of-the-Art Models Across Key Metrics

Model	Accuracy (%)	AUC (%)	Recall (%)	Prec. (%)	F1 (%)	Model Training Time (s)
LSTM [6]	90.50	88.12	86.99	91.00	88.00	916.65
Random Forest Classifier [19]	81.25	79.47	78.35	80.56	79.77	198.8800
XGBoost [10]	76.80	74.89	75.59	77.59	76.60	230.5565
Extra Trees [18]	68.30	67.48	69.75	70.86	69.87	216.3456
Decision Tree Classifier [15]	60.45	59.62	58.17	60.66	60.64	170.4275
K Neighbors Classifier [19]	52.70	54.88	53.26	52.65	52.77	94.8725
Deep Convolutional Neural Network [20]	92.50	90.93	91.84	91.84	92.91	894.79
Quadratic Discriminant Analysis [12]	88.85	73.14	44.54	79.45	83.83	6.6650
SVM [4]	50.35	48.45	47.20	52.34	49.11	355.9950
GUARDEV	**99.47**	**98.65**	**97.77**	**99.10**	**98.86**	**185.4773**

Continuous Learning and Adaptation GUARDEV's adaptive mechanism is reinforced through continuous learning:

$$W_m \leftarrow \text{UpdateWeights}(M, D_{\text{responses}}) \tag{13}$$

where the model weights are updated based on their performance on recent threats, ensuring GUARDEV evolves with the threat landscape.
Complexity Analysis: The overall complexity of GUARDEV is $O(DFC)$, where D is the dataset volume, F is the number of features, and C encapsulates the complexity of the underlying models. This analysis ensures GUARDEV's scalability in real-world EV scenarios.

These mathematical formulations provide a rigorous foundation for GUARDEV's behaviour and performance characteristics. By integrating ensemble learning with sophisticated mathematical techniques, GUARDEV offers a guided, unified, and adaptive solution for intrusion detection in IoV-based EV networks.

4 Results and Discussion: GUARDEV's Performance in Defending Electric Vehicles

This section presents a comprehensive analysis of the experimental results obtained from the evaluation of GUARDEV (Guided Unified Adaptive Response for Defending Electric Vehicles). The performance of our model is compared against several state-of-the-art models using the CICIoT2023 [16] dataset, underpinning its superiority in accurately detecting and responding to a wide array of network attacks targeting electric vehicles. Table 2 shows the Area Under the Curve (AUC) per Attack Category for GUARDEV, showcasing its robust performance across various attack categories relevant to EV security. Impressively, the overall AUC is recorded at 0.99, as depicted in Table 1. In the subsequent section, we provide a detailed discussion of the findings presented in Table 2 and Table 1, elucidating the efficacy and significance of GUARDEV in providing a guided, unified, and adaptive response to cyber threats in the EV ecosystem.

4.1 GUARDEV's Accuracy and Adaptive Robustness in EV Defense

As demonstrated in Table 1 and Table 2, GUARDEV exhibits exceptional accuracy and adaptive robustness in detecting and responding to various network attacks targeting electric vehicles, surpassing several other models including Random Forest [9], Extra Trees [18], and XGBoost [17]. In the detection and response to 33 overall attack types relevant to EV security, GUARDEV achieved a remarkable accuracy of 99.47%. This showcases a significant improvement over:

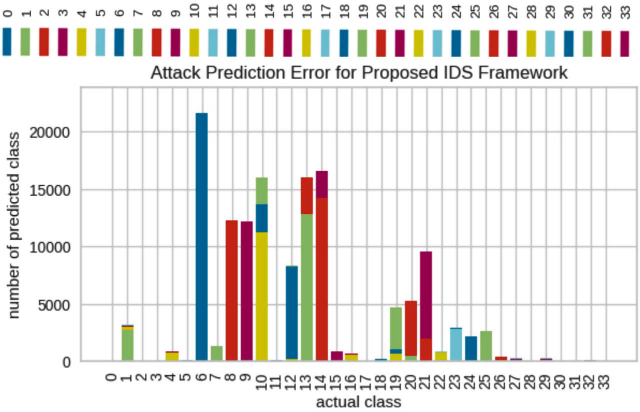


Fig. 3. Number of Attack Prediction Errors by the Proposed IDS Framework

- Random Forest [9]: 81.25% accuracy
- Extra Trees[18]: 68.30% accuracy
- XGBoost [17]: 76.80% accuracy

4.2 GUARDEV's Attack Prediction Error Analysis for Electric Vehicle Security

The performance of GUARDEV (Guided Unified Adaptive Response for Defending Electric Vehicles) was analyzed by examining the number of attack prediction errors across different attack classes relevant to EV security, as depicted by colors with corresponding numbers in Fig. 3. The graph illustrates the distribution of prediction errors made by GUARDEV. The x-axis represents the actual class of the attack, while the y-axis indicates the number of times GUARDEV incorrectly predicted the class of the attack. It is observed that GUARDEV made the highest number of errors on attacks belonging to classes 10, 11, and 12. This suggests that these particular attack classes may pose greater challenges in terms of detection in the EV ecosystem, possibly due to their complexity or GUARDEV's

Table 2. Category-Wise Area Under the Curve (AUC) Evaluation for GUARDEV Across Different Attack Vectors

Attack Category	Attack Sub-Category	AUC (Probability)
DDoS	ACK Fragmentation	0.99
DDoS	UDP Flood	1.00
DDoS	SlowLoris	0.99
DDoS	ICMP Flood	0.98
DDoS	RSTFIN Flood	1.00
DDoS	PSHACK Flood	0.99
DDoS	HTTP Flood	1.00
DDoS	UDP Fragmentation	1.00
DDoS	TCP Flood	1.00
DDoS	SYN Flood	1.00
DDoS	SynonymousIP Flood	0.98
Brute Force	Dictionary Brute Force	1.00
Spoofing	Arp Spoofing	0.99
Spoofing	DNS Spoofing	0.99
DoS	TCP Flood	0.99
DoS	HTTP Flood	1.00
DoS	SYN Flood	1.00
DoS	UDP Flood	0.99
Recon	Ping Sweep	0.98
Recon	OS Scan	0.98
Recon	Vulnerability Scan	0.99
Recon	Port Scan	1.00
Recon	Host Discovery	1.00
Web-based	SQL Injection	1.00
Web-based	Command Injection	1.00
Web-based	Backdoor Malware	1.00
Web-based	Uploading Attack	0.99
Web-based	XSS	0.98
Web-based	Browser Hijacking	0.99
Mirai	GREIP Flood	0.89
Mirai	Greeth Flood	1.00
Mirai	UDPPlain	0.87

limited training on such EV-specific attacks. Despite these errors, GUARDEV demonstrates overall effectiveness in detecting and responding to various attacks targeting electric vehicles. Understanding the nature of these errors is crucial for enhancing GUARDEV's adaptive capabilities and mitigating potential vulnerabilities in EV security.

4.3 Cumulative Gain Curve Analysis for GUARDEV's EV Protection Efficiency

The Cumulative Gain (CG) curve [14], depicted in Figure 4, stems from the evaluation of GUARDEV on a dataset of attack samples targeting electric vehicles. Each sample is categorized into specific attack classes denoted as C_i for $i = 1, 2, ..., N$, representing various cyber threats to EVs. Throughout the evaluation, attacks are sequentially analyzed, and the number of true positives (TPs) and false positives (FPs) for each class are recorded at every step, reflecting GUARDEV's guided and adaptive response capabilities.

Mathematically, at a particular step n, $TP_i(n)$ represents the number of correctly classified attacks belonging to class C_i. The cumulative gain at this step, denoted as $CG(n)$, is computed as:

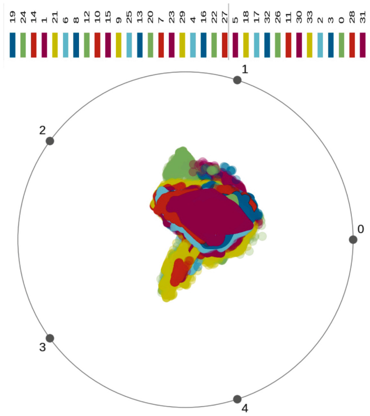


Fig. 4. Cumulative Gain Analysis of GUARDEV for Enhanced EV Cyber Defense

$$CG(n) = \frac{1}{N} \sum_{i=1}^{N} \frac{TP_i(n)}{T_i} \tag{14}$$

where T_i is the total number of attacks in class C_i, and N is the total number of attack classes relevant to EV security. The curve's concavity offers insights into GUARDEV's detection and response efficiency in the context of EV protection:

- **Concave-up Curve:** Indicates that GUARDEV prioritizes high-confidence attack detection, achieving significant initial gains in CG. This suggests that GUARDEV's guided response mechanism is particularly effective in quickly identifying and responding to the most prevalent or easily detectable EV-related cyber threats.
- **Plateaus:** Suggest saturation points where additional analysis yields diminishing returns, implying encounters with more challenging EV-specific attacks later in the analysis. These plateaus help identify areas where GUARDEV's adaptive mechanisms may need further refinement to address complex or evolving threats to electric vehicles.

The analysis of the CG curve furnishes valuable insights into GUARDEV's detection and response efficiency, aiding in the comprehensive assessment of its performance across various attack classes targeting EVs. This analysis is crucial for:

- Fine-tuning GUARDEV's guided response mechanisms for different types of EV-related cyber threats.
- Enhancing the unified approach to address a wider range of attack patterns specific to electric vehicles.
- Improving GUARDEV's adaptive capabilities to evolve with new and emerging threats in the EV ecosystem.

By leveraging these insights, GUARDEV can continually improve its ability to provide a comprehensive, efficient, and adaptive defence strategy for electric vehicles against the ever-evolving landscape of cyber threats.

5 Conclusion

Our study presents GUARDEV, a pioneering Guided Unified Adaptive Response framework for Defending Electric Vehicles within the Internet of Vehicles (IoV) ecosystem. By leveraging an ensemble-based machine learning (ML) approach that synergizes XGBoost, Random Forest, and Extra Trees, GUARDEV effectively mitigates diverse cyber threats prevalent in IoV environments. With an outstanding accuracy of 99.47% on the CICIoT2023 dataset, GUARDEV surpasses existing ML-based Intrusion Detection Systems (IDS) in both accuracy and F1-scores, showcasing its robustness and efficacy in safeguarding IoV networks against a spectrum of cyber threats. The comprehensive evaluation and analysis presented in this paper highlight the superiority of GUARDEV over state-of-the-art models. Notably, our framework demonstrates exceptional accuracy and robustness in detecting various network attacks, outperforming individual models such as Random Forest, Extra Trees, and XGBoost. The guided and adaptive nature of GUARDEV allows it to continuously refine its defence strategies, ensuring optimal protection against evolving threats. Furthermore, the detailed examination of attack prediction errors and the analysis of the Cumulative Gain curve provide valuable insights into GUARDEV's performance and areas for improvement. These insights inform the framework's adaptive capabilities, enabling it to enhance its defense mechanisms over time.

In conclusion, GUARDEV represents a significant advancement in EV cybersecurity within the IoV paradigm by addressing the pressing need for enhanced cybersecurity measures in IoV networks through a guided, unified, and adaptive solution. However, several hard research challenges remain, including the continuous adaptation of the framework to emerging, sophisticated cyber threats, integration of real-time data processing capabilities while maintaining high accuracy and low latency, and ensuring scalability to accommodate the growing number of connected vehicles and the increasing complexity of IoV networks. Future research should focus on developing more robust methods for real-time threat detection and response, exploring the integration of quantum computing techniques for enhanced cryptographic security, and investigating the ethical implications of autonomous decision-making in cybersecurity. Addressing these challenges will pave the way for even safer and more secure EV operations in the evolving landscape of connected vehicles and smart transportation systems, providing a robust and flexible defense mechanism capable of adapting to new and emerging cyber threats.

Acknowledgement. This work is partially supported by the DST IIT Bhilai Innovation and Technology Foundation (IBITF) under the grant IBITF/Note/EIR-PRAYAS/Cohort-03/SanctionLetter/2024-25/0076. We would like to express our gratitude to the foundation for their invaluable support and funding, which has significantly contributed to the successful completion of this research.

References

1. Agrawal, K., Alladi, T., Agrawal, A., Chamola, V., Benslimane, A.: Novelads: A novel anomaly detection system for intra-vehicular networks. IEEE Trans. Intell. Transp. Syst. **23**(11), 22596–22606 (2022)
2. Alamleh, A., Albahri, O., Zaidan, A., Albahri, A., Alamoodi, A., Zaidan, B., Qahtan, S., Alsatar, H., Al-Samarraay, M.S., Jasim, A.N.: Federated learning for iomt applications: A standardization and benchmarking framework of intrusion detection systems. IEEE J. Biomed. Health Inform. **27**(2), 878–887 (2022)
3. Aldhyani, T.H., Alkahtani, H.: Cyber security for detecting distributed denial of service attacks in agriculture 4.0: Deep learning model. Mathematics **11**(1), 233 (2023)
4. Alshammari, A., Zohdy, M.A., Debnath, D., Corser, G.: Classification approach for intrusion detection in vehicle systems. Wirel. Eng. Technol. **9**(4), 79–94 (2018)
5. Arthurs, P., Gillam, L., Krause, P., Wang, N., Halder, K., Mouzakitis, A.: A taxonomy and survey of edge cloud computing for intelligent transportation systems and connected vehicles. IEEE Trans. Intell. Transp. Syst. **23**(7), 6206–6221 (2022)
6. Ashraf, J., Bakhshi, A.D., Moustafa, N., Khurshid, H., Javed, A., Beheshti, A.: Novel deep learning-enabled lstm autoencoder architecture for discovering anomalous events from intelligent transportation systems. IEEE Trans. Intell. Transp. Syst. **22**(7), 4507–4518 (2020)
7. Awajan, A.: A novel deep learning-based intrusion detection system for iot networks. Computers **12**(2), 34 (2023)
8. Chaganti, R., Suliman, W., Ravi, V., Dua, A.: Deep learning approach for sdn-enabled intrusion detection system in iot networks. Information **14**(1), 41 (2023)
9. Chen, D., Yongchareon, S., Lai, E.M., Sheng, Q.Z., Liesaputra, V.: Locally weighted ensemble-detection-based adaptive random forest classifier for sensor-based online activity recognition for multiple residents. IEEE Internet Things J. **9**(15), 13077–13085 (2022)
10. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. pp. 785–794 (2016)
11. Chen, Z., Simsek, M., Kantarci, B., Djukic, P.: All predict wisest decides: A novel ensemble method to detect intrusive traffic in iot networks. In: 2021 IEEE Global Communications Conference (GLOBECOM). pp. 01–06. IEEE (2021)
12. Deolindo, V.M., Dalmazo, B.L., da Silva, M.V., de Oliveira, L.R., Silva, A.d.B., Granville, L.Z., Gaspary, L.P., Nobre, J.C.: Using quadratic discriminant analysis by intrusion detection systems for port scan and slowloris attack classification. In: International Conference on Computational Science and Its Applications. pp. 188–200. Springer (2021)
13. Elmasry, W., Akbulut, A., Zaim, A.H.: Evolving deep learning architectures for network intrusion detection using a double pso metaheuristic. Comput. Netw. **168**, 107042 (2020)
14. Huang, S., Poursafaei, F., Danovitch, J., Fey, M., Hu, W., Rossi, E., Leskovec, J., Bronstein, M., Rabusseau, G., Rabbany, R.: Temporal graph benchmark for machine learning on temporal graphs. Advances in Neural Information Processing Systems **36** (2024)

15. Ingre, B., Yadav, A., Soni, A.K.: Decision tree based intrusion detection system for nsl-kdd dataset. In: International conference on information and communication technology for intelligent systems. pp. 207–218. Springer (2017)
16. Neto, E.C.P., Dadkhah, S., Ferreira, R., Zohourian, A., Lu, R., Ghorbani, A.A.: Ciciot2023: A real-time dataset and benchmark for large-scale attacks in iot environment (2023)
17. Otchere, D.A., Ganat, T.O.A., Nta, V., Brantson, E.T., Sharma, T.: Data analytics and bayesian optimised extreme gradient boosting approach to estimate cut-offs from wireline logs for net reservoir and pay classification. Appl. Soft Comput. **120**, 108680 (2022)
18. Patel, A.M., Suthar, A.: Adaboosted extra trees classifier for object-based multispectral image classification of urban fringe area. Int. J. Image Graph. **22**(3), 2140006:1–2140006:16 (2022)
19. Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A.: Toward generating a new intrusion detection dataset and intrusion traffic characterization. ICISSp **1**, 108–116 (2018)
20. Song, H.M., Woo, J., Kim, H.K.: In-vehicle network intrusion detection using deep convolutional neural network. Vehicular Communications **21**, 100198 (2020)
21. Verma, A., Ranga, V.: Cosec-rpl: detection of copycat attacks in rpl based 6lowpans using outlier analysis. Telecommun. Syst. **75**, 43–61 (2020)
22. Wang, C., Li, Z., Xia, X., Shi, J., Si, J., Zou, Y.: Physical layer security enhancement using artificial noise in cellular vehicle-to-everything (C-V2X) networks. IEEE Trans. Veh. Technol. **69**(12), 15253–15268 (2020)
23. Yang, L., Moubayed, A., Hamieh, I., Shami, A.: Tree-based intelligent intrusion detection system in internet of vehicles. In: 2019 IEEE global communications conference (GLOBECOM). pp. 1–6. IEEE (2019)

Synthetic Vascular Models : Application to Bifurcation Classification and Aneurysm Detection

Rafic Nader[1], Vincent L'Allinec[2], Romain Bourcier[1,3], and Florent Autrusseau[1,4(✉)]

[1] Nantes Université, CHU Nantes, CNRS, INSERM, l'institut du thorax, 44000 Nantes, France
Florent.Autrusseau@univ-nantes.fr

[2] CHU Angers, neuro-radiology department, Angers, France

[3] CHU Nantes, neuro-radiology department, Nantes, France

[4] Ecole Polytechnique de l'Université de Nantes, LTeN U6607 Nantes Université, Nantes, France

Abstract. In this work, we present new synthetic vascular models that tries to mimic various portions of the cerebral vascular tree, as acquired from Magnetic Resonance Angiography - Time of Flight modality - acquisitions. Not only are these vascular models able to replicate the cerebral arteries, but also, the bifurcations formed by the arteries, and furthermore, one option within the models allows to embed an intracranial aneurysm. Our goal in designing this set of tools was to train convolutional neural networks for various pattern recognition tasks; namely, we intend to label the main bifurcations forming the Circle of Willis, or to automatically detect intracranial aneurysms. However, to efficiently train a neural network, the fidelity of the mimicked vascular portions is of paramount importance.

Keywords: Cerebral vascular tree · Cerebral bifurcations · Circle of Willis · synthetic model · intracranial aneurysms

1 Introduction

Various diseases may occur along the vascular tree. Such accidents can be particularly critical when located within the brain. Among the various vascular pathologies, the intracranial aneurysms (ICA) can be particularly devastating [16]. Cerebral aneurysms commonly occur onto the bifurcations forming a central arterial structure named the Circle of Willis (CoW) [3]. More specifically, they arise between the daughter arteries, they are referred to as saccular aneurysms, due to their balloon shape. Intracranial aneurysms occur for 3 to 5% of the world population.

This work was supported by projects "eCAN" #ANR-23-RHUS-0013, #ANR-21-CE17-0006 and INSERM CoPoC #MAT-PI-22155-A-01.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 63–77, 2025.
https://doi.org/10.1007/978-3-031-78383-8_5

While an aneurysm itself may not pose immediate harm, if it bleeds (ruptures), it induces severe consequences such as subarachnoid hemorrhage, resulting in death (35%) or serious cognitive deficits (46%) [13]. Hence, it is crucial to not only, automatically detect the aneurysms, but also, to monitor the portions of the CoW presenting a higher risk (see Fig. 1). Prior to the emergence of deep learning techniques such as Convolutional Neural Networks (CNNs), fewer works focused on probabilistic or traditional machine learning approaches for labeling the Circle of Willis bifurcations [2,17,21]. Recent advances include a deep learning based method for CoW arteries segmentation [7] and a two-step pipeline for detecting CoW vascular bifurcations [14]. In the context of aneurysms detection, various deep learning-based methods have emerged for the segmentation and/or detection of ICAs [9,15,18]. Of particular interest, the ADAM Challenge [20] compared 11 distinct deep learning approaches aimed at detecting and/or segmenting aneurysms. It is crucial to highlight that the majority of existing methods have been developed using private clinical data, which includes meticulously refined manual annotations. Indeed, when it comes to artificial intelligence, there is a recurring burden : the acquisition of manual annotation. To address this issue, the authors in [6] suggested employing "weak" annotations.

In this work, we intend to propose some alternatives to these manual annotations. We have designed a set of synthetic Vascular Models (VaMos)[1] which purpose is specifically to train CNNs. Although one can find synthetic models in the literature, the final aim differs. In [10], the authors used Constrained Constructive Optimization (CCO) for arterial model tree generation, mainly focusing on predicting vascular network growth. Similarly, works in [19] proposed a macroscopic model emphasizing angiogenesis and capillary sprout formation, and authors in [11] exploited CCO to estimate liver vascular network growth. The work in [4] used CCO onto cerebral arteries, and incorporated level set functions for growth estimation. Later, authors in [8,22] proposed *VascuSynth*, a numerical vascular tree generation tool intended to model not only the geometrical layout of the arterial tree, but also simulating the background noise, albeit with limited accuracy in replicating the artery tortuosity. More recently, SimVascular [12] offered advanced 3D mesh modeling for cardiovascular simulation but lacks flexibility in modifying the geometry or modeling background noise. Unfortunately, none of these models could be efficiently exploited along with machine learning methods for Computer-Aided Diagnosis. Another interesting approach was proposed in [5] where the authors generate synthetic blood vessels surfaces. Variational Autoencoders and Generative Adversarial Networks were used to generate mesh like 3D arteries and try to model stenosis; but only onto isolated arteries, neither bifurcations, nor aneurysms were considered.

In this paper, we aim to come up with a highly reliable synthetic vascular model. We aim to generate a substantial (synthetic) image dataset to train various Deep-Learning algorithms. The rationale behind our approach is to demonstrate that, even with a small dataset, if supplemented by VaMos, we can achieve excellent results. In Section 2, we will present VaMos in details. In Section 3, we

[1] Available here : https://gitlab.univ-nantes.fr/autrusseau-f/vamos.

will evaluate the improvements brought by the synthetic data on two distinct tasks: *i)* bifurcations classification, and *ii)* ICA detection. Finally, in Section 4, we conclude our work by summarizing the key findings.

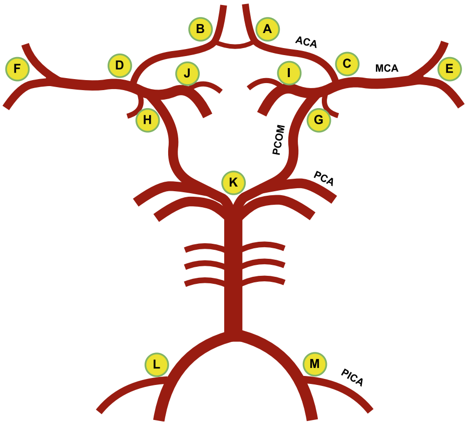


Fig. 1. Bifurcations of Interest along the Circle of Willis (Yellow tags)

2 VaMos : Vasculature Models

Fig. 2 shows the overall structure of the Vasculature Models. Overall, three important vascular components are being modeled : the arteries, the bifurcations and the aneurysms. This means, one could either generate synthetic branches for segmentation purpose, or bifurcations for their classification, or, it would also be possible to generate only synthetic ICA to be merged onto actual MRI images for their detection. Moreover, besides the different geometrical shapes of the vascular tree, the various background matters are also mimicked (gray/ white matters, cerebro-spinal fluid, lateral ventricles, etc.). The upper part in Fig. 2 (yellow shaded block) represents the background noise modeling, whereas the lower part (light blue shaded block), shows the replication of the arterial geometry. In order to generate synthetic images with adjustable resemblance to the ground truth, the VaMos have been designed to modify everything from its basis image portion. Indeed the five green ellipses show all the modifications that can be brought onto an aneurysm bearing bifurcation; namely, we can *i)* tweak the background shape, *ii)* modulate the noise amplitudes, *iii)* adjust the arteries' tortuosity, *iv)* change the diameters, and finally, *v)* add up an ICA (while modifying its shape). So far, our model has been designed, adjusted and tested on Magnetic Resonance Angiography images, with Time-of-Flight modalities (MRA-TOFs).

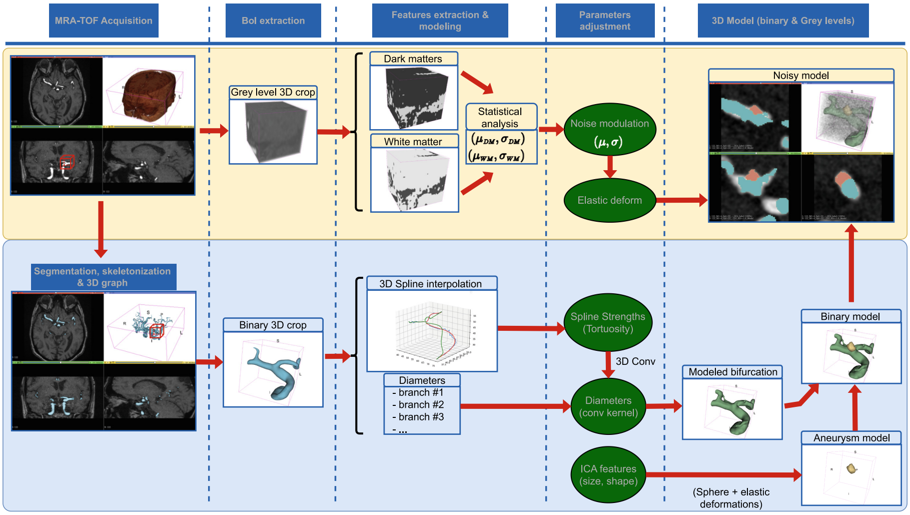


Fig. 2. Structure of the synthetic Vasculature Models.

2.1 Modeling the structure of the bifurcations

From the binary representation of a given MRA-TOF acquisition, the 3D skeleton is first computed, and then, the 3D graph is collected. Each single branch from a given cropped area of interest (typically located around a bifurcation of the CoW) is identified, the voxels along the centerline are represented as a 3D curve, on which 3D B-splines will be fitted. Each branch (artery) can thus be represented by the knots, the B-spline coefficients and the degree of the spline. It is then relatively easy to tweak the 3D spline model by modifying details on B-spline equations fitting. Figure 3 represents two different spline coefficients modifications for a given bifurcation. The blue lines represent the actual bifurcation centerline, the red lines represents the best 3D fit, whereas the green lines stand for the modified 3D splines (left panel : weak modifications and right panel: stronger variation of the 3D splines).

2.2 Modeling the background gray levels

In the aim to generate a 3D noise presenting strong similarities with our target MRA-TOF, our approach consists in producing a higher frequency noise patch, which will be subsequently filtered through a predefined filter kernel, so as to reach the target statistical properties. In other words, if a Gaussian noise patch of standard deviation σ_0 goes through a Gaussian filter of standard deviation σ_f, the resulting filtered noise will present a standard deviation σ_G, such that:

$$\sigma_G \approx \frac{\sigma_0}{(2\sigma_f\sqrt{\pi})} \tag{1}$$

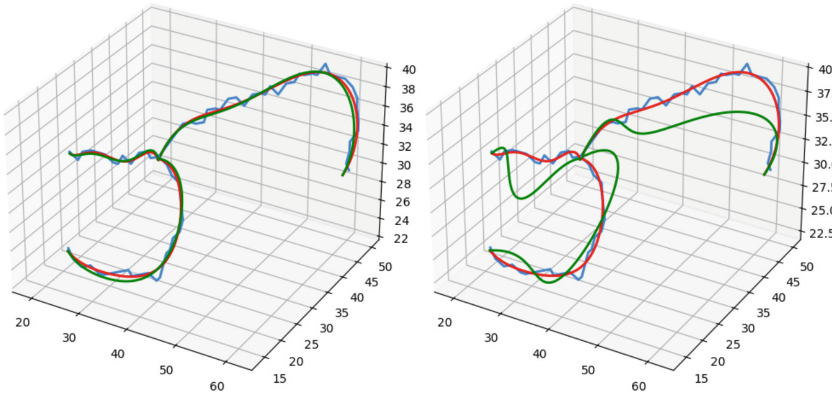


Fig. 3. 3D spline fit of the three arteries forming a bifurcation for two distinct spline modification parameters.

Indeed, when an image, composed of Gaussian noise of standard deviation σ_0 is being filtered by a Gaussian filter of standard deviation σ_G, the so-obtained filtered image ends up with a standard deviation of σ_f. according to the eq. 8. For our particular purpose, we intend to determine which Gaussian filter (of standard deviation σ_G) shall be used on the input image so as to obtain a filtered image with a given target statistics (σ_f), and hence $\sigma_G \approx \sigma_0/(2\sigma_f\sqrt{\pi})$.

This allows us to generate a high frequency noise of average set to our target 3D crop. This noise will then be smoothed by a Gaussian filter (σ_G). The resulting image (of standard deviation σ_f) will thus present strong statistical similarities with the target portion of the MRA-TOF being modeled. Evidently, our model allows to target slightly different statistical noises as the one extracted from the ground truth cropped area.

An input image $I(x, y)$ is Gaussian filtered as :

$$O(x,y) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} \frac{1}{2\pi\sigma_G^2} e^{-\frac{i^2+j^2}{2\sigma_G^2}} I(x+i, y+j) \tag{2}$$

According to the Bienaymé's identity :

$$Var\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n} Var(X_i) + \sum_{i,j=1,i\neq j}^{n} Cov(X_i, X_j) \tag{3}$$

And thus, the variance is:

$$Var\left(\sum_{i=1}^{n} c_i X_i\right) = \sum_{i=1}^{n} c_i^2 Var(X_i) + 2 \times \sum_{i,j=1,i\neq j}^{n} c_i c_j Cov(X_i, X_j) \tag{4}$$

However, if $X_i, ..., X_n$ are pairwise independent ($Cov(X_i, X_j) = 0$, $\forall(i \neq j)$):

$$Var\left(\sum_i c_i X_i\right) = \sum_i c_i^2 Var(X_i) \tag{5}$$

(c_i being constants). We consider that the variance of $I(x, y)$ is $Var\left[I(x+i, y+i)\right] = \sigma_0^2$; we estimate the variance of the output image $Var\left[O(x, y)\right] = \sigma_f^2$. Thus,

$$\sigma_f^2 = \sigma_0^2 \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} \left(\frac{1}{2\pi\sigma_G^2} e^{-\frac{i^2+j^2}{2\sigma_G^2}} \right)^2 \tag{6}$$

For large σ_G, the sum can be approximated as:

$$\begin{aligned} \sigma_f^2 &\approx \sigma_0^2 \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \left(\tfrac{1}{2\pi\sigma_G^2} e^{-\frac{i^2+j^2}{2\sigma_G^2}} \right)^2 di.dj \\ &= \tfrac{\sigma_0^2}{4\pi\sigma_G^2} \end{aligned} \tag{7}$$

and thus,

$$\sigma_f \approx \frac{\sigma_0}{2\sigma_G\sqrt{\pi}} \tag{8}$$

2.3 Adding an intracranial aneurysm

Besides the generation of highly similar and yet easily tunable arteries, our models allow to embed an intracranial saccular aneurysm between the daughter arteries. The ICA is located onto the bisector of the two daughter arteries, at a distance $\mathcal{D}$ from the bifurcation center such that:

$$\mathcal{D} = r \times \gamma + \sqrt{\left(\frac{R}{tan(\Theta/2)}\right)^2 + R^2} \tag{9}$$

R being the average artery radius, Θ the angle between the daughter arteries, r is the ICA radius, and γ, a growth factor allowing to push/pull the aneurysm inward/outward the bifurcation.

Figure 4 shows a possible configuration of a bifurcation bearing an aneurysm. The distance $\mathcal{D}$ separating the aneurysm center and the bifurcation node can be computed as shown in the equation (9). The various constituents of this formula are represented in Fig.4.

2.4 Synthetic Model Evaluation

We have conducted a thorough evaluation of the model features in terms of both bifurcation anatomical evaluation, and aneurysm shape. We present on Fig. 5 a comparison between the Ground Truth (GT) bifurcations and those generated by the Synthetic Model (SM). We hereby compare the angles formed by the three arteries, their diameters, as well as their tortuosity.

We can observe that the modeled branches exhibit strong similarities with their respective ground truths. For this comparison experiment, 100 cropped portions of the bifurcation #C (see Fig.1) were evaluated. Similarly, we have

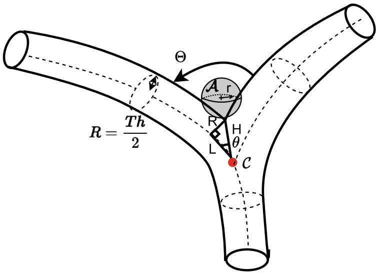


Fig. 4. Position of the synthetic aneurysm along the bisector between the daughter arteries.

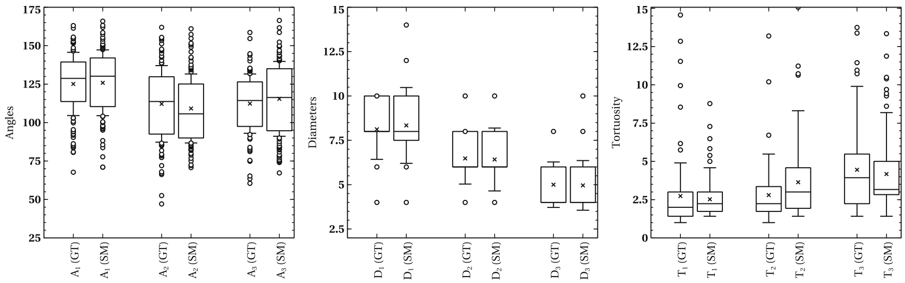


Fig. 5. Evaluation of the bifurcation model with respect to the angles (A), the diameters (D) and branches tortuosity (T).

evaluated various geometrical properties of the aneurysmal sacs, namely, the ICA volume, the outer surface, as well as the sphericity, elongation and flatness coefficients.

We show on Fig. 6 how the various features of the modeled aneurysm actually match quite accurately the ground truth features. Indeed, we can notice the strong similarities between the MRA-TOF and the modeled features. We can expect the model to be efficiently used to train neural networks. The next section is dedicated to the experimental results.

3 Experiments and results

Let us now present the increased performances brought by using VaMos along with neural networks on two specific tasks: CoW bifurcations classification (Task 1) and aneurysms detection (Task 2) on MRA-TOFs. We demonstrate the efficiency of using the synthetic model alone or as a form of data augmentation for a small dataset.

For each experiment, the images were manually labeled by a trained operator, and validated by an expert neuro-radiologist. The MRI images used in the experiments were collected from different French institutions and were acquired

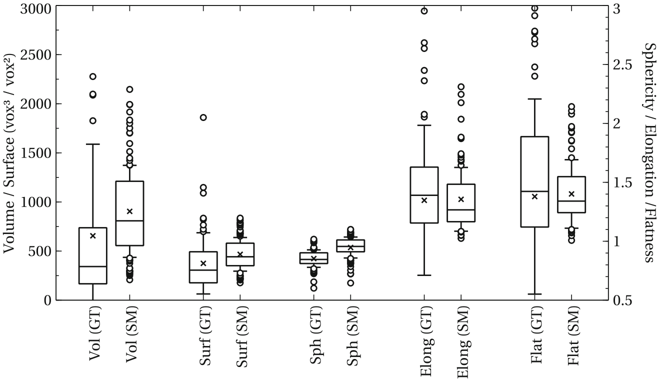


Fig. 6. Evaluation of the modeled aneurysms in terms of sac volume (Vol), outer surface (Surf), Sphericity (Sph), Elongation (Elong) and Flatness(Flat).

using 19 different MRI scanners from Siemens Healthcare, GE Medical systems, Philips Medical systems and Fujifilm (see Table 1).

3.1 Task 1: Classification of CoW bifurcations

This section is devoted to the classification of 3D patches encompassing the Bifurcations of Interest (BoI) along the CoW via 3D CNNs. In our research, we focused on the 13 BoI being associated with the highest risk of aneurysm occurrence [17]. These specific bifurcations are depicted in Fig. 1.

Dataset: For Task 1, we have selected 154 MRA-TOF images. For the training phase, a total of 110 images were used, while an independent test dataset was composed of 44 images. To ensure consistency in image dimensions and voxel spacing across the dataset, we re-sampled all MRA-TOFs to a uniform voxel spacing of 0.4 mm^3.

Data annotation and data generation: To annotate the Ground Truth patches, we used a 3D skeleton computed on the vessel segmentation, along with its corresponding 3D undirected graph. The graph nodes help us identifying the center of each bifurcation and extract 32 × 32 × 32 voxel patches with an isotropic voxel size of 0.4 mm around these coordinates, resulting in a total of 1180 bifurcations in the training set and 386 in the test set. To assess the potential enhancements brought by VaMos, we have generated 40 synthetic models for each BoI. These models were designed to replicate the features of the actual bifurcations, with slight modifications (diameters, tortuosity, etc.)

Neural network and evaluation protocol: Building upon our previous research [14], we employed a 3D-CNN for classification purposes (see Fig. 7).

Table 1. Summary of the Time of flight (TOF) magnetic resonance imaging (MRI) dataset used in the study.

Dataset	**Maker**	**MR device**	**MFS (T)**
Set-1	GE	Optima MR450W	1.5
Set-2	GE	Optima MR360W	1.5
Set-3	GE	Discovery MR750W	3.0
Set-4	GE	Signa HDxt	1.5
Set-5	GE	Signa HDxt	3.0
Set-6	GE	Signa Artist	1.5
Set-7	SIEMENS	Aera	1.5
Set-8	SIEMENS	Skyra	3.0
Set-9	SIEMENS	Avanto	1.5
Set-10	SIEMENS	Prisma	3.0
Set-11	SIEMENS	Sonata	1.5
Set-12	SIEMENS	Verio	3.0
Set-13	SIEMENS	Magnetom Sola	1.5
Set-14	SIEMENS	Magnetom Amira	1.5
Set-15	Philips	Ingenia	3.0
Set-16	Philips	Ingenia Edition X	3.0
Set-17	Philips	Achieva	3.0
Set-18	Philips	Achieva	1.5
Set-19	Fujifilm	Echelon Oval	3.0

The model is composed of nine convolutional layers, each with a kernel size of 3 and a stride of 1. These layers are organized into five convolutional blocks with 32, 64, 128, 256, 512 respective feature channels. The last 3 layers are fully connected. Dropout layer is used for regularization. We implemented the neural networks considered in this work using Tensorflow framework (2.9.0).Training and inference were performed on an NVIDIA RTX A5000 GPU with 16 GB of memory.

We have evaluated the improvements using three different training dataset sizes: D1 (48 TOFs), D2 (82 TOFs), and D3 (110 TOFs). Each dataset was exploited in two distinct experiments: Exp. #B1 involved training exclusively with actual TOF patches, while Exp. #B2 used only synthetic patches for training. Each dataset was partitioned into five folds for cross-validation. The models were trained using categorical cross-entropy loss, Adam optimizer and a learning rate of 0.0001. Training was conducted for 100 epochs, selectively saving the model with the best performance on the validation fold. After training, model evaluation involved using a holdout test set, with final predictions derived by averaging predictions from the five-fold models.

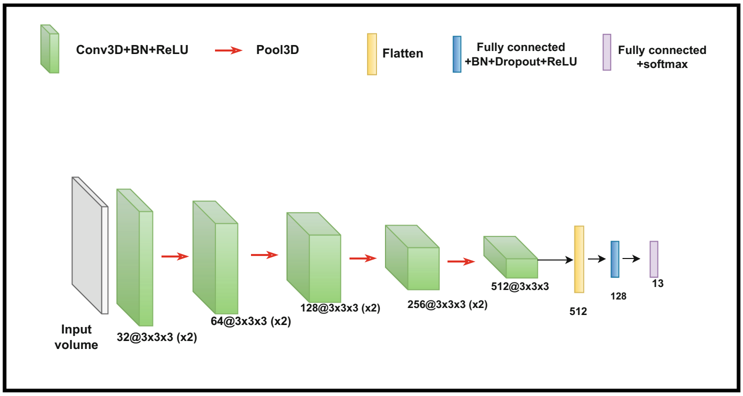
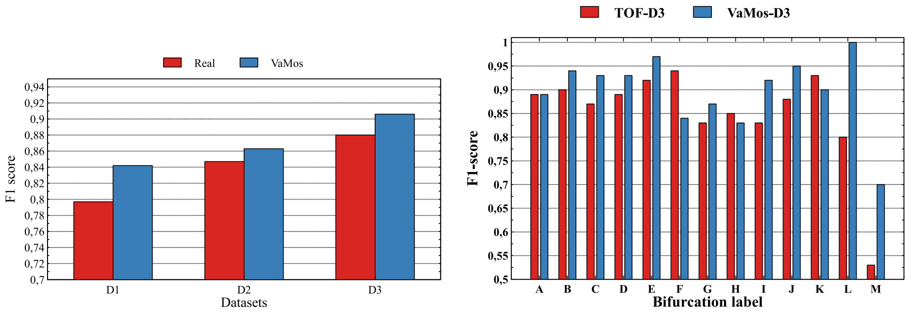


Fig. 7. CNN architecture used for BoIs classification



(a) F1-score using different training sets. (b) F1-score by class using dataset D3.

Fig. 8. F1-Score improvements brought by using the VaMos synthetic patches (on the test set).

Classification results: Our evaluation aimed to assess the impact of using VaMos across various training dataset sizes. The overall performance is evaluated by computing the F1-score across all the samples for each experiment. The results are shown in Fig. 8a. When using actual TOF patches, we notice a notable increase of F1-score, from 79.7% to 88% when the dataset size increases (from D1 to D3). Similarly, when using VaMos, the F1-scores also significantly increase, from 84.2% to 90.6%, although the degree of variation between cases is less pronounced. Across all datasets, the performances are consistently increased when using VaMos. The improvements are more pronounced for D1 (up to 4.5%) than D2 and D3 (up to 2%). We also report the score for each class (bifurcation label) when using D3 in Figure 8b. We observe on this plot that when VaMos patches are included (blue bars, labeled “VaMos-D3”), a better F1-score is achieved for 9 of the 13 classes, compared to a training using only TOF crops (red bars labeled “TOF-D3”).

3.2 Task 2: Aneurysms detection

Let us now evaluate the possible improvements brought by using VaMos in an intracranial aneurysms detection scenario.

Dataset: For Task 2, we have collected 105 scans with unruptured ICAs (distinct from those used in Task 1). The dataset was split into 70 training images, used for both training and validation, and a separate test set of 35 images. Each image contains from 1 to 4 aneurysms, totaling 138 aneurysms with a mean radius of 2.57 ± 0.89 mm.

Data sampling and generation: To address the ICA detection task, we adopted a 3D U-Net implementation used in [6]. This latter was chosen to effectively perform the aneurysm segmentation and its subsequent detection. We use small patches ($64 \times 64 \times 64$ voxels, *i.e.* 25.6 mm wide). For positive samples (with ICA), we extract 10 copies for each aneurysm by shifting its position within the patch. For negative patches, we extract 20 samples for each volume, selected to encompass cerebral arteries (but aneurysm-free). For data generation, 134 aneurysm-free patches from Task 1 served as a basis to generate 998 synthetic patches containing an ICA. Various aneurysms shapes and sizes were simulated by manipulating the radius parameter and applying elastic deformations that emulate the characteristics of original scans, as shown in Fig 6. Moreover for a complete comparison, we apply traditional data augmentation techniques on positive patches, namely rotations within the interval $[-15°, +15°]$ and $(90°, 180°, 270°)$, as well as horizontal and vertical flipping and Gaussian noise addition.

Training and evaluation protocol: The U-Net was optimized using a loss function combining Dice loss and binary Cross-entropy loss along with Adam optimizer and a learning rate of 0.0001. To assess potential improvements brought by VaMos, three distinct experiments were conducted. In Exp. #A1, we trained a baseline model using 50 TOFs (630 positive patches). In Exp. #A2, the training dataset was augmented with patches via 7 traditional data augmentation operations (with a total of 5040 patches $(630 + 7 \times 630)$. In Exp. #A3, the training dataset was augmented with the 998 VaMos patches (total of 1628 positive patches). We used the remaining 20 TOFs as a validation data for all three experiments and the separate test set composed of 35 TOFs (with a total of 50 aneurysms) for inference. During the inference stage, patches centered around cerebral bifurcations, identified through an automated vessel segmentation [14] and subsequent 3D undirected graph generation, are selectively retained to target regions most susceptible to aneurysm development, enhancing the accuracy of the results.

ICA detection performance: The performances were assessed on the test set using two key metrics: lesion-level sensitivity and False Positive (FP) rate (FP per TOF), as outlined in [1]. To calculate the evaluation metrics, each predicted connected component (CC) is treated as a potential detection. If the center of gravity of a CC is present in a ground truth CC, it is classified as a true positive detection. Otherwise, it is labeled as a false positive.

Table 2. The results of aneurysm detection task.

Methods	Sensitivity (%)	FPs/case
Exp. #A1	72	0.37
Exp. #A2	76	1.31
Exp. #A3	88	1.45

In Exp. #A1, the CNN successfully detected 36 aneurysms within a dataset comprising 50 instances, resulting in a lesion-level sensitivity of 72%. This sensitivity increased to 76% in Exp. #A2 with 38 correctly detected aneurysms. However, when incorporating VaMos patches, the lesion level sensitivity reaches 88% with 44 detected aneurysms. Moreover, within Exp. #A1, the network displayed a low false-positive rate of 0.37, which respectively increased to 1.31 and 1.45 for Exp #A2 and Exp #A3 upon incorporating data augmentation (see Table 2).

4 Discussion and Conclusion

In this section, we delve into the impact of the synthetic vasculature model, which effectively mimics portions of MRA-TOF images. The synthetic model encompasses several processes, it accurately models the geometry of cerebral arteries and their bifurcations, it introduces surrounding noise, and incorporates aneurysms of diverse sizes and shapes. Our goal is to provide a comprehensive dataset that can enhance the performance of various deep learning tasks, such as the classification of bifurcations forming the Circle of Willis or the detection of cerebral aneurysms. In this study, we deliberately opted to employ basic deep learning architecture with straightforward optimization techniques. Our primary focus was on the contribution brought by VaMos.

Regarding Task 1, we chose to exclusively use VaMos as the source of training patches. A direct comparison was made with a training solely performed on actual TOF patches. The primary finding is that using hundreds of synthetic patches enhances the classification accuracy across all dataset sizes. This improvement is particularly notable when considering a small dataset (up to a 4.5% increase in F1-score). On average, achieving similar performance is possible by annotating only 82 TOFs supplemented with VaMos, resulting in comparable F1-score (0.863 for VaMos-D2, 0.88 TOF-D3 and 0.906 for VaMos-D3). Our model demonstrates superior performance across most classes, with exceptions observed for classes M, H, and J. The challenges in modeling M may stem from its complex anatomical structure, potentially resulting in trifurcations or quadrifurcations. Uncertainties encountered by neuroradiologists in annotating the MCA branch likely contribute to discrepancies in class H, while hypoplastic cases of PCOM arteries present challenges for modeling J. Ongoing works focus on adapting VaMos to address these specific cases.

A prominent achievement of our research is the successful generation of synthetic aneurysms seamlessly integrated into MRA scans lacking aneurysms, significantly enriching limited dataset. The primary outcome of Task 2 reveals that training the CNN using VaMos patches as Data augmentation yielded a marked improvement in sensitivity for detecting intracranial aneurysms compared to training solely on actual TOFs. Whereas the latter missed 28% of lesions in the test data, for the former, the CNN only missed 12% aneurysms. Traditional data augmentation techniques yielded only a modest improvement in sensitivity (4%) compared to VaMos (16%). This is consistent with the limitations inherent in these traditional methods. These techniques primarily enhance the model's ability to generalize from existing examples without fundamentally diversifying the range of anatomical structures represented in the training data as they do not introduce variations in critical features such as aneurysm shape and size, which are essential for effectively training models to recognize a wide variety of aneurysms across different patients. In contrast, advanced generation models like VaMos are designed to create more complex and diverse synthetic examples that include significant variations in aneurysm characteristics. This was accomplished with a slight increase in false positive rate highlighting the efficacy of our approach. Missed detection in Exp. #A3 predominantly stemmed from small aneurysms. Out of the 6 missed aneurysms, 4 had a radius below 1.5 mm. For such aneurysms, these missed detection can also be attributed to the uncertainty in the initial expert labeling. Nonetheless, our forthcoming endeavors will specifically target this subset of aneurysms, by generating more synthetic aneurysms with small radius.

References

1. Adams, W., Laitt, R., Jackson, A.: Time of flight 3d magnetic resonance angiography in the follow-up of coiled cerebral aneurysms. Interv. Neuroradiol. **5**(2), 127–37 (1999). https://doi.org/10.1177/159101999900500203
2. Bogunovic, H., Pozo, J.M., Cardenes, R., Roman, L.S., Frangi, A.F.: Anatomical labeling of the circle of willis using maximum a posteriori probability estimation. IEEE Trans. Med. Imaging **32**(9), 1587–1599 (2013)
3. Brown, R.D., Broderick, J.P.: Unruptured intracranial aneurysms: epidemiology, natural history, management options, and familial screening. The Lancet Neurology **13**(4), 393–404 (2014)
4. Bui, A.V., Manasseh, R., Liffman, K., Šutalo, I.D.: Development of optimized vascular fractal tree models using level set distance function. Medical engineering & physics **32**(7), 790–794 (2010)
5. Danu, M., Nita, C.I., Vizitiu, A., Suciu, C., Itu, L.M.: Deep learning based generation of synthetic blood vessel surfaces. In: 2019 23rd International Conference on System Theory, Control and Computing (ICSTCC). pp. 662–667. IEEE (2019)

6. Di Noto, T., Marie, G., Tourbier, S., Alemán-Gómez, Y., Esteban, O., Saliou, G., Cuadra, M.B., Hagmann, P., Richiardi, J.: Towards automated brain aneurysm detection in tof-mra: Open data, weak labels, and anatomical knowledge. Neuroinformatics **21**(1), 21–34 (2023)
7. Dumais, F., Caceres, M.P., Janelle, F., Seifeldine, K., Arès-Bruneau, N., Gutierrez, J., Bocti, C., Whittingstall, K.: eicab: A novel deep learning pipeline for circle of willis multiclass segmentation and analysis. Neuroimage **260**, 119425 (2022)
8. Hamarneh, G., Jassi, P.: Vascusynth: Simulating vascular trees for generating volumetric image data with ground truth segmentation and tree analysis. Comput. Med. Imaging Graph. **34**(8), 605–616 (2010). https://doi.org/10.1016/j.compmedimag.2010.06.002
9. Joo, B., Ahn, S.S., Yoon, P.H., Bae, S., Sohn, B., Lee, Y.E., Bae, J.H., Park, M.S., Choi, H.S., Lee, S.K.: A deep learning algorithm may automate intracranial aneurysm detection on mr angiography with high diagnostic performance. Eur. Radiol. **30**, 5785–5793 (2020)
10. Karch, R., Neumann, F., Neumann, M., Schreiner, W.: A three-dimensional model for arterial tree representation, generated by constrained constructive optimization. Comput. Biol. Med. **29**(1), 19–38 (1999)
11. Kretowski, M., Rolland, Y., Bezy-Wendling, J., Coatrieux, J.L.: Physiologically based modeling of 3-d vascular networks and ct scan angiography. IEEE Trans. Med. Imaging **22**(2), 248–257 (2003). https://doi.org/10.1109/TMI.2002.808357
12. Lan, H., Updegrove, A., Wilson, N.M., Maher, G.D., Shadden, S.C., Marsden, A.L.: A re-engineered software interface and workflow for the open-source simvascular cardiovascular modeling package. J. Biomech. Eng. **140**(2), 024501 (2018)
13. Ma, N., Feng, X., Wu, Z., Wang, D., Liu, A.: Cognitive impairments and risk factors after ruptured anterior communicating artery aneurysm treatment in low-grade patients without severe complications: A multicenter retrospective study. Front. Neurol. **12**, 613785 (2021)
14. Nader, R., Bourcier, R., Autrusseau, F.: Using deep learning for an automatic detection and classification of the vascular bifurcations along the circle of willis. Medical Image Analysis p. 102919 (2023).https://doi.org/10.1016/j.media.2023.102919, https://www.sciencedirect.com/science/article/pii/S1361841523001792
15. Park, A., Chute, C., Rajpurkar, P., Lou, J., Ball, R.L., Shpanskaya, K.S., Jabarkheel, R., Kim, L.H., McKenna, E., Tseng, J., Ni, J.C., Wishah, F., Wittber, F., Hong, D.S., Wilson, T.J., Halabi, S.S., Basu, S., Patel, B.N., Lungren, M.P., Ng, A., Yeom, K.W.: Deep learning–assisted diagnosis of cerebral aneurysms using the headxnet model. JAMA Network Open **2** (2019), https://api.semanticscholar.org/CorpusID:174811733
16. Pascalau, R., Padurean, V.A., Bartos, D., Bartos, A., Szabo, B.A.: The geometry of the circle of willis anatomical variants as a potential cerebrovascular risk factor. Turk. Neurosurg. **29**(2), 151–158 (2019)
17. Robben, D., Türetken, E., Sunaert, S., Thijs, V., Wilms, G., Fua, P., Maes, F., Suetens, P.: Simultaneous segmentation and anatomical labeling of the cerebral vasculature. Med. Image Anal. **32**, 201–215 (2016)
18. Shi, Z., Miao, C., Schoepf, U.J., Savage, R.H., Dargis, D.M., Pan, C., Chai, X., Li, X.L., Xia, S., Zhang, X., et al.: A clinically applicable deep-learning model for detecting intracranial aneurysm in computed tomography angiography images. Nat. Commun. **11**(1), 6090 (2020)
19. Szczerba, D., Székely, G.: Macroscopic modeling of vascular systems. In: Dohi, T., Kikinis, R. (eds.) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2002, pp. 284–292. Springer, Berlin Heidelberg, Berlin, Heidelberg (2002)

20. Timmins, K., van der Schaaf, I., Bennink, E., et al.: Comparing methods of detecting and segmenting unruptured intracranial aneurysms on TOF-MRAS: The ADAM challenge. Neuroimage **238**, 118216 (2021)
21. Wang, X., Liu, Y., Wu, Z., Mou, X., Zhou, M., Ballester, M.A.G., Zhang, C.: Automatic labeling of vascular structures with topological constraints via hmm. In: Medical Image Computing and Computer-Assisted Intervention- MICCAI 2017: 20th International Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part II 20. pp. 208–215. Springer (2017)
22. Zhao, M., Hamarneh, G.: Bifurcation detection in 3d vascular images using novel features and random forest. In: 2014 IEEE 11th International Symposium on Biomedical Imaging (ISBI). pp. 421–424. IEEE (2014)

ENS-RFMC: An Encrypted Network Traffic Sampling Method Based on Rule-Based Feature Extraction and Multi-hierarchical Clustering for Intrusion Detection

Liang-Chen Chen[1,2,3,4,5], Shu Gao[1](✉), Zi-Xuan Wei[1], Bao-Xu Liu[3,5](✉), and Xu-Yao Zhang[2]

[1] School of Computer Science and Artificial Intelligence, Wuhan University of Technology, Wuhan 430063, China
gshu@whut.edu.cn

[2] Institute of Automation, NLPR, Chinese Academy of Sciences, Beijing 100190, China
{liangchen.chen,xyz}@nlpr.ia.ac.cn

[3] Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China
liubaoxu@iie.ac.cn

[4] School of Computer, China University of Labor Relations, Beijing 100048, China

[5] Key Laboratory of Network Assessment Technology, Chinese Academy of Sciences, Beijing 100093, China

Abstract. Efficiently and quickly identifying malware in encrypted network traffic is a major challenge. This research presents and evaluates a novel approach for sampling encrypted network traffic, called ENS-RFMC. Initially, the method employs a rule-based strategy for extracting relevant metadata features of encrypted traffic from network connections, SSL and certificates log files. Subsequently, a multi-hierarchical clustering algorithm is utilized to divide the dataset into smaller clusters, with benign clusters subsequently eliminated for streamline data processing. Classifier models such as Random Forests, XGBoost and SVM are then utilized to detect attacks and assess performance. Experimental results indicate that the proposed ENS-RFMC sampling method is effective, with the encrypted traffic intrusion detection model employing ENS-RFMC sampling exhibits enhanced accuracy and efficiency in identifying attacks.

Keywords: Encrypted Traffic Sampling · Rule-based Feature Extraction · Multi-hierarchical Clustering · Intrusion Detection

1 Introduction

In the detection of network encrypted attacks, the vast majority of traffic data in the network is normal data, and only a small amount of attack traffic appears when attacked. This means that network intrusion detection is a classification problem under large-scale data conditions, with the aim of focusing on the security threats posed to the entire network by these small amounts of attack traffic [1]. However, the amount of normal traffic

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 78–92, 2025.
https://doi.org/10.1007/978-3-031-78383-8_6

data is too large, resulting in low storage and analysis efficiency when dealing with a small number of attack behaviors. Encrypted traffic data sampling involves collecting analyzable and testable encrypted traffic from the overall dataset, and the effectiveness of these sampling methods significantly impacts malware detection outcomes. Traditional network traffic sampling methods are not suitable for encrypted traffic, and most existing methods for sampling encrypted traffic are inherently inaccurate [1]. How to sample encrypted traffic data effectively without losing useful information to improve the performance has become a research hotspot in the field of encrypted malware detection.

This research investigates the features of encrypted traffic related to malicious activities, and proposes a sampling method using rule-based feature extraction and multi-hierarchical clustering algorithm (ENS-RFMC) for encrypted intrusion detection. The ENS-RFMC method can sample encrypted traffic from multiple perspectives, thereby reducing the large-scale encrypted traffic in intrusion detection. More importantly, it improves the efficiency of flow-based encrypted intrusion detection. The major contributions of this paper can be summarized as follows.

1. We employ a rule-based feature mining strategy to obtain metadata features of encrypted network traffic. Through log rule association analysis, three categories of features related to malicious activities were examined and extracted: connection features, SSL features, and certificate features.
2. We present a three-stage data sampling method, called ENS-RFMC, which employs rule-based feature extraction and the multi-hierarchical clustering algorithm. This method extracts important data from encrypted traffic for further deep detection.
3. We combine the proposed ENS-RFMC sampling method with machine learning methods to conduct experiments on two real encrypted traffic datasets, and evaluate the performance by comparing it with other widely used data sampling methods. Experimental results indicate that ENS-RFMC can effectively reduce normal traffic, and can more accurately and efficiently detect encrypted malicious traffic.

2 Related Works

According to distinct techniques for processing encrypted traffic, encrypted intrusion detection methods can be divided into active detection and passive detection. Active detection includes methods that involve traffic decryption and those based on searchable encryption. Its research focuses on the application of security measures, including trusted execution environments and controllable transmission protocols. The earliest one was the use of Man in the Middle (MitM) technology for decryption detection [2]. Some network security enterprises have developed detection products based on this technology, which can be applied to the latest version of TLS protocol. Carnavalet X et al. [3] proposed a framework for evaluating client TLS proxies by analyzing 14 antivirus products based on MitM technology, raising doubts about similar security products. Han J et al. [4] proposed an SGX-Box system, which facilitates secure detection of encrypted traffic through the SGX trusted execution environment. Goltzsche et al. [5] proposed a distributed intermediate box system, termed EndBox, which is deployed at the network's edge and offers security services to clients based on SGX trusted hardware technology. Lan C et al. [7] proposed the Embark system, capable of detecting packet header information through range queries or prefix matching. Conversely, passive detection refers to the identification of encrypted malicious traffic without user's perception and without performing any encryption or decryption operations. Its research focuses on feature selection and construction, primarily analyzing relevant detection methods based on three categories of features: side channel features, plaintext features, and original traffic features. Anderson et al. [8] first comprehensively elaborated on a passive detection method for encrypted malicious traffic, marking the emergence of this topic as a significant area within network security. Shekhawat et al. [9] extracted traffic log files and three categories of 38 encrypted traffic features using open-source traffic analysis software, including: connections, sessions and certificates. Liu JY et al. [10] proposed the MalDetect architecture, which can detect malicious traffic by using only the first 8 packets of each encrypted traffic. Hou J [2] compared representative research results and concluded that traditional machine learning algorithms are suitable for constructing lightweight online detection models, while deep learning-based detection models exhibit superior performance when dealing with high-dimensional features.

The traffic data sampling method in network intrusion detection can be categorized into three distinct levels: byte-level sampling, packet-level sampling, and flow-level sampling. Claffy et al. [11] introduced the performance and importance of data sampling methods related to network traffic. He et al. [12] proposed a biased sampling method that estimates the mean more accurately and reduces sampling overhead compared to static sampling and simple random sampling. Raspall et al. [13] proposed a non-uniform random sampling strategy that considers packet size to avoid adverse effects on sampling, thereby improving the efficiency of network traffic analysis. Duffield et al. [14] proposed a heuristic sampling method that selects a suitable sampling function based on the distribution of network flow length, adjusting this function through heuristic approaches. Su et al. [15] applied a hierarchical sampling method to identify benign and malicious clusters within raw network traffic, subsequently analyzing these clusters for filtering and further malware detection. Chen LC et al. [1] proposed a three-stage encrypted traffic sampling model based on an improved density peak clustering algorithm, which extracts 36 features from encrypted traffic and samples the data for further attack detection. However, most existing encrypted traffic sampling methods are either ineffective or inherently inaccurate. In this paper, we propose an improved three-stage hierarchical sampling method for encrypted malicious traffic detection based on reference [1].

3 Rule-Based Feature Extraction Method

Feature extraction constitutes the initial and critical step in the process of data sampling. Therefore, it is necessary to conduct feature extraction on the original input data prior to the reduction of encrypted traffic [16]. Figure 1 illustrates the encrypted traffic feature extraction framework proposed in this research. Firstly, Zeek IDS is used to parse the original encrypted traffic pcap files into multiple log files. Then, protocol related features are extracted by analyzing the association rules.

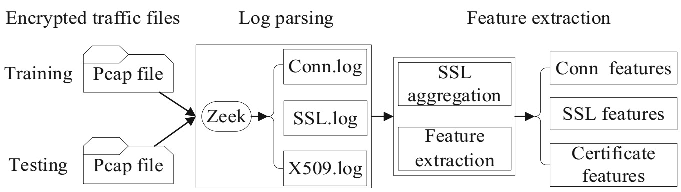
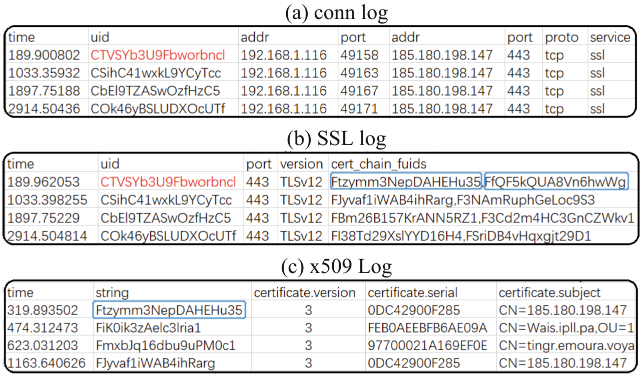


Fig. 1. Encrypted traffic feature extraction process

3.1 Correlation Analysis of Encrypted Traffic Logs

The network data was initially captured using tools such as Wireshark. We employed Zeek to parse each pcap file of malicious encrypted traffic, and generated various related network log files, including dns.log, http.log, conn.log, etc. Then, we assessed the types of generated log files and selected three primary log files for further examination: the connection log file (conn.log), the SSL log file (ssl.log) and the certificate log file (x509.log). The feature information of malicious encrypted traffic is mainly contained within these three log files.

This research employs these three types of traffic log files to establish data associations, and extract relevant feature information based on their associations. And there is a correlation exists among these log files. The internal structure of the log files is illustrated in Fig. 2, where each row contains a unique key used to link rows from other logs. In the ssl. Log, SSL session information uses a unique key and certificate ID to identify the corresponding records in the conn.log and ssl.log. The unique key in conn.log record can be linked to the unique key in the ssl.log for two records. Additionally, the ID key values, which are separated by commas in ssl.log, can be associated with the corresponding certificate records in the x509.log.


(a) conn log

time	uid	addr	port	addr	port	proto	service
189.900802	CTVSYb3U9Fbworbncl	192.168.1.116	49158	185.180.198.147	443	tcp	ssl
1033.35932	CSihC41wxkL9YCyTcc	192.168.1.116	49163	185.180.198.147	443	tcp	ssl
1897.75188	CbEl9TZASwOzfHzC5	192.168.1.116	49167	185.180.198.147	443	tcp	ssl
2914.50436	COk46yBSLUDXOcUTf	192.168.1.116	49171	185.180.198.147	443	tcp	ssl

(b) SSL log

time	uid	port	version	cert_chain_fuids
189.962053	CTVSYb3U9Fbworbncl	443	TLSv12	Ftzymm3NepDAHEHu35,FfQF5kQUA8Vn6hwWg
1033.398255	CSihC41wxkL9YCyTcc	443	TLSv12	FJyvaf1iWAB4ihRarg,F3NAmRuphGeLoc9S3
1897.75229	CbEl9TZASwOzfHzC5	443	TLSv12	FBm26B157KrANN5RZ1,F3Cd2m4HC3GnCZWkv1
2914.504814	COk46yBSLUDXOcUTf	443	TLSv12	Fl38Td29XslYYD16H4,FSriDB4vHqxgjt29D1

(c) x509 Log

time	string	certificate.version	certificate.serial	certificate.subject
319.893502	Ftzymm3NepDAHEHu35	3	0DC42900F285	CN=185.180.198.147
474.312473	FiK0ik3zAelc3lria1	3	FEB0AEEBFB6AE09A	CN=Wais.ipll.pa,OU=1
623.031203	FmxbJq16dbu9uPM0c1	3	97700021A169EF0E	CN=tingr.emoura.voya
1163.640626	FJyvaf1iWAB4ihRarg	3	0DC42900F285	CN=185.180.198.147

Fig. 2. Encrypted traffic log association

By utilizing the interrelationship among these three categories of log files, a substantial array of distinct features can be extracted. Each SSL/TLS connection requires a server certificate, so each record in the ssl.log contains at least one unique certificate ID used by the server to authenticate its signature chain, which corresponds to the certificate records in the x509.log. In this study, we only extract the first certificate ID from ssl.log, as it corresponds to the end user certificate, while the remaining IDs correspond to the intermediate certificate and root certificate.

3.2 Multi-dimensional Feature Extraction Method

Through SSL aggregation of data from conn.log, ssl.log and x509.log, Connection-tetrad records are extracted based on the association rules among log files. Then 4-tuples are generated, and encrypted traffic is analyzed utilizing 4-tuples as sample units to extract corresponding data flow connection features, SSL features and certificate features, thereby obtaining multiple preset features through aggregation.

(1) Connection-tetrad Generation

Definition 1 (SSL aggregation record). An SSL aggregation record consists of a TCP connection record, an SSL record and a certificate record concatenated to store information about encrypted network flow. The connection records, SSL records, and x509 records are sourced from conn.log, ssl.log and x509.log, respectively. Each SSL record has a connection record, and x509 records are only available when a certificate exists in this SSL session.

Definition 2 (Connection-tetrad). A connection-tetrad is composed of a set of SSL aggregated records that collectively share a 4-tuple: source IP, destination IP, destination port, and transport protocol, which is the unique key for each connection-tetrad. Connection-tetrad can be used to describe malicious encrypted traffic, as well as to capture and express benign encrypted traffic patterns.

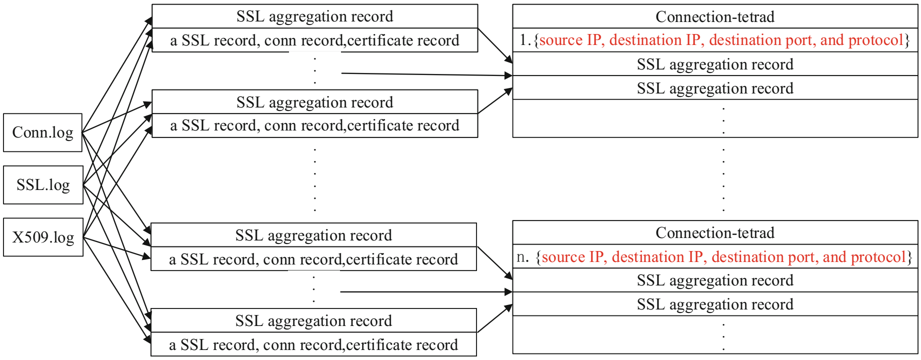

SSL aggregation record
a SSL record, conn record,certificate record
SSL aggregation record
a SSL record, conn record,certificate record
Conn.log
SSL.log
X509.log
SSL aggregation record
a SSL record, conn record,certificate record
SSL aggregation record
a SSL record, conn record,certificate record
Connection-tetrad
1.{source IP, destination IP, destination port, and protocol}
SSL aggregation record
SSL aggregation record
Connection-tetrad
n. {source IP, destination IP, destination port, and protocol}
SSL aggregation record
SSL aggregation record

Fig. 3. SSL aggregation record and Connection-tetrad generation

As shown in Fig. 3, the connection-tetrad comes from three log files: conn.log, SSL.log and X509.log. Firstly, it associates conn.log with ID in SSL.log, and then the associated certificate path is re-linked with the ID in X509.log to generate the SSL aggregation record. Within the obtained SSL aggregation records, aggregation operations are conducted based on the same 4-tuple data to obtain the connection-tetrad. Following this, feature extraction is performed on each connection-tetrad.

(2) Feature extraction

Following the generation of numerous connection-tetrads, it is essential to calculate and extract features from SSL aggregation records with the same 4-tuple in each connection-tetrad, and obtain multiple preset features to form an initial feature set.

For each 4-tuple connection, we extracted 36 features, most of which were created based on our professional knowledge in the field and thorough analysis of our malware data. These 36 features can be organized into feature vectors, and the set of feature vectors corresponding to all connection-tetrads constitutes the final extracted features. We will divide these features extracted from three log files into three distinct groups: connection features, SSL features and certificate features. Each feature is computed by synthesizing information from a single connection-tetrads record, with each feature representing a complex information extraction and aggregation. These features are frequently employed in supervised learning training and have excellent ability to distinguish encrypted malicious traffic.

4 Encrypted Traffic Sampling Method

Reducing encrypted traffic data is the initial task of employing machine learning methods for detecting encrypted malware. This study investigates and extracts three types of encrypted traffic features through network log correlation analysis, then uses an improved multi-hierarchical clustering algorithm to cluster the encrypted traffic, finally performs clustering analysis and filtering. The complete reduction process is shown in Fig. 4.

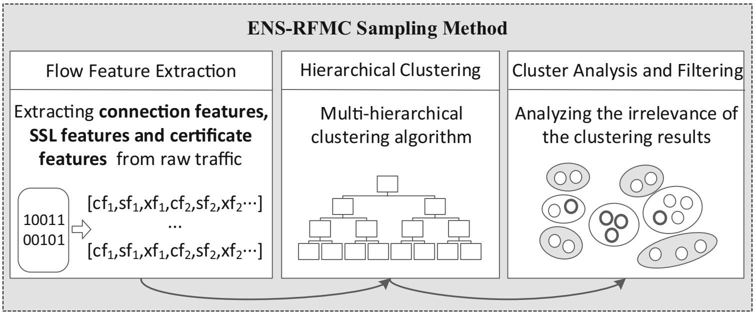

ENS-RFMC Sampling Method
Flow Feature Extraction
Extracting **connection features, SSL features and certificate features** from raw traffic
10011
00101
[cf1,sf1,xf1,cf2,sf2,xf2···]
···
[cf1,sf1,xf1,cf2,sf2,xf2···]
Hierarchical Clustering
Multi-hierarchical clustering algorithm
Cluster Analysis and Filtering
Analyzing the irrelevance of the clustering results

Fig. 4. The three-stage ENS-RFMC sampling process

4.1 Encrypted Traffic Feature Extraction

The connection features describe the common patterns of communication flow unrelated to certificates and encryption, the SSL features describe the SSL handshake and encrypted communication information, and the certificate features describe the certificate information provided during the SSL handshake. Table 1 enumerates the 12 connection features extracted from conn.log, including the number of aggregated and connected records, mean of connection duration and standard deviation of connection duration, etc.

Additionally, the 12 SSL features extracted from ssl.log include the ratio of SSL records to non-SSL records, the ratio of TLS to SSL, and the proportion of SSL records with SNI, etc. Furthermore, the 12 certificate features obtained from x509.log, include the average length of the certificate's public key, the average certificate validity period, the standard deviation of the certificate validity period, and the count of different certificates, etc.

Table 1. Features extracted from conn.log, ssl.log and x509.log.

No	Connection features	SSL features	Certificate features
1	no_of_flows	ssl_ratio	avg_key_len
2	avg_of_duration	tls_version_ratio	avg_of_cert_len
3	stand_devi_of_duration	SNI_ssl_ratio	stand_devi_cert_length
4	percent_sd_of_duration	self_signed_ratio	is_valid_certificate
5	size_of_orig_flows	SNI_equal_DstIP	amount_diff_certificate
6	size_of_resp_flows	differ_SNI_in_ssl_log	no_of_domains_in_cert
7	ratio_of_sizes	differ_subject_in_ssl_log	certificate_ratio
8	percent_of_estab_states	differ_issuer_in_ssl_log	no_of_cert_path
9	inbound_pckts	ratio_of_same_subjects	x509_ssl_ratio
10	outbound_pckts	ratio_of_same_issuer	SNIs_in_SAN_dns
11	periodicity_average	is_SNI_top_level_domain	is_SNs_in_SAN_dns
12	period_stand_deviation	ratio_missing_cert	avg_certificate_age

4.2 Multi-hierarchical Clustering of Encrypted Traffic

To achieve higher quality and smaller clusters while effectively lowering computational costs, we implemented a multi-hierarchical density peak clustering method (HDPC-GS) that integrates three distinct categories of encrypted traffic features. As illustrated in Fig. 5, the process initiates with coarse-grained clustering based on the connection features of encrypted traffic, followed by medium-grained clustering utilizing the SSL features, and finally a fine-grained clustering phase is conducted using the certificate features.

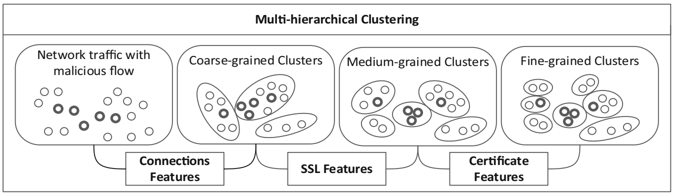


Fig. 5. The multi-hierarchical clustering process

All coarse-grained, medium-grained, and fine-grained clustering processes are completed utilizing the DPC-GS-MND algorithm, as detailed in Reference [19]. This algorithm represents an improved density peaks clustering method that incorporates grid screening, custom center decision values and mutual neighborhood degree for network anomaly detection.

4.3 Normal Encrypted Traffic Cluster Reduction

After undergoing multi-hierarchical clustering processing, the original encrypted traffic is divided into numerous small clusters, which may or may not contain attacks. Cluster analysis and reduction are aimed at filtering out the clusters that consist solely of normal encrypted traffic, while preserving the remaining clusters that contain malicious traffic, for further examination by the network intrusion detection system. This achieves the goal of data reduction, thereby further effectively improving the accuracy of attack detection.

Typically, the cluster center of each cluster is usually considered to represent the entire cluster, so existing sampling methods employ cluster center and related sampling to determine whether a cluster is malicious or normal. Here, the ENS-RFMC sampling method analyzes whether the cluster is normal or malicious by utilizing the cluster center alongside several sampling samples that are both closest and farthest from the cluster center. The cluster analysis and reduction process are shown in Fig. 6.

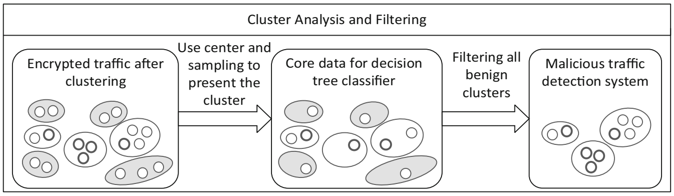

Cluster Analysis and Filtering
Encrypted traffic after clustering
Use center and sampling to present the cluster
Core data for decision tree classifier
Filtering all benign clusters
Malicious traffic detection system

Fig. 6. The cluster analysis and filtering process

After three-layer clustering, the original encrypted traffic will be classified utilizing a lightweight decision tree classifier to determine whether the cluster is malicious or normal. The classifier is trained with a small amount of training data. If the cluster center, as well as the closest and farthest sampling data from the cluster center are identified as exhibiting attack, the entire cluster will be classified as malicious and retained for further analysis in the network attack detection system. On the contrary, if all data are classified as normal traffic, it is considered that the entire cluster data can be reduced.

5 Evaluation

To comprehensively evaluate the feasibility and reduction performance of the three-stage encrypted network traffic sampling method based on rule-based feature extraction and multi-hierarchical clustering (ENS-RFMC) in complex network environments, this section constructs various experiments utilizing a real-world encrypted network traffic dataset, and compares the results. The evaluation and analysis reveal that ENS-RFMC can achieve better data reduction and malicious traffic detection results than other existing network traffic sampling methods. The experiments employed several widely used supervised learning classifiers as encrypted malicious traffic detection models, including Random Forest, XGBoost and SVM algorithms.

5.1 Experimental Data

In this study, the experimental dataset is composed of the Stratosphere IPS dataset [17] and the MTA dataset [18]. The Stratosphere IPS dataset is a research outcome implemented by the ATG group at the Czech Technical University, which contains encrypted normal traffic and malware traffic captured in a real-world network environment, with malware executing under bandwidth limitation and spam filtering restrictions. The MTA dataset focuses on sharing various types of malware traffic data for analysis, covering malware types such as ransomware and exploit kits. Given that the research objective is the detection of encrypted malicious traffic, therefore, only normal and malicious traffic encrypted using the SSL/TLS protocol were extracted from the pcap files. Finally, the dataset selected 13 benign traffic captures and 35 captures generated by four malware families, and extracted 8724 normal connection tuples and 449 malicious connection tuples from them. Table 2 lists the malware families and their distribution information in the experimental dataset.

Table 2. Malware families and distribution information.

Malware family	Malware type	Flows no	4-tuples no
Benign	Normal	69461	8724
Dridex	Trojan	62710	72
Hancitor	Ransomware	3695	247
WannaCry	Ransomware	785	34
Zeus	Trojan	17890	96

After collecting the dataset, the first step is data pre-processing. The values of most features in the dataset have varying ranges. To eliminate the dimensional relationships between network traffic and make the data comparable, we have also performed a unified normalization process. The feature values are converted into a range of [0,1] after data normalization.

5.2 Experimental Setup

Our experiments are running on Intel Xeon CPU E5–2680 @ 2.4 GHz, 64GB of RAM. Python 3.2, NetworkX 1.9.1 and Sklearn 0.24.2 running on the Ubuntu 17.04 64-bit OS, are applied as software frameworks. In addition, a Nvidia Titan GPU is used as a model training accelerator.

In the experiments, the evaluation indicators for encrypted network traffic sampling and attack detection include attack retention rate (ARR) and attack detection accuracy (ADA). They are defined as following.

$$\mathrm{ARR} = \frac{\mathrm{NAAS}}{\mathrm{NABS}} * 100\% \tag{1}$$

$$\mathrm{ADA} = \frac{\mathrm{TP}}{\mathrm{NABS}} * 100\% \tag{2}$$

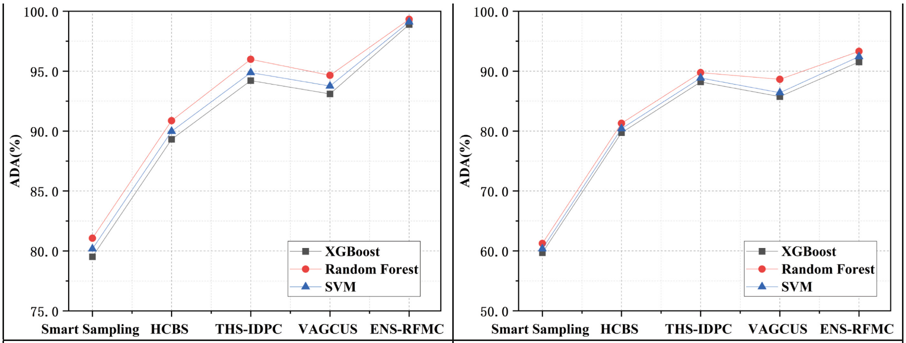
where NAAS (number of attacks after sampling) represents the number of reduced attack traffic; NABS (number of attacks before sampling) represents the number of attack traffic before sampling; TP (true positive) represents the number of samples that the detection model correctly identifies the type of attack.

5.3 Experimental Results

5.3.1 Evaluate and Comparative Experiments

In this section, smart sampling [14], HCBS [15], THS-IDPC [1] and VAGCUS [20] sampling methods are selected to compare and evaluate the effectiveness of the ENS-RFMC sampling method in data reduction. And the experiments employ Random Forest, XGBoost and SVM machine learning algorithms as backend classification models to identify attacks and evaluate performance.

Figure 7 compares the attack detection rates obtained when using XGBoost, Random Forest, and SVM algorithms as classification models at approximately 25% and 50% data reduction rates. The experimental results indicate that the random forest algorithm achieves the best attack detection effect. Therefore, the random forest algorithm is used by default for subsequent experiments. Additionally, the ENS-RFMC reduction method demonstrates the highest attack detection performance regardless of which machine learning algorithm is used as the classification model.



(a) Attack detection rate (25% reduction rate) (b) Attack detection rate (50% reduction rate)

Fig. 7. Performance comparison under different machine learning models

Table 3 demonstrates the attack retention rates with applying various data sampling techniques, as well as the attack detection effectiveness when using the random forest algorithm as the classification model, at approximately 25% and 50% data reduction rates.

From the experimental results shown in Table 3, we can clearly observe that ENS-RFMC successfully retains 99.55% and 96.21% of the attack traffic when the reduction rates are approximately 25% and 50%, respectively. This attack retention rate is significantly higher than the other four sampling methods, demonstrating the excellent performance of ENS-RFMC in reducing network traffic. As the reduction rate increases, the attack retention rate decreases. Among them, when the reduction rate is around 25%, ENS-RFMC improves the attack retention rates by about 18%, 8%, 4% and 5% compared with smart sampling, HCBS, THS-IDPC and VAGCUS reduction methods, respectively. When the reduction rate is around 50%, ENS-RFMC increases the attack retention rate by approximately 35%, 15%, 6%, and 8%, respectively. Furthermore, the experimental results also demonstrate that the ENS-RFMC method outperforms other comparison methods in terms of attack detection rate. Moreover, with the ENS-RFMC sampling method, attack detection with 25% sampling rate can achieve a level closest to 50% sampling rate.

Table 3. ARR and ADA with different data sampling methods.

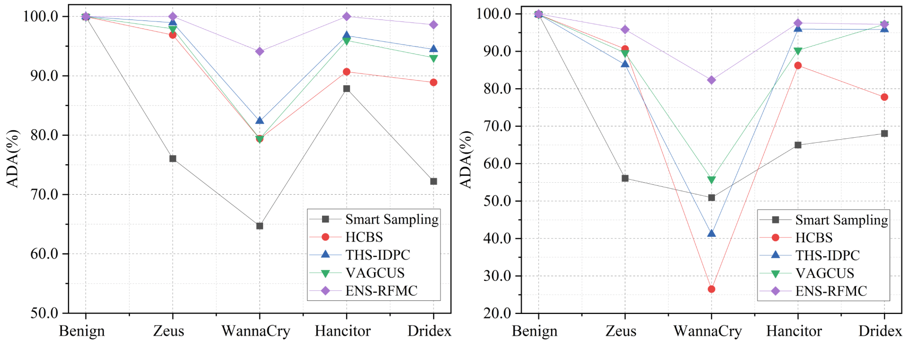
Sampling method	about 25%		about 50%	
	ARR	ADA	ARR	ADA
Smart Sampling	81.29	81.06	61.47	61.25
HCBS	91.09	90.87	81.52	81.29
THS-IDPC	95.99	95.99	89.97	89.75
VAGCUS	94.87	94.65	88.86	88.64
ENS-RFMC	**99.55**	**99.33**	**96.21**	**95.99**

Figure 8 compares the detection effectiveness of different data reduction methods for various types of attacks at approximately 25% and 50% data reduction rates. The experimental results show that, compared with smart sampling, HCBS, THS-IDPC and VAGCUS reduction methods, ENS-RFMC demonstrates the optimal attack detection effectiveness for each kind of attack.

5.3.2 Analysis of the Improved Clustering method

To verify the impact of the HDPC-GM clustering method on data reduction efficiency and attack detection performance, we compared the attack retention rate and attack detection accuracy of different clustering algorithms by using the ENS-RFMC reduction method. The comparison clustering methods include Average-linkage, DPC-DLP [21], and SDPC [22].

Figure 9 displays the results of attack retention and attack detection using HDPC-GM and comparative clustering methods when the total data reduction is approximately 25%. As can be seen in Fig. 9, HDPC-GM achieves better attack retention rate and attack



(a) Attack detection rate (25% reduction rate) (b) Attack detection rate (50% reduction rate)

Fig. 8. Attack detection rate with different data reduction methods

detection accuracy compared to the other three comparative clustering methods. This indicates that the HDPC-GM clustering method outperforms the comparative clustering methods in terms of the detection performance of encrypted traffic attacks.

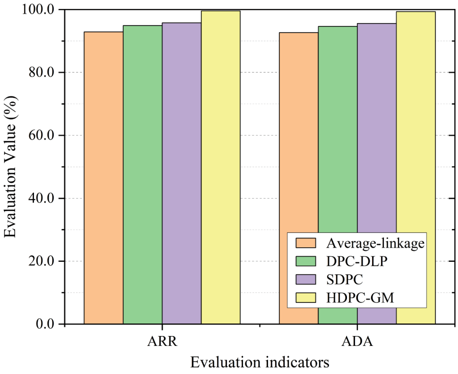


Fig. 9. With different clustering algorithms

The use of different combinations of connection features, SSL features, and certificate features in the HDPC-GM hierarchical clustering method can directly affect the clustering effect, which in turn influences the results of data reduction and attack detection. Figure 10 presents the impact of different combinations of connection features, SSL features, and certificate features on data reduction and network attack detection results using HDPC-GM when the total data reduction is approximately 50%. In the table, conn, ssl, and x509 represent connection features, SSL features, and x509 features, respectively. By comparing the experimental results of six feature combination orders, it can be seen that the first feature combination order achieves the best attack retention rate and attack detection accuracy.

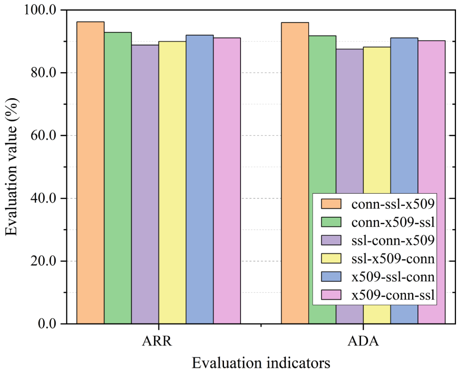


Fig. 10. With different orders of features

5.3.3 Analysis of the Improved Cluster Reduction Algorithm

The ENS-RFMC method uses the cluster center, a certain number of samples closest to the cluster center, and a certain number of samples farthest from the cluster center to determine whether the cluster is normal or malicious. Figure 11 compares the effectiveness of ENS-RFMC with the other three cluster determination methods when the total reduction is approximately 50%. Method 1: Use the cluster center to determine whether the cluster is normal or malicious. Method 2: Use the cluster center and a certain number of samples closest to the cluster center to determine whether the cluster is normal or malicious. Method 3: Use a certain number of samples closest to the cluster center and a certain number of samples farthest from the cluster center to determine whether the cluster is normal or malicious. Based on the results in Fig. 11, the cluster determination method used in this study exhibits the best performance in terms of attack retention rate and attack detection accuracy.

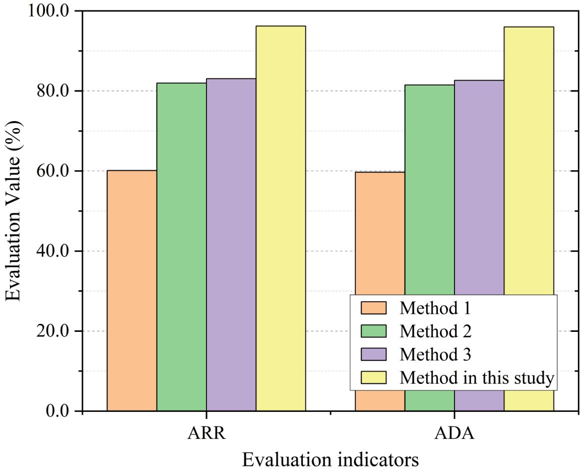


Fig. 11. Attack detection effectiveness using different cluster determination methods

Figure 12 demonstrates the changes of reduction rate under different cluster numbers. According to the experimental results, using the ENS-RFMC reduction method, when a smaller number of clusters is selected, the reduction rate of the encrypted flows gradually increases with the increase of the number of clusters. When the number of clusters approaches 50, the reduction rate begins to rapidly rise. Once the number of clusters reaches around 70, the reduction rate starts to decline dramatically, and then slowly rises and stabilizes.

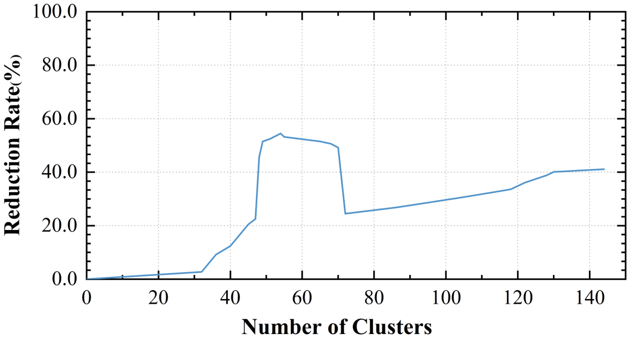


Fig. 12. Relationship between cluster number and reduction rate

6 Conclusions

In network intrusion detection, excessive normal data leads to a decrease in the storage and processing efficiency of a few attack categories of data during the processing. How to reduce encrypted traffic as much as possible without losing useful information to improve the performance of malware detection is an urgent problem in the field of encrypted malware detection. This study proposes an improved three-stage encrypted traffic sampling method for intrusion detection. The method extracts meaningful information from network connections, SSL and certificates log files, and applies a multi-hierarchical clustering method to divide the dataset into small clusters, and then removes the normal clusters. Through comparative analysis with smart sampling, HCBS, THS-IDPC and VAGCUS algorithms, experiments show that the encrypted traffic sampling method ENS-RFMC proposed in this paper can effectively reduce normal data while retaining network attack behavior features. The encrypted intrusion detection model using ENS-RFMC sampling can detect encrypted attacks more accurately and efficiently.

Acknowledgement. This research has been supported by the National Key Research and Development Program of China (No.2023YFB2603800), the National Statistical Science Research Project of China National Bureau of Statistics (No. 2022LY005), Research Funds for Key Laboratory of Network Assessment Technology of Chinese Academy of Sciences (No. KFKT2022–003), General Research Project of China University of Labor Relations (No. 23XYJS016), Teaching Reform Project of China University of Labor Relations (No. JG24037). This research was also partially supported by Key Laboratory of Network Assessment Technology, Chinese Academy of Sciences and Beijing Key Laboratory of Network Security and Protection Technology.

References

1. Chen LC, Gao S, Liu BX, et al. THS-IDPC: A three-stage hierarchical sampling method based on improved density peaks clustering algorithm for encrypted malicious traffic detection[J]. The Journal of Supercomputing, 2020, 76(9): 7489-7518
2. Hou J, Lu H, Liu FA, et al. Detection and countermeasure of encrypted malicious traffic: A survey[J]. Journal of Software, 2023, 35(1): 333-355
3. De Carné X, Mannan M. Killed by proxy: Analyzing client-end TLS interception software[C]. the 23rd Annual Network and Distributed Systems Security Symp. NDSS, 2016
4. Han J, Kim S, Ha J, et al. SGX-Box: Enabling visibility on encrypted traffic using a secure middlebox module[C]. the 1st ACM Asia-Pacific Workshop on Networking, 2017: 99–105
5. Goltzsche D, Rüsch S, Nieke M, et al. EndBox: Scalable middlebox functions using client-side trusted execution[C]. The 48th Annual IEEE/IFIP Int Conf. on Dependable Systems and Networks. 2018: 386–397
6. Justine S, Lan C, Popa RA, et al. BlindBox: Deep packet inspection over encrypted traffic[J]. ACM SIGCOMM Computer Communication Review, 2015, 45(4): 213-226
7. Lan C, Sherry J, Popa RA. Embark: Securely outsourcing middleboxes to the cloud[C]. the 13th USENIX Conf. on Networked Systems Design and Implementation. 2016: 255–273
8. B. Anderson, S. Paul, D. McGrew. Deciphering malware's use of TLS (without decryption) [J]. Journal of Computer Virology and Hacking Techniques, 2018, 14(3): 195211
9. Shekhawat AS, Troia FD, Stamp M. Feature analysis of encrypted malicious traffic[J]. Expert Systems with Applications, 2019, 125: 130-141
10. Liu JY, Zeng YZ, Shi JY, et al. MalDetect: A structure of encrypted malware traffic detection[J]. Computers, Materials & Continua, 2019, 60(2): 721-739
11. Claffy K, Polyzos G, Braun H. Application of sampling methodologies to network traffic characterization[C]. ACM SIGCOMM Comput Commun Rev. 1993, 23(4): 194-203
12. He G, Hou JC. On sampling self-similar internet traffic[J]. Computer Networks, 2006, 50 (16): 2919-2936
13. Raspall F. Efficient packet sampling for accurate traffic measurements[J]. Computer Networks, 2012, 56(6):1667-1684
14. Duffield N, Lund C. Predicting resource usage and estimation accuracy in an IP flow measurement collection infrastructure[C]. the 3rd ACM SIGCOMM conference, 2003: 179–191
15. Su L, Yao Y, Li N, et al. Hierarchical clustering based network traffic data reduction for improving suspicious flow detection[C]. The 17th IEEE TrustCom/BigDataSE Conference,
16. -753.
17. 16. Wang Z, Fok KW, Thing VLL. Machine learning for encrypted malicious traffic detection: Approaches, datasets and comparative study[J]. Computers & security, 2022,113:102542
18. Stratosphere IPS. Malware Capture Facility Project. URL https://www.stratosphereips.org/datasets-malware
19. A source for pcap files and malware samples, 2013. Retrieved March 13, 2020, from https://www.malware-traffic-analysis.net/
20. 18. Chen LC, Gao S, Liu BX. An improved density peaks clustering algorithm based on grid screening and mutual neighborhood degree for network anomaly detection[J]. Scientific Reports, 2022,12(1):1-14
21. 19. Lu Y, Chai S, Suo Y, et al. Intrusion detection for industrial internet of things based on deep learning[J]. Neurocomputing, 2024, 564(7): 126886
22. 20. Seyedi SA, Lotfi A, Moradi P, et al. Dynamic graph-based label propagation for density peaks clustering[J]. Expert Syst Appl, 2019, 115: 314-328
23. 21. Ding S, Li C, Xu X, et al. A sampling-based density peaks clustering algorithm for large-scale data[J]. Pattern Recognition, 2023, 136: 109238

An Extension of Random Forest-Clustering Schemes Which Works with Partition-Level Constraints

Manuele Bicego(✉) and Hafiz Ahmad Hassan

Computer Science Department, University of Verona, Strada le Grazie 15,
37135 Verona, Italy
manuele.bicego@univr.it

Abstract. Many classical clustering algorithms, like K-Means, spectral clustering, or hierarchical approaches, have been adapted to work with constraints; surprisingly, the literature completely lacks constrained versions of Random Forest Clustering (RFC) schemes, a class of methods whose usefulness has been shown in different scenarios. In this paper, we take one step to fill this gap, proposing a simple extension of RFC which works in the presence of partition-level constraints. In particular, the proposed approach exploits the modularity of RFC schemes, which all start from a Random Forest (RF) trained on available (unlabelled) data, by integrating in this first step the a priori knowledge given by the constraints, leaving the remaining part of the pipeline unchanged. We show the feasibility of our simple extension on three different RFC schemes, employing 18 datasets of small and moderate size. We also positively compare the obtained constrained RFCs with respect to some literature alternatives.

Keywords: Random Forest Clustering · Constrained Clustering · Decision Trees

1 Introduction

Clustering represents a widely investigated and applied exploratory data tool whose usefulness has been assessed in different scenarios (e.g., [21–23]). Clustering is definitely a challenging problem due to its unsupervised nature: actually, in clustering, no labels are available, and the natural groups (i.e., clusters) should be extracted directly from data. In some practical scenarios, however, some extra information is available, which can be used to derive better results. One example is *constrained clustering* [10,14,18], a paradigm which belongs to the widely investigated family of approaches for semi-supervised learning ([45]).

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78383-8_7.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 93–108, 2025.
https://doi.org/10.1007/978-3-031-78383-8_7

In constrained clustering, the a priori information is provided in the form of constraints. Such constraints can be of different types and work at different levels, like constraints on clusters (e.g., imposing a minimum size on clusters), on pairs of instances (e.g., knowing pairs of objects that must or must not be in the same cluster), or directly on partition (e.g., having a set of objects for which labels are known) – for more info see e.g., [14].

Due to the scientific relevance and the practical usefulness of constrained clustering solutions, many classical clustering algorithms have been adapted to the constrained case: the most striking example is the K-means algorithm [28], for which several constrained versions have been proposed, starting from the COP-Kmeans approach of [48] up to more recent versions [9,19,47]. Other examples include constrained extensions of spectral clustering [36,49], density-based clustering [24], and hierarchical clustering [11].

Surprisingly, the literature completely lacks constrained versions for Random Forest Clustering (RFC) schemes, a clustering paradigm which exploits Random Forests (RFs) [7], typically employed in supervised settings, to perform clustering [3,4,6,16,29,34,38,39,41,50,52]. This paper takes one step to fill this gap, presenting a simple extension of RFC schemes that work in the presence of partition-level constraints. We focus on the most important class of RFC approaches, which are often simply referred to as *the* Random Forest Clustering approach [3,6,16,38–40,52]. Within this paradigm, the clustering is obtained in three steps: i) a RF is created to unsupervisedly model the data; ii) a distance between objects is derived through the learned RF and iii) the final clustering is obtained through a standard distance-based algorithm, such as hierarchical clustering or spectral clustering – the differences among various RFC schemes typically lie in the definition of the RF-distance. Our starting intuition is that it is possible to straightforwardly embed the constraints in the first step of the pipeline, i.e., in the learning of the forest. Please note that in RFC the unsupervised learning of the forest is still an unsolved issue (see e.g., discussions in [3]), since no labels are available: the standard solutions imply generating a synthetic negative class, and to train a classification Forest (done e.g., in [6,39,52]), or to use Extremely Randomized Trees (ERT – [15]), which can be trained without labels (done e.g., in [3,29,41]), and have shown to be very effective in deriving powerful distances [5]. In this paper, we describe a constrained extension of these RFC methods, focusing on partition-level constraints, in which the constraints are given in terms of a subset of labels for the objects [14]. This represents a classic scenario [27], very often also used to derive instance-level constraints (i.e., Must-Link and Cannot-Link constraints, which are derived from a subset of labelled objects) – for some discussions on the different forms of constraints interested readers can refer to [10,31,35].

The proposed extension is very simple and suggests replacing the unsupervised training of the forest with a *supervised* training, done with the labelled objects, discarding all the unlabelled data. Our idea starts from the observation that, in RFC schemes, the training of RFs is problematic (no labels [3]); however, the RF is not used to directly perform clustering, but only to define the tests used to compute the distance (i.e., as a *feature extractor* – [6]); therefore an RF encoding the constraints, even if not describing all the objects of our problem, would be definitely descriptive, providing more focused tests able to characterize the different clusters.

The presented approach has been thoroughly evaluated with 18 datasets, employing 3 different versions of RFC: the original one of Shi and Horvath [39], the approach of Zhu and colleagues [52], and the very recent RatioRF method [6]. Results show that our simple extension is drastically beneficial, especially for datasets of moderate size. We also present a comparison of constrained RFCs with some literature approaches, showing that the proposed methods represent a viable alternative to standard as well as advanced constrained clustering methods.

The remainder of the paper is organized as follows: in section 2, we summarize the RFC scheme, while in section 3, we introduce the proposed extension; section 4 contains the empirical evaluation, whereas section 5 concludes the paper.

2 Random Forests and Random Forest Clustering

In this section, we will provide a brief overview of the class of RFC which we consider, starting from Decision Trees (DT) and RF – mainly to set up the notation. In the more general formulation of [8], given a vectorial representation of d features, a DT t is a *complete* binary tree, where each internal node j has associated a test $\theta_j = (\nu_j, f_j)$, with ν_j being a threshold on a feature f_j; the two children represent the two possible results of the binary test $\theta_j = (\nu_j, f_j)$: more in detail, an object $\mathbf{x} = [x_1, .., x_d]$ goes to the left child if $x_{f_j} < \nu_j$, to the right one otherwise. Given an object $\mathbf{x}$, and a tree t, let us denote as $\ell_t(\mathbf{x})$ the leaf of the tree where the object $\mathbf{x}$ falls, and as $P_t(\mathbf{x})$ denotes the set of tests on the path $\mathbf{x}$ is taking from the root to the leaf $\ell_t(\mathbf{x})$.

Typically, DTs are learned starting from a training set $\mathbf{X}$, used to determine, for all nodes j, the optimal tests $\theta_j = (\nu_j, f_j)$. More in detail, the training follows a recursive procedure: in a given node, i) the best pair (ν_j, f_j) is found according to an optimality criterion evaluated using the objects arrived at that node, and ii) the objects are propagated to the left or the right node according to the result of the test. This recursive procedure starts from the root, where all objects are used, and is recursively iterated until nodes contain a single object or a maximum depth is reached. The optimality criterion used inside a node depends on the task: for example, in classification DTs [8], where labels are available, the best rule is the one that maximizes the separability of the classes of the objects reaching that node, such as the Gini criterion. RFs [7] represent a robust ensemble of Decision Trees, obtained with a randomization mechanism in the learning of the different trees: in the typical scenario, M different Decision Trees are built starting from random subsamples of the problem training set, and the final decision is taken by averaging decisions of the different trees.

2.1 Random Forest Clustering

In recent years, a great interest has arisen in the exploitation of RFs beyond the classical regression and classification scenarios, especially in unsupervised contexts such as outlier detection [25,26] or clustering. In this last scenario, there are some approaches that exploit RFs (or RF-like schemes) to directly devise clustering algorithms, such as [4,29,34,41,50]. However, as said in the introduction, the most important trend is the so-called distance-based RF clustering (often simply referred to as *the* Random Forest Clustering method [3,6,16,38–40,52]), which exploits the data description abilities of RF to derive a *distance* between points, to be employed with classic distance-based clustering algorithm, such as spectral clustering or hierarchical clustering. In this class of approaches, clustering is performed in three steps, briefly summarized in the following.

Step 1. Training of the Random Forest. A forest is trained on the available data. Since labels are not available (clustering), unsupervised methods should be derived. The typical options are two:

- **Negative Sampling**: in this option, we train a classical classification forest, e,g, based on CART [8], in which the set of points to be clustered represents the positive class, while the negative class is synthetically generated. This last step is typically performed by random sampling from the product of empirical marginal distributions of the dataset, in order to create a negative class of the same size of the dataset. This training scheme represents the first and most employed option in RFC – e.g. used in [6,16,38–40,52].
- **Extremely Randomized Trees**: in this option, less investigated than the previous one, a forest of Extremely Randomized Trees [15] is used. Within these trees, randomization is taken to the extreme: in every node, the rule is found by randomly selecting a feature, and then by selecting the threshold by uniformly sampling a value from the domain of the objects of the training set arrived at that node. Examples of RFC methods using this option can be found in [3,29,41].

Step 2. Distance computation. In this step, a distance between all pairs of objects to be clustered is computed through the learned RF. The different RFC schemes typically differ in the way this distance is computed, exploiting different concepts like path overlaps [39,52], probability masses [1,42] or axiomatic definitions of similarity [6]. However, in all cases, the idea is that a good measure of similarity between two objects can be defined by comparing the way they answer to the tests of the trees of the forest. The typical scheme is to define a similarity on a single tree, to be aggregated and transformed to distance at the Forest level. In more detail, the general formulation of a RF-distance, defined on a forest with T trees, is:

$$d^{RF}(\mathbf{x}, \mathbf{y}) = \sqrt{1 - \sum_{t=1}^{T} s^t(\mathbf{x}, \mathbf{y})} \tag{1}$$

where $s^t(\mathbf{x}, \mathbf{y})$ defines the tree-similarity between $\mathbf{x}$ and $\mathbf{y}$. Here we studied the following tree similarities, which lead to different RFC schemes:

- **RFC-Shi**. In this case the similarity between two objects $\mathbf{x}$ and $\mathbf{y}$ in a tree t of the forest is defined as:

$$s^t_{Shi}(\mathbf{x},\mathbf{y}) = \begin{cases} 1 & \text{if}\ell_t(\mathbf{x}) = \ell_t(\mathbf{y}) \\ 0 & \text{if}\ell_t(\mathbf{x}) \neq \ell_t(\mathbf{y}) \end{cases} \tag{2}$$

  This similarity, aggregated at the forest level, measures the number of times, over the total number of trees of the forest, the two objects reach the same leaf – i.e., they provide the same answers to all questions in the path. This represents the original measure proposed by [7], firstly used for clustering in [39].
- **RFC-Zhu**. This represents a tree-similarity introduced in [52][1]:

$$s^t_{Zhu}(\mathbf{x},\mathbf{y}) = \frac{\mathrm{depth}(\mathrm{lca}(\ell_t(\mathbf{x}),\ell_t(\mathbf{y}))}{\max\{\mathrm{depth}(\ell_t(\mathbf{x})),\mathrm{depth}(\ell_t(\mathbf{y}))\}} \tag{3}$$

  where $\mathrm{lca}(\ell_t(\mathbf{x}),\ell_t(\mathbf{y}))$ is the *least common ancestor* of $\ell_t(\mathbf{x})$ and $\ell_t(\mathbf{y})$. In this case, the idea is that the larger the overlap between the two paths $\mathbf{x}$ and $\mathbf{y}$ are following in the tree t, the larger their similarity.
- **RFC-RatioRF**. This represents the very recent RatioRF measure, introduced in [6] by following the axiomatic definition of similarity given by Tversky [44]. This approach has been shown to outperform all the alternatives in the clustering scenario [6], also in the presence of missing data [37]. In this case:

$$s^t_{RatioRF}(\mathbf{x},\mathbf{y}) = \frac{|P_t(\mathbf{x}) \mathbin{\dot{\cap}} P_t(\mathbf{y})|}{|P_t(\mathbf{x}) \mathbin{\dot{\cap}} P_t(\mathbf{y})| + |P_t(\mathbf{x}) \mathbin{\dot{-}} P_t(\mathbf{y})| + |P_t(\mathbf{y}) \mathbin{\dot{-}} P_t(\mathbf{x})|} \tag{4}$$

  where i) $P_t(\mathbf{x})$ denotes the set of tests on the path of $\mathbf{x}$, $P_t(\mathbf{x}) \mathbin{\dot{\cap}} P_t(\mathbf{y})$ is the set of tests on which $\mathbf{x}$ and $\mathbf{y}$ agree, among the tests in $P_t(\mathbf{x}) \cup P_t(\mathbf{y})$, and ii) $P_t(\mathbf{x}) \mathbin{\dot{-}} P_t(\mathbf{y})$ (or, equivalently, $P_t(\mathbf{y}) \mathbin{\dot{-}} P_t(\mathbf{x})$) is the set of tests on the path of $\mathbf{x}$ ($\mathbf{y}$) on which $\mathbf{y}$ ($\mathbf{x}$) disagrees.

Step 3. Clustering. The final clustering is then obtained using any distance-based clustering algorithm, such as Spectral Clustering.

Let us conclude this section with a couple of very general considerations on the complexity of the RFC scheme. The computational load of the first step mainly depends on the size of the dataset, which, within the Negative Sampling, is also doubled; however, this represents a classic and widely studied issue with classification RFs, and very fast options have been proposed – e.g., [43]. When using ERTs, however, training is very fast, since no optimization is required. The second step (i.e. the computation of the RF-similarities) typically represents the bottleneck of RFC schemes, since the similarity should be determined for all pairs of objects to be clustered, and this quadratic complexity often makes the scaling

[1] Among the three variants proposed in [52], we employed here the second one, representing the best compromise between clustering accuracy and computational requirements, as also suggested in [3].

to very large datasets too computationally demanding. For what concerns the computation of a single similarity, the complexity mainly depends on the depth of the tree (since objects have to traverse all trees): however, experiments in [2] have shown that very short trees (with max depth 7 or 8) already permit to get excellent results. Given the distance, a standard distance-based method is used in the third step; the optimization of such methods is again a widely studied topic, with many optimized variants already present in the literature – e.g. [51].

3 The proposed approach

In this section, we introduce the variant of the RFC schemes described in the previous section, which permits the integration of partition-level constraints. Let us start with the definition of the problem, which follows [27]. Given a dataset $\mathbf{X} = \{\mathbf{x}_1, \cdots \mathbf{x}_N\}$ ($\mathbf{x}_i \in \Re^d$) to be clustered in K clusters, and a value $p \in (0, 1)$, let us define the p-partition level constraints as a set of labels from 1 to K, assigned by an expert, to pN objects of $\mathbf{X}$. Let us define as $\mathbf{X}^C \subset \mathbf{X}$ these objects, Y^C the corresponding labels, and $\mathbf{X}^U \subset \mathbf{X}$ the remaining objects. Clearly $|\mathbf{X}^C| = pN, |\mathbf{X}^U| = (1-p)N$, and $\mathbf{X}^U \cup \mathbf{X}^C = \mathbf{X}$. The goal is to cluster the objects in $\mathbf{X}$ in K clusters, starting from the representation $\mathbf{Z} = \{\mathbf{X}^U, \mathbf{X}^C, Y^C\}$.

In our constrained modification of the RFC, the main intuition is that constraints can be easily integrated into the first phase of the pipeline, namely in the learning phase. Our modification is extremely simple, but permits us to get promising results in the experiments. We start from two observations. The first is that the trained forest is not used for clustering, but simply to derive the distance measure for clustering: actually, as described in the previous section, the distance is computed by letting all objects in $\mathbf{X}$ traverse trees of the RF, subsequently comparing their paths. In this sense, the forest can also be trained with a subset of objects, or, in principle, also with objects from a hold-out set. Second observation: if we consider the basic version of the distance [39], where two objects are similar if they end in the same leaf in several trees of the forest, it is clear that a good forest is composed of trees in which objects belonging to different clusters follow different paths, ending in different leaves (by the way, this is true also for other RF measures). But this is actually what is looked for when building a RF with classification trees like CART [8], i.e. to get trees in which objects of different classes are grouped in different parts of the trees. Putting these two facts together, we have our simple approach to constrained RFC: to train a classification RF using only $\{\mathbf{X}^C, Y^C\}$, and to leave steps 2 and 3 of the RFC pipeline unchanged. The proposed approach is sketched in Fig. 1: in red the novel ingredients with respect to the standard RFC.

Let us stress that working on the learning stage of the RFC pipeline has different advantages: i) we can work on the weakest part of the RFC pipeline, for which a well-established option is still missing; ii) it seems reasonable that inserting the a priori information in the early stage of the pipeline would be more beneficial than inserting in later stages: actually, with better forests –

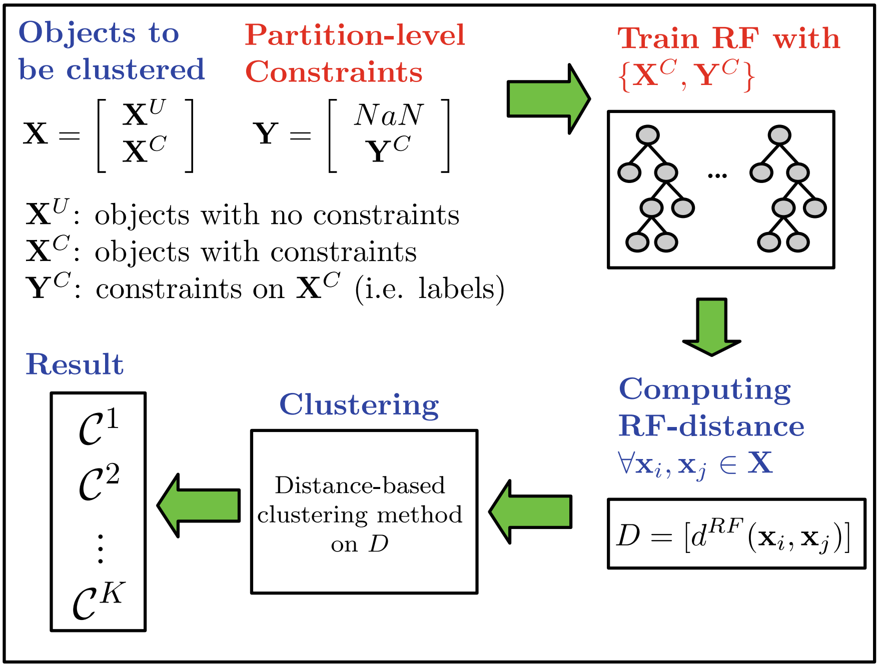

Objects to be clustered
$\mathbf{X} = \begin{bmatrix} \mathbf{X}^U \\ \mathbf{X}^C \end{bmatrix}$
Partition-level Constraints
$\mathbf{Y} = \begin{bmatrix} NaN \\ \mathbf{Y}^C \end{bmatrix}$
$\mathbf{X}^U$: objects with no constraints
$\mathbf{X}^C$: objects with constraints
$\mathbf{Y}^C$: constraints on $\mathbf{X}^C$ (i.e. labels)
Train RF with $\{\mathbf{X}^C, \mathbf{Y}^C\}$
...
Computing RF-distance $\forall \mathbf{x}_i, \mathbf{x}_j \in \mathbf{X}$
$D = [d^{RF}(\mathbf{x}_i, \mathbf{x}_j)]$
Clustering
Distance-based clustering method on D
Result
$\mathcal{C}^1$
$\mathcal{C}^2$
$\vdots$
$\mathcal{C}^K$

Fig. 1. The proposed approach. In red the modification introduced with respect to the standard RFC scheme.

built exploiting the constraints – we would have better distances, which would lead to a better clustering result; iii) the computational overhead is very limited, and related only to training; indeed, with our approach, we are actually *reducing* the computational overhead with respect to the unconstrained case of "Negative Sampling", since we are training a classification forest with a reduced number of points (only pN objects, whereas for Negative Sampling we need $2N$ objects, since typically the generated negative class has the same cardinality of the dataset).

4 Experimental evaluation

In this section, the proposed approach is evaluated and compared with some alternatives. The evaluation is based on 18 datasets, which are listed in Tab. 1. The full description of these datasets, together with the source and the preprocessing, is provided in the Section B of the supplementary material. We considered datasets of both small and moderate size (these latter marked with an "(*)" in Tab. 1), which represents the situation where tree-based approaches have been shown to be most useful [17]. As done in most of the clustering experiments, the evaluation is done by comparing the cluster assignment with the true labels. In particular, we used the standard Adjusted Rand Index (ARI – [20]), which employs a contingency table between the obtained grouping and the true one to quantify the quality of the result (the higher this index, the better the clustering), also performing a correction to consider the chance of the formation of the clusters.

Table 1. Datasets used for evaluation. The set of datasets denoted with "(*)" represents the "moderate size" datasets.

Name	Objects	Features	Clusters
Iris	150	4	3
Glass	214	9	4
Ecoli	336	7	8
Seeds	210	7	3
Cryotherapy	90	6	2
AutoMpg	398	6	2
Imox	192	8	4
StoneFlakes	70	8	3
Pima	768	8	2
Nose	358	128	5
Microarray	90	100	5
Flickr(*)	1000	87	5
Isolet(*)	1200	617	4
EEGEyeState(*)	1280	14	2
UAVIntDet-1(*)	1100	54	2
UAVIntDet-2(*)	1100	54	2
UAVIntDet-3(*)	1100	54	2
UAVIntDet-4(*)	1100	18	2

Concerning RFC approaches, we considered the three different schemes described in Section 2.1, namely RFC-Shi, RFC-Zhu, and RFC-RatioRF, evaluating their extension to the constrained case. In all experiments, we used forests with 100 trees, sampling 50% of features in each node, and building each tree using a random 50% of the training set – for datasets of moderate size, we followed suggestions in [6], using 128 random objects for each tree. Each tree was built until its maximum depth; finally, for classification trees we used the Gini Criterion. Given the distance, the final clustering is obtained with Spectral Clustering using the Ng-Jordan-Weiss normalized version [46], and repeating the inner k-means 20 times, as done in most of the RFC schemes [3,6,52]. The Matlab code of the proposed approach is available at https://profs.scienze.univr.it/~bicego/code.html.

Constrained clustering experiments were performed following the classic protocol for partition level constraints [27]: for a given level of supervision p, we

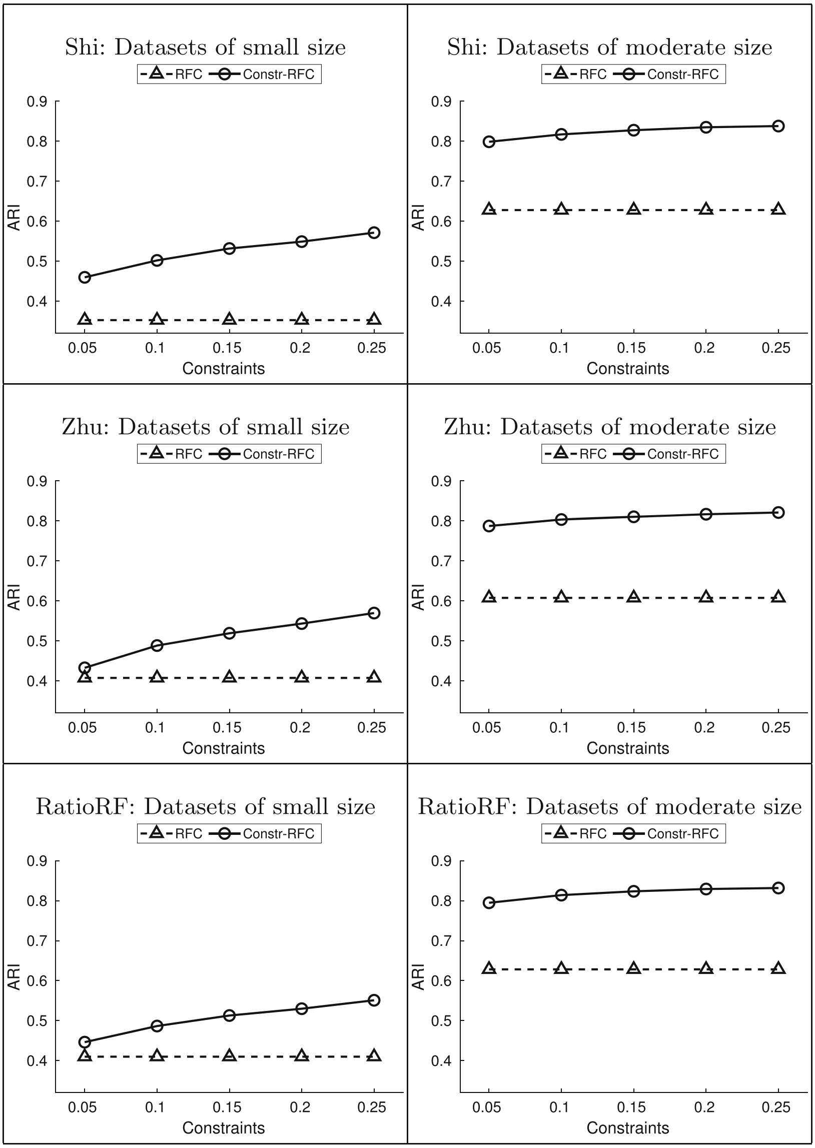


Fig. 2. Comparison between unconstrained and constrained strategies for different RF Clustering schemes: first row: average over datasets of small size, second row: average over datasets of moderate size.

randomly sampled pN objects from the dataset, considering their labels as the constraints. We investigated different levels of p from 0.05 to 0.25, with step 0.05, and we repeated the whole pipeline 100 times. In each of the 100 runs, the constraints set was kept fixed and used as input to all the constrained versions.

In the following, after reporting the comparison of the constrained versions of RFC with the corresponding unconstrained versions, we compare the proposed schemes with some literature alternatives in terms of both the ARI index and the Negative Effects of Constraints (NEC) ratio [12].

4.1 Comparison with the unconstrained RFC

The comparison with the unconstrained RFC is reported in Fig. 2. Due to lack of space, we reported here only the averages over the datasets of small size (first row) and of moderate size (second row), leaving the results dataset per dataset to the Section C of the supplementary material. In the figure, the three columns represent the results with the RFC-Shi, RFC-Zhu, and RFC-RatioRF, respectively. In every plot, we report the average of the ARI criterion (the higher, the better) over the 100 repetitions and over the group of datasets.

From the plots, it is evident that the exploitation of the constraints is, on average, very beneficial for the RFC schemes, especially for datasets of moderate size. To assess the statistical significance of the comparison reported in the Figure, we performed a paired t-test, comparing, for each RF scheme, and each level of constraints, all the repetitions over the different groups of datasets of the constrained version with the corresponding unconstrained scheme. All tests reported a statistically significant difference, according to a level of 0.05.

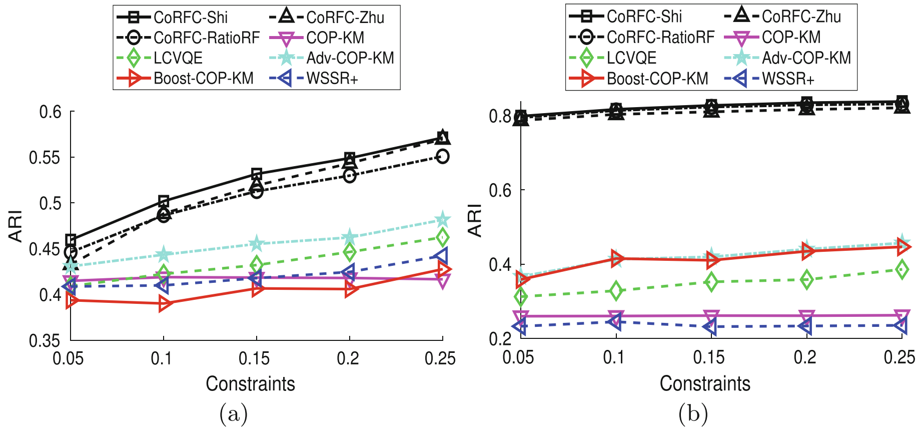


Fig. 3. Comparison between the proposed approach and some alternatives in terms of ARI for different levels of constraints: a) average over small datasets, b) average over datasets of moderate size.

4.2 Comparison with other standard constrained clustering methods

To get an idea of the potentialities of the proposed approach, we compare our constrained RFC schemes (RFC-Shi, RFC-Zhu, and RFC-RatioRF) with a few standard literature alternatives. In particular, we first considered some standard and widely used algorithms: two versions of the COP-Kmeans algorithm, namely **COP-KM**, the original extension of [48] and **Adv-COP-KM**, a more advanced one proposed in [30]); another standard approach, i.e., the **LCVQE** method of [32]; then we considered some more advanced and recent approaches, having some far commonalities with our approach: i) **Boost-COP-KM**, a recent algorithm based on ensemble learning (as RF), introduced in [30]; ii) **WSSR+**, a very recent constrained spectral clustering algorithm, proposed in [33], which learns a similarity measure from the constraints (as in RFC), to be clustered with spectral clustering (which we used in our experiments). We evaluate all these methods with the same protocol, i.e., computing the ARI values; to have a fair comparison, in all repetitions, the starting point was the same: the dataset and the same constraints used for our constrained RFC methods. The description of these methods, together with the implementation details, is provided in the Section D of the supplementary material. Comparative results are shown in Figure 3. Also in this case we show results averaged over datasets of small (part (a) of fig. 3) and moderate (part (b)) size, whereas the complete results, dataset per dataset, are reported in section D of the supplementary material.

It is interesting to observe that the constrained RFC-schems, on average, outperform alternatives, especially when working with large datasets and with a large level of supervision. This is somehow expected since the labels of the constraints are used to train the initial forest, which is then used to derive the distance used for clustering: the larger this Set, the better the forest is trained. In order to assess the statistical significance of the comparison, we employ a Friedman test followed by a post-hoc Nemenyi test. The Friedman test represents a common non-parametric test, employed when comparing more than two approaches [13]. The test is based on the ranking of the compared approaches, and permits the assessment of the presence of a significant difference among them; if the null hypothesis is rejected, a Nemenyi test is applied, assessing via a critical value which pairs of methods are different with a statistical significance. The critical value represents the required minimum difference between the ranks of the approaches. The final output of the whole procedure can be represented via a critical diagram [13], which shows the ranks (from highest to lowest): a line connecting two or more methods indicates that there is not a statistically significant difference between them. We applied this procedure to the comparison between the constrained RFC schemes and the literature alternatives, using as significance level $\alpha = 0.05$. The obtained critical diagram is shown in Fig. 4(a), from which it can be observed that the three constrained RFC schemes outperform all the alternatives with a statistical significance.

To have a better understanding of the proposed approach, also in comparison with literature alternatives, we provide here an analysis of the so-called negative effects of constraints (NEC). As firstly hypothesized by [12], it may be the case that the introduction of constraints does not lead to an improvement of the clustering, reducing the actual accuracy of the clustering results. Following [12], we quantified such effect by counting how many times, among the 100 repetitions,

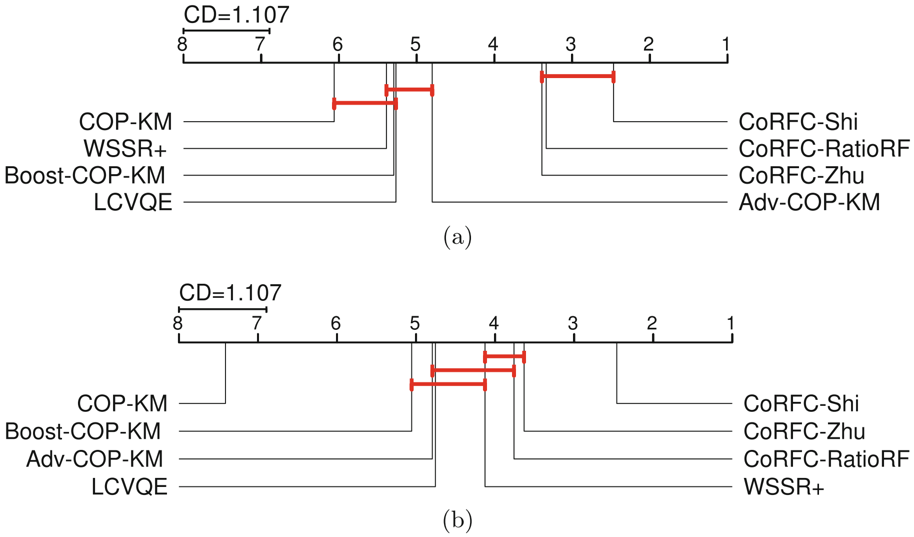
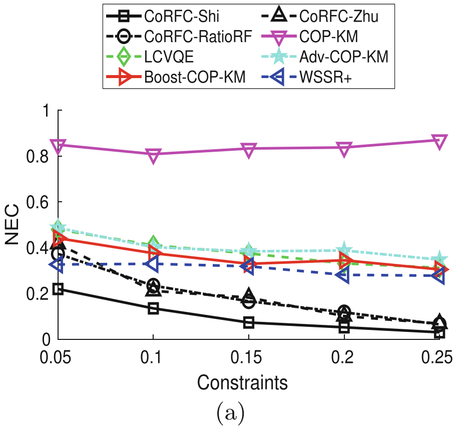


Fig. 4. Statistical analysis of the comparison between the proposed approach and some alternatives: a) critical diagram for ARI, b) critical diagram for NEC.

using the constraints does not improve the clustering accuracy with respect to not using them. This measure, which we refer to as *NEC*, is aimed at measuring how effective are the different constrained clustering methods in exploiting the constraints so that there is an increase in the clustering accuracies.

The results are shown in Figure 5, for our RFC schemes and for the literature alternatives. Please note that the lower this measure, the better the exploitation of the constraints. Also in this case we show results averaged over datasets of small (part (a) of fig. 5) and moderate (part (b)) size, whereas detailed results dataset per dataset are reported in section D of the supplementary material. Also in this case, we can observe that the proposed approach compares reasonably well with literature alternatives, especially for large levels of supervision and for datasets of small size.

Also in this case we analyzed the statistical significance of the results via a critical diagram, shown in part (b) of Fig. 4. CoRFC-Shi is better than all alternatives with a statistical significance, whereas CoRFC-RatioRF and CoRFC-Zhu are equivalent to the very recent WSSR+. It is interesting to note that the best constrained variant is obtained when using the oldest RFC scheme (with the Shi-Breiman distance); in the unconstrained case, RatioRF and Zhu outperform Shi, as also confirmed in other works [6,52]. Probably, when a better RF, trained by exploiting the constraints, is provided, the simple Shi-Breiman distance with the 0/1 mechanism is very adequate, and more sophisticated strategies, like RatioRF or Zhu, may introduce overtraining in the whole process. On the contrary, without constraints, worse RFs are derived, and more clever strategies should be used. This is confirmed by the theoretical characterization of [5], which shows



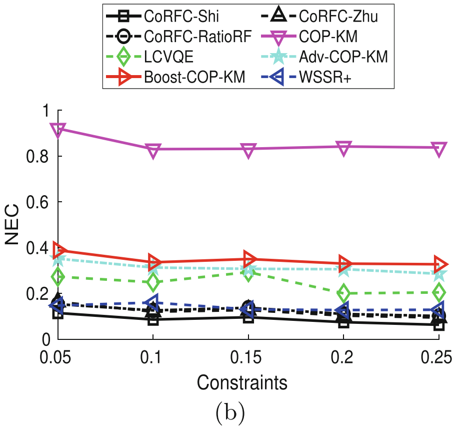


Fig. 5. Comparison between the proposed approach and some alternatives in terms of NEC ratio for different levels of constraints: a) average over small datasets, b) average over datasets of moderate size.

that, when starting from Forest built with completely random trees (Extremely Randomized Trees), the RatioRF distance has a better theoretical behaviour (i.e., better bounds) than the Shi-Breiman distance.

5 Conclusions

In this paper, we presented a simple approach to extend RFC schemes to work with partition-level constraints. The approach is very straightforward, simply modifying the first (and weakest) step of the RFC scheme. Obtained results are promising, showing that i) constraints can be easily and fruitfully integrated into different RFC schemes and ii) the obtained constrained RFC schemes represent a valid alternative to standard as well advanced constrained clustering approaches. Future works will include in the experimental evaluation larger real-world datasets and applications, in order to show the practical relevance of the proposed method.

References

1. Aryal, S., Ting, K., Washio, T., Haffari, G.: A comparative study of data-dependent approaches without learning in measuring similarities of data objects. Data Min. Knowl. Disc. **34**(1), 124–162 (2020)
2. Bicego, M., Cicalese, F., Mensi, A.: RatioRF: A novel measure for random forest clustering based on the tversky's ratio model. IEEE Tr. on Knowledge and Data Engineering **35**(1), 830–841 (2023)
3. Bicego, M., Escolano, F.: On learning random forests for random forest-clustering. In: Proc. Int. Conf. on Pattern Recognition. pp. 3451–3458. IEEE (2021)

4. Bicego, M.: K-random forests: a k-means style algorithm for random forest clustering. In: Proc. Int. Joint Conf. on Neural Networks. pp. 1–8. IEEE (2019)
5. Bicego, M., Cicalese, F.: On the good behaviour of extremely randomized trees in random forest-distance computation. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 645–660. Springer (2023)
6. Bicego, M., Cicalese, F., Mensi, A.: RatioRF: a novel measure for random forest clustering based on the Tversky's ratio model. IEEE Trans. Knowl. Data Eng. **35**(1), 830–841 (2023)
7. Breiman, L.: Random forests. Mach. Learn. **45**, 5–32 (2001)
8. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth (1984)
9. Covoes, T.F., Hruschka, E.R., Ghosh, J.: A study of k-means-based algorithms for constrained clustering. Intelligent Data Analysis **17**(3), 485–505 (2013)
10. Davidson, I., Basu, S.: A survey of clustering with instance level constraints. ACM Trans. on Knowledge Discovery from data **1**(1-41), 2–42 (2007)
11. Davidson, I., Ravi, S.: Agglomerative hierarchical clustering with constraints: Theoretical and empirical results. In: Proc. Europ. Conf. on Principles of Data Mining and Knowledge Discovery. pp. 59–70 (2005)
12. Davidson, I., Wagstaff, K.L., Basu, S.: Measuring constraint-set utility for partitional clustering algorithms. In: Proc. Europ. Conf. on Principles of Data Mining and Knowledge Discovery. pp. 115–126 (2006)
13. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. The Journal of Machine Learning Research **7**, 1–30 (2006)
14. Gançarski, P., Dao, T.B.H., Crémilleux, B., Forestier, G., Lampert, T.: Constrained clustering: Current and new trends. A Guided Tour of Artificial Intelligence Research: Volume II: AI Algorithms pp. 447–484 (2020)
15. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Mach. Learn. **63**(1), 3–42 (2006)
16. Gray, K.R., Aljabar, P., Heckemann, R.A., Hammers, A., Rueckert, D.: Random forest-based similarity measures for multi-modal classification of alzheimer's disease. Neuroimage **65**, 167–175 (2013)
17. Grinsztajn, L., Oyallon, E., Varoquaux, G.: Why do tree-based models still outperform deep learning on typical tabular data? Adv. Neural. Inf. Process. Syst. **35**, 507–520 (2022)
18. Grossi, V., Romei, A., Turini, F.: Survey on using constraints in data mining. Data Min. Knowl. Disc. **31**, 424–464 (2017)
19. Hong, Y., Kwong, S.: Learning assignment order of instances for the constrained k-means clustering algorithm. IEEE Trans. on Systems, Man, and Cybernetics, Part B (Cybernetics) **39**(2), 568–574 (2008)
20. Hubert, L., Arabie, P.: Comparing partitions. Journal of Classification pp. 193–218 (1985)
21. Jain, A.K.: Data clustering: 50 years beyond k-means. Pattern Recogn. Lett. **31**(8), 651–666 (2010)
22. Jain, A.K., Dubes, R.C.: Algorithms for clustering data. Prentice-Hall, Inc. (1988)
23. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM computing surveys (CSUR) **31**(3), 264–323 (1999)
24. Lelis, L., Sander, J.: Semi-supervised density-based clustering. In: Proc. Int. Conf. on Data Mining. pp. 842–847 (2009)
25. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation forest. In: Proc. Int. Conf. on Data Mining. pp. 413–422 (2008)

26. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation-based anomaly detection. ACM Trans. on Knowledge Discovery from Data (TKDD) **6**(1), 1–39 (2012)
27. Liu, H., Fu, Y.: Clustering with partition level side information. In: Proc. Int. Conf. on Data Mining. pp. 877–882 (2015)
28. Lloyd, S.: Least squares quantization in pcm. IEEE Trans. on information theory **28**, 129–137 (1982)
29. Moosmann, F., Triggs, B., Jurie, F.: Fast discriminative visual codebooks using randomized clustering forests. In: Advances in neural information processing systems. pp. 985–992 (2006)
30. Okabe, M., Yamada, S.: Clustering using boosted constrained k-means algorithm. Frontiers in Robotics and AI **5**, 18 (2018)
31. Pei, Y., Fern, X.Z., Tjahja, T.V., Rosales, R.: Comparing clustering with pairwise and relative constraints: A unified framework. ACM Trans. on Knowledge Discovery from Data **11**(2), 1–26 (2016)
32. Pelleg, D., Baras, D.: K-means with large and noisy constraint sets. In: Proc. European Conference on Machine Learning. pp. 674–682 (2007)
33. Peng, H., Pavlidis, N.G.: Weighted sparse simplex representation: a unified framework for subspace clustering, constrained clustering, and active learning. Data Min. Knowl. Disc. **36**(3), 958–986 (2022)
34. Perbet, F., Stenger, B., Maki, A.: Random forest clustering and application to video segmentation. In: Proc. of British Machine Vision Conference. pp. 1–10 (2009)
35. Qian, P., Jiang, Y., Wang, S., Su, K.H., Wang, J., Hu, L., Muzic, R.F.: Affinity and penalty jointly constrained spectral clustering with all-compatibility, flexibility, and robustness. IEEE Trans. on neural networks and learning systems **28**(5), 1123–1138 (2016)
36. Rangapuram, S.S., Hein, M.: Constrained 1-spectral clustering. In: Artificial Intelligence and Statistics. pp. 1143–1151. PMLR (2012)
37. Raniero, M., Bicego, M., Cicalese, F.: Distance-based random forest clustering with missing data. In: Proc. Int. Conf. on Image Analysis and Processing. pp. 121–132. Springer (2022)
38. Rennard, S.I., Locantore, N., Delafont, B., Tal-Singer, R., Silverman, E.K., Vestbo, J., Miller, B.E., Bakke, P., Celli, B., Calverley, P.M., et al.: Identification of five chronic obstructive pulmonary disease subgroups with different prognoses in the eclipse cohort using cluster analysis. Ann. Am. Thorac. Soc. **12**(3), 303–312 (2015)
39. Shi, T., Horvath, S.: Unsupervised learning with random forest predictors. J. Comput. Graph. Stat. **15**(1), 118–138 (2006)
40. Shi, T., Seligson, D., Belldegrun, A., Palotie, A., Horvath, S.: Tumor classification by tissue microarray profiling: Random forest clustering applied to renal cell carcinoma. Mod. Pathol. **18**, 547–557 (2005)
41. Shotton, J., Johnson, M., Cipolla, R.: Semantic texton forests for image categorization and segmentation. In: Proc. Int. Conf. on Computer Vision and Pattern Recognition. pp. 1–8 (2008)
42. Ting, K., Zhu, Y., Carman, M., Zhu, Y., Zhou, Z.H.: Overcoming key weaknesses of distance-based neighbourhood methods using a data dependent dissimilarity measure. In: Proc. Int. Conf. on Knowledge Discovery and Data mining. pp. 1205–1214 (2016)
43. Tiwari, M., Kang, R., Lee, J., Piech, C., Shomorony, I., Thrun, S., Zhang, M.J.: Mabsplit: Faster forest training using multi-armed bandits. Adv. Neural. Inf. Process. Syst. **35**, 1223–1237 (2022)
44. Tversky, A.: Features of similarity. Psychol. Rev. **84**(4), 327 (1977)

45. Van Engelen, J.E., Hoos, H.H.: A survey on semi-supervised learning. Mach. Learn. **109**(2), 373–440 (2020)
46. von Luxburg, U.: A tutorial on spectral clustering. Stat. Comput. **17**(4), 395–416 (2007)
47. Vouros, A., Vasilaki, E.: A semi-supervised sparse k-means algorithm. Pattern Recogn. Lett. **142**, 65–71 (2021)
48. Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S.: Constrained k-means clustering with background knowledge. In: Proc. Int. Conf. on Machine Learning. vol. 1, pp. 577–584 (2001)
49. Wang, X., Qian, B., Davidson, I.: On constrained spectral clustering and its applications. Data Min. Knowl. Disc. **28**, 1–30 (2014)
50. Yan, D., Chen, A., Jordan, M.: Cluster forests. Computational Statistics & Data Analysis **66**, 178–192 (2013)
51. Zhu, W., Nie, F., Li, X.: Fast spectral clustering with efficient large graph construction. In: 2017 IEEE international conference on acoustics, speech and signal processing (ICASSP). pp. 2492–2496. IEEE (2017)
52. Zhu, X., Loy, C., Gong, S.: Constructing robust affinity graphs for spectral clustering. In: Proc. Int. Conf. on Computer Vision and Pattern Recognition. pp. 1450–1457 (2014)

Enhancing Protein Classification with Graph Convolutional Neural Networks

Abderrahim Mechache(✉) and Hamamache Kheddouci

Laboratoire LIRIS, UMR 5205 CNRS, Université Claude Bernard Lyon 1,
69100 Villeurbanne, France
{abderrahim.mechache,hamamache.kheddouci}@univ-lyon1.fr

Abstract. Proteins are essential components of our lives. This has made their classification a very active research field. Most research focuses on classifying proteins based on their structure, overlooking the surface, which is a challenging aspect of research due to its complexity. However, by relying only on the structural aspect, relevant information will be lost. The main objective of this paper is to propose a new machine learning method for protein classification using Graph Convolutional Neural Networks (GCNNs) applied to 3D files. The use of GCNNs will enable us to efficiently combine these informations with structure information extracted from Protein Data Bank (PDB) files to improve protein representation and enhance accuracy, and then train a classifier to organize and classify the proteins. Additionally, we devide our system into three different modules to ensure the interpretation of the classification results, and ensuring that it works even when one type of information is missing. Our method achieves an average accuracy of 94% on the SHREC19 dataset, and also demonstrates efficient execution times. Finally, we employed a feature importance technique to highlight the most relevant information for deeper insight.

Keywords: Protein Classification · bioinformatics · Machine Learning · Protein Data Bank (PDB) · Graph Convolutional Neural Networks (GCNNs)

1 Introduction

Proteins are vital components found in every aspect of life, from the simplest to the most complex organisms. They play a key role in our health, food, and medicine. The significance of proteins has made protein research a very active field, enhancing our understanding and providing new solutions to simplify our lives. Proteins are often grouped into families, each consisting of proteins that share similar characteristics such as topology and function. In the SCOP database [1], existing proteins are grouped into families with hierarchical levels, facilitating analysis to produce new medicinal solutions. For example, grouping similar viruses helps us to understand their behavior, so we can suggest drugs for untreated viruses based on drugs used for others in the group.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 109–124, 2025.
https://doi.org/10.1007/978-3-031-78383-8_8

Proteins can be grouped using two primary methods: structural and surface-based. Structural methods establish relationships and hierarchies among proteins based on their three-dimensional structures, such as amino acid sequences, molecular weight, and atomic coordinates, simplifying their organization. On the other hand, surface-based methods focus on classifying proteins by their surface characteristics. This includes analyzing the contours and chemical properties of the surface, which play pivotal roles in molecular interactions and biological functionality. However, Relying solely on surface or structural information may be insufficient. Focusing only on structural similarities can miss important differences on their surfaces that matter for how they function, potentially causing mistakes by grouping proteins that are structurally similar but functionally different. On the other hand, grouping proteins based on surface similarity can include proteins that appear similar externally but are functionally different. For example, some proteins might have similar surface features but perform different functions in the body. This could lead to errors in grouping them together. Additionally, combining both types of data is a complex task in terms of resources, as it typically requires the use of two different architectures, one for each type of information.

In this work, we propose a new approach that takes both structural and surface information into account. We first represent the protein by its surface and use a Graph Convolutional Neural Networks (GCNNs) to extract the information relevant to classification. Secondly, we enrich this representation with information from the protein's structure and run the whole through a deep learning classifier, thus guaranteeing the use of a more complete representation of the protein. By using GCNNs in this way, we eliminate the need for two separate modules (one for surface information and one for structural information), making the process more efficient.

In the second section, we review the state-of-the-art approaches in the literature, then we present our approach in detail in Section 3. The implementation details used to validate our approach are presented in Section 4, and the results obtained are discussed in Section 5. We end the paper with a conclusion and some future works.

2 Background

In the literature, grouping proteins can be divided into structure-based classifications and surface-based classifications.

In the first classification type, proteins are classified according to their structures. Generally, proteins are represented in textual files such as PDB files found in the Protein Data Bank [2], which contain various information on the protein's structure, such as the 3D coordinates of atoms, the types of residues, the sequence of amino acids, secondary structure elements like alpha helices and beta sheets, and the chemical bonds between atoms. These files are often not directly usable by computational methods, but can be made usable through techniques that create several types of exploitable representations, such as linear representations from which structural information (also called descriptors) is extracted and presented in a vector [3]. One such method is TM-Align [4], which identifies the best structural alignment between protein pairs independently from their sequences. It generates an optimized residue-to-residue alignment based on structural similarity to calculate the TM-score [5] to scale the structural similarity. Machine learning methods are also used in structure-based classification. Like the SVC (Support Vector Classifier), which will try to find a hyperplane that best separates the different protein categories. Additionally, neural networks can also be used, taking in structural information to efficiently classify proteins into their respective categories. Another representation that is widely used in structural classifications is the use of graphs. In general, these representations can be used in several 3D tasks, such as classification and segmentation [6]. As for proteins, the graph representation provides a more structured approach, offering various modeling possibilities, such as considering each atom as a node and each chemical bond linking each two atoms as an edge, or considering amino acids as nodes by taking just the Cα atoms (the alpha carbon atom, which is a central atom in each amino acid and can serve as a unique identifier for amino acids in protein structures) and add other types of edges, such as contact edges or correlation edges [7]. The choice of models often depends on the study being carried out, such as the analysis of protein dynamics, the identification of active zones and folding zones [8]. These models offer deeper access by adding information (also known as features) to each element of the graph, like chirality, degree, formal charge, number of hydrogens, etc., for the atom, and like type, aromatic stereochemistry, and conjugation for the bonds. The aim of these models is to offer a more detailed understanding of the protein, and thus the possibility of applying graph theory to classify proteins, such as the spectral graph which seeks to identify proteins with similar values in order to classify them in the same group, or to apply deep learning methods such as GCNN, which consists of training a neural network to: (i) perform graph embedding [9], i.e. transform the structural graph to a linear representation while retaining as much information as possible, and (ii) train a deep neural network model to classify these embeddings [10]. Graphical modeling has enjoyed enormous success in protein analysis, notably the prediction of protein-protein interactions [11], protein-molecule interactions in drug discovery [12], and the prediction of protein function [13].

In the second classification type, the surface of proteins is emphasized due to its critical role in their functionality, determining how they interact with other molecules and their environment. The surface classification involves generating 3D surfaces of proteins using methods that create the 3D surface representation. At this stage, the methods used in this type of classification aim to extract information and features from proteins, then compare these surfaces to calculate their similarity. These methods usually transform the protein surface into a vector called 3D descriptors. Then, The similarity between two given protein surfaces is quantified by calculating the distance between their respective descriptors. In addition to these traditional methods, machine learning techniques such as 3D-CNN (3D Convolutional Neural Networks) are also used [14]. Where the protein is represented in a cubic grid format, allowing the network to learn and extract features from the 3D surface for classification.

Using classification based on either the surface or the structure alone can already give us good results. However, focusing solely on structural information might overlook crucial surface details that play a significant role in the protein's biological function. For instance, proteins with similar structural frameworks can have different surface features depending on the species they are found in, leading to different functions. Conversely, relying only on surface information might miss the fundamental structural similarities that define protein families. For example, homologous proteins can have similar surface features but different underlying structures, which can lead to errors in classification if they are grouped in the same class based only on surface similarity. In this work, we represent the protein surfaces with meshes and apply a Graph Convolutional Neural Network (GCNN) to extract surface features. To avoid focusing solely on surface information, we enrich these features with structural information extracted from PDB files before running the combined data through a deep neural network classifier). This approach provides a more complete representation of the protein. We demonstrate that this architecture achieves excellent results in protein classification. To the best of our knowledge, this work represents the first attempt to integrate 3D surface data with structural information using Graph Convolutional Neural Networks (GCNNs) for protein classification.

3 Proposed Approach

3.1 Data Representation

As mentioned in Section 2, classification depends on the type of data used to represent proteins. In this work, since we will be using both types of information, we will be using two types of files. The first type are *.pdb* files, which are used to store three-dimensional structural information about molecules, such as the nature and positioning of atoms, together with other additional information about the protein. These data are organized in lines with a specific format, simplifying data retrieval. For example, data beginning with ATOM describes information about the atoms in the molecule. The details of the structural information and how it was calculated are shown in Table 1.

For surface information, there are basically two types of files: files such as *.xyz*, in which the 3D object is represented as a point cloud, i.e. A list of the coordinates of the geometric points on the protein surface. Files such as *.obj* and *.off*, in which the 3D objects are represented with meshes (points connected to each other), offering a better visualization of the object. In these files, we find two lists, one like that found in *.xyz* files, and another list of triplets (3 points in the first list) indicating links between these points to form triangles enabling clearer visualization of complex objects. By decomposing these triplets into pairs (link ABC becomes links AB, AC, and BC), we obtain an adjacency list, and thus our graphs [15]. An example is illustrated in Figure 1.

Table 1. Information retrieved from .pdb files used in our approach.

Information	Description
Number of Atoms	Total number of atoms in the structure.
Number of Residues	Total number of residues in the protein.
Molecular Weight	Calculated from the amino acid sequence.
Gyration Radius	Measures mass distribution around the central axis.
Average B-Factor	Average of B-factors for all atoms.
Amino Acid Composition	Percentage of each amino acid in the protein.

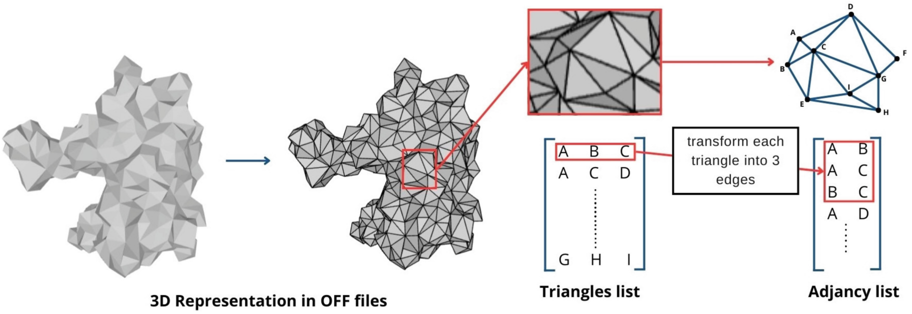

D
A
C
F
B
G
I
E
H
A B C
A C D
G H I
transform each triangle into 3 edges
A B
A C
B C
A D
3D Representation in OFF files
Triangles list
Adjancy list

Fig. 1. Graph Construction from OFF files

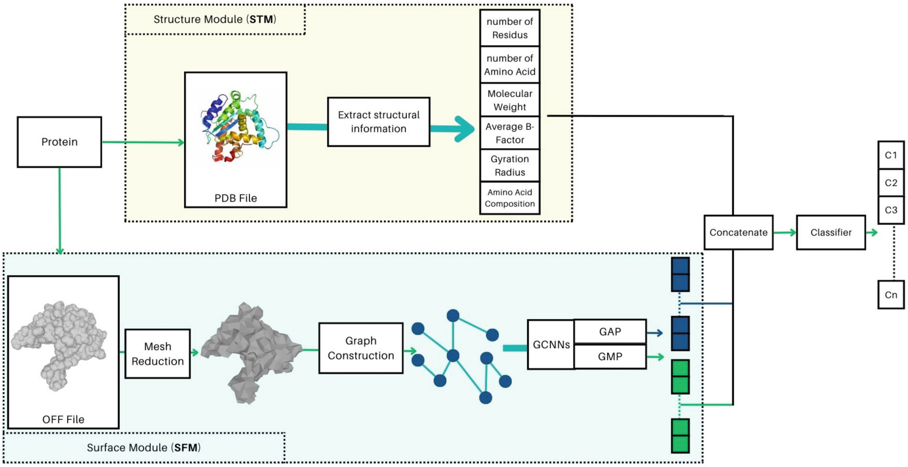


Fig. 2. Structure-Surface Module (SSM) Architecture: Structural information is generated directly from PDB files, and surface information is extracted from graphs obtained from OFF files using GCNNs with two types of pooling. The different pieces of information will be concatenated to train the classifier, which will output a vector of 17 values at the protein level and 54 values at the species level, representing the probability of belonging to each class.

3.2 Architecture

Our system architecture is described in Figure 2, where we have presented each step of the classification process and the three distinct modules.

The information has been extracted separately by retrieving both the *.pdb* and *.off* files corresponding to the same protein. We go through the .pdb files to extract the information presented in Table 1. The resulting vector therefore represents the protein's structural information. As for the surface, the representation will not be calculated or extracted directly like the structural information, but will be obtained using deep learning. The *.off* files will be transformed into graphs $G(X, V)$ where X is a matrix of size $(n, 3)$ also called features where n represents the number of points of each object and 3 refers to the x, y, and z coordinates of each point. And V is a matrix of size $(m, 2)$, also known as an adjacency list, where m represents the number of links in the meshes (which are obtained by decomposing triangles into edges). These graphs will be entered into GCNNs to obtain embedding graphs by following a few steps, as shown in Figure 3: (i) each node in the graph (point) will send and receive messages, which are the features of each point, to and from neighboring nodes (connected points), (ii) the messages received will be aggregated by a derivable function to be used (iii) with another derivable function to update the information of the node in question. The role of the GCNNs is to find the right parameters for these functions that minimize the prediction errors [12]. (i), (ii) and (iii) are grouped together in a step called *message passing*, which boils down to the transformation of each node's information from a 3-dimensional space (x y z coordinates) to a larger space called *embedding*, which aims to group together nodes with similarities in the new dimension. After completing these steps, the information stored in each node no longer represents the coordinates, but rather a deeper positioning of each node relative to the graph. At the end of message passing step, each graph X will be of size (n, e), where e is the chosen embedding size. This means that each graph will have a different size X matrix, which will prevent the training of the classifier. At this point, a pooling method is used, which guarantees to transform all graphs to the same dimension while retaining as much information as possible. Several types of pooling are possible. Once pooling has been applied, all graphs will have the same size, and this representation will be concatenated with the structural representation to form the final representation of the protein, which now enables the classifier to be applied. The classifier used is a single-layer dense neural network. Which takes as input the information of protein and gives as the probabilities of belonging to each class. The advantage of using neural networks is their ability to learn patterns and features at different levels of abstraction. This enables them to capture complex, non-linear relationships in the data, which is often the case with protein structures. And also to adapt to a wide variety of classification problems.

Regarding the pooling in our architecture, We have employed a combined strategy that utilizes both Global Average Pooling (GAP), which averages the features across all nodes, and Global Max Pooling (GMP), which selects the maximum value among the node features. The advantage of using these two types of pooling together lies in their ability to capture different facets of protein graphs. By integrating these two pooling strategies, our aim is to achieve a richer embedding graphs. Although the transformation of the matrix into a linear representation may cost us some data, the use of a classifier in the sequel allows us to avoid a large loss. Indeed, if the classifier fails to make good predictions on the vectors resulting from pooling, the GCNN parameters will be adjusted and updated to improve performance. This capacity for self-improvement is one of the major advantages of using GCNNs in combination with pooling.

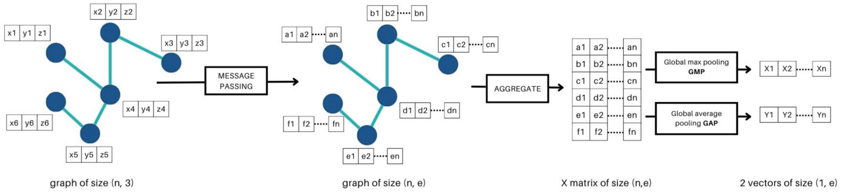


Fig. 3. Graph Convolutional Neural Network (GCNN) Message Passing and Pooling Operations.

In our approach, we did not just want to propose an approach that integrates structural and surface data, but also to determine the importance of each type of data in proteins classification. To do this, we also tested each type of data separately. As illustrated in Figure 2, we therefore call SFM (***S***ur***F***ace ***M***odule) the model associated with surface information, and STM (***ST***ructure ***M***odule) the model applied to structural information, and finally SSM (***S***tructure and ***S***urface ***M***odule) in which both types of information are taken. For each module, we will apply the same classifier with the same parameters to guarantee a reliable comparison. In STM, the classifier takes as input a 25-dimensional vector, representing the structural information used. For the SFM approach, we apply four message-passing steps corresponding to four GCN layers, each with an embedding size of 64. After applying double pooling, we obtain a 128-dimensional vector, which serves as the classifier's input. In the SSM case, both vectors are combined, resulting in a 156-dimensional vector (128 + 28). The model uses the tanh activation function, CrossEntropy as the loss function, and is trained over 100 epochs with a learning rate of 0.007.

4 Implementation Details

4.1 Dataset

To evaluate our model, we will use the SHREC19 Protein Shape Retrieval Contest Dataset [16], which contains 5,298 proteins, each represented by its solvent-excluded molecular surface. The authors used EDTSurf to generate these surfaces, with EDTSurf leveraging the Euclidean Distance Transform - EDT [17] to create the triangular mesh surfaces. SHREC19 is a track where they used the Structural Classification of Proteins extended (SCOPe) database [1],[18] and [19]. Which are classified at 2 levels, (1) at the protein level with 17 classes (the ability to retrieve conformations from orthologous proteins, i.e. independently of the species), and (2) at the species level with 54 classes (the ability to retrieve conformations from the protein of a given species) [16]. thus offering two versions of datasets to validate our approach.

4.2 Method Evaluation

In our study, we adopted a mesh simplification strategy similar to the one used in the SHREC19 track [20], reducing the size of the meshes (from .off files) by 80%, leaving us with 20% of the original data. This reduction process is designed to decrease the number of points in a way that preserves the overall structure of the protein. This approach significantly cuts down on execution time, allowing us to compare our methods with others efficiently and fairly.

In our implementation, we have divided the dataset into 3 parts: the first, representing 60%, will be used for training, the second, representing 20%, will be used for validation, and the last remaining part will be used for testing, ensuring each set maintains the same class distribution for consistency. To enable comparison with other approaches applied on the two SHREC19 datasets [16], the test set will not be used in training to avoid our models having already seen the data. On the other hand, we have decided to apply the same classifier in all three proposed modules of our approach (STM, SFM, and SSM) to ensure a fair comparison and better interpret the importance of each type of information.

We compare our method to those presented in the SHREC19 Protein Shape Retrieval Contest [16]. The ConvLDSNet approach employs a 3D neural network trained on protein surface meshes [21], the 3D Zernike Moments (3DZM) and 3D Zernike Descriptors (3DZD) methods capture global surface characteristics of proteins using Zernike polynomials [22] [23], the Histogram of Area Projection Transform (HAPT) technique differentiates itself by generating histograms that reflect spherical symmetries within protein structures [24], aiming to identify unique shape features, the Framework towards Protein Shape Singularity Characterization (Ft-PSSC) investigates a blend of feature extraction techniques [25][26], the TM-Align method [4] which identifies the best structural alignment between proteins. And we also compared our approach with two methods presented in [27], namely AE, which utilizes an autoencoder for feature dimensionality reduction, and FS, which employs feature selection techniques to enhance classification performance. These last two methods have just been tested on protein levels.

We have chosen the F1 score and the mean average precision (MAP) as metrics for evaluating model performance. The F1 score is a harmonic mean of precision and recall, offering a balance between the model's ability to correctly label true positives while penalizing false positives and false negatives. It is calculated using the equation:

$$F1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{1}$$

where precision is the ratio of true positive predictions to the total number of positive predictions (true positives plus false positives), and recall is the ratio of true positive predictions to the actual number of positive cases (true positives plus false negatives). This metric is particularly useful in our context because it provides a more accurate measure of performance across classes that do not have an equal number of proteins (at proteins level, Class 1 has the highest data count (1160) and Class 13 the lowest (25). At species level, Class 1 has the highest data count (1049), while Class 53 has the fewest data points (3)). the MAP is a measure that evaluates how accurately a system ranks relevant items in its output (Table 2 & Table 3). At species levels only the MAP metric is available, which is why we just compare this metric.

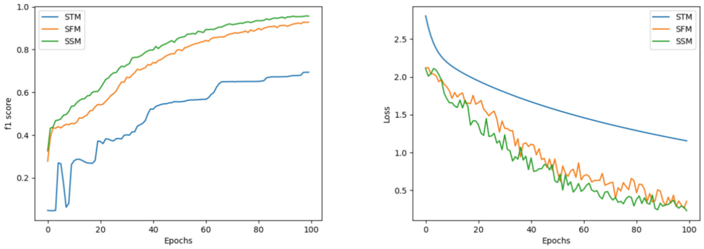
5 Results and Discussion

In this section, we will present and discuss the results our modules achieved during training and on the test set, and compare them with other methods. We start by presenting the metrics at both protein and species levels. Then, for a better understanding, we apply the feature importance technique to interpret stucture module STM module behavior. Finally, we compare the execution times of each method.

5.1 Proteins Level

In Figure 4a and 4b, we present the F1 score and loss evaluation of each model at the protein level. These figures help us understand the learning process of each module, demonstrating how effectively they are improving over time. SFM and SSM showed more relevant progressions (with a small superiority of SSM) compared to STM. We can see that the F1 score tends towards 1 and the loss tends towards 0 for both models, while for STM which also gave good results. This shows that at this level of classification, by enriching surface informations with structural informations we achieve the fastest learning, thus confirming our initial hypothesis that integrating both types of information makes the system more effective.

Table 2 shows the results obtained on the test set at the protein level. The surface-structure module SSM gives the best results with an F1 score of **0.94** compared to the other modules. This means that integrating both types of information provides the module with a more complete representation of the proteins. As for the surface module SFM, it achieved the highest MAP score of **0.918**, slightly higher than the MAP obtained by the SSM module at **0.909**. This indicates that the SFM module is capable of accurately ranking the relevant



(a) (b)

Fig. 4. Loss and F1 score progression for each Module at proteins level.

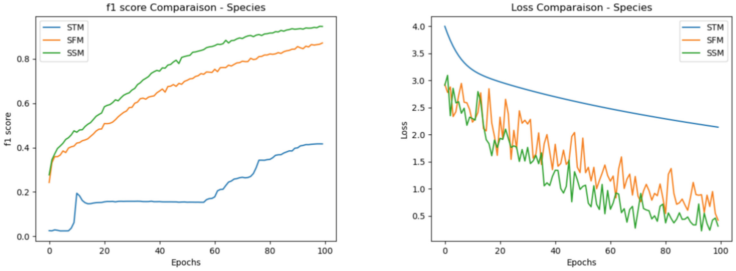
Table 2. F1 score and MAP scores at protein level.

Methods	**MAP**	**F1**
FS [27]	0.880	0.830
AE [27]	0.857	0.800
3DZD [22]	0.720	0.490
3DZDM [23]	0.649	0.350
ConvLDSNet [21]	0.329	0.270
Ft-PSSC [25]	0.417	0.280
HAPT [24]	0.666	0.470
TM-Align [4]	0.79	-
Our Module STM	**0.88**	**0.68**
Our Module SFM	**0.918**	**0.864**
Our Module SSM	**0.909**	**0.94**

instances (demonstrating its effectiveness in precise ranking), but it did not surpass the SSM in F1 score. While the SFM excels at ranking predictions perfectly, it is the SSM module that makes more accurate final predictions overall. In our case, the final predictions, as reflected by the F1 score, are more critical, which is why the SSM module is ultimately more valuable for our task. And this means that using only surface information does not achieve as good predictions as when both types of information are used. The STM module achieved a MAP of **0.88** and an F1 score of **0.68**, indicating that while it performs well in ranking relevant instances, it falls short in achieving high prediction accuracy. The compared methods utilize either structural information or surface information. When comparing our surface module to surface-based methods, it surpasses the best one, Auto Encoder with Feature Selection [27], with an F1 score of **0.864** compared to **0.83**. In comparison with the structure-based method TM-Align, although the F1 score is not provided, our STM module excels in MAP with a score of **0.88**. Finally, our SSM module outperforms all these methods in prediction accuracy.

5.2 Species Level

Like in the first level, the modules showed good progress, but this time there was a remarkable difference between the SFM and SSM modules compared to the STM module (Figures 5a & 5b). This can be explained by the fact that in this level of classification structural information is of less importance.



(a) (b)

Fig. 5. Loss and F1 score progression for each Module at species level.

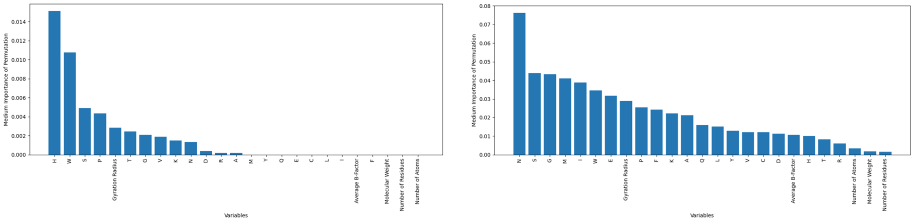
Table 3. F1 score and MAP scores at Species level.

Methods	*MAP*	F1
3DZD 3 [22]	0.610	-
3DZDM [23]	0.604	-
ConvLDSNet [21]	0.355	-
Ft-PSSC [26]	0.405	-
HAPT [24]	0.578	-
TM-Align [24]	0.685	-
Our Module STM	**0.672**	**0.395**
Our Module SFM	**0.867**	**0.632**
Our Module SSM	**0.746**	**0.928**

The results presented in Table 3 clearly highlight the superior performance of our proposed architecture at the species level. However, it was the SFM and SSM modules that performed particularly well. Similar to the first level, the SFM module outperformed the SSM module in terms of MAP, with a more pronounced difference at this level (**+0.121**). This can be attributed to the increased number of imbalanced classes at the species level (54 classes compared to 17 at the protein level). But the F1 score was not as high compared to the SSM module, which achieved an F1 score of **0.928**. These results exceed the highest scores achieved by other methods (MAP = **0.685** by Tm-Align [4]), in contrast to the STM module where the MAP score was **0.672** with an F1 score of **0.395**.

These results can be attributed to the fact that in the hierarchical SCOPe classification, proteins within the same class at the protein level share similar structures and surfaces. At the species level, these similarities are even stronger, meaning that even proteins in different classes can have similar structures and surfaces but are classified differently. Our structure module results suggest that at the species level, a more powerful classifier is needed to differentiate between classes. For the surface classification, using GCNNs in our module effectively identifies differences between classes, even when it's hard to classify proteins because they have similar surface features, highlighting their ability to capture subtle variations in protein surfaces.

5.3 Exploring Feature Importance



(a) Proteins level

(b) Species level

Fig. 6. Feature Importance using feature permutation in knn algorithm.

To better understand why the structure module STM performed differently across the two versions of the dataset, we decided to use the K-nigherst neighbor algorithm, The k-Nearest Neighbors (KNN) is a simple classification algorithm that relies on the principle of proximity to predict. The algorithm identifies the closest proteins in the training data and assigns that protein the most frequent Class among its neighbors. In an iterative process, we randomly shuffle the values of a feature across the test set and measure the effect of this permutation on model performance. If the permutation of a feature results in a significant drop in performance, this indicates that the feature is important for the model. We look at Figure 6, where the importance of each variable is shown.

At the protein level, the variables H and W, where H represents the percentage of the amino acid Histidine and W represents the percentage of the amino acid Tyrosine in the protein sequence, turned out to be the most significant ones by a large margin compared to the others. This means they play a crucial role in helping the model. Proteins in the same class usually have similar values for H and W, which is why the model performs so well in this case.

However, At species level, the variable N (Asparagine) has the most important variable and the difference between it and the other variables is much smaller. This indicates that even though N is significant, the model has to rely more on other variables as well, which explains why the structure module didn't perform as well in this version.

5.4 Run Time

Execution time plays an important role in protein classification tasks. The other methods calculate descriptors (linear vectors) for each protein and then determine similarity by computing distances between these descriptors. To provide a fair comparison, we present the time it takes for our modules to compute the probabilities of each protein belonging to different classes (Figure 2), rather than just predicting the final class. ConvLDSNet displaying the best runtime, followed by our three modules STM, SFM and SSM, respectively. The ConvLDSNet method holds the 1st position with the best runtime; however, it produces insufficient results in terms of accuracy (Table 2 & Table 3).

While ConvLDSNet is the fastest, our modules offer a better balance of speed and predictive performance, providing more accurate and reliable classifications. Although our architecture may seem more complex due to the use of both types of information, the utilization of GCNNs allows us to achieve these results in a relatively short amount of time (Table 4).

Table 4. Comparison of different methods based on SES and Distance metrics.

Methods	***Descriptors calculation***	**Distance Calculation**
Our Module STM	0.58 s (CPU)	0.01 s (i3-1115G4)
Our Module SFM	0.66 s (CPU)	0.01 s (i3-1115G4)
Our Module SSM	0.69 s (CPU)	0.01 s (i3-1115G4)
3DZM	2.95 s (CPU)	0.5 s (CPU)
ConvLDSNet	2.5 ms (GTX1070)	0.002 ms (i7-6700K)
3DZD	3.48 s (NP)	0.1774 s (Titan X)
HAPT	4.98 s / 11.1s / 13.02s (i7-4720HQ)	NP

6 Conclusion

In this research, we propose a new approach based on Graph Convolutional Neural Networks, utilizing both surface and structural information for protein classification. Our method efficiently extracts information from 3D representations of the protein surface, and enriches them with structural information extracted from PDB files to improve the efficiency of protein classficiations. The results demonstrate that integrating these two types of information significantly enhances classification performance, achieving an average accuracy of 94% on the SHREC19 dataset.

The use of three distinct modules (structure module STM, surface module SFM and structure-surface module SSM) has enabled us to interpret the results and to know which type of data is relevant for each level. In fact, our experiments shows that although each type of information is useful at a specific level, the combination of structural and surface information gives the best results overall. The SSM module, in particular, outperformed the other methods, confirming the effectiveness of our integrated approach. Additionally, our modules are not only accurate but also fast, striking a good balance between speed and performance. This balance is essential for real-world applications in bioinformatics, where both accuracy and efficiency are very important.

For further work, we plan to integrate other types of information, such as protein dynamics and functional properties, which will enable us to adapt our system to perform other bioinformatics tasks, such as predicting protein interactions and identifying active sites.

Acknowledgment. ©Copyright Gradient. This project has received support from the French National Agency for Research (ANR) on the reference GRADIENT ANR-22-CE23-0009.

References

1. Fox, N.K., Brenner, S.E., Chandonia, J.M.: SCOPe: Structural Classification of Proteins—extended, integrating SCOP and ASTRAL data and classification of new structures. *Nucleic Acids Research* 42(D1), D304–D309 (2013). https://doi.org/10.1093/nar/gkt1240
2. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., Bourne, P.E.: The Protein Data Bank. *Nucleic Acids Research* 28(1), 235-242 (2000). https://doi.org/10.1093/nar/28.1.235
3. Mohan, A., Rao, M.D., Sunderrajan, S., et al.: Automatic classification of protein structures using physicochemical parameters. *Interdisciplinary Sciences: Computational Life Sciences* 6, 176–186 (2014). https://doi.org/10.1007/s12539-013-0199-0
4. Zhang, Y., Skolnick, J.: Tm-align: a protein structure alignment algorithm based on the tm-score. Nucleic Acids Res. **33**, 2302–2309 (2005)
5. Zhang, Y., Skolnick, J.: Scoring function for automated assessment of protein structure template quality. Proteins: Struct., Funct., Bioinf. **57**, 702–710 (2004)
6. Tang, W., Qiu, G.: Dense graph convolutional neural networks on 3D meshes for 3D object segmentation and classification. *Image and Vision Computing* 114, 104265 (2021). https://doi.org/10.1016/j.imavis.2021.104265
7. Huan, J., Bandyopadhyay, D., Wang, W., Snoeyink, J., Prins, J., Tropsha, A.: Comparing Graph Representations of Protein Structure for Mining Family-Specific Residue-Based Packing Motifs. *Journal of Computational Biology* 12(6), 657-671 (2005). https://doi.org/10.1089/cmb.2005.12.657

8. Vishveshwara, S., Brinda, K.V., Kannan, N.: PROTEIN STRUCTURE: INSIGHTS FROM GRAPH THEORY. *Journal of Theoretical and Computational Chemistry* 01(01), 187-211 (2002). https://doi.org/10.1142/S0219633602000117
9. Goyal, P., Ferrara, E.: Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems* 151, 78-94 (2018). https://doi.org/10.1016/j.knosys.2018.03.022
10. Zhang, S., Tong, H., Xu, J., Maciejewski, R.: Graph convolutional networks: a comprehensive review. *Computational Social Networks* 6(1), 11 (2019). https://doi.org/10.1186/s40649-019-0069-y
11. Baranwal, M., Magner, A., Saldinger, J., et al.: Struct2Graph: a graph attention network for structure based predictions of protein–protein interactions. *BMC Bioinformatics* 23, 370 (2022). https://doi.org/10.1186/s12859-022-04910-9
12. Ruiz Puentes, P., et al.: Modeling Protein-Ligand Interactions with Graph Convolutional Networks for Interpretable Pharmaceutical Discovery. *Research Square* (2022), Preprint version 1. https://doi.org/10.21203/rs.3.rs-1262123/v1
13. Gligorijević, V., Renfrew, P.D., Kosciolek, T., et al.: Structure-based protein function prediction using graph convolutional networks. *Nature Communications* 12, 3168 (2021). https://doi.org/10.1038/s41467-021-23303-9
14. Amidi A, Amidi S, Vlachakis D, Megalooikonomou V, Paragios N, Zacharaki EI. EnzyNet: enzyme classification using 3D convolutional neural networks on spatial representation. *PeerJ* 6:e4750 (2018). https://doi.org/10.7717/peerj.4750
15. Ahmed, E., Saint, A., Shabayek, A.E.R., Cherenkova, K., Das, R., Gusev, G., Aouada, D., Ottersten, B.: Deep Learning Advances on Different 3D Data Representations: A Survey. arXiv preprint arXiv:1808.01462 (2018)
16. Langenfeld, F., Axenopoulos, A., Benhabiles, H., Daras, P., Giachetti, A., Han, X., Hammoudi, K., Kihara, D., Lai, T.M., Liu, H., Melkemi, M., Mylonas, S.K., Terashi, G., Wang, Y., Windal, F., Montes, M.: SHREC'19 Protein Shape Retrieval Contest. *Eurographics Workshop on 3D Object Retrieval* (2019)
17. Xu, D., Zhang, Y.: Generating Triangulated Macromolecular Surfaces by Euclidean Distance Transform. PLoS ONE **4**(12), e8140 (2009)
18. Chandonia, J.M., Fox, N.K., Brenner, S.E.: SCOPe: Manual Curation and Artifact Removal in the Structural Classification of Proteins – extended Database. *Journal of Molecular Biology* 429(3), 348-355 (2017). https://doi.org/10.1016/j.jmb.2016.11.023
19. Chandonia, J.M., Fox, N.K., Brenner, S.E.: SCOPe: classification of large macromolecular structures in the structural classification of proteins—extended database. *Nucleic Acids Research* 47(D1), D475-D481 (2018). https://doi.org/10.1093/nar/gky1134
20. Benhabiles, H., Aubreton, O., Barki, H., Tabia, H.: Fast simplification with sharp feature preserving for 3D point clouds. In: 2013 11th International Symposium on Programming and Systems (ISPS). pp. 47-52 (2013). https://doi.org/10.1109/ISPS.2013.6581492
21. Maturana, D., Scherer, S.: VoxNet: A 3D Convolutional Neural Network for real-time object recognition. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 922-928 (2015). https://doi.org/10.1109/IROS.2015.7353481
22. Novotni, M., Klein, R.: 3D zernike descriptors for content based shape retrieval. In: Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications. pp. 216–225. Association for Computing Machinery, New York, NY, USA (2003). https://doi.org/10.1145/781606.781639

23. Canterakis, N.: 3D Zernike moments and Zernike affine invariants for 3D image analysis and recognition. In: Proc. 11th Scandinavian Conf. on Image Analysis. pp. 85–93 (1999)
24. Giachetti, A., Lovato, C.: Radial Symmetry Detection and Shape Characterization with the Multiscale Area Projection Transform. *Computer Graphics Forum* 31(5), 1669-1678 (2012). https://doi.org/10.1111/j.1467-8659.2012.03172.x
25. Lima, D.M., Teichrieb, V.: An Efficient Global Point Cloud Descriptor for Object Recognition and Pose Estimation. In: 2016 29th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI). pp. 56–63 (2016). https://doi.org/10.1109/SIBGRAPI.2016.017
26. Jégou, H., Perronnin, F., Douze, M., Sánchez, J., Pérez, P., Schmid, C.: Aggregating Local Image Descriptors into Compact Codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34(9), 1704-1716 (2012). https://doi.org/10.1109/TPAMI.2011.235
27. Madi, K., Paquet, E.: 3D Deformable Protein Shapes Classification based on Triangles-Stars and Composite Deep Neural Networks. Umanis Research and Innovation, Levallois-Perret 92300, France, and National Research Council, 1200 Montreal Road Ottawa, Ontario, Canada (2022)

Adaptive Nearest Neighbor Density Peak Clustering Based on Fuzzy Logic

Houshen Lin, Jian Hou(✉), and Huaqiang Yuan

School of Computer Science and Technology, Dongguan University of Technology,
Dongguan 523808, China
houjian@dgut.edu.cn

Abstract. Density Peak Clustering (DPC) has attracted widespread attention in the recent decade. However, traditional DPC algorithms still have shortcomings such as difficulty in describing data distribution and sensitivity to parameters and allocation strategies. To address these shortcomings, this paper introduces an adaptive shared nearest neighbor density peak clustering algorithm based on fuzzy logic. Our algorithm makes use of the concepts of natural nearest neighbor and shared nearest neighbor, and provides an effective method for estimating neighbor distance and local density. The number K of nearest neighbors is selected adaptively based on the dataset itself. Instead of manually selecting cluster centers from a decision graph, our algorithm automatically determines them. In addition, based on the idea of fuzzy logic, non-central data points are divided into different types and assigned using different methods, which improves the robustness and accuracy of assigning non-cluster central data points. With both synthetic and real datasets in experiments, our algorithm is shown to be effectiveness when compared with recent DPC-based algorithms.

Keywords: natural nearest neighbors · density peak · shared nearest neighbors

1 Introduction

Data clustering divides the samples of the dataset into different clusters, and is used frequently in the fields including social networks [21], pattern recognition [12], and image segmentation [3]. Numerous clustering methods have been presented, including several classic ones like K-means [20], GMM [4], and DBSCAN [7]. Traditional algorithms include density-based [1], partition-based [22], hierarchical clustering [34] and distribution-based [37]. Algorithms published in recent years, such as spectral clustering [25], affinity propagation [28], ensemble clustering [31], multi-view clustering [8] and subspace clustering [5], have also received much attention.

The density Peak Clustering (DPC) algorithm [23] estimates local density ρ in the first step, then finds the distance δ to the nearest neighbor with higher

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 125–139, 2025.
https://doi.org/10.1007/978-3-031-78383-8_9

density, and constructs a decision graph based on the product of the first two to select cluster centers. After completing the above steps, each non-central data point is allocated to the cluster of its nearest neighbor of larger density, thereby distinguishing different clusters. This algorithm is efficient, does not involve iteration, and is also applicable to complex, non-convex datasets. Therefore, DPC has received widespread attention since its publication.

The DPC algorithm inevitably has some problems that need to be improved [27]. First, the local density ρ and the neighbor distance δ are easily contaminated by noise when facing data sets with complex data distribution. Second, the algorithm is sensitive to parameters, and prior knowledge is required to determine the specific value. In addition, the cluster center is affected by the local density ρ and the neighbor distance δ, which requires additional manual selection and is full of uncertainty. Finally, the allocation strategy may produce a "domino effect", that is, if a single data point is allocated incorrectly, other data points may also be forced to be allocated incorrectly.

In view of the above limitations of DPC, we present an adaptive shared nearest neighbor DPC algorithm based on fuzzy logic. By introducing the concept of shared nearest neighbor, the algorithm redesigns ρ and δ and proposes a new calculation formula. However, the number k of the shared nearest neighbor often requires prior knowledge to determine. Combined with the natural nearest neighbor [38], the number k of nearest neighbors is selected appropriately for a given dataset. Based on the superiority of the design of ρ and δ, the product operation results of the local density and the neighbor distance are sorted in descending order, and then the cluster centers are selected in turn. In data sets with overlapping or unclear cluster boundaries, such as cross-stacked and variable density data sets, the original allocation strategy may be difficult to accurately allocate data points to a unique cluster. Combined with fuzzy logic, we introduce the concept of membership. That is, the probability that non-cluster center data points belong to different clusters. We use it to determine the final allocation of data points, which can effectively reduce the error allocation rate compared with the original allocation strategy. Therefore, we divide the non-clustering centers into three categories of density data points and design a fuzzy logic-based allocation strategy for each category of data points. The above steps not only overcome the problem of no manual intervention and parameter adjustment, but also improve the clustering accuracy.

This paper has the following contributions.

(1) We introduce shared nearest neighbor and natural nearest neighbor into DPC and present a method to calculate ρ and δ so as to more easily identify cluster centers. In addition, the parameters can be determined according to the situation of the dataset itself, thus avoiding the influence of human selection errors.

(2) Inspired by fuzzy logic, based on the original DPC algorithm allocation strategy, we design appropriate allocation strategies according to data points of different density types to ensure the robustness of the allocation process of data points of different density types.

Section 2 reviews the DPC algorithm and introduces recent works related to the DPC algorithm. Our algorithm is presented in details in Section 3. In section 4, we verify the proposed algorithm through experiments, and in section 5, we summarize.

2 Related Work

Here we present the implementation idea of DPC and introduce the latest works based on this algorithm.

2.1 Density Peak Clustering

The key of DPC lies in two assumptions. First, cluster centers should have larger local density than neighboring points. Second, cluster centers are relatively distant from each other.

The original DPC algorithm calculates local density based on the cutoff and Gaussian kernels. Both density kernels are very sensitive to the parameter d_c. [23] suggests that d_c should be determined with 1% to 2% of all data points.

The parameter δ refers to the distance between a point x_i and its nearest neighbor of larger density. On one hand, if the data point is not the point with the highest local density among its neighboring data points, the δ value of the current data point should be the distance between its nearest high-density neighbor. On the other hand, if the data point is the point with the highest local density compared to its neighboring data points, the value of δ should be the maximum of all δ's among all points.

Data points that satisfy the requirement that both local density ρ and relative distance δ should be as large as possible are called cluster centers. In addition, we need to draw a decision graph based on local density and relative distance to help identify which data points are cluster centers. Finally, based on the assumption that a data point and its nearest high-density neighbor should be in the same cluster. We assign the unassigned points to the same cluster as its nearest neighbor of larger density.

2.2 Recent Works on DPC

The DPC algorithm also faces some difficulties, especially when dealing with real datasets. Therefore many improved algorithms have been proposed in recent years, namely, new density kernels, new cluster center identification methods, and new non-cluster center data allocation methods.

The DPC-KNN [6] algorithm is based on KNN and designs a new density kernel to better describe the local structure of the datasets. This idea is also applied to the DPC-DLP [24] algorithm. The FKNN-DPC [30] algorithm is also based on KNN and designs a density kernel calculation method that combines the local density of data points with the information of their neighbors. The MDPC+ cite GuanJY23 algorithm adopts a similar approach to FKNN-DPC in the local density calculation part. The DPC-DBFN cite LotfiA20 algorithm selects fuzzy kernels to calculate local density, thereby improving the accuracy of clustering algorithms and reducing the influence of individual noise points. In addition, the DPC-FSC [16] algorithm also redesigns the density kernel based on the introduction of fuzzy ideas. The RDO-DPC [15] algorithm uses the neighborhood information of sample points to define the relative density of sample data, and finds and identifies the density peaks of non-uniform distribution as cluster centers. The adaptive local density formula proposed by the ERK-DPC [29] algorithm requires the use of the similarity between data points in the dataset to measure the local density of each point to improve clustering density, where the similarity is calculated using the Euler cosine distance. BC-DPC [36] is based on balancing density and connectivity to eliminate the density differences between different clusters to accurately identify the cluster center.

Secondly, complex distributed datasets may have multiple density peaks or one cluster center is not far from the non-center data in the decision graph, making it difficult to determine the correct cluster center. The MDPC+ [9] algorithm constructs a decision graph by constructing an adjacent density peak graph and selects data points with main peak features as cluster centers. The DPC-FSC [16] algorithm redesigns the neighbor semantic distance to replace the neighbor distance, thereby constructing a clearer decision graph to select the correct cluster center. The PFD-DPC [39] algorithm introduces the concept of potential field and proposes a density measure based on potential field diffusion, which can accurately select cluster centers. The G-KNN-DPC [13] algorithm first finds the optimal value of parameter $d_c$based on the Gini coefficient, and then uses the k-nearest neighbor idea to identify more accurate clustering centers, avoiding the occurrence of incorrect selection of clustering centers. The RDPC-DSS [32] algorithm proposes a density clustering index (DCI) method instead of drawing decision maps to automatically monitor the number of cluster centers. In addition, density sensitive similarity is proposed to avoid the problem of algorithm parameter sensitivity. The DADC [2] algorithm proposes clustering and fragmenting the dataset. Then, density peaks with similar values are identified, and finally the initial cluster centers are automatically extracted and the fragmented clusters are merged to complete the clustering.

The DPC-DLP [24] algorithm assigns labels to data points based on graph label propagation, which can effectively assign true labels to data points located in the boundary and overlapping areas. The DPC-DBFN [18] algorithm uses a density-based kNN graph to propagate the label of each dense point to its dense neighbors, and finally assigns each boundary point to the cluster to which the nearest backbone point belongs. The FKNN-DPC [30] algorithm adopts a two-step allocation strategy, which assigns labels to most data points by breadth first searching for the nearest neighbors of the cluster center, and then assigns labels to the remaining data points based on fuzzy weighted nearest neighbors. In DPCSA [35], we first assign points that meet the boundary conditions, and the remaining data points are assigned based on the nearest neighbor assignment strategy that destroys the local density decreasing assignment order. The KS-

FDPC [14] algorithm adopts a partition merging strategy to reduce the impact of high-density points on subsequent assigned points by dividing the data into multiple subclusters. The FDPC [33] algorithm uses support vectors to perform clustering based on density peak clustering, and then recursively merges clusters according to the feedback value between each two clusters to obtain accurate clustering results. The DPC-LMST algorithm [26] is based on the idea of local minimum spanning tree, which considers each identified potential cluster center as an initial cluster. Before completing the final clustering result, a sub cluster merging factor needs to be introduced to aggregate similar sub clusters.

The algorithm proposed in this article is different from the related work on DPC in recent years. Unlike distance based density kernel calculations or KNN based approaches, we have improved the DPC algorithm based on the idea of sharing nearest neighbors. By considering the distribution of shared neighbors among data points to measure their similarity, it can more accurately reflect the density distribution of data points in their local neighbors. Regarding the value of the neighbor parameter k, unlike the practice of giving a priori default values or traversing parameters, our algorithm combines the natural nearest neighbor and can dynamically and adaptively determine the optimal value of the neighbor parameter k according to the local characteristics of the data. Similar to the DPC-FSC algorithm, we use the compensation distance method to improve the effectiveness of the neighbor distance. However, we no longer manually determine cluster centers through decision graphs, but instead achieve automatic identification of cluster centers in the dataset. Different from the fuzzy weighting strategy adopted by the FKNN-DPC algorithm for boundary points, we divide the non-cluster center into three categories of density data points, and design a fuzzy logic-based allocation strategy for each category of density data points, which is more targeted.

3 Our Algorithm

We present an adaptive shared nearest neighbor density peak clustering algorithm based on fuzzy logic to solve some problems of DPC. Firstly, we use the ideas of natural nearest neighbors and shared nearest neighbors to redesign the density kernel and relative distance, overcoming the problem of parameter sensitivity. In addition, we divide the non-cluster center data points into data points of different density types, and design allocation strategies based on fuzzy logic for allocation. The details are presented in the following.

3.1 Density-Distance Estimation

Before calculating the local density ρ and neighbor distance δ, we need to initialize the dataset, such as normalization or t-sne [19] dimensionality reduction. This can handle complex nonlinear relationships and enhance clustering effects. Next, we introduce the SNN idea to redesign the density kernel calculation formula.The basic idea of sharing nearest neighbors is that the more similar two points are, the greater the probability that they come from the same cluster.

The similarity of data points lies in the degree to which they share the same nearest neighbor. When this value is large, it means that the measured data points may be located in a highly dense common neighborhood, thereby avoiding the possibility of noise points affecting the local density ρ and neighbor distance δ results of high-density data points. In addition, the SNN process does not rely on a key, globally fixed distance threshold parameter, but is controlled by the neighbor parameter k to control the shared neighborhood range.

Before calculating the shared nearest neighbors of all data points, we first use the Euclidean distance metric to list the k nearest neighbors corresponding to each point in the dataset, where d is the distance between data points. For example, for data points x_i and x_j in dataset X, their respective k nearest neighbor sets are $\mathrm{KNN}(x_i))$ and $\mathrm{KNN}(x_j))$. The shared neighbor set of data points x_i and x_j is their common neighbor set, that is, $\mathrm{SNN}(i, j) = \mathrm{KNN}(x_i) \cap \mathrm{KNN}(x_j)$.

Inspired by [17], we use the same method to calculate SNN similarity. When data points x_i and x_j cannot be found in each other's k-neighborhood set, it can be said that the similarity between them is 0. When the above conditions are not met, the specific formula is as follows:

$$\mathrm{Sim}(i, j) = \frac{|\mathrm{SNN}(i, j)|^2}{\sum_{p \in \mathrm{SNN}(i,j)} (d_{ip} + d_{pj})} \tag{1}$$

The above SNN similarity can better reflect the distribution of various types of data to a certain extent by checking shared neighbors, so we use this similarity to calculate the local density of data points. Then we can assume that the data point x_i is a point in the data set X, and we find the k data points with the highest similarity to this point, provided that these data points are also in the data set X. We organize all eligible data points into the set $\mathrm{L}(i) = \{x_1, x_2, \cdots, x_k\}$, and then define the density of one point x_i as the sum of the similarities with the k points with the highest similarity. The specific local density calculation formula is as follows:

$$\rho_i = \sum_{j \in \mathrm{L}(i)} \mathrm{Sim}(i, j) \tag{2}$$

The advantage of the new local density kernel is that it can better highlight the data distribution of the current data set. For example, if the distance between x_i and x_j to each other's shared neighbors is smaller, it means that the density of point x_i is greater. In addition, if the number of shared neighbors between x_i and x_j is large, it means that the data point x_i is more likely to belong to the same cluster as most of the points around it, which also proves that the density of data point x_i should be larger. All of these can prove the superiority of local density calculation itself.

However, to overcome the problem of selecting the key neighbor parameter values in the SNN idea, we optimize the algorithm based on the natural nearest neighbor idea. The natural nearest neighbor is an adaptive nearest neighbor method. The key idea is that points located in sparse areas should have fewer neighbors, while points located in dense areas should have a large number of neighbors. In the implementation of the natural nearest neighbor method, no additional input parameters are required. The most important thing is that the optimal number of neighbors is determined adaptively for each dataset.

The implementation premise of the NNN algorithm is the k-nearest neighbor and reverse k-nearest neighbor ideas, which continuously expand the scope of neighborhood search. Each search calculates the number of reverse nearest neighbors of each data point until the number of data points without reverse neighbors no longer changes. That is, if $(\forall x_i)(\exists x_j)(r \in \mathbb{N}) \wedge (x_i = x_j) \rightarrow (x_i \in \mathrm{KNN}_r(x_j)) \wedge (x_j \in \mathrm{KNN}_r(x_i))$ is satisfied, the algorithm is considered to have reached the natural searching state and the searching is terminated. The experiments in [38] show that the natural nearest neighbor eigenvalue λ obtained in the experiment is the optimal K value for the current experimental datasets, so we choose this value as the parameter value in the density kernel.

Next, we adopt a compensation mechanism based on neighborhood distance and redesign the neighborhood distance δ to highlight the difference between cluster centers and non-cluster centers and enhance the fairness of the δ value. First, we assume that a data point in the datasets X is x_i and a data point x_j whose local density is greater than x_i. Then calculate the distance between the two data points and multiply it by the distances of the two data points to their respective nearest neighbors. Finally, the minimum value of the product operation is the δ value of the data point x_i. When the local density of a data point is the highest, the delta value of the data point is the largest delta value among other points. The specific calculation formula for δ is:

$$\delta_i = \min_{j:\rho_j > \rho_i} \{ d_{ij} * (\sum_{p \in KNN(i)} d_{ip} + \sum_{q \in KNN(j)} d_{jq}) \} \tag{3}$$

Due to the superiority of our newly proposed local density kernel and relative distance calculation, we no longer draw decision diagrams to manually determine the cluster center, but select the cluster center in descending order according to the γ of the data point. We calculate γ as

$$\gamma = \rho * \delta. \tag{4}$$

3.2 Fuzzy Allocation Strategy

After obtaining n_c cluster centers, we initialize a one-to-one corresponding cluster set $\{C_j\}$, which also means that the data in the dataset will be divided into n_c clusters. Among them, $j = 1, \ldots, n_c$ represents the jth cluster in $\{C_j\}$. To overcome the disadvantage of error propagation of DPC, we propose to divide the non-cluster centers into high-density, medium-density and low-density data points, and design appropriate allocation strategies based on fuzzy logic respectively.

First, we assign the high-density data points in the dataset. We can make a judgment when the following two conditions are met, that is, when the data point x_i is assigned and the data point x_o is not assigned. If and only if $|\mathrm{SNN}(i, o)| \geq$

$k/2$ is satisfied, we believe that the two data points are highly similar, and the probability that x_o and x_i should be assigned to the same cluster is very high. In other words, if at least half of the nearest neighbors of two data points are shared with each other, then they should belong to the same cluster.

Next, we initialize a queue of length n_c and store the cluster centers in the queue in order. We find the head data point x_i in the current queue and find its shared neighbor set. According to the above assumptions, point x_i is selected as the cluster center and has been assigned to a cluster. When the data point x_j in the shared neighbor set has not been assigned and meets the conditions, we assign x_j to the same cluster $\{C_j\}$ as x_i.

After the current allocation operation is completed, we choose to store x_j at the end of the queue. In addition, after traversing the shared neighbor set of data point x_i, we take out the data point x_i from the queue and regard it as completing the iteration of data point x_i. When the elements in the queue are empty, we end the iteration and the allocation of the current type of data points is completed.

Then we proceed to the second step of allocation. We first operate on the similarity matrix $A \in R^{m*n_c}$, set all its values to empty, and modify the values in the similarity matrix A through continuous iteration. Here A_{lj} in the matrix represents the similarity relationship of an unassigned data point x_l being assigned to the j-th cluster $\{C_j\}$. The greater the similarity between the data point x_l and the cluster C_j, the greater the probability that x_l belongs to the cluster C_j.

We believe that the similarity of A_{lj} depends on the similarity between it and its assigned data neighbor points. In the nearest neighbors of x_i, we find the neighbor point x_l that is eligible to be assigned to cluster C_j, that is, $x_l \in N^k_{x_i} \cap C_j$. Of course, when calculating the similarity, we will consider the following aspects.

First, we believe that the larger the distance d_{li} between x_l and x_i, the smaller the probability that x_l belongs to C_j. Second, a larger ρ_i means that cluster C_j will be more attractive to data point x_l. This is consistent with the assumption that a data belongs to the same cluster as its nearest higher density neighbor. Finally, the allocation strategy for medium similarity data points roughly allocates data points in descending order of local density, which means that local density ρ_l plays a positive role. According to the above description, our specific calculation formula is as follows:

$$A_{lj} = \sum_{x_l \in N^k_{x_i} \cap C_j} \frac{\rho_l * \rho_i}{d_{li}} \tag{5}$$

Next, we believe that the largest value in the similarity matrix $A_{i^*j^*}$ often means that the data point x_{i^*} is most likely to be assigned to the cluster C_{j^*}. After we have assigned x_{i^*}, to distinguish the assigned data points from the unassigned points, let the value of the i^*th row of the membership matrix be 0, indicating that the current data point x_{i^*} has been assigned. There is a situation where the k nearest neighbor data points x_i* of the unassigned data point x_m,

Algorithm 1 The proposed algorithm.

```
Input: $X = \{1, \ldots, n\}$ , number of clusters $n_c$
Output: data labels $Y = \{Y_1, \ldots, Y_n\}$
Initializes the dataset X and calculates the distance matrix of the dataset.
Implement the natural nearest neighbor algorithm to obtain the natural neighbor eigenvalue $\lambda$ of the data set itself.
Calculate the local density $\rho_i$ using Eq.(2).
Calculate the neighbor distance $\delta_i$ using Eq.(3).
Calculate the $\gamma$ using Eq.(4) to select cluster centers.
Initialize queue $Q_1$ to store cluster centers.
while $Q_1$ is not empty do
    Find the data points that meet the conditions in the header point's neighbor set and assign them to the corresponding clusters.
    Push up the allocated data points to the end of the queue.
    Pop up the header point.
end while
Calculate the similarity matrix $A$ using Eq.(5).
Initialize queue $Q_2$ to store unallocated points.
while $Q_2$ is not empty do
    Find the point with the highest similarity degree and assign it to the corresponding cluster.
    Update $A$ with Eq. (6).
end while
Find the k nearest data points in different clusters for each unassigned data point and calculate $\zeta$.
Assign points to the cluster $C_{j^*}$ with the smallest $\zeta$ value.
Return $Y$.
```

and $x_i* \in N_{x_m}^k$, then the value of this data point in the membership matrix is affected and needs to be reset. The specific operations are as follows.

$$A_{mj} = A_{mj} + \frac{\rho_m * \rho_{i^*}}{d_{mi^*}} \tag{6}$$

When all elements in the matrix are zero, the iteration ends and the assignment of medium-similarity data points is completed. The remaining low-similarity data points are usually outliers far away from most data. Such data points are difficult to assign through the relationship between data points in the region. Therefore, we designed a simple method to assign.

Specifically, for each unassigned data point x_i, we find its k nearest neighbors in the cluster, ζ_{ij} represents the average distance between x_i and the nearest neighbor in the jth cluster. Finally, the point x_i will be allocated to the cluster C_{j^*} with the minimum ζ value, that is, $j_* = argmin_{j \in \{1, \ldots, n_c\}} \zeta_{ij}$.

The implementation details of our algorithm are shown in Algorithm 1.

4 Experiments Result

In the comparative experiments, we use datasets with different sizes, dimensions and manifold structures, as shown in Table 1. We use accuracy (ACC), normalized mutual information (NMI) and adjusted Rand index (ARI) to evaluate the clustering results.

Table 1. Datasets information used in the experiment

Datasets	Instances	Attributes	Clusters	Datasets	Instances	Attributes	Clusters
Compound	399	2	6	DryBean	13611	16	7
D31	3100	2	31	Dutchnumeral	2000	649	10
Pathbased	300	2	3	Libras	360	90	15
Spread-2-10	1000	2	10	Olivertti	400	28	40
Spread-10-20	2000	10	20	Seeds	210	7	3
Spread-20-35	3500	20	35	Segment	2310	19	7
Spread-50-50	5000	50	50	Waveform(noise)	5000	40	3
Unbalance	6500	2	8	Wine	178	13	3

4.1 Baselines and parameter settings

To show the performance of our algorithm, we make a comparison with 9 recent representative DPC-based clustering optimization algorithms, including DPCSA [35], DPC-DBFN [18], FHC-LDP [10], DPC-FSC [16], MDPC+ [9], and ICDKP [11]. The parameter descriptions of these methods are shown in Table2. When it comes to the experimental parameters, all algorithms have the number of nearest neighbors as a parameter, which is set as the ground truth. Unlike the original paper of the DPCSA algorithm, which has given the values of all parameters, our algorithm has no other parameter values that need to be set. Finally, the parameter values that need to be set for the other comparison algorithms are shown in Table 2.

Table 2. Algorithm parameter values in the experiment

Method	Paramerters
DPC-DBFN	k in kNN, $k \in (2:1:50)$
DPC-FSC	$\alpha \in (0.1:0.1:0.9)$
FHC-LDP	k in kNN, $k \in (2:1:50)$
MDPC+	the attenuation-coefficient $\lambda \in (1:0.1:5)$
ICDKP	k in kNN, $k \in (2:1:50)$

4.2 Experiments with synthetic datasets

In this set of experiments, we can see from Table 3 that on the Spread series datasets, the algorithms involved in the experiment can achieve very good clustering results, especially DPC-DBFN, FHC-LDP, DPC-FSC and our algorithm can achieve a clustering accuracy of 1 under different indicators. However, on the Pathbased dataset, other algorithms perform unsatisfactorily, while our algorithm can still maintain a good clustering accuracy. Under the ACC, ARI and NMI indicators, the clustering results of our algorithm are 0.8933, 0.7275 and 0.7895 respectively, ranking first.

In addition, on the Compound and Unbalance datasets, the performance of the algorithms involved in the experiment varies greatly, while our algorithm performs robustly on these datasets and has higher clustering results. For example, under the ACC indicator, our proposed algorithm performs best in the Compound dataset with a result of 0.9098; under the ARI indicator, our proposed algorithm performs best with a clustering result of 0.8699; under the NMI indicator, the FHC-LPD algorithm performs best with a result of 0.9020. Finally, the average clustering result of our proposed algorithm ranks first in all indicators. This also shows that our algorithm has excellent clustering ability, which is no less than the current excellent improved algorithm.

This set of experimental datasets are all synthetic datasets, which are usually generated according to predefined distributions and rules. The separation between clusters is high and the clusters are easy to identify. However, when processing such datasets, the disadvantage of algorithm parameter sensitivity will be more obvious. It is not difficult to see that in comparison with other parameter-sensitive methods, our approach has better average clustering results. Different from the default parameter value of the DPCSA algorithm, this paper determines the number k of nearest neighbors adaptively after introducing the natural nearest neighbor idea, which not only avoids errors caused by manual parameter setting, but also effectively improves the accuracy of clustering.

4.3 Experiments on real datasets

In this group of experiments, it can be seen from Table 4 that the performance of each algorithm on the real dataset is worse than that on the synthetic dataset. On the Dutchnumeral and Seeds datasets, our algorithm and the comparative experimental algorithm can show a high clustering effect, especially our algorithm is better than other algorithms. For example, when processing Seeds, it ranks first in ACC, ARI and NMI indicators. On the DryBean and Olivertti datasets, the differences between the algorithms are large. Our algorithm performs robustly on these datasets and has a high clustering result.

In addition, on the Libras and Segment datasets, the algorithms involved in the comparative experiments all performed poorly. For example, in the Segment dataset, under the ACC and ARI indicators, the best performing DPC-DBFN clustering results are 0.5385 and 0.3470 respectively; under the NMI indicator, the best performing DPCSA algorithm result is 0.5083; and the clustering results

Table 3. Experimental results with synthetic datasets

Dataset	Index	DPCSA	DPC-DBFN	FHC-LPD	DPC-FSC	MDPC+	ICKDP	Ours
Compound	ACC	0.8396	0.8546	0.8822	0.6692	0.4411	0.6366	**0.9098**
	ARI	0.8284	0.8087	0.8483	0.5352	0.2460	0.5131	**0.8699**
	NMI	0.8439	0.8465	**0.9020**	0.7670	0.5881	0.7600	0.8977
D31	ACC	0.9681	**0.9732**	0.9713	0.9677	0.6981	0.9706	0.9710
	ARI	0.9360	**0.9477**	0.9421	0.9351	0.6642	0.9409	0.9416
	NMI	0.9578	**0.9635**	0.9611	0.9570	0.9011	0.9605	0.9602
Pathbased	ACC	0.8233	0.7233	0.7467	0.8367	0.5667	0.6967	**0.8933**
	ARI	0.6133	0.4454	0.5629	0.6309	0.3540	0.4230	**0.7275**
	NMI	0.7311	0.5314	0.7144	0.7398	0.4778	0.5145	**0.7895**
Spread-2-10	ACC	0.9000	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9000	**1.0000**
	ARI	0.8971	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.8971	**1.0000**
	NMI	0.9694	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9694	**1.0000**
Spread-10-20	ACC	0.9500	**1.0000**	**1.0000**	**1.0000**	0.8905	0.9500	**1.0000**
	ARI	0.9493	**1.0000**	**1.0000**	**1.0000**	0.8525	0.9493	**1.0000**
	NMI	0.9884	**1.0000**	**1.0000**	**1.0000**	0.9680	0.9884	**1.0000**
Spread-20-35	ACC	0.9714	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9714	**1.0000**
	ARI	0.9711	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9711	**1.0000**
	NMI	0.9944	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9944	**1.0000**
Spread-50-50	ACC	0.9800	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9800	**1.0000**
	ARI	0.9798	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9798	**1.0000**
	NMI	0.9965	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9965	**1.0000**
Unbalance	ACC	**1.0000**	0.9860	**1.0000**	0.5331	0.5562	0.9998	**1.0000**
	ARI	**1.0000**	0.9988	**1.0000**	0.4729	0.5683	**1.0000**	**1.0000**
	NMI	**1.0000**	0.9795	**1.0000**	0.6416	0.7671	0.9994	**1.0000**
Mean	ACC	0.9291	0.9421	0.9500	0.8758	0.7691	0.8881	**0.9718**
	ARI	0.8969	0.9001	0.9192	0.8218	0.7106	0.8343	**0.9424**
	NMI	0.9352	0.9151	0.9472	0.8882	0.8378	0.8979	**0.9559**

of MDPC+, which performs the worst under the three indicators, are 0.2879, 0.0238, and 0.2283 respectively; the results of our proposed algorithm under these three indicators are 0.5152, 0.3269, and 0.4941, respectively, proving that it still maintains a high clustering performance when facing complex and difficult real datasets. In the real data set, the average clustering results of our proposed algorithm under different indicators are 0.7208, 0.5396, and 0.6367, respectively, also ranking first.

All experimental datasets in this group are real datasets, and real data usually do not have obvious density distribution. The shape, size, density, etc. of the clusters are very diverse and very challenging. Our algorithm provides structured information based on neighbor relationships after introducing the idea of shared nearest neighbors, which can effectively capture local and global structural information, thereby identifying complex cluster shapes and distributions in real datasets. In addition, after introducing the idea of fuzzy logic, the algorithm can estimate probabilities of points being assigned to clusters, handle the fuzzy attribution of data points between different clusters, and more accurately reflect the actual distribution of the data. Therefore, it has excellent clustering performance with real datasets.

Table 4. Experimental results with real datasets

Dataset	Index	DPCSA	DPC-DBFN	FHC-LPD	DPC-FSC	MDPC+	ICKDP	Ours
DryBean	ACC	0.6246	0.7050	0.4570	0.7376	0.6142	0.6105	**0.7755**
	ARI	0.4589	0.4984	0.3004	**0.5778**	0.4232	0.4306	0.5706
	NMI	0.6085	0.5965	0.5477	**0.7120**	0.5915	0.6342	0.6787
Dutchnumeral	ACC	0.7860	0.8015	**0.8325**	0.8265	0.7450	0.7040	0.7730
	ARI	0.6672	0.6869	**0.7176**	0.7133	0.6341	0.6249	0.6776
	NMI	0.7879	0.7699	0.7811	0.7866	0.7741	0.7372	**0.7885**
Libras	ACC	0.4722	0.4583	0.4917	**0.5111**	0.3861	**0.5111**	**0.5111**
	ARI	0.3512	0.3402	0.3681	**0.3906**	0.3332	0.3826	0.3774
	NMI	0.6282	0.6168	0.6334	**0.6510**	0.6031	0.6465	0.6486
Olivertti	ACC	0.7175	0.7125	0.7650	0.7600	0.1750	0.7100	**0.7725**
	ARI	0.6660	0.6403	0.6854	0.6917	0.1562	0.6100	**0.7076**
	NMI	0.8954	0.8797	0.9010	0.9008	0.6444	0.8599	**0.9016**
Seeds	ACC	0.5810	0.8571	0.8429	0.8429	0.5810	0.8429	**0.9095**
	ARI	0.3981	0.6317	0.5952	0.5840	0.3981	0.5840	**0.7477**
	NMI	0.5336	0.6362	0.6307	0.6237	0.5336	0.6237	**0.7054**
Segment	ACC	0.4502	**0.5385**	0.5087	0.5108	0.2879	0.4654	0.5152
	ARI	0.2518	**0.3470**	0.1827	0.2919	0.0238	0.2868	0.3269
	NMI	**0.5083**	0.4944	0.4424	0.4573	0.2283	0.4029	0.4941
Waveform (nosie)	ACC	0.7110	0.5762	0.7170	0.6642	0.4868	0.7158	**0.8072**
	ARI	0.3524	0.2694	0.3656	0.3307	0.0995	0.3621	**0.5060**
	NMI	0.4004	0.3423	0.4087	0.4093	0.1736	0.4059	**0.4383**
Wine	ACC	0.7247	**0.7303**	**0.7303**	0.7135	0.5449	0.7135	0.7022
	ARI	0.4007	**0.4154**	0.4061	0.4029	0.3184	0.4066	0.4029
	NMI	0.3948	0.4473	**0.4536**	0.4302	0.4146	0.4288	0.4388
Mean	ACC	0.6334	0.6724	0.6681	0.6958	0.4776	0.6591	**0.7208**
	ARI	0.4433	0.4787	0.4526	0.4979	0.2983	0.4610	**0.5396**
	NMI	0.5947	0.5979	0.5998	0.6214	0.4954	0.5924	**0.6367**

5 Conclusions

We present an adaptive shared nearest neighbor density peak clustering algorithm based on fuzzy logic. First, we introduce natural nearest neighbor to overcome the parameter selection problem and avoid errors caused by inappropriate parameter selection. Next, we adopt the concept of shared nearest neighbors for not only accurately calculating local density, but also identifying cluster centers. In addition, we designed a suitable allocation strategy for data points of different density types to overcome the shortcomings in the original method. In experiments with datasets of different data distributions and types, our algorithm is shown to be superior to some recent algorithms.

Acknowledgment. This work is supported by the National Natural Science Foundation of China under Grants No. 62176057.

References

1. Bhattacharjee, P., Mitra, P.: A survey of density based clustering algorithms. Front. Comp. Sci. **15**, 1–27 (2021)

2. Chen, J., Yu, P.S.: A domain adaptive density clustering algorithm for data with varying density distribution. IEEE Trans. Knowl. Data Eng. **33**, 2310–2321 (2021)
3. Coleman, G.B., Andrews, H.C.: Image segmentation by clustering. Proc. IEEE **67**, 773–785 (1979)
4. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. J. Roy. Stat. Soc.: Ser. B (Methodol.) **39**, 1–22 (1977)
5. Deng, T., Yang, G., Huang, Y., Yang, M., Fujita, H.: Adaptive multi-granularity sparse subspace clustering. Inf. Sci. **642**, 119143 (2023)
6. Du, M., Ding, S., Jia, H.: Study on density peaks clustering based on k-nearest neighbors and principal component analysis. Knowl.-Based Syst. **99**, 135–145 (2016)
7. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: kdd. vol. 96, pp. 226–231 (1996)
8. Fang, U., Li, M., Li, J., Gao, L., Jia, T., Zhang, Y.: A comprehensive survey on multi-view clustering. IEEE Transactions on Knowledge and Data Engineering **35** (2023)
9. Guan, J., Li, S., He, X., Chen, J.: Clustering by fast detection of main density peaks within a peak digraph. Inf. Sci. **628**, 504–521 (2023)
10. Guan, J., Li, S., He, X., Zhu, J., Chen, J.: Fast hierarchical clustering of local density peaks via an association degree transfer method. Neurocomputing **455**, 401–418 (2021)
11. Guo, W., Chen, W., Liu, X.: Density peak clustering by local centers and improved connectivity kernel. Inf. Sci. **666**, 120439 (2024)
12. Jain, A.K., Duin, R.P.W., Mao, J.: Statistical pattern recognition: A review. IEEE Trans. Pattern Anal. Mach. Intell. **22**, 4–37 (2000)
13. Jiang, D., Zang, W., Sun, R., Wang, Z., Liu, X.: Adaptive density peaks clustering based on k-nearest neighbor and gini coefficient. IEEE Access **8**, 113900–113917 (2020)
14. Li, C., Ding, S., Xu, X., Hou, H., Ding, L.: Fast density peaks clustering algorithm based on improved mutual k-nearest-neighbor and sub-cluster merging. Inf. Sci. **647**, 119470 (2023)
15. Li, C., Zhang, Y.: Density peak clustering based on relative density optimization. Math. Probl. Eng. **2020**, 1–8 (2020)
16. Li, Y., Sun, L., Tang, Y.: Dpc-fsc: An approach of fuzzy semantic cells to density peaks clustering. Inf. Sci. **616**, 88–107 (2022)
17. Liu, R., Wang, H., Yu, X.: Shared-nearest-neighbor-based clustering by fast search and find of density peaks. Inf. Sci. **450**, 200–226 (2018)
18. Lotfi, A., Moradi, P., Beigy, H.: Density peaks clustering based on density backbone and fuzzy neighborhood. Pattern Recogn. **107**, 107449 (2020)
19. der Maaten, L.V., Hinton, G.: Visualizing data using t-sne. J. Mach. Learn. Res. **9**, 2579–2605 (2008)
20. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability. vol. 1, pp. 281–297 (1967)
21. Mishra, N., Schreiber, R., Stanton, I., Tarjan, R.E.: Clustering social networks. In: International Workshop on Algorithms and Models for the Web-Graph. pp. 56–67 (2007)
22. Mittal, M., Sharma, R.K., Singh, V.P., Kumar, R.: Adaptive threshold based clustering: a deterministic partitioning approach. International Journal of Information System Modeling and Design (IJISMD) **10**, 42–59 (2019)

23. Rodriguez, A., Laio, A.: Clustering by fast search and find of density peaks. Science **344**, 1492–1496 (2014)
24. Seyedi, S.A., Lotfi, A., Moradi, P., Qader, N.N.: Dynamic graph-based label propagation for density peaks clustering. Expert Syst. Appl. **115**, 314–328 (2019)
25. Tang, C., Li, Z., Wang, J., Liu, X., Zhang, W., Zhu, E.: Unified one-step multi-view spectral clustering. IEEE Trans. Knowl. Data Eng. **35**, 6449–6460 (2022)
26. Wang, R., Zhu, Q.: Density peaks clustering based on local minimal spanning tree. IEEE Access **7**, 108438–108446 (2019)
27. Wang, Y., Qian, J., Hassan, M., Zhang, X., Zhang, T., Yang, C., Zhou, X., Jia, F.: Density peak clustering algorithms: A review on the decade 2014–2023. Expert Syst. Appl. **328**, 121860 (2023)
28. Wei, Z., He, D., Jin, Z., Liu, B., Shan, S., Chen, Y., Miao, J.: Density-based affinity propagation tensor clustering for intelligent fault diagnosis of train bogie bearing. IEEE Transactions on Intelligent Transportation Systems **24** (2023)
29. Wu, Q., Zhang, Q., Sun, R., Li, L., Mu, H., Shang, F.: Adaptive density peak clustering based on dimension-free and reverse k-nearest neighbours. Information Technology and Control **49**, 395–411 (2020)
30. Xie, J., Gao, H., Xie, W., Liu, X., Grant, P.W.: Robust clustering by detecting density peaks and assigning points based on fuzzy weighted k-nearest neighbors. Inf. Sci. **354**, 19–40 (2016)
31. Xu, J., Li, T., Zhang, D., Wu, J.: Ensemble clustering via fusing global and local structure information. Expert Syst. Appl. **237**, 121557 (2024)
32. Xu, X., Ding, S., Wang, L., Wang, Y.: A robust density peaks clustering algorithm with density-sensitive similarity. Knowl.-Based Syst. **200**, 106028 (2020)
33. Xu, X., Ding, S., Xu, H., Liao, H., Xue, Y.: A feasible density peaks clustering algorithm with a merging strategy. Soft. Comput. **23**, 5171–5183 (2019)
34. Yu, B., Zheng, Z., Dai, J.: K-dghc: A hierarchical clustering method based on k-dominance granularity. Inf. Sci. **632**, 232–251 (2023)
35. Yu, D., Liu, G., Guo, M., Liu, X., Yao, S.: Density peaks clustering based on weighted local density sequence and nearest neighbor assignment. IEEE Access **7**, 34301–34317 (2019)
36. Zhang, Q., Dai, Y., Wang, G.: Density peaks clustering based on balance density and connectivity. Pattern Recogn. **134**, 109052 (2023)
37. Zhang, R., Ma, X., Zhan, J., Yao, Y.: 3wc-d: A feature distribution-based adaptive three-way clustering method. Appl. Intell. **53**, 15561–15579 (2023)
38. Zhu, Q., Feng, J., Huang, J.: Natural neighbor: A self-adaptive neighborhood method without parameter k. Pattern Recogn. Lett. **80**, 30–36 (2016)
39. Zhuang, H., Cui, J., Liu, T., Wang, H.: A physical model inspired density peak clustering. PLoS ONE **15**, 1–30 (2020)

Zero-shot Automated Class Imbalanced Learning

Zhaoyang Wang and Shuo Wang(✉)

School of Computer Science, University of Birmingham, Birmingham B15 2TT, UK
zxw180@student.bham.ac.uk, s.wang.2@bham.ac.uk

Abstract. Given a new class imbalanced dataset D and limited computational resources, the challenge arises of selecting promising class imbalanced learning (CIL) pipelines that include resampling methods, classification models, and their corresponding hyperparameters. To address this challenge, we study Zero-shot Automated Machine Learning and propose a new approach aiming at class imbalanced data, called Zero-shot Automated Class Imbalance Learning (ZAutoCIL). ZAutoCIL employs domain-independent meta-learning to develop a zero-shot surrogate model for automated class imbalanced learning. This model aims to recommend effective CIL pipelines for new unseen imbalanced datasets without requiring additional search. Specifically, we meta-train a two-tower model to serve as the surrogate model, adapted from recommender systems, using a pairwise ranking loss on the meta-dataset gained from collecting performance data across a wide range of CIL pipelines and a comprehensive repository of class imbalance datasets. We perform extensive experiments on 100 datasets grouped in 4 parts based on their imbalance ratio. The experimental results demonstrate the efficacy of our approach in automating the recommendation of CIL pipelines given any target imbalanced datasets.

Keywords: Class imbalance · Automated machine learning · Meta-learning · Recommender systems

1 Introduction

Class imbalanced datasets with a skewed distribution in one or more class labels. This kind of imbalanced datasets has some classes of data significantly underrepresented compared to the others. It commonly exists in various domains, such as fault diagnosis in the monitoring system for manufacturing industry, fraud detection in banking, and medical diagnostics in the healthcare [11,27,29]. A class imbalanced distribution can degrade a model's performance significantly. The negative impact of class imbalance on real-world applications has spurred the development of numerous methods in imbalanced domain learning, encompassing both data-level and algorithm-level approaches [7]. Given a class imbalanced dataset with a limited computational budget, it is unclear how to choose which

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 140–155, 2025.
https://doi.org/10.1007/978-3-031-78383-8_10

algorithm to use, nor how to configure its hyperparameters in class imbalanced domain learning. This challenge can be considered as a Combined Algorithm Selection and Hyperparameter Optimization (CASH) problem that Automated Machine Leaning (AutoML) is dedicated to solving [6,10,23].

However, most conventional solutions in CASH typically neglect class imbalanced learning. Moreover, they often search across vast algorithm configuration spaces and repeatedly train pipelines from scratch for each new dataset in a time-consuming trial-and-error form (see Fig. 1 left) [16,26]. Consequently, identifying an effective CIL pipeline for class imbalanced datasets might require significant computational resources. This challenge is particularly pronounced when there are constraints on time and budget. In addition, researchers and practitioners continuously release methods and datasets in class imbalanced domain learning, making them accessible to the public. With a plethora of class imbalanced datasets available in open sources, the question arises: How can we effectively utilize these existing datasets to recommend a Class Imbalanced Learning (CIL) pipeline when faced with a new and unseen imbalanced task, without incurring search cost on the target dataset?

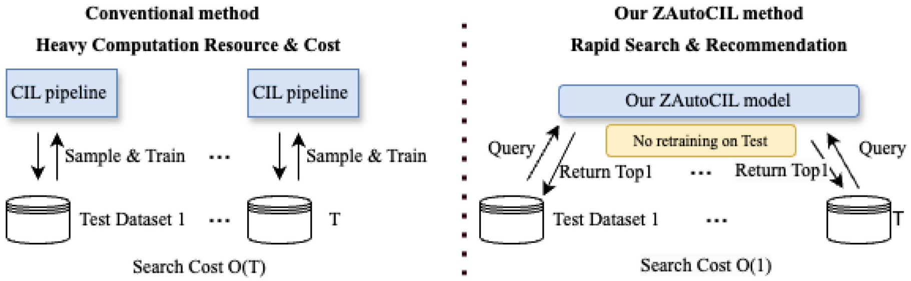


Fig. 1. Comparison between conventional approaches and our proposed method (**Left** vs. **Right**): Traditional approaches require to repeatedly train CIL pipeline on each test dataset, resulting in tremendous total search time on multiple class imbalanced datasets ($O(T)$). In contrast, our method ZAutoCIL (**Right**) leverages a two-tower surrogate model only meta-trained on meta-datasets once and then retrieves CIL pipeline for unseen class imbalanced datasets without additional CIL pipeline training. This reduces the search cost for training CIL pipeline across diverse class imbalance datasets from $O(T)$ to $O(1)$.

In this paper, we introduce zero-shot AutoML for class imbalanced learning, called ZAutoCIL. Zero-shot refers to the capability of selecting a well-performing CIL pipeline effectively without conducting exploratory evaluations for test tasks [17]. We build a surrogate model capable of robust generalization across various imbalanced datasets. We achieve this by leveraging accumulated meta-knowledge (i.e. class imbalanced datasets, CIL pipelines, and their corresponding performance) to effectively adapt to new test tasks in class imbalanced scenarios. This surrogate model is a two-tower model, an advanced recommender system method to understand the complex mapping between pipelines and datasets, with great flexibility, scalability, and the ability to solve the cold start issue [32]. We further improve its performance by employing a pairwise ranking loss, which prioritizes the relative performance of pipelines in the meta-dataset. ZAutoCIL reduces the search time complexity from $O(T)$ to $O(1)$ for multiple class imbalanced datasets because of no training on test datasets (as illustrated in Fig. 1 Right).

The contributions of this work are threefold: (1) We extend AutoML to best exploit class imbalanced learning pipeline by meat-learning to select the well-performing CIL pipeline conditional on dataset meta-features. We introduce a meta-dataset with the performance of 100 CIL pipelines across 100 class imbalanced datasets. (2) We introduce Zero-shot Automated Class Imbalanced Learning, treating it as a recommendation task that considers the embeddings of both class imbalanced datasets and CIL pipelines. We meta-train a two-tower model using a pairwise loss. (3) We conduct experimental evaluations on 100 class imbalanced datasets against baseline methods to demonstrate the efficacy of our approach.

The rest of the paper is organized as follows: Sect. 2 discusses related work of class imbalanced learning and Zero-shot hyperparameter optimization. Section 3 presents the problem definition. Section 4 describes the ZAutoCIL method, including meta-dataset construction and meta-training. Section 5 gives observations from experiments and analyzes the experimental results. Finally, Sect. 6 presents the conclusions and future work.

2 Related work

2.1 Class Imbalanced Learning

A class imbalanced dataset is characterized by a skewed distribution in the number of samples among its classes, with one or more classes being underrepresented (referred to as minority classes) and others as majority classes [8]. Research has demonstrated the effectiveness of data-level (i.e. resampling methods) and algorithm-level approaches in addressing class imbalance [3,14,19]. Resampling methods aim to generate balanced datasets and are categorized into three groups: undersampling (e.g. TomekLinks), oversampling (e.g. SMOTE), and combined-sampling (e.g. SMOTETomek). In contrast, algorithm-level methods focus on modifying classification algorithms to address imbalanced datasets. Due to the abundance of choices among these methods and their respective hyperparameters, several studies have investigated automated class imbalanced learning to address CASH issue for class imbalanced datasets [16,21,26,33]. However, these traditional methods involve exhaustive searches across vast algorithm configuration spaces and training pipelines from scratch for each new dataset. This leads to significant computational resource to identify an effective CIL pipeline given a new class imbalanced dataset.

To address this issue, recent studies have employed prior knowledge in automated class imbalanced learning. Vieira et al. [25] utilize the Frobenius norm to assess similarity among class imbalanced datasets and recommend a combination of resampling and classification methods from 220 search spaces based on accumulated knowledge. However, the hyperparameters of these algorithms are ignored. Similarly, Automated imbalanced datasets learning (AutoIDL) [34] focuses on selecting suitable combinations of resampling methods and classification algorithms. It exploits user-based collaborative filtering to recommend a CIL pipeline. Both of these approaches [25,34] rely on similarity measures on datasets to make their recommendations given class imbalanced datasets, achieving low search costs. Depending solely on similarity measures on datasets can lead to recommendations that overlook interactions between components (i.e. CIL pipelines and class imbalanced datasets), resulting in increasing the likelihood of overfitting. To fill in the gap, we introduce an advanced method from recommender systems: the two-tower model. This model enhances the ability to capture intricate interactions and dependencies between class imbalanced datasets and CIL pipelines, thereby enabling precise recommendations of effective CIL pipelines for unseen class imbalanced datasets in a zero-shot paradigm.

2.2 Transfer Hyperparameter Optimization (HPO) and Zero-shot HPO

Transfer hyperparameter optimization (HPO) exploits knowledge from previous experiments to build a robust surrogate model with only a few observations on the target dataset [31]. To further reduce the search cost, zero-shot HPO has been proposed. Traditional zero-shot learning refers to a machine learning paradigm where a model is trained to recognize classes not present in the training data [28]. In recent developments, zero-shot HPO has been considered a more efficient approach that does not require any observations of the response on the target dataset [17,22,30]. A closely related study by Ozturk et al. [17] employs zero-shot HPO methods to address zero-shot AutoML. They utilize meta-features of datasets and pipelines as joint feature vectors, inputting these into a neural network optimized using a ranking loss across datasets. However, their focus on pretrained models overlooks the issue of class imbalance. To fill in this gap, we introduce Zero-shot Automated Class Imbalanced Learning (ZAutoCIL). In contrast to previous research, we approach ZAutoCIL as a recommendation problem and introduce the two-tower model as a novel surrogate model that can embed separate representations of class imbalance datasets and CIL pipelines. This model is selected for its flexibility, scalability, and robust performance in the recommender system domain.

3 Problem Definition

Let $\mathcal{P} := \{P_k\}_{k=1}^{K}$ denote a set of Class Imbalanced Learning (CIL) pipelines. Each specific CIL pipeline is a combination of a resampling algorithm and a classification classier, including their respective hyperparameters. For example, this could be SMOTE with k-neighbours $= 5$ and Adaboost classifier with the size of base classifier with $n = 50$. Let $\mathcal{D} = \{D_t\}_{t=1}^{T}$ represents a set of T class imbalanced datasets collected from open-source repositories. The descriptive features of each class imbalanced dataset are defined as meta-features ξ_t, such as the imbalance ratio (IR), dataset size, and various statistical characteristics.

The cost of a class imbalanced learning pipeline P_k on dataset D_t is defined as the $C(P_k, D_t) = \mathcal{L}_k(P_k, D_t)$, where $\mathcal{L}_k$ denotes the loss incurred by P_k on D_t. Given a set of K class imbalanced learning (CIL) pipelines $\mathcal{P} := \{P_k\}_{k=1}^{K}$ and a collection of T class imbalanced datasets $\mathcal{D} = \{D_t\}_{t=1}^{T}$, along with a $K \times T$ matrix of costs $C(P_k, D_t)$, the objective of Zero-shot AutoML with Class Imbalanced Learning (ZAutoCIL) pipelines is to determine a mapping f that minimizes the expected cost across the class imbalanced dataset collection $\mathcal{D}$:

$$\mathbf{w}^* = \arg\,\min_{\mathbf{w}}\, \mathbb{E}_{t\sim\mathcal{T}}[C(f_{\mathbf{w}}(\xi_t), D_t)] \tag{1}$$

where $\mathbf{w}$ represents the parameters of the meta-model. After training, $f_{\mathbf{w}^*}$ is employed during the meta-test phase to rapidly identify a high-performing CIL pipeline $\mathcal{P}$ given a unseen class imbalanced dataset.

4 Methodology: Zero-shot Automated Class Imbalanced Learning

We develop Zero-shot AutoML [17] for the domain of class imbalanced learning. Current methods cannot be directly applied to the field of imbalanced learning, as has been demonstrated in related work [25,34]. Therefore, we need to design a zero-shot learning approach specifically for the imbalanced domain. The Zero-shot Automated Class Imbalanced Learning (ZAutoCIL) aims to rapidly recommend high-performing CIL pipelines for a class imbalanced dataset by leveraging prior knowledge learned from a meta-dataset. We detail the preparation of our new meta-dataset designed for class imbalanced learning in Sect. 4.1. Afterward, we present optimization details for our surrogate model, which employs pairwise ranking loss during the meta-training phase, as outlined in Sect. 4.2. An overview of our solution framework is illustrated in Fig. 2.

4.1 Meta-Dataset Construction and Pipeline Hubs Design for ZAutoCIL

In this section, we present a new meta-dataset for imbalance domain learning in order to perform Zero-shot Automated Class Imbalanced Learning (ZAutoCIL). This meta-dataset encompasses a comprehensive collection of class imbalanced datasets along with their meta-features, a diverse range of CIL pipelines, and their respective performance metrics. Specifically, we gather a set of 100 class imbalanced datasets and compute the meta-features for each of them. We then define a space of CIL pipelines and identify strong instantiations within this space for each class imbalanced dataset. Finally, we evaluate the performance of these instantiations in all 100 datasets, resulting in a 100 × 100 matrix of performance.

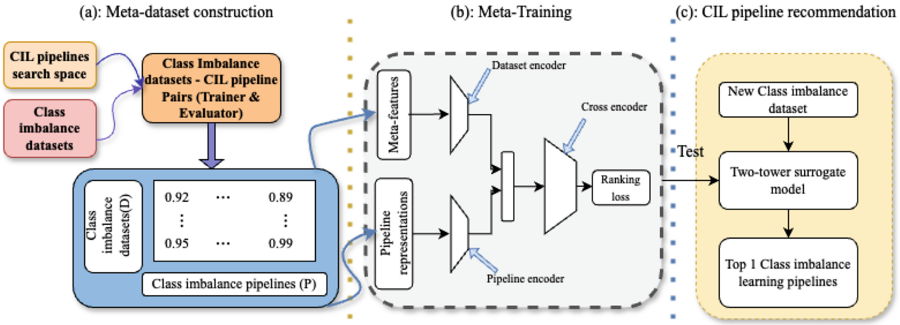

(a): Meta-dataset construction
(b): Meta-Training
(c): CIL pipeline recommendation
CIL pipelines search space
Class imbalance datasets
Class Imbalance datasets - CIL pipeline Pairs (Trainer & Evaluator)
Class imbalance datasets(D)
0.92
0.89
0.95
0.99
Class imbalance pipelines (P)
Meta-features
Pipeline representations
Dataset encoder
Cross encoder
Pipeline encoder
Ranking loss
Test
New Class imbalance dataset
Two-tower surrogate model
Top 1 Class imbalance learning pipelines

Fig. 2. Overview of our proposed Zero-shot Automated Class Imbalanced Learning (ZAutoCIL) framework. (a) Meta-dataset Construction Phase. We construct a meta-dataset comprising triples, i.e. class imbalance dataset - Pipeline (CIL) - performance metric. (b) Meta-training phase. We exploit a two-tower model to encode the CIL pipeline P and meta-features ξ, and use MLP to encode the combination of the outputs of two previous encoders. The framework is optimized by the pairwise ranking loss. (c) Meta-test phase. Our optimized framework recommends the top-1 CIL pipeline given a new class imbalanced dataset during the meta-test phase.

Class Imbalanced datasets for ZAutoCIL The selection of class imbalanced datasets should prioritize public availability and representativeness, which forms the foundation of the prior knowledge for the ZAutoCIL framework. We opt for the widely-used imbalance domain learning repository, i.e. KEEL – Knowledge Extraction on Evolutionary Learning [1], renowned for its diverse class imbalanced datasets organized by class imbalance ratio. From the Imbalance datasets for classification in KEEL, we retrieved 100 class imbalanced datasets $\mathcal{D}$ that have been categorized into four groups: imbalance ratio between 1.5 and 9, and imbalance ratio higher than 9 - Part I, Part II, and Part III. Given the space restrictions, detailed descriptions of the class imbalanced datasets can be found on the KEEL website. The selected data sets vary in sample size from 92 to 5472 and cover a wide range of domains. Additionally, the class imbalance ratios within these datasets range between 1.82 to 100.14, with varying degrees of imbalance across the groups.

To represent a class imbalanced dataset D_t, we extract meta-features that characterize the dataset's properties. These meta-features are precomputed and serve as inputs for training the surrogate model. Several previous studies have conducted surveys and evaluations of dataset meta-features [15,20,24]. From these studies, we identify a set of common, cost-effective, and validated performance meta-features. We utilize the open-source meta-feature extraction library PyMFE [2] to implement this process, focusing on its general and statistical meta-feature groups. General meta-features of a dataset include the number of instances and attributes, imbalance ratio, etc. Statistical meta-features encompass the interquartile range (IQR) of each attribute, the correlation among numeric attributes, skewness, etc. Due to space limitations, detailed descriptions

Table 1. Search Space of CIL Pipelines: Resampling Techniques and Ensemble Classifiers.

	Model	Hyperparameter	Search range
Undersampling	ClusterCentroids	-	-
	EditedNearestNeighbour	k_neighbors	(3,7,11,15)
	NearMiss	k_neighbors	(3,7,11,15)
	RandomUnderSampler	-	-
	TomekLinks	-	-
Oversampling	SMOTE	k_neighbors	(3,7,11,15)
	Borderline SMOTE	k_neighbors	(3,7,11,15)
	RandomOversampler	-	-
	SVMSMOTE	k_neighbors	(3,7,11,15)
OverUnderSampling	SMOTEENN	-	-
	SMOTETomek	-	-
Ensemble	EasyEnsemble	n_estimators	(50, 100, 150)
	RUSBoost	learning_rate	(0.01, 0.05, 0.1, 0.5, 1.0)
		n_estimators	(50, 100, 150)
	BalancedBagging	n_estimators	(50, 100, 150)
		max_features	(0.5, 0.75, 1.0)
		max_samples	(0.5, 0.75, 1.0)
	BalancedRandomForest	n_estimators	(50, 100, 150)
		max_depth	(5, 10, 20)
		min_samples_split	(2, 4, 8, 16, 32)
	AdaBoost	learning_rate	(0.01, 0.05, 0.1, 0.5, 1.0)
		n_estimators	(50, 100, 150)
	GradientBoosting	learning_rate	(0.01, 0.05, 0.1, 0.5, 1.0)
		n_estimators	(50, 100, 150)

of the general and statistical meta-feature groups can be found in the PyMFE library.

CIL Pipeline Design Space for ZAutoCIL on Imbalanced Datasets An overview of the CIL search space is presented in Table 1. Our aim is to select a diverse range of CIL pipelines that can achieve high performance across the class imbalanced datasets, with the goal of identifying well-performing pipelines for each dataset. To obtain robust pipelines, we choose effective resampling techniques and ensemble classifiers that are compatible and available in both the imbalance library and the scikit-learn library [12].

Specifically, the discrete search space includes resampling techniques such as undersampling, oversampling, and over-undersampling, with a focus on the hyperparameters for the number of neighbors. In terms of classifiers, it encompasses ensemble methods such as boosting, bagging, and random forests, each with their respective essential hyperparameters.

Selection and Evaluation of CIL Pipelines for Meta-dataset Construction With the collection of 100 class imbalanced datasets and our defined CIL pipeline search space, we now describe the process of generating the CIL pipeline hub, which is pre-evaluated across these datasets. Instead of employing uniform or random sampling, we perform grid search with 10-fold cross-validation to identify an optimal Class Imbalance Learning (CIL) pipeline for each individual class imbalanced dataset. The optimization metric focuses on the common and crucial class imbalanced learning metric – AUC-ROC. As a result, this process produces one optimized CIL pipeline for each class imbalanced dataset, resulting in a total of $T = D = 100$ CIL pipelines that are defined as the CIL pipeline hub in our study. This CIL pipeline hub is denoted as $\mathcal{P}$ in our ZAutoCIL framework, as detailed in Sect. 3. Given the CIL pipeline hub $\mathcal{P}$ and the set of 100 class imbalanced datasets $\mathcal{D}$, we run each CIL pipeline in $P \in \mathcal{P}$ on each class imbalanced dataset $D \in \mathcal{D}$. The AUC-ROC performance score is calculated as an average over ten runs to mitigate noise. We record the average-of-ten AUC-ROC score for each pair (dataset & CIL pipeline) in the performance matrix. We obtain every pairs' average-of-ten AUC-ROC score in the performance matrix 100×100. This matrix is visualized in Fig. 3.

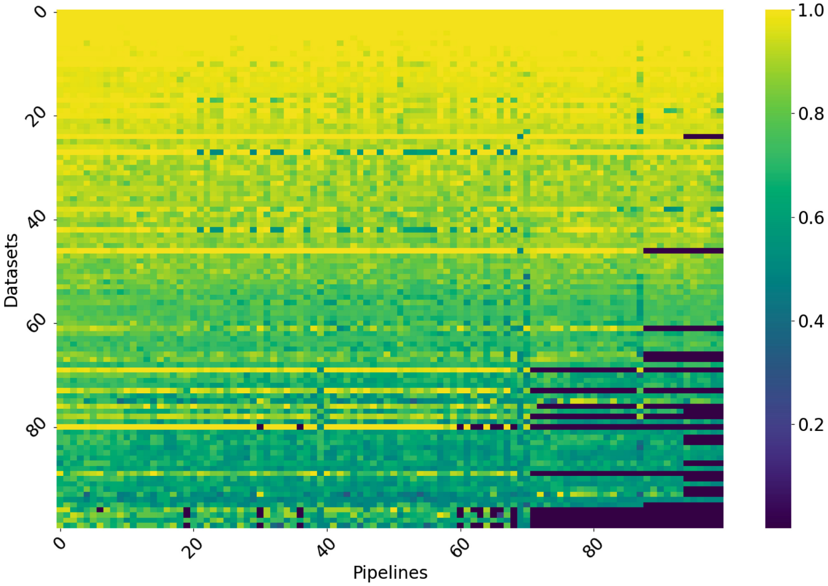


Fig. 3. Performance matrix displayed as a heat map with color representing the AUC-ROC score. It is evident that certain class imbalanced datasets are more challenging, and some CIL pipelines generalize worse than others.

We observe that some datasets, particularly those at the top, pose relatively fewer challenges for the CIL pipelines, since most CIL pipelines perform well on them. Conversely, datasets at the bottom present significant challenges for the CIL pipelines. In terms of CIL pipeline performance, robust CIL pipelines on the left consistently achieve good results across a wide range of class imbalanced datasets. In contrast, some pipelines perform well on only a few datasets, suggesting specialized strengths but limited applicability across diverse class imbalance scenarios. Besides, horizontal striping indicates variability in how different pipelines perform on specific datasets. A dataset exhibiting horizontal striping means that some pipelines handle it effectively while others do not, revealing differences in pipeline effectiveness for that dataset. Vertical striping indicates that a pipeline performs well on some datasets but poorly on others, demonstrating inconsistency in its effectiveness across varying dataset conditions. These observations underscore the importance of selecting CIL pipelines based on dataset characteristics (i.e., dataset-dependent manner in ZAutoCIL), as performance can vary significantly depending on the class imbalanced dataset.

4.2 Training for Surrogate Model in ZAutoCIL

Zero-shot automated class imbalanced learning (ZAutoCIL) can be addressed using the zero-shot HPO method, where the CIL pipelines we want to select can be envisioned as points within a geometric space and serve as the search space for HPO. To enable meta-learning for ZAutoCIL, we represent the predefined CIL pipeline hub as $P_k := \{R_k, \lambda_k^R, M_k, \lambda_k^M\}$. Specifically, R_k and M_k are encoded as one-hot vectors. λ_k^R and λ_k^M are represented as variable vectors defined by their respective hyperparameter values. As a result, the CIL pipelines are mapped to a geometric space defined by $\mathcal{P}$ and can be viewed as a configuration space.

We introduce a two-tower model, denoted as f_w parameterized by w, which serves as a surrogate model to estimate the performance observed by the CIL pipeline P_k on dataset D_t with meta-features ξ_t. It is able to embed the meta-features of class imbalanced datasets and CIL pipelines into a latent space. By leveraging joint learnable representations in the geometric space conditioned on datasets, the model can capture similarities and relationships between class imbalanced datasets and CIL pipelines. As a result, it enables efficient search in a zero-shot paradigm.

We are interested in identifying high-performance CIL pipelines based on their relative performance that can help distinguish the best CIL pipeline among candidates pipelines. Predicting the absolute value of a specific metric is less critical in this context. Approximating the absolute performance for every instance in the meta-dataset might result in selecting lower-performing pipelines for new class imbalanced datasets. To overcome this issue, we optimize the two-tower model as a surrogate model to rank CIL pipelines by a pairwise [4] loss based on the fact that the order matters: one CIL pipeline is preferred to another. In the learning-to-rank literature [4], the pairwise loss is a common choice to rank the candidate space. In this study, we choose the pairwise loss is formulated as:

$$
\begin{aligned}
-\frac{1}{|\boldsymbol{\varepsilon}|} \sum_{(i,j)\in\boldsymbol{\varepsilon}} & \left(y_{i,j} \log \sigma\left(f_w(P_i, \xi_t) - f_w(P_j, \xi_t)\right)\right. \\
& \left. + (1 - y_{i,j}) \log\left(1 - \sigma\left(f_w(P_i, \xi_t) - f_w(P_j, \xi_t)\right)\right)\right)
\end{aligned} \tag{2}
$$

where $\boldsymbol{\varepsilon} = \{(i,j) | P_i \in \mathcal{P}, P_j \in \mathcal{P}, \text{and } P_i \neq P_j\}$, and $y_{i,j}$ is defined as:

$$
y_{i,j} = \begin{cases} 1, & \text{if } r_i > r_j \\ 0.5, & \text{if } r_i = r_j \\ 0, & \text{if } r_i < r_j \end{cases}
$$

$\sigma(\cdot)$ is the sigmoid function and r is the performance metric of P.

To highlight the advantages of utilizing a pairwise loss, we also compare the performance of the same two-tower model optimized using a least-squares loss (pointwise) [13]:

$$
\arg\min_{w} \sum_{t=1}^{T} \sum_{k=1}^{K} \left(f_w(P_k, \xi_t) - y(P_k, \xi_t)\right)^2 \tag{3}
$$

5 Experiments

This section presents a detailed experimental evaluation. The aim of our experiments is to provide empirical evidence demonstrating the effectiveness of our method – Zero-shot automated class imbalanced learning (ZAutoCIL). We focus on answering the following research questions:

1. Does our method demonstrate improved performance in automated class imbalanced learning compared to established baseline approaches?
2. Does our method based on pairwise loss outperform pointwise loss in our zero-shot automated class imbalanced learning approach?
3. How effectively does the ZAutoCIL method infer missing values in the performance matrix?

5.1 Evaluation and Baselines

Evaluation We evaluate the effectiveness of our ZAutoCIL method based on the two-tower surrogate model using a leave-one-core-dataset-out protocol [17]. Specifically, we meta-train our surrogate model method on 99 out of the 100 datasets and test it on the held-out dataset. After obtaining a CIL pipeline recommendation from ZAutoCIL for each class imbalanced dataset, we conduct the CIL pipeline ten runs on the dataset and average the AUC-ROC metric. The AUC-ROC [5] metric is widely recognized in the class imbalance domain to evaluate the performance of the classifier. We report the resulting average performance for each dataset group. We ran all experiments using a Mac M1 Pro chip with 16 GB of RAM as the experimental environment.

Table 2. Average AUC-ROC and the Ranking for each group of all approaches

Metric	IR	Random	Similarity	Pointwise	**ZAutoCIL**
AUC-ROC	1.5−9	0.8732 ± 0.0140	0.8750 ± 0.0126	0.8722 ± 0.0125	**0.8811** ± 0.0120
	Part I	0.7599 ± 0.0255	0.8089 ± 0.0193	0.8174 ± 0.0172	**0.8535** ± 0.0117
	Part II	0.8402 ± 0.0090	0.8457 ± 0.0089	0.8547 ± 0.0083	**0.8677** ± 0.0052
	Part III	0.7936 ± 0.0327	0.8021 ± 0.0372	0.8231 ± 0.0316	**0.8631** ± 0.0203
	Overall	0.8150 ± 0.0229	0.8297 ± 0.0218	0.8400 ± 0.0188	**0.8660** ± 0.0129
Rank	1.5−9	2.50	2.54	2.63	**2.04**
	Part I	2.75	2.76	2.54	**1.40**
	Part II	2.85	2.68	2.31	**2.09**
	Part III	2.66	2.81	2.09	**1.69**
	Overall	2.68	2.71	2.36	**1.79**

Baselines To evaluate the performance of ZAutoCIL, we compare it with several baseline approaches. For a novel class imbalanced dataset and our 100 carefully designed CIL pipelines in the pipeline hub, the random selection baseline uniformly samples one CIL pipeline from these pipelines. Additionally, we assess our method against existing similarity measure methods used to recommend suitable pipelines in the class imbalanced learning domain [25]. Furthermore, we compare the performance of our method with the same two-tower model optimized using a least-squares loss (pointwise).

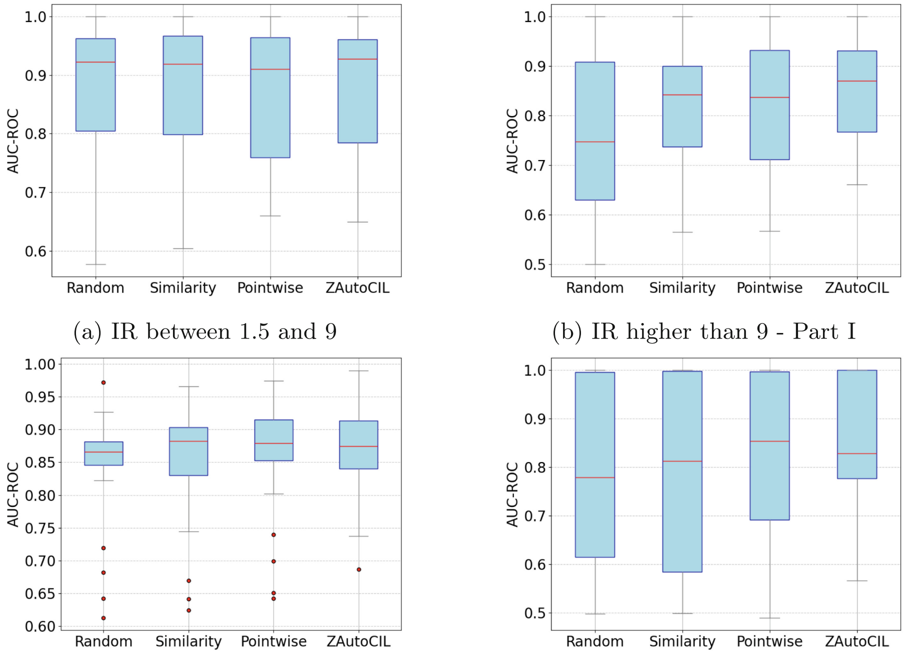
Meta-trained surrogate model For our meta-trained surrogate model (i.e., two-tower model), we utilize the Tensorflow Learning to Rank library [18]. The Adam optimizer optimizer is employed with a learning rate of 0.001 for meta-training over 1,000 epochs, determined by the meta-validation datasets. For both the CIL pipeline encoder and the class imbalanced datasets encoder, we adopt two hidden MLP layers with 32 and 16 dimensions, respectively. The across encoder uses two hidden MLP layers with 64 and 32 hidden dimensions, respectively. Additionally, we use mini-batch [9] training with a batch size of 32.

5.2 Results on Imbalanced Datasets for ZAutoCIL

The results of the average performance of each method and their ranking for each class imbalance dataset group are shown in Table 2. These groups include four categories: datasets with imbalance ratio between 1.5 and 9 (22 datasets), datasets with imbalance ratio higher than 9 – Part I (22 datasets), Part II (22 datasets) and Part III (34 datasets). These groups are sourced from the KEEL website as described in Sect. 4.1.

As shown in Table 2, our method consistently outperforms both random selection and similarity-based methods. ZAutoCIL achieves the highest AUC-ROC scores in all four groups. Moreover, ZAutoCIL achieves the lowest rank across all parts (lower is better), reaffirming its superior performance compared to other methods. These findings highlight the robust performance of ZAutoCIL. The superior performance of ZAutoCIL can be attributed to its sophisticated two-tower model architecture, which separately embeds both the meta-features of class imbalanced datasets and CIL pipelines into distinct latent spaces. It enables ZAutoCIL to learn latent interactions between class imbalanced datasets and pipelines, capturing non-linear relationships. In contrast, simple similarity measures often overlook these complex relationships and interactions between datasets and pipelines. They typically rely on straightforward metrics that may not adequately capture nuanced interactions. In addition, our method based on pairwise loss outperforms the pointwise (i.e. mean squared error) approach, verifying that focusing on relative performance and ranking-aware methods is more effective than predicting absolute values.

In Fig. 4, we present box plots to visualize the distribution of AUC-ROC scores across each group. These plots offer valuable insights into both the variability and the central tendencies of the performance metric within each dataset group. ZAutoCIL shows competitive median AUC-ROC scores and generally exhibits narrower interquartile ranges compared to random selection and simi-



(a) IR between 1.5 and 9

(b) IR higher than 9 - Part I

(c) IR higher than 9 - Part II

(d) IR higher than 9 - Part III

Fig. 4. The distribution of AUC-ROC scores across the four groups for random selection, similarity measure, pointwise, and ZAutoCIL.

Table 3. Average AUC-ROC and the Ranking for each group of all approaches across three sparse cases.

Metric	IR	ZAutoCIL	25%	50%	75%
AUC-ROC	1.5−9	0.8811 ± 0.0120	0.8820 ± 0.0113	0.8874 ± 0.0114	0.8850 ± 0.0108
	Part I	0.8535 ± 0.0117	0.8575 ± 0.0110	0.8480 ± 0.0123	0.8288 ± 0.0169
	Part II	0.8677 ± 0.0052	0.8701 ± 0.0059	0.8415 ± 0.0080	0.8652 ± 0.0071
	Part III	0.8631 ± 0.0203	0.8595 ± 0.0204	0.8643 ± 0.0217	0.8600 ± 0.0251
	Overall	0.8660 ± 0.0129	0.8664 ± 0.0128	0.8611 ± 0.0142	0.8601 ± 0.0158
Rank	1.5−9	2.40	2.63	2.45	2.23
	Part I	2.22	2.04	2.59	2.66
	Part II	2.54	2.31	2.55	2.45
	Part III	2.15	2.33	2.03	2.22
	Overall	**2.31**	2.33	2.36	2.37

larity measure. This suggests ZAutoCIL's robust performance in terms of recommending CIL pipeline across class imbalanced datasets with less variability. The pointwise loss version of ZAutoCIL has shown good results on several class imbalanced datasets; however, it also exhibits multiple outliers within dataset group Part-II and the lowest performance within dataset group Part-III. These visualizations provide complementary insights to the numerical results presented in Table 2, highlighting the effectiveness of ZAutoCIL in our experimental evaluation.

5.3 Results on Sparse Performance Matrix for ZAutoCIL

To assess the robustness of ZAutoCIL in managing missing values within the performance matrix, we perform experiments involving the random removal of 25%, 50%, and 75% entries. This setup aims to further examine the source of improvement in our method (i.e. the inference capabilities of ZAutoCIL), and to simulate realistic scenarios where data completeness is often compromised.

As shown in Table 3, ZAutoCIL consistently maintains strong performance at varying levels of sparsity. Specifically, even with only 25% of entries remaining, ZAutoCIl exhibit competitive AUC-ROC scores across different groups. This finding shows ZAutoCIL's robustness in recommending effective CIL pipelines for class imbalanced datasets, highlighting its resilience to missing data. This effectiveness of ZAutoCIL can be attributed to the architecture of the two-tower model. It is essentially a hybrid system that combines collaborative filtering and content-based filtering by embedding both the meta-features of class imbalanced datasets and the characteristics of CIL pipelines into a low-dimensional latent space. This approach allows ZAutoCIL to capture intricate correlations across CIL pipelines and leverage its generalization capability to accurately infer missing values within the performance matrix.

Specifically, CIL pipelines with similar configuration values tend to exhibit similar performance on the same class imbalanced datasets. Even when the performance of some CIL pipelines is missing, the two-tower neural network model can infer these missing values by leveraging the known performance of other similar CIL pipelines on the same dataset. Likewise, datasets with similar meta-features often require the similar effective CIL pipelines or exhibit similar performance distributions of CIL pipelines. If some datasets lack performance data for certain pipelines, ZAutoCIL can predict these missing values by considering the performance of those pipelines on similar datasets. Consequently, even when the meta-training dataset is sparse, the surrogate model is still able to infer the missing values effectively, allowing ZAutoCIL to maintain superior performance and robustly recommend CIL pipelines. The ranking information among these sparse cases is similar, which further verifies that ZAutoCIL can achieve good inference performance even under conditions of sparse data. In addition, Table 2 shows the ranking results of all the methods based on the full meta-training dataset, whereas the sparse experiment focuses on ZAutoCIL with the same architecture but varying levels of sparsity. This explains the differences in ranking results between Table 2 and Table 3 in the sparse experiment.

6 Conclusion and future work

In this paper, we extend the realm of Zero-shot AutoML to address class imbalanced learning on unseen class imbalanced datasets. We formalize this problem as Zero-shot Automated Class Imbalanced Learning (ZAutoCIL), leveraging knowledge from a meta-dataset of various CIL pipelines evaluated on a set of class imbalanced datasets. Specifically, we propose an approach using a two-tower model trained with a ranking objective to recommend CIL pipelines. Our findings are summarized as: (1) The performance of CIL pipelines varies significantly across different class imbalanced datasets, necessitating a dataset-dependent selection manner. (2) Our method outperforms established baseline approaches by effectively capturing dataset-pipeline relationships and correlations, while simple similarity measures struggle to grasp. (3) The use of pairwise loss in our approach proves superior to pointwise loss, emphasizing the importance of relative performance assessment over absolute regression-based methods in ZAutoCIL. (4) Our method demonstrates robust performance across three degrees of sparse performance matrices, owing to the generalization and inference capabilities of the two-tower model in ZAutoCIL.

We believe that our work represents a significant advancement for ZAutoCIL through the application of recommender system methods in real-world scenarios, where we need to handle diverse class imbalanced datasets while minimizing the search cost. However, there is considerable room for further enhancement. Future research should focus on adapting to other metrics in class imbalance domain learning, such as F1-score, and Recall. In addition, exploring a wider range of algorithms in imbalance learning can further enhance its applicability and effectiveness across different scenarios. Prioritizing multi-class imbalance learning is also a promising direction for future work.

Acknowledgements. This work is supported by the EPSRC Early Career Researchers International Collaboration Grants [EP/Y002539/1].

References

1. Alcalá-Fdez, J., Sanchez, L., Garcia, S., del Jesus, M.J., Ventura, S., Garrell, J.M., Otero, J., Romero, C., Bacardit, J., Rivas, V.M., et al.: Keel: a software tool to assess evolutionary algorithms for data mining problems. Soft. Comput. **13**, 307–318 (2009)
2. Alcobaça, E., Siqueira, F., Rivolli, A., Garcia, L.P.F., Oliva, J.T., de Carvalho, A.C.P.L.F.: Mfe: Towards reproducible meta-feature extraction. Journal of Machine Learning Research **21**(111), 1–5 (2020), http://jmlr.org/papers/v21/19-348.html
3. Chawla, N.V.: Data mining for imbalanced datasets: An overview. Data mining and knowledge discovery handbook pp. 875–886 (2010)
4. Chen, W., Liu, T.Y., Lan, Y., Ma, Z.M., Li, H.: Ranking measures and loss functions in learning to rank. Advances in Neural Information Processing Systems **22** (2009)
5. Erickson, B.J., Kitamura, F.: Magician's corner: 9. performance metrics for machine learning models (2021)
6. Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., Hutter, F.: Efficient and robust automated machine learning. Advances in neural information processing systems **28** (2015)
7. Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., Bing, G.: Learning from class-imbalanced data: Review of methods and applications. Expert Syst. Appl. **73**, 220–239 (2017)
8. He, H., Garcia, E.A.: Learning from imbalanced data. IEEE Trans. Knowl. Data Eng. **21**(9), 1263–1284 (2009)
9. Hinton, G., Srivastava, N., Swersky, K.: Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. Cited on **14**(8), 2 (2012)
10. Hutter, F., Kotthoff, L., Vanschoren, J.: Automated machine learning: methods, systems, challenges. Springer Nature (2019)
11. Krawczyk, B.: Learning from imbalanced data: open challenges and future directions. Progress in artificial intelligence **5**(4), 221–232 (2016)
12. LemaÃŽtre, G., Nogueira, F., Aridas, C.K.: Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. J. Mach. Learn. Res. **18**(17), 1–5 (2017)
13. Li, H.: A short introduction to learning to rank. IEICE Trans. Inf. Syst. **94**(10), 1854–1862 (2011)
14. Liu, X.Y., Zhou, Z.H.: Ensemble methods for class imbalance learning. Imbalanced learning: Foundations, algorithms, and applications pp. 61–82 (2013)
15. Moniz, N., Cerqueira, V.: Automated imbalanced classification via meta-learning. Expert Syst. Appl. **178**, 115011 (2021)
16. Nguyen, D.A., Kong, J., Wang, H., Menzel, S., Sendhoff, B., Kononova, A.V., Bäck, T.: Improved automated cash optimization with tree parzen estimators for class imbalance problems. In: 2021 IEEE 8th international conference on data science and advanced analytics (DSAA). pp. 1–9. IEEE (2021)
17. Öztürk, E., Ferreira, F., Jomaa, H., Schmidt-Thieme, L., Grabocka, J., Hutter, F.: Zero-shot automl with pretrained models. In: International Conference on Machine Learning. pp. 17138–17155. PMLR (2022)

18. Pasumarthi, R.K., Bruch, S., Wang, X., Li, C., Bendersky, M., Najork, M., Pfeifer, J., Golbandi, N., Anil, R., Wolf, S.: Tf-ranking: Scalable tensorflow library for learning-to-rank. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 2970–2978 (2019)
19. Rezvani, S., Wang, X.: A broad review on class imbalance learning techniques. Appl. Soft Comput. **143**, 110415 (2023)
20. Rivolli, A., Garcia, L.P., Soares, C., Vanschoren, J., de Carvalho, A.C.: Characterizing classification datasets: a study of meta-features for meta-learning. arXiv preprint arXiv:1808.10406 (2018)
21. Singh, P., Vanschoren, J.: Automated imbalanced learning. arXiv preprint arXiv:2211.00376 (2022)
22. Tornede, A., Wever, M., Hüllermeier, E.: Extreme algorithm selection with dyadic feature representation. In: International Conference on Discovery Science. pp. 309–324. Springer (2020)
23. Truong, A., Walters, A., Goodsitt, J., Hines, K., Bruss, C.B., Farivar, R.: Towards automated machine learning: Evaluation and comparison of automl approaches and tools. In: 2019 IEEE 31st international conference on tools with artificial intelligence (ICTAI). pp. 1471–1479. IEEE (2019)
24. Vanschoren, J.: Meta-learning: A survey. arXiv preprint arXiv:1810.03548 (2018)
25. Vieira, P.M., Rodrigues, F.: An automated approach for binary classification on imbalanced data. Knowledge and Information Systems pp. 1–21 (2024)
26. Wang, K., Xue, Q., Lu, J.J.: Risky driver recognition with class imbalance data and automated machine learning framework. Int. J. Environ. Res. Public Health **18**(14), 7534 (2021)
27. Wang, S., Yao, X.: Multiclass imbalance problems: Analysis and potential solutions. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **42**(4), 1119–1130 (2012)
28. Wang, W., Zheng, V.W., Yu, H., Miao, C.: A survey of zero-shot learning: Settings, methods, and applications. ACM Transactions on Intelligent Systems and Technology (TIST) **10**(2), 1–37 (2019)
29. Wang, Z., Wang, S.: Online automated machine learning for class imbalanced data streams. In: 2023 International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE (2023)
30. Winkelmolen, F., Ivkin, N., Bozkurt, H.F., Karnin, Z.: Practical and sample efficient zero-shot hpo. arXiv preprint arXiv:2007.13382 (2020)
31. Wistuba, M., Grabocka, J.: Few-shot bayesian optimization with deep kernel surrogates. arXiv preprint arXiv:2101.07667 (2021)
32. XU, S., Wang, J.: On strong convergence of the two-tower model for recommender system (2021)
33. Yang, F., Zou, Q.: maml: an automated machine learning pipeline with a microbiome repository for human disease classification. Database **2020**, baaa050 (2020)
34. Zhang, J., Sun, Z., Qi, Y.: Autoidl: Automated imbalanced data learning via collaborative filtering. In: International Conference on Knowledge Science, Engineering and Management. pp. 96–104. Springer (2020)

Online Automated Imbalanced Learning via Adaptive Thompson Sampling

Zhaoyang Wang and Shuo Wang(✉)

School of Computer Science, University of Birmingham, Birmingham B15 2TT, UK
zxw180@student.bham.ac.uk, s.wang.2@bham.ac.uk

Abstract. Existing Automated machine learning (AutoML) systems have achieved considerable success in offline machine learning. Nonetheless, they are not applicable to data streams that requires real-time training and prediction, not to mention class imbalanced data streams. This paper proposes an Online Automated framework designed for imbalanced data streams learning, called OAutoIDSL. Firstly, we adopt and improve Thompson Sampling (TS) with an imbalanced reward design for combined algorithm selection and hyperparameter tuning (CASH) that enables online learning and optimisation. Secondly, we introduce two mechanisms to further tackle data non-stationarity and class imbalance – discounting outdated knowledge and adaptive weight tuning in imbalanced rewards. The effectiveness of our approach is demonstrated through the empirical evaluation on a set of synthetic imbalanced data streams encompassing various stationary and non-stationary scenarios, along with four real-world data sets.

Keywords: Class imbalance · Online AutoML · ensemble learning · data streams · multi-armed bandits

1 Introduction

Automated machine learning (AutoML) aims at Combined Algorithm Selection and Hyperparameter tuning (CASH) and has achieved significant success in recent studies [4]. As more and more real-world applications collect data over time in the form of data streams or sequences, most existing offline AutoML systems become unsuitable in an online setting because no pre-collected data are available and decisions must be made in real-time [17,18].

A few recent attempts have been made on online AutoML, aiming to automatically configure online learning models and their hyperparameters. OAML [2] and OnlineAutoClust [3] utilize a two-stage procedure that involves a search phase and an online learning phase. However, they do not work strictly online, as the framework learning process is temporarily interrupted to conduct time-consuming algorithm searches to adapt to new data distribution. Moreover, the associated search cost is not considered. In contrast, EvoAuoML [6] consistently

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 156–172, 2025.
https://doi.org/10.1007/978-3-031-78383-8_11

evolves the search space at a regular interval time step through an evolutionary optimization method and maintains the real-time ensemble prediction. At each time step, ChaCha [18] continually identifies a Champion and schedules a set of Challengers according to sample complexity bounds, achieving the sublinear regret by relying on a ConfigOracle. Even though these approaches were proposed for online learning algorithms, they made a default assumption in classification problems that the class distribution is relatively balanced. They neglect the fact that the skewed distribution can degrade a classifier's performance significantly and hence affect the choice of the best algorithm and its hyperparameters. Moreover, data distribution can be changing dynamically over time (a.k.a concept drift) in data streams, which makes the problem more challenging. This paper fills in this gap by developing an online automated framework that can perform searching and training at every time step, and handle class imbalance and data non-stationarity simultaneously. It includes five most popular online ensemble learning algorithms as the candidate learning algorithms in the framework search space, which are UOB, OOB, UnerOverBagging, CSB2 and AdaC2 [14,15]. The performance of these methods is heavily dependent on the choice of hyperparameters.

To achieve our goal, we formulate the online CASH task as a Multi-Armed Bandit (MAB) problem, as detailed in Sect. 3. Thompson Sampling (TS) is introduced to address online CASH, which has been widely recognized for its strong empirical performance among the solutions to the MAB problem [1]. In order to handle class imbalance in data, we improve TS and propose Imbalanced Thompson Sampling (ITS) by imbalance reward design. It automatically selects from a set of candidate online ensemble models designed to proficiently handle imbalanced data streams on-the-fly. In this context, each configuration of the online algorithm is treated as an arm, with its corresponding performance serving as the reward. The objective is to maximize the cumulative rewards derived from the execution of the selected candidate algorithms. Then, we add new mechanisms to ITS, discounting old knowledge and adaptive weight tuning for imbalanced rewards, as well as their combined effects. These investigate whether they can further benefit non-stationary and class imbalanced data streams. In total, four versions of ITS are compared: ITS, Discounting ITS (DITS), Weighted ITS (WITS) and Weighted Discounting ITS (WDITS). This is the pioneering online automated framework – OAutoIDSL crafted for online ensemble learning algorithms that are capable of handling imbalanced data streams on-the-fly. We provide extensive experimental evaluations on synthetic and real-world data sets to validate the effectiveness of our approach. The results show that: a) In stationary imbalanced data streams, ITS consistently obtains the highest performance, especially in high degree of class imbalance. b) In non-stationary scenarios, ITS demonstrates superior performance with changing Imbalance Ratio (IR) and fixed concepts, while DITS competes well with a fixed IR and changing concepts. c) For real-world datasets, ITS performs strongly in G-mean across all real data sets (IR below 30%). WITS consistently excels in minority class performance for most situations.

2 Related work

2.1 AutoML for online learning

Several recent studies have looked into AutoML within an online setting. In the Online Automated Machine Learning (OAML) system [2], asynchronous genetic programming has been employed to facilitate online classifier searching for data streams. OnlineAutoClust utilizes Bayesian optimization (BO) - Tree of Parzen Estimators (TPE) to execute CASH in the context of online clustering [3]. Both of these approaches assume the availability of an initial data batch in advance. Moreover, they do not perform pipeline search strictly online, which is only triggered after data distribution changes, and the search cost is not considered. This infringes upon the real-time prediction requirements for online learning. Additionally, leveraging BO or similar cutting-edge offline methods is impractical for real-time decisions in online learning [5]. EvoAutoML is another recently proposed online automated framework. It continuously and adaptively selects models and hyperparameters, sustaining an ensemble prediction through periodic updates using a straightforward evolutionary optimization method [6]. ChaCha consistently makes hyperparameter decisions based on statistical test results derived from sample complexity bounds, taking into account resource limitations [18]. It dynamically balances the interaction between a champion and a range of challengers at each time step. Although the strong assumption about ConfigOracle aids in achieving sublinear results in online learning, obtaining it in real-world applications is challenging. The key component in an online automated framework lies in the methods for handling online CASH. Recent research closely related to the domain of online CASH encompasses areas such as online hyperparameter tuning [5,7] and online model selection [12]. Although many of these studies provide theoretical insights, they primarily focus on very specific hyperparameters or models. Furthermore, it's noteworthy that all the aforementioned frameworks and methods overlook conducting online CASH for ensemble machine learning algorithms designed to handle online imbalanced data streams.

2.2 AutoML for Class Imbalanced Data

Some offline AutoML approaches have integrated techniques to tackle class imbalance problems in data. Vieira et al. [13] utilize the Frobenius norm to measure similarity among class imbalanced datasets and recommend a combination of resampling and classification methods from 220 search spaces based on accumulated knowledge given a target dataset. Automated imbalanced datasets learning (AutoIDL) [19] selects suitable combinations of resampling methods and classification algorithms. AutoIDL exploits user-based collaborative filtering to recommend a CIL pipeline. The tree Parzen estimator approach demonstrates superior performance with five candidate classification algorithms and 21 resampling approaches [10]. The AutoBalance framework considers a wider search space, leveraging the modified GAMA framework [11]. All of these approaches focus on offline settings, which means that it is a one-off optimization procedure and cannot handle data streams. In summary, none of the existing automated framework approaches can learn from data streams online and handle CASH for ensemble algorithms designed in online imbalanced data streams simultaneously. Bridging this gap constitutes the core objective of our work.

3 Preliminaries

In this section, we provide an explicit definition of online CASH in the context of one-by-one data stream learning. Subsequently, we reformulate online CASH as a multi-armed bandits (MAB) problem that motivates our approach proposed in this paper.

3.1 Online CASH for Data Streams

The online CASH is able to be treated as an iterative decision making problem over time in a data stream [12,18]. Instances are drawn from the instances space $\mathcal{I} = \mathcal{X} \times \mathcal{Y}$, and a set of candidate algorithms $\mathcal{A} = \{A^1, ..., A^m\}$ (every algorithm has a hyperparameter space $\{\Lambda^1, ..., \Lambda^m\}$). The instances arrive continually over time. An algorithm and its hyperparameter $s(h_t, i_t) = A_t(\Lambda_t) \in \mathcal{A}$ are selected by the meta-learner to handle the current instance $i_t \in \mathcal{I}$, i.e. $s : \mathcal{H} \times \mathcal{I} \rightarrow \mathcal{A}$. Here, $h_t \in \mathcal{H}$ refers to the history of the choosing process, which includes $\{(i_j, A_j(\Lambda), l_j)\}_{j=1}^{t-1}$, i.e., accumulated instances observed so far, algorithms applied corresponding to hyperparameters, as was well as the corresponding losses evaluated ($l_j = l(i_j, A_j(\Lambda))$). As a result, the objective of online CASH is to minimize the average loss, defined as $\mathcal{L}(s) = T^{-1} \sum_{t=1}^{T} l(i_t, s(h_t, i_t))$. The optimal meta-learner is represented as: $s^*(h_t, i_t) := \arg\min_{A(\Lambda)\in\mathcal{A}} \mathbb{E}[l(i_t, A(\Lambda))) \mid h_t]$.

3.2 Online CASH as a MAB problem

The online CASH can be modeled as a MAB problem consisting of a set of candidate arms/algorithms denoted as $A(\Lambda)$ in a data stream. The meta-learner is provided with $K = |A(\Lambda)|$ arms. At each time step $t = 1, 2, ..., T$, it is requested to choose one of the arms $A(\Lambda)_t$ for the instance i_t, which is consider as pulling one of the arms in MAB community. Simultaneously, the meta-learner incurs a loss with this decision. We denote the arm that is pulled at time step t as m_t. Ideally, the meta-learner pick an arm with the smallest expected loss for the given instance at each time step t, i.e., $A_t^*(\Lambda) \in \arg\min_{A(\Lambda)\in\mathcal{A}} \mathbb{E}[l(i_t, A(\Lambda))]$, which is referred to optimal strategy and serves as a benchmark.

4 Online Automated Imbalanced Data Streams Learning (OAutoIDSL)

4.1 Reward Design in MAB for Imbalanced Data Streams

Most of the reward distribution designs in the traditional MAB are often single when the environment interacts with the learner. Such designs are impractical when addressing online CASH for imbalanced stream learning. Several essential factors need to be taken into consideration. In a stationary imbalanced scenario, the rewards revealed for the meta-learner need to guide exploration and exploitation based on the performance of two classes of each candidate algorithm (i.e., a combination of online model and hyperparameters) within the search space. This aids the meta-learner in selecting a candidate algorithm with good trade-off performance of two classes, preventing the selection of an algorithm biased toward the majority class during the trade-off exploration and exploitation process over time. In a non-stationary imbalanced scenario, the class imbalance can change over time [16]. This necessitates the meta-learner to automatically tune or adaptively select candidate algorithms for the current time step and new data distributions, taking into account the dynamic properties inherent in imbalanced data streams. Therefore, the rewards disclosed to the meta-learner should reflect the recent performance of candidate algorithms, and the bandit strategy based on these rewards should automatically fine-tune to adapt to the new imbalanced data streams. In other words, the meta-learner needs to choose appropriate models and hyper-parameters on the fly according to the designed rewards and bandit strategy at the current time step. Hence, the meta-learner for online CASH is imperative to be kept updated for selecting good models and hyperparameters so that it can maintain high performance on the current minority class without compromising performance on the current majority class.

4.2 Thompson Sampling Improvement

We adopt Thompson Sampling (TS) and improve it into four versions through the new mechanisms (i.e., discounting old knowledge and adaptive weight tuning). The combined impacts are also explored. Firstly, Imbalance Thompson Sampling (ITS) is proposed to address class imbalance in online CASH. We then further extend it into Discounting Imbalance Thompson Sampling (DITS), Weighted Imbalance Thompson Sampling (WITS) and Weighted Discounting Imbalance Thompson Sampling (WDITS).

Algorithm 1 Thompson Sampling

```
Input: $M = A(\Lambda)$ candidate algorithms and hyperparameters. Initialize: For each $i \in [M]$, $a^i = b^i = 1$.
for time steps $t = 1, ..., T$ do
    For each $i \in [M]$, sample $\hat{\theta}^i \sim Beta(a^i, b^i)$ .
    Choose an online algorithm $m_t \leftarrow \arg\max_{i \in [M]} \hat{\theta}^i$ and observe a real-valued payoff $r_{m_t}$.
    Update posterior distributions:
        $a^i \leftarrow a^i + r_{m_t}, b^i \leftarrow b^i + (1 - r_{m_t})$
end for
```

ITS The algorithm 1 illustrates the original TS [1]. When applied to online model selection and hyperparameter tuning for classification issues, TS assumes a Bernoulli reward. In this context, the reward (r_{m_t}) is assigned as 0 (representing a wrong algorithm prediction) or 1 (indicating a correct algorithm prediction), with the parameters adhering to a Beta distribution. TS initially assumes that algorithm i follows a $Beta(1,1)$ prior. At time t, after observing a^i successes and b^i failures, each candidate algorithm has the distribution $Beta(a^i, b^i)$ (as shown in line 2). TS samples from these distributions, and the algorithm i with the largest sample value is chosen, followed by the revelation of the reward for this choice (as seen in Line 3). When the algorithm is not chosen, a^i and b^i remain the same as the last time step, and for the chosen algorithm r_{m_t}, the corresponding posterior distributions is updated in Line 4. However, TS faces limitations in handling online CASH under imbalance cases. The use of a single reward fails to adequately balance the trade-off between exploration and exploitation, particularly concerning the performance of the two classes within an algorithm. Consequently, TS tends to favor algorithms with superior performance in the majority class. To address this issue, TS is improved to accommodate imbalanced MAB with Bernoulli distributions into a new approach called Imbalanced Thompson Sampling (ITS), as outlined in Algorithm 2. In line 2 of the ITS Algorithms, a_0^i signifies the times in which algorithm i attains a reward of 1 for class 0, while b_0^i represents failure prediction times for class 0. a_1^i denotes the times that algorithm i receives a reward of 1 for class 1, b_1^i symbolizes failure prediction times for class 1. Next, ITS samples simultaneously from $\hat{\theta}_0^i$ and $\hat{\theta}_1^i$ for each algorithm i. ITS selects an online algorithm m_t with the highest sampled sum-reward between classes (as indicated in line 3). Furthermore, both rewards $r_{m_t}(0)$ and $r_{m_t}(1)$ are revealed. Subsequently, for the chosen algorithm, ITS proceeds to update the posterior distributions of the two events (as outlined in Line 4). As is evident, ITS straightforwardly trades off exploration and exploitation among a set of algorithm $|M|$ options based on their performance across two classes. This method is intuitive, easy to implement and efficient.

Algorithm 2 ITS and DITS

```
Input: $M = A(\Lambda)$ candidate algorithms and hyperparameters.
Initialize: For each $i \in [M]$, $a_0^i = b_0^i = 1$, $a_1^i = b_1^i = 1$
1: for time steps $t = 1, ..., T$ do
2:     For each $i \in [M]$, sample $\hat{\theta}_0^i \sim Beta(a_0^i, b_0^i)$ and $\hat{\theta}_1^i \sim Beta(a_1^i, b_1^i)$.
3:     Choose an online algorithm $m_t \leftarrow \arg\max_{i \in [M]} (\hat{\theta}_0^i + \hat{\theta}_1^i)$ and observe two real-valued payoff $r_{m_t}(0)$ and $r_{m_t}(1)$.
4:     For ITS: Update posterior distributions
       $a_0^i \leftarrow a_0^i + r_{m_t}(0), b_0^i \leftarrow b_0^i + (1 - r_{m_t}(0))$
       $a_1^i \leftarrow a_1^i + r_{m_t}(1), b_1^i \leftarrow b_1^i + (1 - r_{m_t}(1))$
5:     For DITS: Update posterior distributions
       $a_0^i \leftarrow \gamma a_0^i + r_{m_t}(0), b_0^i \leftarrow \gamma b_0^i + (1 - r_{m_t}(0))$
       $a_1^i \leftarrow \gamma a_1^i + r_{m_t}(1), b_1^i \leftarrow \gamma b_1^i + (1 - r_{m_t}(1))$
6: end for
```

DITS We propose DITS as an extension of the ITS approach, aiming to adapt it to handle non-stationary data streams. To achieve this, we employ a discounting mechanism, specifically applied to a_0^i, b_0^i, a_1^i, and b_1^i in line 5 of Algorithm 2, enabling the method to forget outdated knowledge related to class 0 and class 1. The discounting factor plays a crucial role in addressing the dynamic changes inherent in data streams within non-stationary scenarios by mitigating the influence of past observations. This forgetting strategy empowers DITS to select algorithm i based on recent performance, thereby enhancing its adaptability to evolving data streams.

Algorithm 3 WITS and WDITS

```
Input: $M = A(\Lambda)$ candidate algorithms and hyperparameters, time-decay factor $\alpha$.
Initialize: For each $i \in [M]$, $a_0^i = b_0^i = 1$, $a_1^i = b_1^i = 1$
for time steps $t = 1, ..., T$ do
    For each $i \in [M]$, sample $\hat{\theta}_0^i \sim Beta(a_0^i, b_0^i)$ and $\hat{\theta}_1^i \sim Beta(a_1^i, b_1^i)$
    Calculate $s_0^t = \alpha s_0^{t-1} + I_{y^t=c_0}(1-\alpha)$ and $s_1^t = \alpha s_1^{t-1} + I_{y^t=c_1}(1-\alpha)$
    if $y_t = 1$ and $s_0^t < s_1^t$ then
        set $l = s_0^t / s_1^t$ and $p_t^i = (1-l)\hat{\theta}_0^i + l\hat{\theta}_1^i$
    else if $y_t = 0$ and $s_0^t > s_1^t$ then
        set $l = s_1^t / s_0^t$ and $p_t^i = l\hat{\theta}_0^i + (1-l)\hat{\theta}_1^i$
    else
        $p_t^i = \hat{\theta}_0^i + \hat{\theta}_1^i$
    end if
    Choose an online algorithm $m_t \leftarrow \arg\max_{i \in [M]} p_t^i$ and observe two real-valued
    payoff $r_{m_t}(0)$ and $r_{m_t}(1)$.
    For WITS: Update posterior distributions
    $a_0^i \leftarrow a_0^i + r_{m_t}(0), b_0^i \leftarrow b_0^i + (1 - r_{m_t}(0))$
    $a_1^i \leftarrow a_1^i + r_{m_t}(1), b_1^i \leftarrow b_1^i + (1 - r_{m_t}(1))$
    For WDITS: Update posterior distributions
    $a_0^i \leftarrow \gamma a_0^i + r_{m_t}(0), b_0^i \leftarrow \gamma b_0^i + (1 - r_{m_t}(0))$
    $a_1^i \leftarrow \gamma a_1^i + r_{m_t}(1), b_1^i \leftarrow \gamma b_1^i + (1 - r_{m_t}(1))$
end for
```

WITS and WDITS We further propose an adaptive weight tuning mechanism according to the imbalance information within online environment and integrate it into the ITS and DITS. WITS and WDITS are shown in Algorithm 3. The pivotal feature of adaptive weight tuning mechanism works on sampled rewards from two classes for each candidate algorithm, with making real-time adjustments based on imbalance information at the recent data stream. This entails allocating more weight to candidate algorithms within the search space that perform well in the minority class over time. To actively monitor real-time imbalance information in an online setting, we employ the time-decay class probability [15]. This approach allows us to identify the minority/majority class and to quantify the current class size ratio between classes in the recent time step. At each time step t, the size of each class is incrementally updated according to (1).

$$s_k^t = \alpha s_k^{t-1} + I_{y^t=c_k}(1-\alpha) \tag{1}$$

In this equation, s_k^t denotes the decaying size of class c_k at time step t. The indicator function $I_{y^t=c_k}$ is equal to 1 if the true class label of x_t is c_k and 0 otherwise. The parameter α $(0 < \alpha < 1)$ serves as a predetermined time-decay factor, with the intention of emphasizing the percentage of current class over time while mitigating the influence of older instances. In our research, we assign the labels 0 and 1 to c_0 and c_1, respectively. The time-decay class sizes for c_0 and c_1 are represented by s_0^t and s_1^t, respectively.

The adaptive weight mechanism is presented in line 3 to 10 of Algorithms 3. At any given time t, the time-decay class size, s_0^t and s_1^t are calculated (as illustrated in Line 3), reflecting the recent class percentage for each class over time in an data stream. This information is employed to assign weights. For example, in the case of class 1, if the current instance belongs to class 1 $y_t = 1$ and $s_0^t < s_1^t$ (as highlighted in line 4), it signifies that class size 1 has a higher percentage at the current time step. Consequently, the weight l is adjusted to a smaller value, leading to a reduction in the weight assigned to the sampled rewards for this class, denoted as $\hat{\theta}_1^i$ (as illustrated in line 5). In the case of WITS (as shown in line 12), the posterior update is the same as in ITS. For WDITS (as shown line 13), the posterior update involves forgetting outdated knowledge.

5 Experiment

In this section, we design several experiments to investigate the effectiveness of OAutoIDSL, focusing on three key research questions (RQ): (RQ1) How do they perform on stationary data streams with different levels of class imbalance? (RQ2) How do they perform on non-stationary data streams? (RQ3) How do they perform on real-world data sets?

5.1 Experimental Setup

Data sets Two frequently employed synthetic data generators, SINE and SEA [16], were utilized in our experiments. We designed diverse scenarios involving class imbalances, both with and without drifts, to assess the effectiveness of the proposed methods. Furthermore, our experiment incorporates four real-world data sets. These data sets span diverse domains, including finance banking, weather, and environment [8,16]. Specifically: 1) The Credit Card Frauds (Frauds) data set comprises 492 frauds out of 284,807 transactions with IR–0.172%. 2) The task of The Given Me Some Credit (GMSC) is to determine whether a loan should be granted. The minority class constitutes approximately 7% of all borrowers in 120,269, excluding missing instances. 3) The Weather data set aims to predict whether it will rain or not, with 5,698 instances of rain (minority class) and an IR of around 30%. 4) The Forest cover type (Covtype) data set is transformed into a binary format known as Covtype34, where class 3 is designated as the majority (3,5754 instances) and class 4 as the minority (2747 instances), resulting in an IR of approximately 7%.

Metrics and Evaluation The geometric mean (G-mean), renowned for its insensitivity to class imbalance, is commonly utilized as a performance metric in imbalance learning [16]. In the context of binary classification, the G-mean is expressed as $\sqrt{(Recall\_0) \times (Recall\_1)}$, where $Recall\_0$ represents the recall for class 0 and $Recall\_1$ signifies the recall for class 1. A higher G-mean suggests that the algorithm achieves greater accuracy across all classes. The prequential evaluation, also known as test-then-train, is commonly employed to evaluate and compare methods in data streams. The instance is first used to test algorithms before proceeding to update itself. Additionally, Wilcoxon Sign Rank tests assesses the statistical significance of methods based on 30 runs in this paper, with a significance level set to 0.05.

Candidate Ensemble Classification Algorithms in the Search Space The search space of OAutoIDSL includes five prevalent online ensemble models specialised in imbalanced data stream learning. These algorithms are the variations of online Bagging (UOB, OOB, OnlineUnderOverBagging) or online Boosting (OnlineCSB2 and OnlineAdaC2) [14,15]. The key hyperparameters of each of the algorithms are listed in Table 1. The performance of these models is particularly sensitive to the types of base classifiers employed (i.e., Logistic Regression-LR, Perceptron, Hoeffding Trees-HT, and Naive Bayes-NB) and resampling techniques (i.e., time-decay factor, sampling rate, cost of negative-C_p, and cost of false positive-C_n). They form part of the search space. The default values of base classifiers are implemented within the online learning library River [9].

Algorithms and Baselines Currently, there are no automated framework specifically tailored to address the challenges of model selection and hyperparameter tuning for online algorithms designed to handle online imbalanced data streams. Consequently, we conduct a comparative analysis, evaluating OAutoIDSL against various intuitive alternatives. We customize EvoAutoML's search space by using the classifiers in Table 1, for a fair comparison [6]. At each time step, the algorithm with the highest performance is chosen for making predictions. This assesses whether OAutoIDSL built upon TS-based techniques outperforms EvoAutoML based on evolutionary optimization techniques. To study the effectiveness of the four TS techniques, We integrate all four versions into OAutoIDSL respectively, plus the original TS. Therefore, there are 5 versions of OAutoIDSL in total that join the comparison.

Table 1. The models and hyperparameters in the search space

Models	Hyperparameters	Values

UOB	base classifier
	time decay factor
		{LR, Perceptron, HT, NB}
		{0.5, 0.9}

OOB	base classifier
	time decay factor
		{LR, Perceptron, HT, NB}
		{0.5, 0.9}

UnderOverBagging	base classifier
	sampling rate
		{LR, Perceptron, HT, NB}
		{2, 5}

OnlineCSB2	base classifier
	$\{(C_p,C_n)\}$
		{LR, Perceptron, HT, NB}
		{(1, 0.1), (1, 0.5)}

OnlineAdaC2	base classifier
	$\{(C_p\ C_n)\}$
		{LR, Perceptron, HT, NB}
		{(1, 0.1), (1, 0.5)}

5.2 Stationary Data Streams

The objective of this experiment is to answer RQ1. SINE and SEA are used to generate data streams characterized by four imbalance levels (i.e., 0.5%, 1%, 5%, and 10%), respectively. Each data stream consists of $100,000$ instances, with class 1 designated as the minority class in this section. The decay factor γ and α are set to 0.9 based on our initial experiments in this paper. We compare the minority class recall and G-mean obtained in the final step across all methods. Each approach is repeated 30 times on every data stream. The average recall and G-mean in the final step showcase the ultimate prequential performance after these methods process all instances over 30 runs. Their means and standard deviations are shown in Table 2 and Table 3.

Table 2. The Final Step Recall_1 and G-mean on four data streams with varying IR for SINE.

		SINE-0.5%	SINE-1%	SINE-5%	SINE-10%
Recall_1	EvoAutoML	0.5552±0.0484	0.6397±0.0276	0.8755±0.0218	0.7628±0.0631
	TS	0.8501±0.0395	0.9164±0.0203	0.9781±0.0097	0.9907±0.0031
	ITS	**0.9071±0.0149**	**0.9591±0.0055**	**0.9898±0.0028**	**0.9964±0.0006**
	WITS	0.9013±0.0106	0.9450±0.0138	**0.9913±0.0016**	**0.9939±0.0029**
	DITS	0.5832±0.0617	0.7109±0.0558	0.9595±0.0264	0.9936±0.0025
	WDITS	0.6102±0.0442	0.6980±0.0403	0.9819±0.0083	0.9916±0.0032
G-mean	EvoAutoML	0.6837±0.0240	0.7290±0.0134	0.7845±0.0145	0.7814±0.0169
	TS	0.9045±0.0331	0.9547±0.0125	0.9884±0.0053	0.9950±0.0016
	ITS	**0.9508±0.0089**	**0.9789±0.0030**	**0.9945±0.0015**	**0.9969±0.0009**
	WITS	0.9364±0.0060	0.9549±0.0108	0.9944±0.0009	**0.9957±0.0020**
	DITS	0.6803±0.0505	0.7300±0.0408	0.9726±0.0186	0.9953±0.0021
	WDITS	0.7233±0.0376	0.7664±0.0295	0.9870±0.0063	0.9937±0.0023

Table 3. The Final Step Recall_1 and G-mean on four data streams with varying IR for SEA.

		SEA-0.5%	SEA-1%	SEA-5%	SEA-10%
Recall_1	EvoAutoML	0.6439±0.0479	0.6934±0.0141	0.9352±0.0014	0.9746±0.0017
	TS	0.8246±0.0403	0.8598±0.0186	0.9645±0.0145	0.9777±0.0093
	ITS	**0.9013±0.0132**	**0.9603±0.0055**	**0.9861±0.0023**	**0.9946±0.0012**
	WITS	0.8938±0.0177	0.9470±0.0069	**0.9811±0.0076**	**0.9962±0.0006**
	DITS	0.5050±0.0672	0.8291±0.0541	0.9718±0.0143	0.9907±0.0022
	WDITS	0.7693±0.0469	0.8358±0.0395	0.9779±0.0074	0.9907±0.0061
G-mean	EvoAutoML	0.7681±0.0007	0.7964±0.0091	0.9466±0.0002	0.9541±0.0002
	TS	0.8942±0.0292	0.9255±0.0102	0.9809±0.0078	0.9877±0.0050
	ITS	**0.9434±0.0090**	**0.9794±0.0031**	**0.9895±0.0021**	**0.9958±0.0010**
	WITS	0.9288±0.0113	0.9562±0.0070	0.9852±0.0046	**0.9976±0.0004**
	DITS	0.6314±0.0554	0.8432±0.0441	0.9775±0.0096	0.9929±0.0021
	WDITS	0.7918±0.0341	0.8627±0.0282	0.9781±0.0086	0.9948±0.0032

In Table 2 and Table 3, it is evident that ITS consistently achieves high performance in Recall_1 and G-mean. This finding is supported by our statistical tests. For instance, at a 0.05 significance level, Wilcoxon Sign Rank tests on Recall_1 in SINE-0.5% reject the null hypothesis that ITS and WITS performance similarly with a p-value of 0.0206. This suggests that ITS achieves better Recall_1 than WITS. Due to space constraints, p-values will be presented for tests only when the methods we are concerned with show no significant differences. WITS also performs well in several cases. For example, no significant difference is found between ITS and WITS (p-value 0.9310) in terms of Recall_1 in SINE-5% and for Recall_1 (p-value 0.7188) in SINE-10%. Similarly, no significant difference is observed in SEA-5% for Recall_1 (p-value 0.9183)) and in SEA-10% for Recall_1 (p-value 0.0507)and for G-mean (p-value 0.0519). While WITS obtains similar good performance in several cases, the weight mechanism in the stationary scenario shows limited improvement. This implies that both ITS and WITS tend to select similar models and hyperparameters over time.

In contrast to ITS, TS demonstrates inferior performance with increasing severity of class imbalance. This phenomenon is attributed to its exclusive focus on exploring and exploiting a single reward. Consequently, TS tends to favor algorithms with strong performance in the majority class over time. Hence, the imperative necessity of imbalance reward component design becomes apparent in the domain of online CASH, particularly under the influence of imbalanced data streams. Furthermore, the evolutionary optimization employed in EvoAutoML manifests significantly inferior performance compared to other approaches in most cases. This discrepancy might arise from EvoAutoML maintaining a small pool and updating it at regular intervals via random evolution, leading to the accumulation of more errors during the early exploration stage. In contrast, the sequential decision-making employed in TS-based methods, which observes the search space one-by-one over time, can alleviate this issue.

5.3 Non-stationary data streams

In this section, we design two non-stationary scenarios to answer RQ2.

Data with a changing IR and fixed concepts Each data stream is still fixed to have 100,000 instances. The IR is changed at time step 50,000 and from time step 50,001, and we change the IR at a different speed (either an abrupt drift where the old distribution is completely taken over by a new one, or a gradual drift where the change lasts for 30,000 time steps). We consider a severe severity where $P(y = 1)$ changes from 0.01 to 0.99 because the changing IR does not impact the true decision boundary of algorithms within the search space and this ensures a more transparent evaluation of their impact on all approaches. The subscripts a and g denote abrupt and gradual drift, respectively. Table 4 presents a comparison of recall and G-mean for all algorithms in the new data distribution, specifically assessing performance during the time steps 50,001-100,000 for abrupt changes and 80,001-100,000 for gradual changes. This analysis explores whether our methods still maintain effectiveness in a changing IR and fixed concepts environment. To ensure accurate assessments, the values are reset to zero when the change starts and ends. In particular, class_0 transitions to a minority class status after the changes in all cases.

In terms of G-mean, we can see that ITS attains the highest performance, indicating its effectiveness in balancing the trade-off between the minority and majority classes in the new data distribution. WITS excels in the performance of the minority class in SINEa. Both DITS and WDITS demonstrate weak performance, forgetting outdated knowledge showing limited improvement on the changing imbalance. This suggests that the old configuration remains effective in the new data streams distribution. The reason may be attributed to the decision boundary of models within search spaces remaining unchanged, rendering new exploration and adaptation unnecessary. Conversely, EvoAutoML achieves the worst performance, potentially due to passively continual evolution in the optimization process, where excessive adaptation may not be necessary for these types of changes. Additionally, we note that abrupt drifts have a more severe impact than gradual changes on all methods.

Data with a fixed IR and changing concepts In this setting, we investigate real concept drift with a fixed IR, signifying the coexistence of real concept drift and class imbalance in data streams. The IR is set to a fixed 10% given that we place a greater emphasis on the impact of concept changes. Class 1 is designated as the minority class. SINEa involves a concept swap, while SINEg represents a probabilistic occurrence of change. For a mild change, the data distribution in SEAa is altered by controlling the threshold θ from 7 to 9.5. In SEAg, this threshold moves linearly with continual changes. The SEAa and SEAg data streams are less severe than SINEa and SINEg, as instances of the new data distribution in the SEA data set still encompass some previous instances after the threshold change ends.

Table 4. Performance of all algorithms with a changing IR and fixed concepts: means and standard deviation of Recall (class 0 and class 1) and G-mean over the new data distribution.

	Methods	Recall_1	Recall_0	G-mean
SINEa	EvoAutoML	0.9978±0.0004	0.0632±0.0104	0.2320±0.0319
	TS	0.9893±0.0009	0.8711±0.0398	0.9171±0.0269
	ITS	0.9998±0.0000	0.9857±0.0096	**0.9923±0.0052**
	WITS	0.9831±0.0056	**0.9932±0.0038**	0.9881±0.0041
	DITS	0.9533±0.0188	0.9041±0.0185	0.9250±0.0153
	WDITS	0.9163±0.0256	0.9074±0.0264	0.9096±0.0251
SINEg	EvoAutoML	0.9971±0.0010	0.0950±0.0251	0.2617±0.0539
	TS	0.9987±0.0003	0.9895±0.0045	0.9940±0.0023
	ITS	0.9999±0.0000	**0.9981±0.0009**	**0.9990±0.0004**
	WITS	0.9487±0.0198	0.9870±0.0086	0.9656±0.0122
	DITS	0.9575±0.0197	0.9081±0.0312	0.9235±0.0256
	WDITS	0.9514±0.0218	0.9538±0.0209	0.9522±0.0209
SEAa	EvoAutoML	0.9996±0.0001	0.6679±0.0166	0.8165±0.0101
	TS	0.9958±0.0004	0.9140±0.0206	0.9519±0.0117
	ITS	0.9998±0.0000	**0.9855±0.0110**	**0.9921±0.0060**
	WITS	0.9198±0.0324	0.9689±0.0107	0.9408±0.0228
	DITS	0.9942±0.0016	0.9300±0.0244	0.9588±0.0143
	WDITS	0.9723±0.0150	0.9493±0.0197	0.9602±0.0169
SEAg	EvoAutoML	0.9992±0.0002	0.8893±0.0074	0.9426±0.0039
	TS	0.9999±0.0000	0.9930±0.0021	0.9964±0.0011
	ITS	0.9999±0.0000	**0.9981±0.0007**	**0.9990±0.0003**
	WITS	0.9936±0.0013	0.9959±0.0017	0.9947±0.0015
	DITS	0.9958±0.0015	0.9900±0.0042	0.9929±0.0028
	WDITS	0.9933±0.0036	0.9885±0.0062	0.9908±0.0049

Table 5. Performance of all algorithms with a fixed IR and changing concepts: means and standard deviation of Recall(class 0 and class 1) and G-mean over the new data distribution.

	Methods	Recall_1	Recall_0	G-mean
SINEa	EvoAutoML	0.7190±0.0295	0.7756±0.0329	0.7386±0.0080
	TS	0.8752±0.0229	0.9471±0.0037	0.9081±0.0140
	ITS	0.8499±0.0083	0.8553±0.0084	0.8526±0.0083
	WITS	0.7007±0.0179	0.7434±0.0147	0.7216±0.0162
	DITS	**0.9719±0.0126**	0.9768±0.0105	**0.9743±0.0114**
	WDITS	**0.9766±0.0119**	0.9739±0.0107	**0.9752±0.0113**
SINEg	EvoAutoML	0.7796±0.0938	0.6467±0.1512	0.6859±0.0526
	TS	0.3860±0.0456	0.9668±0.0020	0.5714±0.0405
	ITS	0.7638±0.0296	0.8211±0.0177	0.7876±0.0234
	WITS	0.2428±0.0404	0.2775±0.0301	0.2469±0.0323
	DITS	**0.8815±0.0299**	0.8103±0.0286	**0.8358±0.0247**
	WDITS	0.6457±0.0071	0.4554±0.0238	0.5380±0.0146
SEAa	EvoAutoML	0.9578±0.0017	0.9383±0.0010	0.9480±0.0012
	TS	0.7646±0.0176	0.9989±0.0002	0.8722±0.0101
	ITS	0.9148±0.0093	0.9945±0.0031	0.9534±0.0050
	WITS	0.9121±0.0058	0.9886±0.0023	0.9494±0.0033
	DITS	**0.9878±0.0018**	0.9950±0.0015	**0.9914±0.0016**
	WDITS	**0.9861±0.0043**	0.9890±0.0053	**0.9875±0.0048**
SEAg	EvoAutoML	0.9658±0.0024	0.9458±0.0010	0.9557±0.0014
	TS	0.7312 ±0.0224	0.9985±0.0004	0.8514±0.0133
	ITS	0.9288±0.0177	0.9972±0.0014	0.9608±0.0100
	WITS	0.9646±0.0097	0.9880±0.0072	0.9760±0.0077
	DITS	**0.9886±0.0030**	0.9933±0.0019	**0.9909±0.0024**
	WDITS	0.9753±0.0048	0.9761±0.0054	0.9757±0.0049

Table 5 presents a comprehensive comparison of Recall and G-mean across all methods in the new data distribution. As we anticipated, DITS constantly achieves superior performance in all changing data scenarios. This superiority might be attributed to the discounting of outdated knowledge mechanism in DITS, enabling it to forget old knowledge and dynamically select the appropriate algorithm for new concepts after drifts. The real concept drift significantly influences the decision boundaries of models within the search space. As a result, the evolving performance of the models requires the online optimization method to adaptively balance exploration and exploitation as the data concepts change over time. In addition, there is no significant difference observed between DITS and WDITS in SINEa, as indicated by p-values of 0.9913 for Recall_1 and 0.1156 for G-mean. Similarly, in SEAa, statistical tests between DITS and

Table 6. Final step performance on the Real-world data sets displaying the mean and standard deviation.

	Methods	Frauds	GMSC	Weather	Covety34
Recall_1	EvoAutoML	0.2152±0.0542	0.3250±0.0411	0.5472±0.0110	0.7577±0.0438
	TS	0.7701±0.0394	0.1794±0.0066	0.7163±0.0056	0.6614±0.0421
	ITS	0.7880±0.0185	0.7285±0.0109	0.7146±0.0028	**0.9691±0.0100**
	WITS	**0.8084±0.0163**	**0.7525±0.0092**	**0.8191±0.0074**	0.9485±0.0158
	DITS	0.7269±0.0197	0.4721±0.0326	0.7016±0.0031	0.8742±0.0281
	WDITS	0.7378±0.0188	0.4737±0.0145	0.7498±0.0028	0.8117±0.0238
G-mean	EvoAutoML	0.3771±0.0365	0.4487±0.0211	0.6101±0.0037	0.76547±0.0091
	TS	0.8702±0.0255	0.4179±0.0085	**0.7693±0.0048**	0.7852±0.0360
	ITS	**0.8819±0.0111**	**0.7532±0.0048**	0.7434±0.0066	**0.9815±0.0062**
	WITS	0.8273±0.0104	0.4643±0.0095	0.6531±0.0055	0.9474±0.0167
	DITS	0.8419±0.0137	0.5927±0.0168	0.6754±0.0026	0.9046±0.0202
	WDITS	0.8169±0.0086	0.3965±0.0107	0.6805±0.0025	0.8358±0.0197

WDITS reveal non-significant differences with p-values of 0.4223 for Recall_1 and 0.6883 for G-mean. Although WDITS also performs well in SINEa and SEAa, its adaptive weight mechanism does not yield noticeable improvements in other non-stationary environments. One possible reason is that we only consider the fixed IR in new data distribution.

5.4 Real-World Data Sets

The aim of this experiment is designed to answer RQ3. We look into the performance of all approaches on the four real-world data sets, as detailed in Sect. 5.1. The means and standard deviations of the final step Recall_1 and G-mean, averaged over 30 runs of all methods, are shown in Table 6. According to Table 6, we can see that ITS achieves superior performance across most data sets in terms of G-mean. TS achieves peak performance in the Weather data set, which can be attributed to its less imbalanced degree (i.e., approximately 30%). Similar results observed in Sect. 5.2 indicate that an imbalanced reward design might not be necessary when the data set tends to be balanced. With regard to minority class performance, WITS consistently outperforms other methods in most cases. Although the adaptive weight tuning mechanism emphasizes the selection of suitable model performance and hyperaprametrs for the minority class, it may come at the cost of sacrificing majority class performance.

6 Conclusions

In this paper, we propose a new online automated framework for handling class imbalanced and non-stationary data streams, called OAutoIDSL. It integrates improved TS to enable one-by-one learning and real-time predictions, while keeping the classifiers robust to class imbalanced and time varying distribution. We evaluate the proposed approaches against EvoAutoML and TS by answering three research questions outlined in Sect. 5. The key findings are as follows: a) For stationary imbalanced data streams, ITS consistently obtains highest performance, particularly when confronted with a high degree of class imbalance. b) In non-stationary scenarios, ITS demonstrates significantly superior performance for G-mean in settings involving a changing IR and fixed concepts. DITS achieves the competitive performance with a fixed IR and changing concepts. c) For real-world data sets, ITS surpasses other methods in G-mean across all real data sets (IR below 30%), while WITS consistently exhibits superior performance in minority class outcomes in most cases. In future work, we would like to extend our research to a broader search space using online optimization methods and investigate multi-class imbalance in online settings.

Acknowledgements. This work is supported by the EPSRC Early Career Researchers International Collaboration Grants [EP/Y002539/1].

References

1. Agrawal, S., Goyal, N.: Analysis of thompson sampling for the multi-armed bandit problem. In: Conference on learning theory. pp. 39–1. JMLR Workshop and Conference Proceedings (2012)
2. Celik, B., Singh, P., Vanschoren, J.: Online automl: An adaptive automl framework for online learning. Mach. Learn. **112**(6), 1897–1921 (2023)
3. El Shawi, R., Rozgonjuk, D.: Onlineautoclust: A framework for online automated clustering. In: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management. pp. 3870–3874 (2023)
4. Feurer, M., Eggensperger, K., Falkner, S., Lindauer, M., Hutter, F.: Auto-sklearn 2.0: Hands-free automl via meta-learning. The Journal of Machine Learning Research **23**(1), 11936–11996 (2022)
5. Kang, Y., Hsieh, C.J., Lee, T.: Online continuous hyperparameter optimization for contextual bandits. arXiv preprint arXiv:2302.09440 (2023)
6. Kulbach, C., Montiel, J., Bahri, M., Heyden, M., Bifet, A.: Evolution-based online automated machine learning. In: Advances in Knowledge Discovery and Data Mining: 26th Pacific-Asia Conference, PAKDD 2022, Chengdu, China, May 16–19, 2022, Proceedings, Part I. pp. 472–484. Springer (2022)
7. Liu, Y., Li, Y., Schiele, B., Sun, Q.: Online hyperparameter optimization for class-incremental learning. arXiv preprint arXiv:2301.05032 (2023)
8. Malialis, K., Panayiotou, C.G., Polycarpou, M.M.: Online learning with adaptive rebalancing in nonstationary environments. IEEE Transactions on Neural Networks and Learning Systems **32**(10), 4445–4459 (2020)
9. Montiel, J., Halford, M., Mastelini, S.M., Bolmier, G., Sourty, R., Vaysse, R., Zouitine, A., Gomes, H.M., Read, J., Abdessalem, T., et al.: River: machine learning for streaming data in python. The Journal of Machine Learning Research **22**(1), 4945–4952 (2021)
10. Nguyen, D.A., Kong, J., Wang, H., Menzel, S., Sendhoff, B., Kononova, A.V., Bäck, T.: Improved automated cash optimization with tree parzen estimators for class imbalance problems. In: 2021 IEEE 8th international conference on data science and advanced analytics (DSAA). pp. 1–9. IEEE (2021)

11. Singh, P., Vanschoren, J.: Automated imbalanced learning. arXiv preprint arXiv:2211.00376 (2022)
12. Tornede, A., Bengs, V., Hüllermeier, E.: Machine learning for online algorithm selection under censored feedback. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 36, pp. 10370–10380 (2022)
13. Vieira, P.M., Rodrigues, F.: An automated approach for binary classification on imbalanced data. Knowl. Inf. Syst. **66**(5), 2747–2767 (2024)
14. Wang, B., Pineau, J.: Online bagging and boosting for imbalanced data streams. IEEE Trans. Knowl. Data Eng. **28**(12), 3353–3366 (2016)
15. Wang, S., Minku, L.L., Yao, X.: Resampling-based ensemble methods for online class imbalance learning. IEEE Trans. Knowl. Data Eng. **27**(5), 1356–1368 (2014)
16. Wang, S., Minku, L.L., Yao, X.: A systematic study of online class imbalance learning with concept drift. IEEE transactions on neural networks and learning systems **29**(10), 4802–4821 (2018)
17. Wang, Z., Wang, S.: Online automated machine learning for class imbalanced data streams. In: 2023 International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE (2023)
18. Wu, Q., Wang, C., Langford, J., Mineiro, P., Rossi, M.: Chacha for online automl. In: International Conference on Machine Learning. pp. 11263–11273. PMLR (2021)
19. Zhang, J., Sun, Z., Qi, Y.: Autoidl: Automated imbalanced data learning via collaborative filtering. In: International Conference on Knowledge Science, Engineering and Management. pp. 96–104. Springer (2020)

Phase-Encoded Cascaded Autoencoders based Correlation Filters and Adaptive Thresholding for Illumination Invariant Face Recognition

Pradipta K. Banerjee(✉)

Department of CSE-AIML, Future Institute of Technology, Kolkata, India
pradipta.kr.banerjee@teamfuture.in

Abstract. This study presents a novel method for constructing class-oriented correlation filters, utilizing a unique integration of representation learning with phase-only autoencoders. Our proposed model employs a specialized autoencoder architecture designed to handle phase data effectively. The network is composed of dense layers and convolution layers, arranged to optimally condense phase information into a compact, low-dimensional representation, which is then used to reconstruct the input while preserving essential phase relationships. This processed information from the latent space is exploited to develop a correlation filter that is specifically tuned for class recognition, enhancing discriminant phase-encoded features. To further refine the classification threshold, kernel density estimation was employed, allowing for an empirical determination of decision boundaries based on the density functions derived from high and low peak to sidelobe ratio values. We rigorously evaluate our approach across benchmark datasets , YaleB and PIE, demonstrating its superiority in classification accuracy over state-of-the-art frequency domain and feature extraction methods. The effectiveness of our approach is affirmed through extensive testing on face recognition under varying lighting conditions.

Keywords: Autoencoders · Correlation Filter · Face Recognition

1 Introduction

Traditional representation learning techniques[1], especially those based on autoencoders[23, 28, 29], have laid a robust foundation for feature extraction and dimensionality reduction in image processing tasks, including face recognition. However, these systems frequently encounter significant challenges under poor lighting conditions[14, 15], a prevalent issue that can drastically affect the reliability and accuracy of identification processes.

Autoencoders, by design, are powerful tools for learning efficient representations through unsupervised learning, ideally preserving essential information

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 173–189, 2025.
https://doi.org/10.1007/978-3-031-78383-8_12

while minimizing reconstruction loss[2]. Standard autoencoder architectures predominantly focus on amplitude information, which is highly susceptible to variations in image intensity and contrast[20]. This susceptibility often leads to degraded performance when these models are deployed in environments with inconsistent or poor lighting[21].

Recent studies[6,8–10] have highlighted the potential of integrating additional layers of complexity into autoencoder networks to address these illumination challenges. For instance, advancements in convolutional neural networks (CNNs) and their integration into autoencoder[31] frameworks have shown promise in enhancing feature stability across varying lighting conditions. Nevertheless, these approaches still fundamentally rely on amplitude-based features, leaving them vulnerable to the inherent limitations of such data under suboptimal lighting conditions.

The phase spectrum of an image, conversely, carries crucial structural and identity-specific information that remains comparatively invariant under changes in illumination[21]. This aspect of image data is often underutilized in traditional autoencoder designs. Motivated by this, our study proposes a novel approach that pivots from the conventional amplitude-focused methods to a phase-oriented strategy[11]. By using the robustness of the phase spectrum through a specialized autoencoder architecture, our model aims to mitigate the typical difficulties associated with poor lighting.

1.1 Contributions of the Proposed Method

Unique Integration of Representation Learning with Phase-Only Autoencoders: The unique feature of our approach is the incorporation of phase-only autoencoder architecture into the correlation filter design for each class. Unlike traditional methods that primarily focus on amplitude features, our approach emphasizes the phase spectrum of the images, which often carries critical structural and identity-specific information that amplitude components might miss. This is particularly advantageous in pattern recognition tasks where phase components can provide more discriminatory power.

Specialized Network Architecture Our approach involves two stages of autoencoders; output of the first stage (stacked encoder-decoder, SAE) is phase encoded and applied as input to the second (convolution encoder-decoder,CAE), and hence cascaded. Our proposed system is specifically designed to effectively map the phase spectrum, which inherently spans a $0 - 2\pi$ range, into the operational range of the neural network and back. This ensures that the integrity of the phase information is maintained throughout the process. Final stage utilizes the reconstructed phase spectrum for frequency domain correlatrion.

Enhanced Class Discriminability: The low-dimensional, compact representation of phase information yielded a correlation filter that is well-tuned for class detection, providing enhanced discriminant characteristics, which is evident from our comparison trials.

Hierarchical Feature Extraction: The cascading architecture allows for a layered approach to feature extraction. First Level : focuses on capturing and compressing the general features of the input images into a compact, class-specific representation. The simplicity of this AE allows it to efficiently distill the essential features without being biased towards the complex variations within the class. Second Level: after the initial compression and feature distillation, the phase spectrum of these representations is extracted and processed through a convolutional autoencoder. The SAE excels in handling spatial hierarchies and local features in data. It also excels at refining phase information, which is vital for identifying *similarities* and *differences* in *within* and *between* classes.

Optimal Threshold Determination In traditional approaches to classification within correlation filter frameworks, a fixed threshold[12,17] is often employed to distinguish between classes. Despite being simple, this strategy can result in inferior classification accuracy since it ignores the variability and overlap in distribution between various classes. In order to overcome this constraint, we provide a new approach in which the appropriate threshold is dynamically determined from the distribution of average high and average low peak-to-sidelobe ratio (PSR)[12] values using kernel density estimation (KDE). Our method uses KDE to analyse the data instead of depending on a hard threshold[12,17,21]. The best threshold for classification is found by using KDE to locate the intersection of the probability density functions of high and low PSR values.

Section 2 elaborates the proposed methodology. Experimental results are given in Sect. 3. The paper concludes in Sect. 4.

2 Proposed Methodology

A good representation is one that will yield a better performing classifier[25]. In this paper we are interested in a good representation of $\mathbf{x}$, the input vector. $\mathbf{x}$ is a d dimensional random vector, in $[0,1]^d$. The primary concern of our work is to find a good higher level representation $\mathbf{y}$ of $\mathbf{x}$, where $\mathbf{y}$ in $\mathcal{R}^{d'}$ and we ensure the under-completeness as $d' < d$. We also consider there is a mapping from $\mathbf{x}$ to $\mathbf{y}$ through some probability distribution $p(\mathbf{y}|\mathbf{x};\theta)$ parameterized by θ, that we want to learn. We further restrict ourselves to a deterministic function $\mathbf{y} = f_\theta(\mathbf{x})$, through which $\mathbf{x}$ is mapped to $\mathbf{y}$. The deterministic mapping f_θ that transforms an input vector $\mathbf{x}$ into hidden representation $\mathbf{y}$ is called the encoder. Its typical form is an affine mapping followed by a nonlinearity:

$$f_\theta(\mathbf{x}) = r(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{1}$$

where, r is the nonlinearity introduced in the affine transformation and taken as ReLU activation. Its parameter set is $\theta = \mathbf{W}, \mathbf{b}$, where $\mathbf{W}$ is a $d' \times d$ weight matrix and $\mathbf{b}$ is an offset vector of dimensionality d'. The resulting hidden representation $\mathbf{y}$ is then mapped back to a reconstructed d dimensional vector $\mathbf{z}$ in input space,

$z = g_{\theta'}(\mathbf{y})$. This mapping $g_{\theta'}$ is called the decoder. It's typical form is again an affine mapping optionally followed by a squashing non-linearity, that is,

$$g_{\theta'}(\mathbf{y}) = s(\mathbf{W}' f_{\theta}(\mathbf{x}) + \mathbf{b}') \tag{2}$$

where, the squashing nonlinearity used here as 'sigmoid' activation function. Here, $\mathbf{z}$ is not just the exact representation of $\mathbf{x}$, but it represents a class representative, probabilistically, the mean of the distribution of $p(\mathbf{x}|\mathbf{z};\theta,\theta')$, where $\mathbf{z}$ acts as a 'generator', which generates $\mathbf{x}$. So the likelihood of $\mathbf{z}$ with respect to $\mathbf{x}$ in $p(\mathbf{x}|\mathbf{z};\theta,\theta')$ should be maximized. This can be done by minimizing the reconstruction error

$$J(\mathbf{x},\mathbf{z}) \approx -\log p(\mathbf{x}|\mathbf{z}) \tag{3}$$

If we set the choice of $p(\mathbf{x}|\mathbf{z})$ as Gaussian, then for real-valued $\mathbf{x}$, that is, $\mathbf{x} \in \mathcal{R}^d : \mathbf{x}|\mathbf{z}\ \mathcal{N}(\mathbf{z},\sigma^2\mathbf{I}), i.e., \mathbf{x}_j|\mathbf{z}\ \mathcal{N}(\mathbf{z}_j,\sigma^2)$. This yields

$$J(\mathbf{x},\mathbf{z}) = L_2(\mathbf{x},\mathbf{z}) = C(\sigma^2)||\mathbf{x}-\mathbf{z}||^2 \tag{4}$$

where, $C(\sigma^2)$ is constant and that can be ignored for the optimization of the squared error loss. Though the interpretation is Gaussian, in our setting, as $\mathbf{z} \in [0,1]^d$, the squashing nonlinearity 'sigmoid' is used in the decoder part. The class specific reconstruction is shown in Fig.(1), where $k = 3$ and randomly selected single image from all K - classes are reconstructed through the model $\Theta^{(k=3)}$, where $\Theta = \{\theta,\theta'\}$ is our previous discussion.

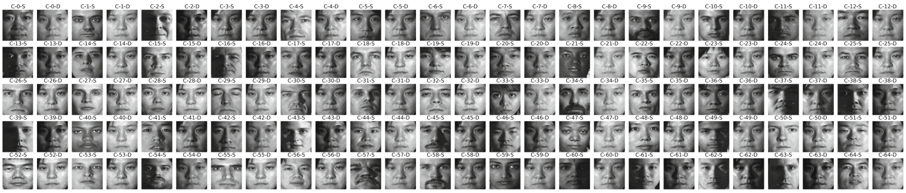


Fig. 1. Class specific reconstruction example. The model $\Theta^{(k)}_{SAE}$ is trained with only class-3 images and all other class images are reconstructed through $\Theta^{(k)}_{SAE}$. $C-l-S$ represents lth class sample and $C-l-D$ stands for lth class decoding. Interesting observation is that, $C-3-S$ and $C-3-D$ are same, as given in first row, 7th and 8th element.

2.1 Phase Encoded Convolutional AE

The phase spectrum in image processing is highly valuable because it encodes the structural information and the relative positioning of objects in an image. It has been shown in [21] that when reconstructing an image from its Fourier transform, the phase information is much more crucial than the magnitude information. Experiments where images are reconstructed using either only the phase or only the magnitude typically show that phase-only reconstructions retain more recognizable features and structures, whereas magnitude-only reconstructions tend to be unrecognizable and blurry. Another advantage of phase encoding is the phase spectrum determines the location of edges and other sharp transitions in the image content, which are essential for identifying objects and their boundaries. Furthermore unlike the amplitude spectrum, which can change significantly with variations in illumination and contrast, the phase spectrum is relatively invariant to such changes. This property makes phase-based features ideal for applications in environments with dynamic lighting conditions or where images are captured under different exposure settings. Motivating with these advantages, the reconstructed image $\mathbf{z}$ obtained from first stage of encoder-decoder combination, is further Fourier transformed:

$$\mathcal{Z} = \mathcal{F}(\mathbf{Z}) \in \mathbb{R}^{d_x \times d_y} \tag{5}$$

and phase spectrum $e^{j\Phi_{\mathcal{Z}}}$ is extracted. The phase angles are further normalized as :

$$\Phi_{\mathcal{Z}n} = \frac{\Phi_{\mathcal{Z}}(u,v) + \pi}{2\pi} \in \mathbb{R}^{d_x \times d_y} \tag{6}$$

This normalization adjusts the phase values, shifting from a range of $[-\pi, \pi]$ to $[0, 2\pi]$ and then scaling it down to $[0, 1]$. Since this is a direct, linear transformation, it does not introduce any additional noise or variability into the data. The characteristics of the noise that might already be present in the phase data remain unchanged. Also, relationships between phases, such as phase differences and alignments, are preserved under this transformation. This preservation is crucial in our application, where primary goal is image reconstruction, The normalized phase spectrum $\Phi_{\mathcal{Z}n}$ is then reshaped for input into subsequent convolutional autoencoder (CAE)[3,18]. This reshaping typically includes adjustments for batch size and the number of channels, depending on the specific architecture of the network being used. The goal is to capture and reconstruct high-quality features from images as well as suppressing the unwanted noise developed by using phase only instances, if there is any. The proposed phase encoded CAE has multiple advantages. CAE enhances the structural details like geometrical and spatial properties. CAE can automatically learn spatial hierarchies[3] of features from input phases ;$\Phi_{\mathcal{Z}n}$, making it an excellent choice for complex patterns inherent in the phase spectrum. Due to the sharing of weights in convolutional layers, CAEs generally preserves the spatial locality as well as require fewer parameters than fully connected networks of similar capacity. Moreover, although phase information is generally robust to Gaussian noise, it can be sensitive to high-frequency noise. This type of noise introduces rapid changes in the phase, especially at higher frequencies where the amplitude might be low, leading to significant distortions in the phase spectrum. The CAE can potentially ignore the high frequency noise , if present in the system (phase instances) and focus on reconstructing the true underlying features of the input instances. This leads to cleaner and more accurate class specific reconstructions. In CAE the reconstruction is hence due to a linear combination of basic image patches based on the latent code[18]. For a mono channel phase information $\Phi_{\mathcal{Z}n}$, the latent representation of the $j-th$ feature map is given by

$$\mathbf{h}_j = r(\mathbf{W}_j * \Phi_{\mathcal{Z}n} + \mathbf{b}_j) \tag{7}$$

A single bias is used for each latent feature map, allowing each filter to specialize in capturing features from the entire input. $*$ denotes the 2D convolution operation. The reconstruction is obtained as

$$\hat{\Phi}_{\mathcal{Z}n} = s(\sum_{j \in L} \Phi_{\mathcal{Z}n} * \mathbf{W}'_j + \mathbf{b}'_j) \tag{8}$$

where, L identifies the group of latent feature maps; $\mathbf{W}'$ identifies the flip operation over both dimensions of the weights. The objective of the CAE is to minimize the difference between the input and the reconstructed phase spectrum, mathematically expressed as:

$$J_2 = \frac{1}{d}\sum_{n=1}^{d}(\Phi_{\mathcal{Z}n} - \hat{\Phi}_{\mathcal{Z}n})^2$$

which is to be minimized.

2.2 Phase Cross Correlation

The cross correlation plane; $\mathbf{G} \in \mathbb{R}^{d_x \times d_y}$, is obtained by the following operation:

$$\mathbf{G} = (-1)^{x+y}\mathcal{F}^{-1}(\mathcal{X} \oplus \mathcal{X}^*) \tag{9}$$

where, $\mathcal{X} = \mathcal{F}(\mathbf{X})$ and $*$ stands for complex conjugate. $\oplus$ represents element wise product. Instead of full frequency domain representation, this work involves only phase spectrum. Hence Eq.(9) will be modified as:

$$\mathbf{G} = (-1)^{x+y}\mathcal{F}^{-1}\{\exp(j(\Phi_{\mathcal{Z}n})) \oplus \exp(-j(\hat{\Phi}_{\mathcal{Z}n})\} \tag{10}$$

For perfect reconstruction, there is exact match between $\Phi_{\mathcal{Z}n}$ and $\hat{\Phi}_{\mathcal{Z}n}$. The cross power spectrum of these two, yield unit flat response in all frequencies. The inverse Fourier transform $\mathcal{F}^{-1}$ with $(-1)^{x+y}$ ensures the $\delta(x, y)$(dirac delta function) representation in spatial domain.

For class specific representation, considering total K -classes, for a specific kth class , all the above formulation will be modified as:

$$\mathbf{z}^{(k)} = s\left(\mathbf{W}^{(k)'} f_{\Theta^{(k)}}\left(\mathbf{x}^{(k)}\right) + \mathbf{b}^{(k)'}\right) \tag{11}$$

where $\Theta^{(k)} = \mathbf{W}^{(k)}, \mathbf{b}^{(k)}$. With class specific representation $\mathbf{z}^{(k)}$, equations corresponding phase reconstruction and cross correlation plane are rewritten as :

$$\hat{\phi}^{(k)}_{\mathcal{Z}n} = s(\sum_{j \in L} \phi^{(k)}_{\mathcal{Z}^{(k)}n} * \mathbf{W}^{(k)'}_j + \mathbf{b}^{(k)'}_j) \tag{12}$$

and

$$\mathbf{G}^{(kl)} = (-1)^{x+y}\mathcal{F}^{-1}\{\exp(j(\Phi^{(k)}_{\mathcal{Z}^{(k)}n})) \oplus \exp(-j(\hat{\Phi}^{(l)}_{\mathcal{Z}^{(l)}n})\} \tag{13}$$

Interestingly, Eq.(13) represents a general correlation plane, where the nature of $\mathbf{G}^{(kl)}$ is governed by $\Theta^{(k)}$ and $X^{(l)}$. This statement can be mathematically expressed as :

$$\begin{aligned} \mathbf{G}^{(kl)} &= \delta(x,y) && \text{if } k = l, \\ &= \text{random} && \text{otherwise.} \end{aligned} \tag{14}$$

Eq.(14) ensures class specific correlation filtering response from phase encoded cascaded autoencoders. Detail block diagram is shown in Fig.(2).

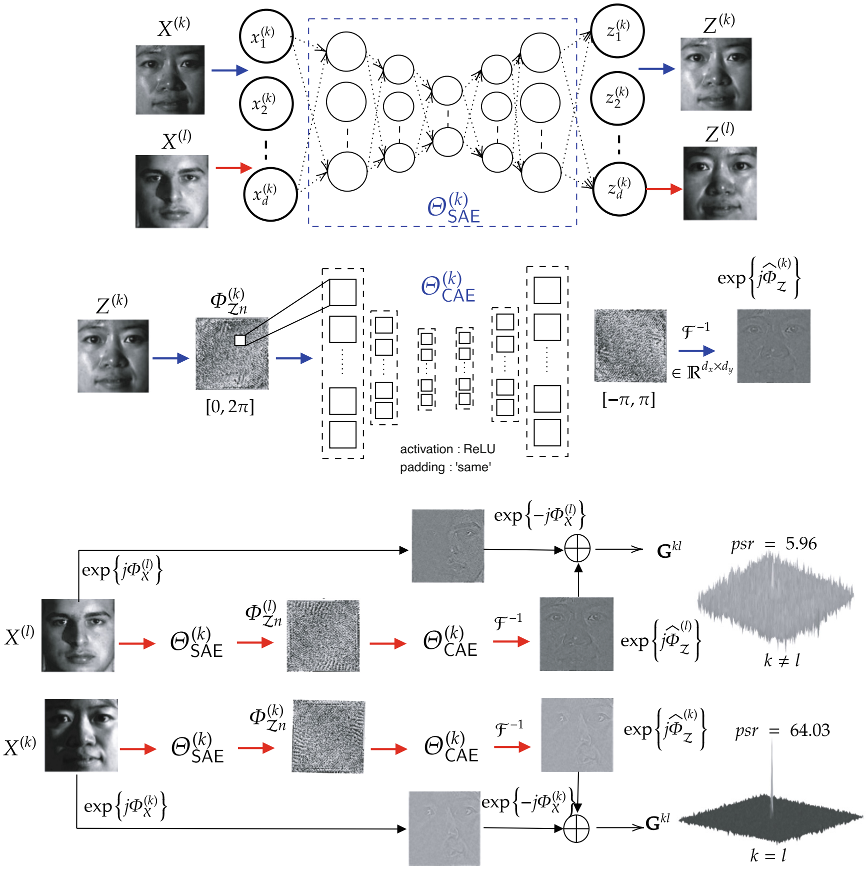

$X^{(k)}$
$X^{(l)}$
$x_1^{(k)}$
$x_2^{(k)}$
$x_d^{(k)}$
$\Theta^{(k)}_{\text{SAE}}$
$z_1^{(k)}$
$z_2^{(k)}$
$z_d^{(k)}$
$Z^{(k)}$
$Z^{(l)}$
$Z^{(k)}$
$\Phi^{(k)}_{\mathcal{Z}n}$
$[0, 2\pi]$
$\Theta^{(k)}_{\text{CAE}}$
activation : ReLU
padding : 'same'
$[-\pi, \pi]$
$\mathcal{F}^{-1}$
$\in \mathbb{R}^{d_x \times d_y}$
$\exp\{j\widehat{\Phi}^{(k)}_{Z}\}$
$\exp\{j\Phi^{(l)}_{X}\}$
$\exp\{-j\Phi^{(l)}_{X}\}$
$\mathbf{G}^{kl}$
$psr = 5.96$
$X^{(l)}$
$\Theta^{(k)}_{\text{SAE}}$
$\Phi^{(l)}_{\mathcal{Z}n}$
$\Theta^{(k)}_{\text{CAE}}$
$\mathcal{F}^{-1}$
$\exp\{j\widehat{\Phi}^{(l)}_{Z}\}$
$k \neq l$
$X^{(k)}$
$\Theta^{(k)}_{\text{SAE}}$
$\Phi^{(k)}_{\mathcal{Z}n}$
$\Theta^{(k)}_{\text{CAE}}$
$\mathcal{F}^{-1}$
$\exp\{j\widehat{\Phi}^{(k)}_{Z}\}$
$psr = 64.03$
$\exp\{j\Phi^{(k)}_{X}\}$
$\exp\{-j\Phi^{(k)}_{X}\}$
$\mathbf{G}^{kl}$
$k = l$

Fig. 2. Block diagram of the proposed methodology. The blue arrow represents the training case, and the red arrow represents the test case.

The decision surface in the correlation plane is based on PSR evaluation. The evaluation of PSR is evaluated according to [13]. In general a hard threshold of

PSR = 10 [12,13,21] is used to evaluate the performance of correlation filtering. Instead of taking hard threshold, a new optimal PSR threshold is proposed here.

2.3 Determination of Optimal PSR Threshold

To objectively determine an optimal threshold for discriminating between high and low PSR values, we employed KDE to model the probability density functions of both distributions. KDE is a non-parametric way to estimate the probability density function of a random variable. Mathematically, KDE for a set of points $\{x_i\}$ is defined as:

$$\hat{f}_h(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

where K is the kernel, a non-negative function that integrates to one and has mean zero, and h is the bandwidth, a positive parameter that controls the degree of smoothing.

For our analysis, Gaussian kernels were utilized for both the high PSR values (representing true class identifications) and the low PSR values (representing false class identifications). The KDE for each distribution was computed over a common range of PSR values, spanning from the minimum to the maximum observed values in our dataset. The intersection points of these two estimated density functions were then calculated. These intersections represent PSR values where the probability densities of the high and low PSR values are equal, providing a natural criterion for setting a threshold. We define the optimal threshold T as the median of these intersection points, mathematically represented by:

$$T_o = \text{median}(\{x \mid |f_{\text{high}}(x) - f_{\text{low}}(x)| < \epsilon\})$$

where ϵ is a small tolerance level, set to 0.01 in our analysis, chosen to fine-tune the precision of intersection detection.

This threshold T_o maximises the classifier's ability to distinguish between these outcomes using PSR data. This is verified through experimental results.

3 Experimental Results

3.1 Database and Setup

The PIE database[22] includes illumination subsets featuring 65 subjects, each with 21 images (Fig. 3). All images are converted to grayscale and resized to 100×100 pixels. The extended YaleB database[7] comprises 30 individuals, each with 64 differently illuminated grayscale frontal face images, resized to 64×64. These images are resized to 64×64 pixels. To train the model $\Theta_{SAE}^{(k)}$, we selected a subset of our dataset containing images from a known class. The training and test sets were created by splitting this subset using an $70 - 30$ train-test split. Due to limited number of training images available in both the dataset, data augmentation technique is exploited to increase the number of training images to train the models $\Theta_{SAE}^{(k)}$ and $\Theta_{CAE}^{(k)}$. The SAE architecture consists of two hidden layers with 512 and 256 neurons respectively. For training the model $\Theta_{CAE}^{(k)}$, we focused on phase spectrum images, which were also resized to 64×64 pixels and normalized. Similar to the SAE, the dataset was split into training and test sets using an $70 - 30$ split. The encoder comprised three convolutional layers with $32, 16$, and 8 filters, each using 3×3 kernels.

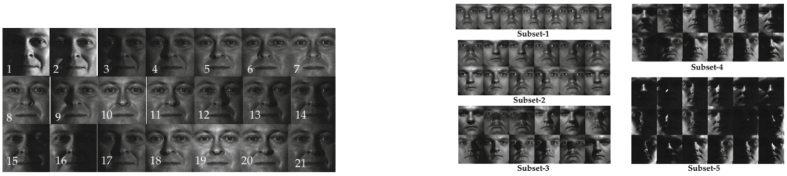


Fig. 3. (a)Sample images of person-3 from the illumination subset of PIE database with no background lighting.(b)Sample images of person-2 from YaleB database divided into five subsets.

3.2 Experiments

A comparative study of the proposed method with other state-of-the-art filtering techniques is reported here. As the decision process of the proposed system is based on frequency domain correlation filtering, we compare the proposed system with frequency domain filters like UMACE[17], MACH[24] and OTMACH[16]; experimented on both PIE and YaleB database. The UMACE filter is designed using the equation $h_{\text{UMACE}} = D^{-1}m$, with parameters $\alpha = 0$, $\beta = 1$, $\gamma = 0$. Similarly, MACH and OTMACH are designed using $h_{\text{MACH}} = S^{-1}m$, with $\alpha = 0.2$, $\beta = 0.5$, $\gamma = 0.3$, and $h_{\text{OTMACH}} = (\alpha C + \beta D + \gamma S)^{-1}m$, with $\alpha = 0.5$, $\beta = 0.3$, $\gamma = 0.2$, respectively.

PSR Distribution Analysis The proposed system is trained separately on both the dataset. For each dataset each class is separately trained and tested on all classes on respective set. Fig.(4) shows the PSR distribution for all classes from PIE and YaleB dataset, while class-3 images are chosen for training for PIE and class-5 for YaleB. The training classes are randomly chosen. PSR plot in Fig.(4) shows the high discrimination capability of the proposed system; high psr values corresponds to kth class (used for training) and low psrs correspond to rest of the classes ($l \neq k$) in the both the dataset. For the proposed system, the classifier performance over all the classes in respective dataset can be evaluated from PSR distribution. Towards this end, we formulated a 3D matrix $\mathbf{T}_{klm}$ (k represents classifier index l represents class index and m represents image index, where each class specific model $(\Theta_{SAE}^{(k)}, \Theta_{CAE}^{(k)})$ is evaluated on each class (30 for YaleB and 65 for PIE) for all images, separately for both the dataset. Here

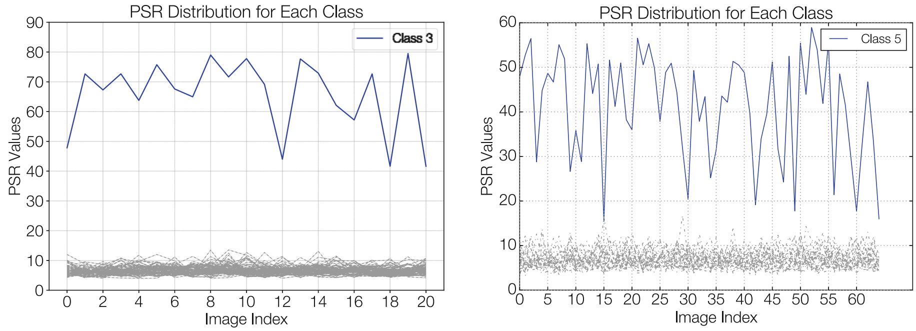


Fig. 4. Left panel:PSR plot for all classes in PIE when class -3 is trained in the proposed system. The demarcation is very much clear between kth class and rest $\forall l$ classes. Right Panel: PSR plot for all classes in YaleB when class -5 is trained in the proposed system.

$k = 65, l = 65, n = 21$ for PIE and $k = 30, l = 30, n = 65$, for YaleB. All the elements of $\mathbf{T}_{klm}$ are the PSR values. Any slice from tensor $\mathbf{T}_{klm}$, is the PSR distribution of k th classifier. For a good classifier, the l th row of k th slice should have very high PSR values iff $l = k$, and other rows of k slice should have very low PSR values. Rest of the experimental results are based on these two tensors $\mathbf{T}_{klm}^{PIE}$ and $\mathbf{T}_{klm}^{YaleB}$. For simplicity we use $\mathbf{T}^{P}$ and $\mathbf{T}^{Y}$ for PIE and YaleB respectively. To measure the average high and average low PSR values across all classifier we use the slices from $\mathbf{T}^{P}$ and $\mathbf{T}^{Y}$. For each slice, of $\mathbf{T}^{P}$ or $\mathbf{T}^{Y}$, the average response of the PSR values are evaluated column wise and then averaged over all the slices. This is done for all the state-of-the-art filters.

Average high and average low PSR values are plotted for both proposed system and state-of-the-art UMACE filter. The distribution of PSRs for true and false classes are well separated in both the case. But the proposed system shows better discriminating capability as the mean squared distance from average high PSRs to average low PSRs is much higher in case of proposed system than standard UMACE filter. The distance obtained for UMACE filter is **90.4539**, whereas for proposed system it is **189.08**, indicates better class separation characteristics of proposed method.

Optimal Threshold Determination and Study of Confusion Matrix To optimize classifier performance in identifying facial features within the Yale B and PIE face database, we applied KDE to analyze the distribution of PSR values derived from proposed algorithm. The histogram, illustrated in Fig.(8) shows the frequency distribution of high and low PSR values. The KDE suggested an optimal threshold of $T_o = 20.36$ for Yale and $T_o = 27.04$ for PIE, where the classifier achieves a balance between sensitivity (true positive rate) and specificity (false positive rate). Instead of taking a hard threshold, the following experiments are performed on the basis of these two optimal threshold values(T_o). Fig.(6) shows the effectiveness of choosing optimal threshold.

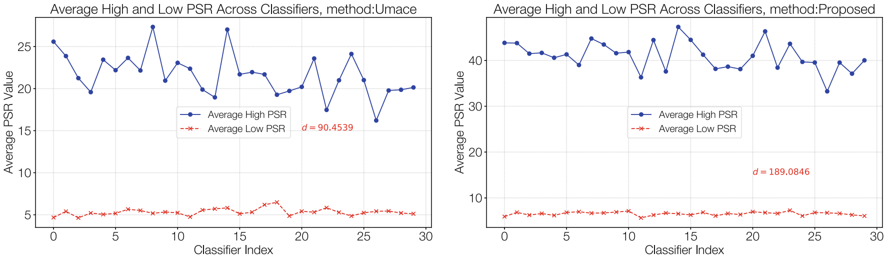


Fig. 5. Average PSR distribution across all the classifiers in YaleB dataset experimented over all 30 classes. The high and low psr distribution is shown for proposed system and it has been seen that it is very much comparable to standard UMACE filter while maitaining a greater margin.

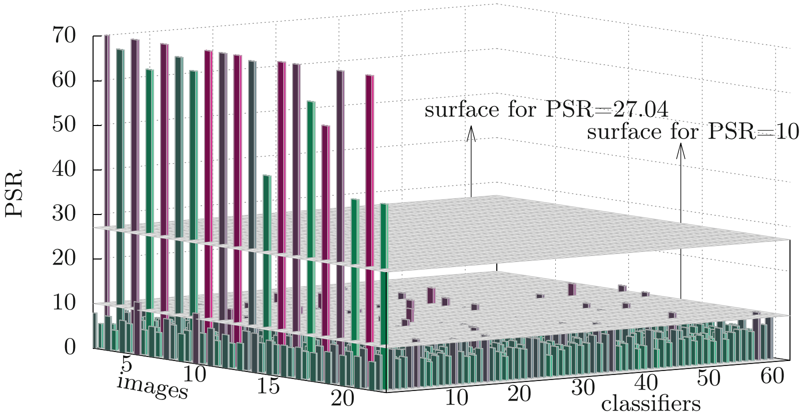
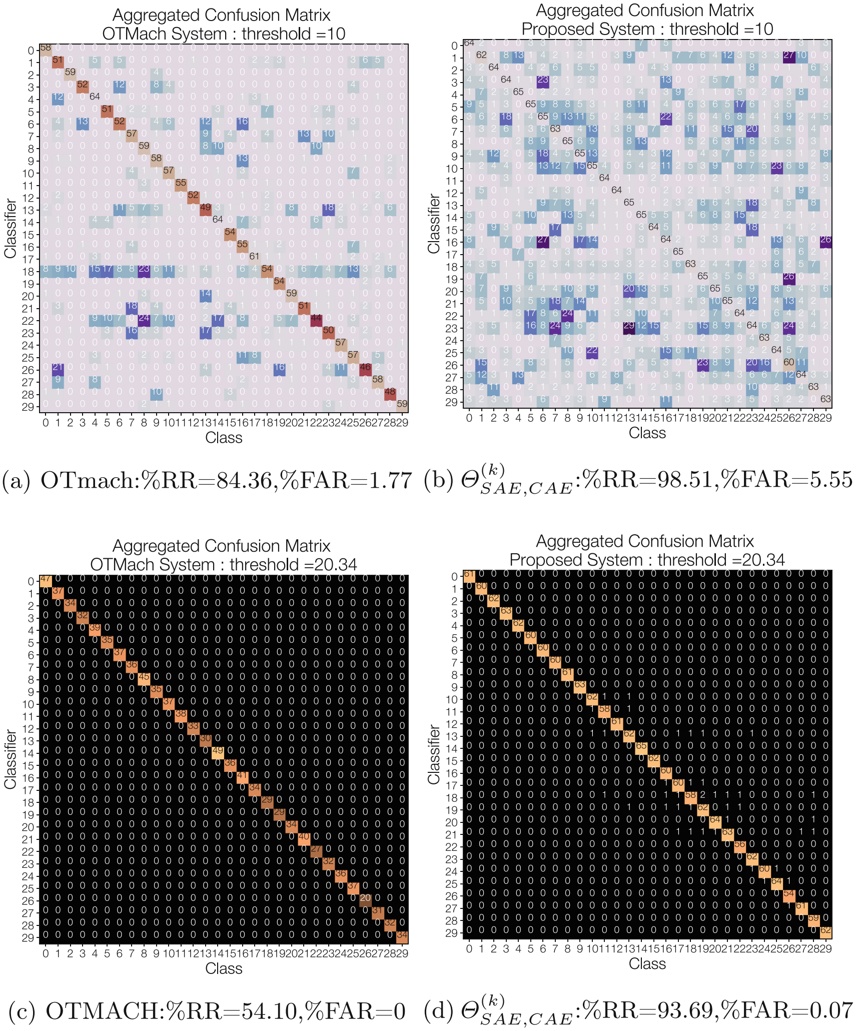


Fig. 6. 3D Visualization of $\mathbf{T}_{3lm}$ for PIE database. False alarms are reduced while taking $T_o = 27.04$. This ensures the effectiveness of adaptive thresholding strategy of the proposed method.

A comparative study has been performed in terms of confusion matrices, with these thresholds. Fig.(7) shows the confusion matrices, with threshold = 10 as in [13]. Each confusion matrix shows the classification performance of each classifier on each class. From Fig.(7), it is observed that the proposed system suffers from high false alarms, and it is obvious as here the confusion matrix is not evaluated on average low PSR values, rather each PSR values are taken into account for calculation of TPR and FPR. Though %FAR =(0.98 to 1.77) is very less for standard filters comparing to the proposed one, the %RR is also poor and can not be appreciated as well. High %RR = 98.51 is obtained for proposed system, but again the %FAR = 5.55 conceded by this is very high. Another experiment is performed for evaluating the confusion matrix, shown in Fig.(7), with $T_o = 20.36$ (for YaleB) and 27.04 for PIE. The observation from Fig.(7) confirms that the proposed correlation filter achieves high recognition

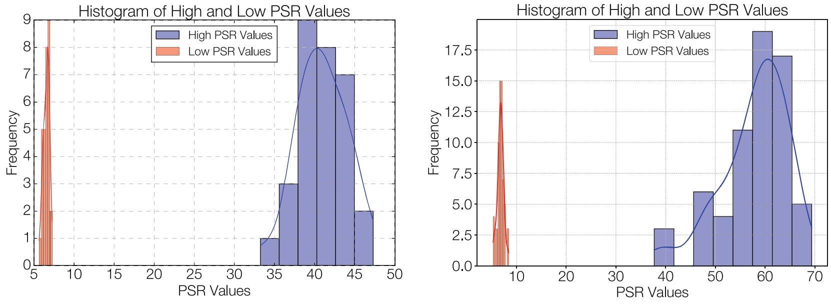


(a) OTmach:%RR=84.36,%FAR=1.77 (b) $\Theta^{(k)}_{SAE,CAE}$:%RR=98.51,%FAR=5.55

(c) OTMACH:%RR=54.10,%FAR=0 (d) $\Theta^{(k)}_{SAE,CAE}$:%RR=93.69,%FAR=0.07

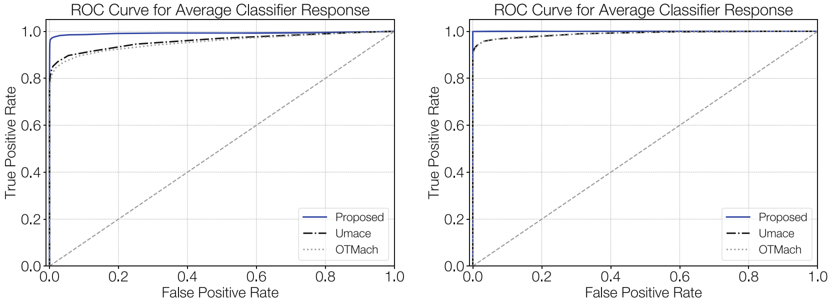
Fig. 7. (a) and (b),Confusion Matrices corresponding hard threshold = 10, for OTmach and Proposed system is given. Proposed system has much false positive alarms, as well as high TPR. Confusion Matrices (c) and (d) corresponding threshold $T_o = 20.36$. High %RR and low %FAR is achieved in proposed method.

rate (%RR=93.81) with a very low false alarm (%FAR = 0.07). In comparison to the proposed system, the standard filters, used in the experiment, are unable to reach a descent %RR. Inference can be drawn from the confusion matrix is that, for the proposed system a very sharp peak is obtained in the correlation plane for the true classes, as well as a flat correlation planes, with no such peaks are generated from false class images.



(a) (b)

Fig. 8. Histogram showing the distribution of high and low PSRs for (a) PIE (b) YaleB. Suggested optimal thresholds obatined from KDE are 27.04 for PIE and 20.36 for YaleB.



(a) (b)

Fig. 9. ROC curves for average classifier response on (a) YaleB and (b) PIE dataset.

Further experiment is conducted to plot receiver operating characteristics. Fig.(9) shows the receiver operating characteristic curves for three different filtering process. A prominent discrimination ability is observed for the proposed approach, where ROC is almost traces a step function comparing to others; provides empirical evidence supporting the robustness of our classifier design and deployment strategy.

Further the performance of the proposed method is compared with different unconstrained correlation filters with only phase extension by setting different optimal trade-off parameters for quad phase UMACE(QPUMACE), and phase only UMACE(POUMACE). Table.(1) shows the %mean recognition rate with %false acceptance rate for randomly selected images from different subsets of Extended YaleB database. POUMACE is obtained as the full phase extension of H_{UMACE} i.e., $H_{\text{POUMACE}} = e^{j\angle H_{\text{UMACE}}}$. In designing the QPUMACE filter, each element in the filter array will take on ±1 for the real component and $\pm j$ for the imaginary component. From Table.(1) it can be concluded that across all sets, the proposed system consistently exhibits higher recognition rates compared to both phase extended unconstrained filters. There is comparatively high %FAR in case of proposed system and that is due to high estimated PSR by adaptive threshold selection.

Table 1. The % mean recognition rate along with the % far obtained by different filters with $thr = 20.36$ as obtained from estimation.

Filters	QPUMACE	POUMACE	Proposed
	(%rec, %far)	(%rec, %far)	(%rec, %far)
Set-1	71.1, 0.00174	76.4, 0.0012	92.22, 0.16
Set-2	69.37, 0.0086	73.9, 0.00257	91.06, 0.02
Set-3	82.96, 0.0086	87.5, 0.002431	94.68, 0.07
Set-4	87.65, 0.0026	91.56, 0.0078	97.65, 0.053
Set-5	73.9, 0.0069	82.03, 0.0023	98.59, 0.089

Table.(2) shows the comparative study where we evaluate the performance of a proposed phase-encoded autoencoder method for different subsets from Extended YaleB, in comparison with several state-of-the-art feature extraction methods. The proposed phase-encoded autoencoder method shows superior performance in most subsets, achieving an overall average recognition rate of 97.59%, which is notably higher than most compared methods.

Table 2. Comparative study of proposed method with state-of-the-art feature extraction methods in terms of recognition rates (%) on Extended Yale-B datasets for different subsets. Best result shows in bold face, second best result shows in blue.

Methods	Subset-5	Subset-4	Subset-3	Subset-2	Subset-1	Average
Zhang,T.[30]	96.5	96.8	88.1	85.9	87.4	90.94
Wang,B.[26]	95.6	94.6	89.5	91.3	93.7	92.94
Chen,W.[5]	95.4	96.5	**99.1**	87.5	86.8	93.06
Chen,T.[4]	**99.8**	98.5	95.4	95.4	92.4	96.3
Wang,H.[27]	95.4	92.6	90.3	95.8	93.7	93.56
Ojala,T.[19]	99.1	**98.8**	96.5	93.7	**94.7**	96.56
Proposed	99.18	98.5	98.3	**98.3**	93.7	**97.59**

Discussion The proposed system uses phase-only reconstruction to enhance face recognition in varying lighting conditions, leveraging the phase spectrum's ability to capture structural information without being affected by illumination changes. This approach provides a more robust and stable method compared to traditional filters that utilize both phase and magnitude. By focusing solely on the phase, our method identifies key features for class differentiation overlooked by amplitude-based techniques. The optimized correlation filter demonstrates superior performance in recognizing distinct facial expressions, outperforming amplitude-based and hybrid methods in comparative tests. This makes it particularly effective for applications like outdoor biometrics and mobile face recognition.

4 Conclusion

In this study, a novel method is proposed for constructing class-oriented correlation filters by integrating representation learning with phase encoded cascaded autoencoders. Our model demonstrated superior classification accuracy compared to state-of-the-art frequency domain methods, as evidenced by the optimal threshold selection. The proposed system showed a significant improvement in class separation, as indicated by a larger mean squared distance between average high and low PSR values compared to standard methods. The proposed method provides a natural criterion for setting a threshold. The limitations of the proposed method primarily arise from its handling of pose variations and head orientations. Although effective at accommodating translational shifts, phase-only reconstructions are less adept at processing other types of transformations, such as rotations, scaling, or deformations. These non-translational variations can disrupt the continuity of phase information, thereby reducing the correlation process's accuracy. This limitation highlights how the method may struggle when dealing with dynamic or complex movements within the visual field. Looking ahead, future enhancements could focus on improving the method's robustness to non-translational changes, such as rotations and scaling, which would broaden its applicability and accuracy in more dynamic environments.

References

1. Bengio, Y., Courville, A., Vincent, P.: Representation learning: A review and new perspectives. IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 1798–1828 (2013)
2. Charte, D., Charte, F., del Jesus, M.J., Herrera, F.: An analysis on the use of autoencoders for representation learning: Fundamentals, learning task case studies, explainability and challenges. Neurocomputing **404**, 93–107 (2020)
3. Chen, M., Shi, X., Zhang, Y., Wu, D., Guizani, M.: Deep feature learning for medical image analysis with convolutional autoencoder neural network. IEEE Transactions on Big Data **7**(4), 750–758 (2017)
4. Chen, T., Yin, W., Zhou, X.S., Comaniciu, D., Huang, T.S.: Total variation models for variable lighting face recognition. IEEE Trans. Pattern Anal. Mach. Intell. **28**(9), 1519–1524 (2006)
5. Chen, W., Er, M.J., Wu, S.: Illumination compensation and normalization for robust face recognition using discrete cosine transform in logarithm domain. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **36**(2), 458–466 (2006)

6. Gao, S., Zhang, Y., Jia, K., Lu, J., Zhang, Y.: Single sample face recognition via learning deep supervised autoencoders. IEEE Trans. Inf. Forensics Secur. **10**(10), 2108–2118 (2015)
7. Georghiades, A.S., Belhumeur, P.N., Kriegman, D.J.: From few to many: Illumination cone models for face recognition under variable lighting and pose. IEEE Trans. Pattern Anal. Mach. Intell. **23**(6), 643–660 (2001)
8. Görgel, P., Simsek, A.: Face recognition via deep stacked denoising sparse autoencoders (dsdsa). Appl. Math. Comput. **355**, 325–342 (2019)
9. Hammouche, R., Attia, A., Akhrouf, S., Akhtar, Z.: Gabor filter bank with deep autoencoder based face recognition system. Expert Syst. Appl. **197**, 116743 (2022)
10. Hashemifar, S., Marefat, A., Joloudari, J.H., Hassanpour, H.: Enhancing face recognition with latent space data augmentation and facial posture reconstruction. Expert Syst. Appl. **238**, 122266 (2024)
11. Horner, J.L., Gianino, P.D.: Phase-only matched filtering. Appl. Opt. **23**(6), 812–816 (1984)
12. Kumar, B.V., Fernandez, J.A., Rodriguez, A., Boddeti, V.N.: Recent advances in correlation filter theory and application. Optical Pattern Recognition XXV **9094**, 5–17 (2014)
13. Kumar, B.V., Mahalanobis, A., Juday, R.D.: Correlation pattern recognition. Cambridge university press (2005)
14. Li, S.Z., Chu, R., Liao, S., Zhang, L.: Illumination invariant face recognition using near-infrared images. IEEE Trans. Pattern Anal. Mach. Intell. **29**(4), 627–639 (2007)
15. Liu, D.H., Lam, K.M., Shen, L.S.: Illumination invariant face recognition. Pattern Recogn. **38**(10), 1705–1716 (2005)
16. Mahalanobis, A., Kumar, B.V.: Optimality of the maximum average correlation height filter for detection of targets in noise. Opt. Eng. **36**(10), 2642–2648 (1997)
17. Mahalanobis, A., Kumar, B.V., Song, S., Sims, S.R.F., Epperson, J.F.: Unconstrained correlation filters. Appl. Opt. **33**(17), 3751–3759 (1994)
18. Masci, J., Meier, U., Cireşan, D., Schmidhuber, J.: Stacked convolutional autoencoders for hierarchical feature extraction. In: Artificial Neural Networks and Machine Learning–ICANN 2011: 21st International Conference on Artificial Neural Networks, Espoo, Finland, June 14-17, 2011, Proceedings, Part I 21. pp. 52–59. Springer (2011)
19. Ojala, T., Pietikainen, M., Maenpaa, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans. Pattern Anal. Mach. Intell. **24**(7), 971–987 (2002)
20. Rizo-Rodríguez, D., Méndez-Vázquez, H., García-Reyes, E.: Illumination invariant face recognition using quaternion-based correlation filters. Journal of mathematical imaging and vision **45**(2), 164–175 (2013)
21. Sawides, M., Kumar, B.V., Khosla, P.K.: " corefaces"-robust shift invariant pca based correlation filter for illumination tolerant face recognition. In: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004. vol. 2, pp. II–II. IEEE (2004)
22. Sim, T., Baker, S., Bsat, M.: The cmu pose, illumination, and expression (pie) database. In: Proceedings of fifth IEEE international conference on automatic face gesture recognition. pp. 53–58. IEEE (2002)
23. Tschannen, M., Bachem, O., Lucic, M.: Recent advances in autoencoder-based representation learning. arXiv preprint arXiv:1812.05069 (2018)
24. Van Nevel, A., Mahalanobis, A.: Comparative study of maximum average correlation height filter variants using ladar imagery. Opt. Eng. **42**(2), 541–550 (2003)

25. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A., Bottou, L.: Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of machine learning research **11**(12) (2010)
26. Wang, B., Li, W., Yang, W., Liao, Q.: Illumination normalization based on weber's law with application to face recognition. IEEE Signal Process. Lett. **18**(8), 462–465 (2011)
27. Wang, H., Li, S.Z., Wang, Y.: Face recognition under varying lighting conditions using self quotient image. In: Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings. pp. 819–824. IEEE (2004)
28. Wang, W., Huang, Y., Wang, Y., Wang, L.: Generalized autoencoder: A neural network framework for dimensionality reduction. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops. pp. 490–497 (2014)
29. Zhang, C., Liu, Y., Fu, H.: Ae2-nets: Autoencoder in autoencoder networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2577–2585 (2019)
30. Zhang, T., Tang, Y.Y., Fang, B., Shang, Z., Liu, X.: Face recognition under varying illumination using gradientfaces. IEEE Trans. Image Process. **18**(11), 2599–2606 (2009)
31. Zhang, Y.: A better autoencoder for image: Convolutional autoencoder. In: ICONIP17-DCEC. Available online: http://users.cecs.anu.edu.au/Tom.Gedeon/conf/ABCs2018/paper/ABCs2018_paper_58.pdf (accessed on 23 March 2017) (2018)

Low-Resource Crop Classification from Multi-Spectral Time Series Using Lossless Compressors

Wei Cheng[1], Hongrui Ye[1], Xiao Wen[1], Jiachen Zhang[1], Jiping Xu[1],
and Feifan Zhang[2](✉)

[1] College of Information and Electrical Engineering, China Agricultural University,
Beijing 100083, China
{weicheng,caucieeyhr,wenxiao,jiachen_zhang,feifanzhang}@cau.edu.cn
[2] College of Science, China Agriculture University, Beijing 100083, China
jipingxu@cau.edu.cn

Abstract. Deep learning models have significantly improved the accuracy of crop classification using multispectral temporal data. However, they have complex structures with numerous parameters, requiring large amounts of data and costly training. In low-resource situations with fewer labeled samples or on low-computing devices, they perform poorly. Conversely, compressors are data-type agnostic, and non-parametric methods do not bring underlying assumptions. Inspired by this insight, we propose a non-training alternative to deep learning models, aiming to address these situations. Specifically, the symbolic representation module is proposed to convert the reflectivity into symbolic representations. The symbolic representations are then cross-transformed in both the channel and time dimensions to generate symbolic embeddings. Next, the Multi-scale Normalised Compression Distance (MNCD) is designed to measure the correlation between any two symbolic embeddings. Finally, based on the MNCDs, high quality crop classification can be achieved using only a k-nearest-neighbor classifier (kNN). The entire framework is ready-to-use and lightweight. Without any training, it outperforms, on average, 6 advanced deep learning models trained at scale on three benchmark datasets. It also outperforms more than half of these models in the few-shot setting with sparse crop labels.

Keywords: Non-training classification · Low-resource · Symbolic representation · Cross-transformation method · Lossless compressors · Nomalised compression distance

1 Introduction

Multi-spectral temporal classification of crops plays a crucial role in growth monitoring, pest forecasting, crop yield estimation, and other agricultural applications. This helps enhance the efficiency and quality of agricultural production [1].

W. Cheng, H. Ye and X. Wen—These authors contributed equally.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 190–205, 2025.
https://doi.org/10.1007/978-3-031-78383-8_13

However, multi-spectral temporal data typically appears as a high-dimensional feature space with numerous features. Redundant information can restrict data fitting and representation capabilities [13].

Deep learning can automatically extract features from high-dimensional data, providing extensive opportunities [15]. In the past decade, various neural network structures have been extensively researched, including networks based on multilayer perceptrons (MLP), convolutional neural networks (CNN), recurrent neural networks (RNN), and Transformer-based networks. Specifically, a method called Long-term Recurrent Convolutional Networks (LRCN) proposed by Donahue et al., which combines the strengths of CNN and RNN. By incorporating Long Short-Term Memory (LSTM) units, it is capable of establishing long-term temporal dependencies [4]. Next, Jia et al. developed a spatiotemporal learning framework based on a dual-memory structure of LSTM. This framework further extends the performance of LSTM. It enables LSTM to establish temporal dependencies and spatial relationships between long-term and short-term events in time and space [9]. Furthermore, Xu et al. developed a deep learning method named DeepCropMapping (DCM) model. This model is based on the Long Short-Term Memory structure with an attention mechanism. The DCM model is trained on ARD time series data, allowing it to learn generalizable features during the training process [24]. Unfortunately, these models require significant computational resources and time, leading to challenges in processing long time series data. To address this challenge, Zhou and colleagues introduced the Informer model. It is based on transformer architecture and significantly improves the speed of long time series prediction [27]. Additionally, Nie et al. created the PatchTST model. The model significantly enhances the classification results of long-term prediction [16]. Meanwhile,Liu et al. introduced the iTransformer model, showcasing superior technical capabilities across various real-world datasets. The iTransformer model demonstrates high performance and strong generalization abilities,and it is highly effective for time series prediction [14]. Currently, in the field of multivariate time series prediction, deep learning models such as CNNs, RNNs, and Transformers have all achieved highly advanced performance. However, in recent research, Zhang et al. introduced the LightTS architecture, which is a method that does not require any convolution or attention mechanisms. This study is the first to demonstrate that an MLP-based structure can also be highly efficient and accurate in multivariate time series prediction [26].

Although these backbone networks and their variants have achieved satisfactory classification accuracy, there are still shortcomings in the following aspects: (1) Deep learning models exhibit high complexity. They require a significant investment of resources during the training process and have a massive number of model parameters [26]; (2) The cost of labeling for deep learning models is high. Obtaining the necessary data requires a significant investment in professional labor and resources. In the few-shot scenario with sparse labels, the performance of deep learning models is not satisfactory [24]. Compressors are data-type agnostic, and non-parametric methods do not bring underlying assumptions [10].

These features can effectively address the aforementioned problems. Inspired by this, the proposed Symbolic Representation Module is used to convert the reflectivity of all pixels into symbol representations. The symbolic representations are then cross-transformed in both the channel and temporal dimensions to generate symbolic embeddings. Next, the Multi-scale Normalised Compression Distance (MNCD) is designed to measure the correlation between any two symbolic embeddings. Finally, based on the MNCDs, classification is implemented using only a kNN. Our method revolves around simple lossless compressors. It offers advantages such as no trainable parameters and a lightweight structure, ensuring its wide practical application.

To sum up, contributions of this research are:

- We treat a pixel as 'text' and introduce compressor-based classification from text classification to multispectral temporal crop classification.
- A Symbolic Representation Module is proposed to convert the reflectivity of pixels into symbolic representations. Based on the symbolic representations, a method of cross-transformation in time and channel dimensions is then proposed to generate symbolic embeddings. The Multi-scale Normalised Compression Distance (MNCD) is then designed to measure the correlation between any two symbolic embeddings for subsequent classification.
- The entire framework is ready-to-use and lightweight. Without any training, it outperforms the average of 6 advanced deep learning models trained at scale on three benchmark datasets. It performs exceptionally well in few-shot scenarios, where the availability of labeled data is limited for effective neural network training.

2 Materials and Methods

2.1 Data Description

Following the previous study [19], we choose three benchmark datasets to evaluate the performance of all methods. The **German dataset** [18] covers a densely cultivated area of 102 × 42 km^2 north of Munich, Germany. It contains 17 different categories with 13 spectral bands. The **T31TFM-1618 dataset** [20] covers a densely cultivated S2 tile in France for the years 2016 to 2018 and includes 20 different categories with 13 bands. The **PASTIS dataset** [6] contains four different regions in France, covering over 4000 km^2 and including 18 crop categories with 33-61 acquisitions and 10 bands.

For each of the three datasets, we identify areas that are not heavily clouded 80% of the time and exclude those that are. We also exclude background categories and those with less than 5 samples. In addition, due to the different resolutions of the different bands in the T31TFM-1618 dataset, we uniformly interpolate all bands to a maximum resolution of 48×48 pixels. Following previous studies on temporal classification [22], 20% of the pixel points from each crop type are randomly selected for the training dataset in the main experiment, while the remaining pixels are assigned to the test dataset. This leaves

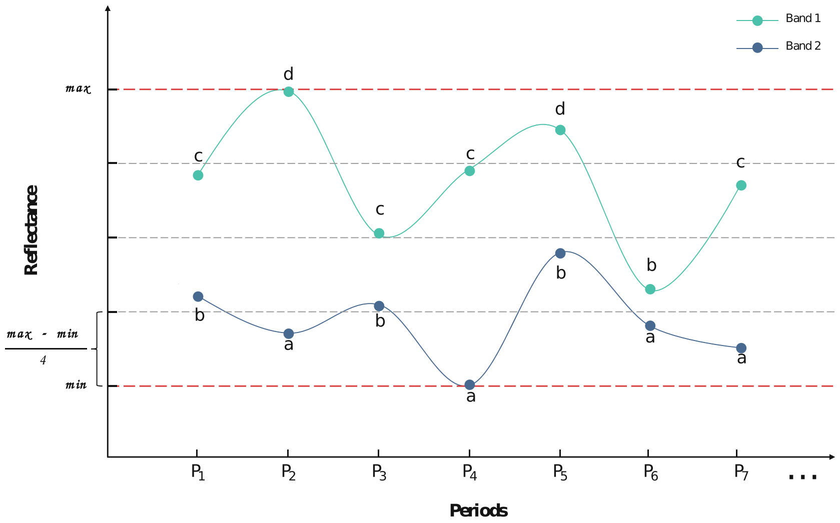


Fig. 1. These time series are discretized by using predetermined breakpoints B to map the reflectivity to symbols. In the example above, with $l = 4$, the time series of band 1 is mapped to **cdcdcbc** and the time series of band 2 is mapped to **bababaa**.

20k training and 80k testing samples for the German dataset, 40k training and 170k testing samples for the T31TFM-1618 dataset, 15k training and 61k testing samples for the PASTIS dataset, covering 266,36 ha land totally. To ensure the reproducibility of these experiments, we use a random seed of 32.

2.2 Design of Our Method

Symbolic Representation Module Lossless compressors are used in data storage and transmission scenarios. Their principle is to identify redundant information (such as repeated strings or symbols) in the data and replace it with a shorter representation, making them more suitable for data in symbolic form. Related research [13] has also demonstrated the effectiveness of converting time series into symbolic representations. Inspired by this, we propose a simple and general module for symbolic representation.

Specifically, we first define an alphabet $\boldsymbol{\alpha}$ with up to 26 lowercase letters and 26 uppercase letters, where α_k represents the k^{th} element of the alphabet, such as $\alpha_1 = a$, $\alpha_2 = b$, ..., $\alpha_{27} = A$. At the same time, the maximum and minimum values of the reflectivity in all pixels are obtained, namely max and min. Then, the reflectivity interval $[max, min]$ is divided into l equal intervals to obtain breakpoints $\boldsymbol{B}$, as shown below:

$$\boldsymbol{B} = [\beta_1, \ldots, \beta_i, \ldots, \beta_{l+1}], \tag{1}$$

where l represents the length of the alphabet $\boldsymbol{\alpha}$, $\beta_1 = min$, $\beta_{n+1} = max$, and $\beta_{i+1} - \beta_i = \frac{max-min}{l}$.

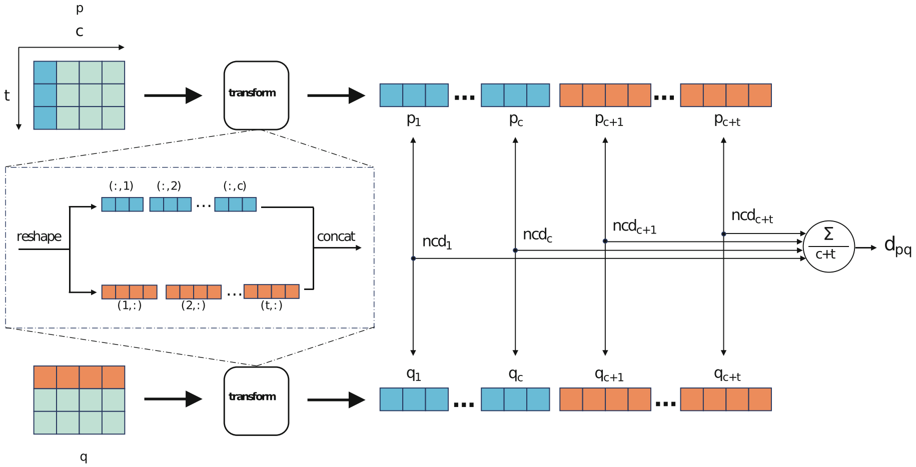


Fig. 2. An illustrative description of cross-transforming in the time and channel dimensions to obtain multi-scale NCD d_{pq} between pixels S_p and S_q after symbolic representation.

Once the breakpoints are determined, any reflectivity can be mapped to a symbolic representation. Given a pixel $\boldsymbol{x} \in \mathbb{R}^{t \times c}$ in the original data $\boldsymbol{X} \in \mathbb{R}^{h \times w \times t \times c}$ (where h is the height of the images and w is the width, t is the time, and c is the number of channels) , the symbolic representation value s_{ij} of any reflectivity x_{ij} can be obtained by the following mapping operation:

$$s_{ij} = \alpha_k, \text{ if } \beta_k \leq x_{ij} < \beta_{k+1}. \tag{2}$$

For example, all reflectivities greater than or equal to the first breakpoint β_1 and less than the second breakpoint β_2 are mapped to the symbol a; all reflectivities greater than or equal to the second breakpoint β_2 and less than the third breakpoint β_3 are mapped to the symbol b. The rest can be inferred by analogy. Figure 1 shows the details of this module. This mapping converts the reflectivity of all pixels into symbolic representations that are more suitable for compressors.

Compressor-Based Crop Classification Previous work combines band data at different time points for a single pixel to obtain information about the pixel [24]. Our method not only continues to use this time-series band combination, but also introduces a new dimension: combining data from the same pixel in different bands in chronological order. Then, by combining these two data sequences, we

can more comprehensively describe the characteristics of a single pixel. Specifically, given a pixel after symbolic representation $\boldsymbol{s} \in \mathbb{R}^{t \times c}$, the following modeling is given:

$$\boldsymbol{S} = f(\boldsymbol{s}) = \text{concat}\left(\boldsymbol{s}^1, \ldots, \boldsymbol{s}^c, \boldsymbol{s}^{c+1}, \ldots, \boldsymbol{s}^{c+t}\right), \tag{3}$$

where the function $f(\cdot)$ is the cross-transformation method of $\boldsymbol{s}$. The function concat$(\cdot)$ denotes the concatenation operation of the symbolic representation. $\boldsymbol{S}$ stands for the symbolic embedding of $\boldsymbol{s}$. For any $\boldsymbol{s}^i$ it holds that:

$$\boldsymbol{s}^i = \begin{cases} [s_{1i}, s_{2i}, \ldots, s_{ti}], & \text{if } 1 \le i \le c \\ \left[s_{(i-c)1}, s_{(i-c)2}, \ldots, s_{(i-c)c}\right], & \text{if } c < i \le c+t. \end{cases} \tag{4}$$

The above cross-transformation can be easily achieved using methods such as `numpy.reshape()` and `numpy.concatenate()`, which are based on numerical computing libraries such as NumPy.

For any two pixels $\boldsymbol{x}_p$ and $\boldsymbol{x}_q$, the regularity between them can be measured by the correlation between their corresponding symbolic embeddings, $\boldsymbol{S}_p$ and $\boldsymbol{S}_q$. And the regularity between two symbolic embeddings can be measured by the Nomalised Compression Distance (NCD) [12] of lossless compressors. Lossless compressors optimise the representation of information by assigning shorter codes to more frequent symbols. The idea is that: (1) compressors are good at capturing regularities; (2) objects within the same category exhibit more regularities than those from different categories [10].

Quantifying this regularity as d_{pq}, its definition is as follows:

$$d_{pq} = g(\boldsymbol{S}_p, \boldsymbol{S}_q) = \frac{1}{c+t} \sum_{k=1}^{c+t} NCD\left(\boldsymbol{s}_p^k, \boldsymbol{s}_q^k\right), \tag{5}$$

where the function $g(\cdot)$ represents the operation of Multi-scale NCD (MNCD) for $\boldsymbol{S}_p$ and $\boldsymbol{S}_q$, and d_{pq} represents the MNCD between $\boldsymbol{S}_p$ and $\boldsymbol{S}_q$. The definition of NCD is as follows:

$$NCD(m, n) = \frac{C(m||n) - \min\{C(m), C(n)\}}{\max\{C(m), C(n)\}}, \tag{6}$$

where $C(m)$ denotes the length of m after compression using a lossless compressor. $C(m||n)$ refers to the compressed length of m and n after being concatenated. This method can compute the MNCD between any symbolic embedding in the training set and symbolic embedding in the test set, and then construct a MNCD matrix. Finally, kNN is used to classify each symbolic embedding in the test set.

It is worth noting that when using a kNN, if $k > 1$, there may be situations where the number of samples from several different categories is the same among the k closest samples. To break the tie, we suggest finding the training set sample with the smallest NCD from the test sample among these categories. Then, the category of this training set sample will be used as the predicted category for the test sample. The code for the entire method is presented as Algorithm 1.

2.3 Implementations Details

The proposed framework uses *gzip* as the compressor. Under the premise of symbolic representation, the alphabet lengths l for the three datasets are all set to 22. The k value of kNN is set to 2. These parameters will be discussed in the subsequent Analyses section.

Algorithm 1 Compressor-Based Crop Classification

```
Require: Test set $\boldsymbol{E}$, Training set $\boldsymbol{T}$ with labels $\boldsymbol{L}$, number of nearest neighbours $k$,
    compression function $C(\cdot)$
Ensure: Predicted labels for all symbolic embeddings in $\boldsymbol{E}$
for each symbolic embedding $\boldsymbol{S}_1$ in the test set $\boldsymbol{E}$ do
    Initialize $\boldsymbol{D}_{\boldsymbol{S}_1}$ as an empty list
    $C_{\boldsymbol{S}_1} \leftarrow C(\boldsymbol{S}_1)$
    for each symbolic embedding $\boldsymbol{S}_2$ in the training set $\boldsymbol{T}$ do
        $C_{\boldsymbol{S}_2} \leftarrow C(\boldsymbol{S}_2)$
        $C_{\boldsymbol{S}_1||\boldsymbol{S}_2} \leftarrow C(\boldsymbol{S}_1||\boldsymbol{S}_2)$
        Compute $d_{12}$ based on the Eq. 5
        Append $(d_{12}, \text{label of } \boldsymbol{S}_2)$ to $\boldsymbol{D}_{\boldsymbol{S}_1}$
    end for
    $\boldsymbol{Y}_k \leftarrow$ labels of the $k$ nearest neighbors by sorting $\boldsymbol{D}_{\boldsymbol{S}_1}$ based on distances
    Compute the frequency of each unique label in $\boldsymbol{Y}_k$: $f_i = \sum_{j=1}^{k} \mathbb{I}(Y_j = i)$, where
    $\mathbb{I}(\cdot)$ is the indicator function
    Find the set of labels with maximum frequency: $\mathcal{M} = \{i : f_i = \max_j f_j\}$
    if $|\mathcal{M}| > 1$ then
        $i^* \leftarrow$ label of the sample in $\mathcal{M}$ with the smallest distance in $\boldsymbol{D}_{\boldsymbol{S}_1}$
    else
        $i^* \leftarrow$ the single element in $\mathcal{M}$
    end if
    Assign $i^*$ as the predicted label for $\boldsymbol{S}_1$
end for
```

2.4 Baselines

Several representative deep learning models are selected for comparative experiments. They are mainly divided into four groups: MLP-based, RNN-based, CNN-based and Transformer-based. We select both classic baselines and state-of-the-art (SOTA) models. The MLP-based category includes MLP and DLinear [25]. The RNN-based category includes Long Short-Term Memory (LSTM) [8] and DeepCropMapping (DCM) [24]. The CNN-based group includes TempCNN [23] and TimesNet [22]. The Transformer-based category includes Transformer [21] and Informer [27]. The parameters specific to these models are as follows and all models have converged in both the complete dataset and the few-shot settings:

For the MLP, we configure it with a single hidden layer of 128 neurons, a batch size of 32, and an initial learning rate of 0.001. We implement early stopping with a patience of 10 iterations and a validation fraction of 0.1, along with a maximum of 100 training epochs. The same configuration is used for the MLP in the few-shot setting.

For the DLinear, DCM, TimesNet and all Transformer-based models, we follow the optimal hyper-parameters recommended in the previous studies and corresponding repositories. Specifically, the batch size is set uniformly to 36, the dropout rate to 0.5, the number of epochs to 100 and the patience for early stopping is set to 10. In terms of model architecture, DLinear consists of three encoder layers and one decoder layer, each featuring a 256-dimensional feed-forward network. DCM, an attention-based bidirectional LSTM model, includes 2 LSTM layers with a hidden size of 256. The Transformer model is supported by three encoder layers and one decoder layer, also emphasizing a 256-dimensional feed-forward network. Similarly, Informer includes three encoder layers and one decoder layer. The same configuration is used for these models in the few-shot setting.

For the LSTM and TempCNN models, we follow the optimal hyperparameters recommended in the paper by Rubwurm et al. [17] and its corresponding repository. Specifically, the bidirectional LSTM stacks 4 layers with 128 hidden units. The learning rate is set to $9.88 \cdot 10^{-3}$ with a decay rate of $5.26 \cdot 10^{-7}$, and the number of epochs is 17. TempCNN uses a kernel size of 7 with 128 hidden units, a learning rate of $2.38 \cdot 10^{-4}$, a decay rate of $5.18 \cdot 10^{-5}$ and 11 epochs. For the few-shot setting, the number of epochs for the three models has been increased to 200 and 100.

2.5 Evaluation Metric

We evaluate the classification performance of each model based on two commonly-used indices, i.e., overall accuracy (OA), Mean Intersection over Union (MIoU). Specifically, the OA is the percentage of all correctly predicted instances in the dataset. The MIoU is the average ratio of overlap to union for predicted and actual segments

3 Results

3.1 Comparison with Deep Learning Models

The quantitative classification results in terms of OA and mIoU are shown in Table 1 for three datasets. Overall, the classification performance of the proposed method is significant. In terms of OA, it outperforms 5, 8 and 5 deep learning models on three datasets respectively. In particular, it outperforms all MLP-based and RNN-based models, demonstrating excellent classification performance. On the German dataset, however, it is not only 2.14% lower on average than the CNN-based model, but also 1.30% lower than the Informer. The performance gap is similar on the PASTIS dataset. A possible reason for this is that both CNN-based and Transformer-based models are able to learn high-level representations over time [22], which is crucial for classification tasks. However, such a result is acceptable given that our method can compete with massively trained deep learning models without any training. This is not always the case on the T31TFM-1618 dataset. The classification performance of our method outperforms all deep learning models. Compared to the CNN-based models and Transformer-based models, it averages 1.34% and 0.51% higher on OA, demonstrating excellent performance.

We also select a representative plot in Figure 3 with many boundaries and small fragmented fields, which thoroughly tests the classification capabilities of

Table 1. Classification results obtained with our method and baseline models. Red highlights the models that outperform our method.

model		trainable parameter	German			T31TFM-1618			PASTIS		
			OA	AA	mIoU	OA	AA	mIoU	OA	AA	mIoU
MLP	MLP	✓	69.62	52.00	34.24	68.49	44.37	31.11	92.06	67.68	60.40
	DLinear	✓	68.41	84.38	40.34	77.18	70.54	63.36	84.58	93.19	64.08
RNN	LSTM	✓	97.41	96.41	91.59	90.99	81.33	75.69	97.37	96.08	92.03
	DCM	✓	97.57	96.32	94.06	81.91	79.00	72.13	98.29	97.05	95.12
CNN	TempCNN	✓	98.61	97.79	96.18	95.74	92.26	88.66	98.67	98.23	96.47
	TimesNet	✓	98.75	98.19	96.56	96.68	95.40	92.87	99.08	97.26	97.03
Transformer	Transformer	✓	97.47	96.37	92.66	96.71	95.73	91.23	98.32	98.20	94.93
	Informer	✓	98.91	98.40	96.82	96.54	95.68	91.10	98.92	98.38	96.81
Ours		✗	97.61	96.89	94.10	96.88	95.42	91.43	98.37	97.64	95.39

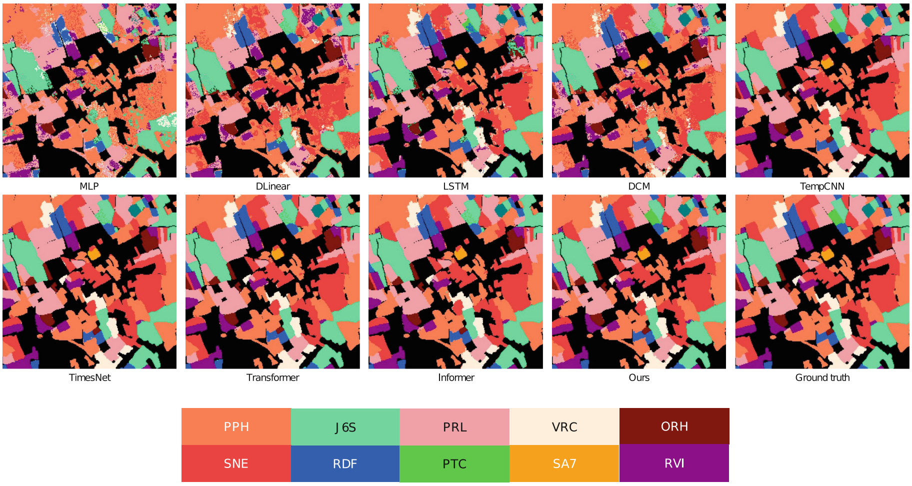


Fig. 3. Classification maps obtained from different models on a typical plot.

models. It is evident that the first four models easily make a lot of misjudgements, resulting in significant salt-and-pepper noise. In addition, the classification performance at the boundaries is rather coarse. In contrast, CNN-based models, Transformer-based models and our method show significant advantages. The boundaries are smoother and more refined, with almost no large-scale misclassifications. Considering that our method achieves classification performance comparable to deep learning models without requiring any training, it demonstrates significant advantages and practical value.

3.2 Few-Shot Learning

We also compare our method with deep learning models in the few-shot setting across three datasets. Following the method of Jiang et al. [11], we use

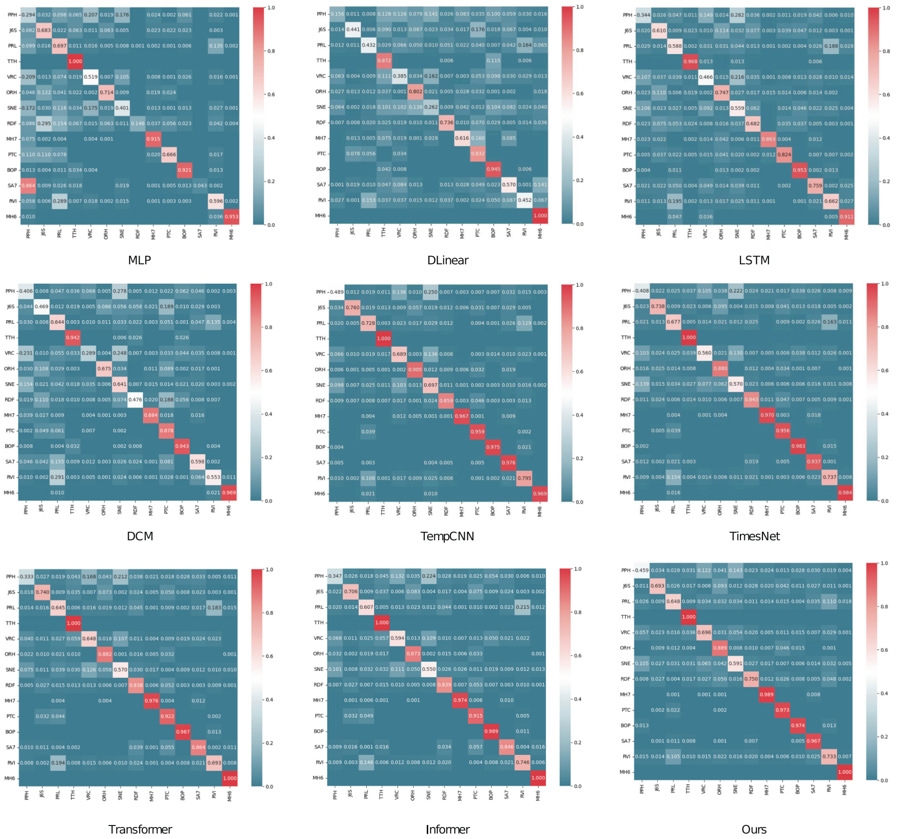


Fig. 4. Confusion matrix for 50-shot crop classification results on the T31TFM-1618 dataset.

n-shot labeled examples per category from the training dataset, where n sets to 50, 20, 10, 5. To ensure the robustness of our results, we run the experiment five times, using a different random seed to select the subset for each run. We then calculate the mean and 95% confidence intervals over the five trials. The five random seeds used are 2024, 21, 32, 400 and 47. The detailed results are listed in the Table 2.

As shown in Figure 4, the proposed method outperforms more than half of the deep learning models. Specifically, when n is set to 50, it significantly outperforms RNN-based and MLP-based models, while its performance is slightly lower than other SOTA models such as TimesNet and Informer. As n decreases to 20, it outperforms the Transformer by 0.13% in mIoU on the German dataset. However, it lags slightly behind other Transformer-based models. As n decreases further to 10, it widens its performance gap with RNN-based and MLP-based models, although it trails CNN-based models in mIoU by an average of 24.51% over the three datasets. This is consistent with the results of the Vision Trans-

Table 2. Results with a 95% confidence interval over five trials at different few-shot settings.

Models	mIoU (%)											
	German				T31TFM-1618				PASTIS			
	50-shot	20-shot	10-shot	5-shot	50-shot	20-shot	10-shot	5-shot	50-shot	20-shot	10-shot	5-shot
MLP	$42.23_{\pm 8.91}$	$26.91_{\pm 6.39}$	$18.93_{\pm 6.31}$	$4.67_{\pm 4.89}$	$24.08_{\pm 1.77}$	$15.31_{\pm 4.50}$	$7.22_{\pm 3.50}$	$2.94_{\pm 1.31}$	$34.45_{\pm 15.21}$	$10.15_{\pm 11.34}$	$2.51_{\pm 2.96}$	$3.17_{\pm 2.94}$
DLinear	$27.25_{\pm 1.14}$	$39.45_{\pm 0.52}$	$27.55_{\pm 20.64}$	$9.24_{\pm 3.06}$	$16.92_{\pm 0.60}$	$9.57_{\pm 0.62}$	$5.59_{\pm 2.03}$	$4.41_{\pm 1.81}$	$47.39_{\pm 3.78}$	$49.23_{\pm 23.63}$	$21.57_{\pm 5.29}$	$23.97_{\pm 27.15}$
LSTM	$52.91_{\pm 7.85}$	$41.08_{\pm 1.46}$	$29.62_{\pm 5.69}$	$19.78_{\pm 2.40}$	$27.00_{\pm 1.86}$	$21.50_{\pm 3.77}$	$17.29_{\pm 2.98}$	$12.57_{\pm 2.36}$	$70.14_{\pm 2.15}$	$54.97_{\pm 5.35}$	$45.75_{\pm 8.28}$	$34.50_{\pm 9.32}$
DCM	$46.12_{\pm 3.36}$	$23.36_{\pm 1.99}$	$12.46_{\pm 4.91}$	$9.21_{\pm 0.97}$	$23.95_{\pm 2.14}$	$17.61_{\pm 0.82}$	$9.77_{\pm 2.60}$	$4.84_{\pm 3.44}$	$59.96_{\pm 4.14}$	$39.32_{\pm 3.40}$	$36.78_{\pm 2.81}$	$16.82_{\pm 4.84}$
TempCNN	$70.66_{\pm 3.16}$	$56.43_{\pm 4.05}$	$43.72_{\pm 5.99}$	$31.40_{\pm 2.08}$	$40.48_{\pm 2.41}$	$30.73_{\pm 3.03}$	$25.02_{\pm 2.55}$	$21.02_{\pm 2.83}$	$75.59_{\pm 1.95}$	$57.51_{\pm 4.15}$	$42.53_{\pm 1.49}$	$29.18_{\pm 3.27}$
TimesNet	$52.99_{\pm 4.73}$	$44.37_{\pm 2.53}$	$57.11_{\pm 19.53}$	$19.42_{\pm 2.79}$	$33.12_{\pm 2.13}$	$26.92_{\pm 2.35}$	$21.02_{\pm 2.50}$	$16.34_{\pm 2.12}$	$71.82_{\pm 3.41}$	$61.57_{\pm 6.87}$	$74.68_{\pm 18.21}$	$34.49_{\pm 5.17}$
Transformer	$54.14_{\pm 7.82}$	$39.34_{\pm 3.09}$	$23.83_{\pm 2.59}$	$16.69_{\pm 1.58}$	$29.52_{\pm 1.18}$	$24.22_{\pm 2.31}$	$19.51_{\pm 1.67}$	$14.13_{\pm 2.04}$	$74.34_{\pm 3.49}$	$58.70_{\pm 4.47}$	$41.10_{\pm 4.72}$	$28.74_{\pm 3.93}$
Informer	$54.03_{\pm 2.08}$	$39.68_{\pm 4.10}$	$24.40_{\pm 3.10}$	$17.60_{\pm 0.76}$	$30.51_{\pm 1.06}$	$24.77_{\pm 2.45}$	$19.24_{\pm 1.28}$	$12.87_{\pm 1.80}$	$72.40_{\pm 3.37}$	$57.80_{\pm 3.25}$	$42.70_{\pm 4.59}$	$28.33_{\pm 2.65}$
Ours	$50.25_{\pm 2.83}$	$39.47_{\pm 2.34}$	$33.78_{\pm 3.77}$	$28.52_{\pm 2.19}$	$32.26_{\pm 0.68}$	$25.36_{\pm 1.87}$	$20.37_{\pm 1.09}$	$16.98_{\pm 1.08}$	$62.63_{\pm 2.21}$	$49.31_{\pm 2.88}$	$41.13_{\pm 2.06}$	$33.93_{\pm 3.57}$

former [5], which shows that the CNN-based models perform better in the few-shot setting. It is worth noting that when n falls to a minimum value of 5, our method outperforms almost all deep learning models on the three datasets. It is on average just under 3% lower than TempCNN or TimesNet. As the value of n gradually decreases, the advantage of our method becomes more pronounced. This may be because compressors are data type agnostic and non-parametric methods have no underlying assumptions [10].

4 Analyses

Due to the low compression speeds resulting from pure numerical mapping and the inherently low compression speeds of the $bz2$ compressor, we randomly sample 20% of the total dataset for all experiments in this section. Half of this subset is divided as the training set and the other half as the test set. To ensure the generalisability of the experiment, we repeat it five times, randomly selecting a different subset each time and then calculating the average of the five results. The five random seeds are 2024, 21, 32, 400, 47.

4.1 Different Alphabet Lengths

In the experiment, classification is performed by one group using pure numerical mapping, while the other group utilizes symbolic representation. The alphabet length starts at 2 and increases in increments of 5 up to 52 (including 26 lower case and 26 upper case letters). Figure 5 illustrates the effect of different mappings and different alphabet lengths on the classification results.

Compared to the symbolic representation mapping, the purely numerical mapping results in a worse classification performance. The mIoU is approximately 46%, 80.71% and 26.34% lower on the three datasets respectively (excluding the case where the length of the alphabet l is equal to 2). One possible reason for this is that the compressor *gzip*, which is widely used in the text domain [11], is more adept at capturing patterns in symbolic representations. However, purely numerical information is not the same as character patterns.

When symbolic representation mappings are used, mIoU initially shows an upward trend. A possible reason for this is that rich features are lost when the

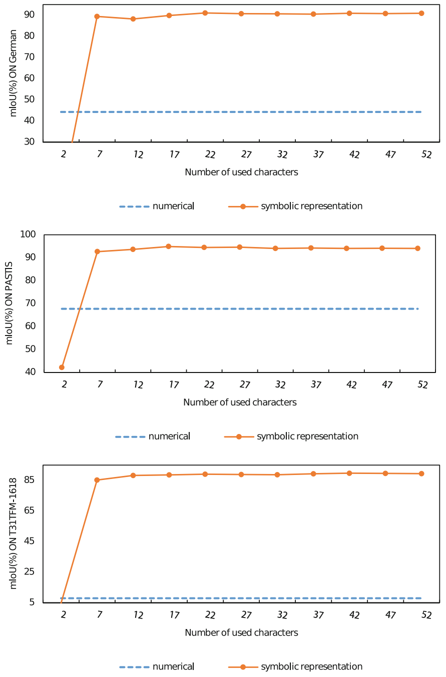


Fig. 5. Results on three datasets for different alphabet lengths.

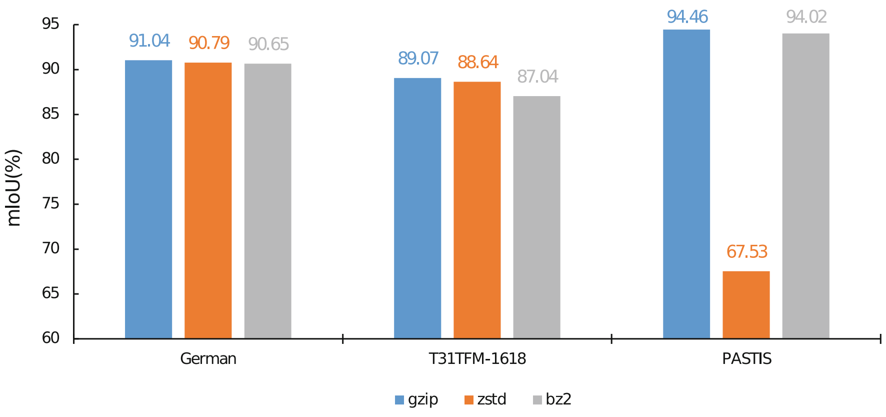


Fig. 6. Results on three datasets for different compressors. ‘zstd’ denotes the *zstandard* compressor.

variety of letters is too small. When the alphabet length exceeds 7, mIoU fluctuates within a range of no more than 3%, indicating relatively stable performance overall. This also demonstrates the robustness of the proposed mapping method.

4.2 Different Compressors

The effect of different compressors on the classification results is shown in Figure 6. On all three datasets, the classification performance of *gzip* is significantly better than all other compressors, with an average mIoU of 91.52%. This is consistent with the results of related research [11]. In comparison, the *bz2*'s mIoUs are on average 0.95% lower. The passable performance is probably due to the high compression ratio [11]. It is worth noting that on the PASTIS dataset, the classification accuracy of *zstandard* is 25% lower than that of other compressors, possibly due to the loss of fine-grained detail and changes in statistical properties during the compression process, which affect the classifier's performance.

5 Discussion

In response to the challenges of compressor-based classification, we have developed a lightweight and plug-and-play multispectral temporal classification method for agricultural crops. However, several challenges remain. First, the kNN algorithm that our method currently employs requires computing the distance between each test sample and all training samples, resulting in a high time complexity of $\mathcal{O}(m * n)$, where m is the number of training samples and n is the number of test samples. Since the method currently works on CPUs, this can lead to significant computation times in extreme cases. Second, our method uses only individual pixel information for classification, without considering the spatial relationships between neighboring pixels, an oversight that may limit potential improvements.

To address these challenges, future work could pursue several optimization directions. First, to improve the efficiency of kNN, spatial indexing structures such as KD trees [2] or ball trees [3] could be implemented to optimize the nearest neighbor search and reduce the time complexity to $\mathcal{O}(\log(m)*n)$. Second, the Local Binary Patterns coding scheme [7] could be used to capture the spatial relationships and textural information between pixels, thereby enriching the feature representation and improving classification performance. Looking to the future, potential advancements include improving the efficiency of the algorithms to handle extremely large datasets and integrating these models into automated systems for real-time crop monitoring and classification.

6 Conclusions

In this research, we introduce a non-training alternative to deep learning models and bring compressor-based classification from text classification to multispectral temporal crop classification. A novel Symbolic Representation Module is proposed to convert the reflectivities of all pixels into symbolic representations. These symbolic representations are then cross-transformed in both the channel and time dimensions to generate symbolic embeddings. Finally, based on the Multi-scale NCDs we have designed, crop classification is implemented using only a kNN. The results show that even without training, the proposed method achieves results comparable to those of large-scale trained deep learning models. It outperforms 5, 8, 5 advanced deep learning models on three benchmark datasets. It also outperforms more than half of these models in the few-shot setting with sparse labels. This lightweight and generalisation advantage contributes to its use in real-world agricultural production.

Acknowledgements. We acknowledge Eren Yeager and Mikasa Ackerman for advising on the design of the model. This work was supported by Mathematics Research Branch Institute of Beijing Association of Higher Education & Beijing Interdisciplinary Science Society.

References

1. Alzhanov, A., Nugumanova, A.: Crop classification using uav multispectral images with gray-level co-occurrence matrix features. Procedia Computer Science **231**, 734–739 (2024)
2. Bentley, J.L.: Multidimensional binary search trees used for associative searching. Commun. ACM **18**(9), 509–517 (sep 1975). https://doi.org/10.1145/361002.361007, https://doi.org/10.1145/361002.361007
3. Dolatshah, M., Hadian, A., Bidgoli, B.M.: Ball*-tree: Efficient spatial indexing for constrained nearest-neighbor search in metric spaces. CoRR **abs/1511.00628** (2015), http://arxiv.org/abs/1511.00628
4. Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T.: Long-term recurrent convolutional networks for visual recognition and description. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2625–2634 (2015)
5. 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)
6. Garnot, V.S.F., Landrieu, L.: Panoptic segmentation of satellite image time series with convolutional temporal attention networks. CoRR **abs/2107.07933** (2021)
7. Guo, Z., Zhang, L., Zhang, D.: A completed modeling of local binary pattern operator for texture classification. IEEE Trans. Image Process. **19**(6), 1657–1663 (2010). https://doi.org/10.1109/TIP.2010.2044957
8. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation **9**(8), 1735–1780 (1997), http://www7.informatik.tu-muenchen.de/~hochreit
9. Jia, X., Khandelwal, A., Nayak, G., Gerber, J., Carlson, K., West, P., Kumar, V.: Incremental dual-memory lstm in land cover prediction. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining. pp. 867–876 (2017)
10. Jiang, Z., Yang, M., Tsirlin, M., Tang, R., Dai, Y., Lin, J.: "low-resource" text classification: A parameter-free classification method with compressors. In: Findings of the Association for Computational Linguistics: ACL 2023. pp. 6810–6828 (2023)

11. Jiang, Z., Yang, M.Y.R., Tsirlin, M., Tang, R., Dai, Y., Lin, J.: "Low-Resource" Text Classification: A Parameter-Free Classification Method with Compressors. In: Findings of the Association for Computational Linguistics: ACL 2023. Association for Computational Linguistics (2023)
12. Li, M., Chen, X., Li, X., Ma, B., Vitányi, P.M.: The similarity metric. IEEE Trans. Inf. Theory **50**(12), 3250–3264 (2004)
13. Lin, J., Keogh, E., Lonardi, S., Chiu, B.: Experiencing sax: a novel symbolic representation of time series. Data Min. Knowl. Disc. **15**(2), 107–144 (2007)
14. Liu, Y., Hu, T., Zhang, H., Wu, H., Wang, S., Ma, L., Long, M.: itransformer: Inverted transformers are effective for time series forecasting. arXiv preprint arXiv:2310.06625 (2023)
15. Mavaie, P., Holder, L., Skinner, M.K.: Hybrid deep learning approach to improve classification of low-volume high-dimensional data. BMC Bioinformatics **24**(1), 419 (2023)
16. Nie, Y., Nguyen, N.H., Sinthong, P., Kalagnanam, J.: A time series is worth 64 words: Long-term forecasting with transformers. arXiv preprint arXiv:2211.14730 (2022)
17. Rubwurm, M., Lefevre, S., Korner, M.: Breizhcrops: A satellite time series dataset for crop type identification. In: Proceedings of the International Conference on Machine Learning Time Series Workshop. vol. 3 (2019)
18. Rußwurm, M., Körner, M.: Multi-temporal land cover classification with sequential recurrent encoders. CoRR **abs/1802.02080** (2018)
19. Tarasiou, M., Chavez, E., Zafeiriou, S.: Vits for sits: Vision transformers for satellite image time series. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10418–10428 (2023)
20. Tarasiou, M., Güler, R.A., Zafeiriou, S.: Context-self contrastive pretraining for crop type semantic segmentation. IEEE Transactions on Geoscience and Remote Sensing **60**, 1–17 (2022https://doi.org/10.1109/TGRS.2022.3198187
21. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. arXiv preprint arXiv:1706.03762 (Aug 2017), https://arxiv.org/abs/1706.03762v7
22. Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J., Long, M.: Timesnet: Temporal 2d-variation modeling for general time series analysis. In: International Conference on Learning Representations. ICLR, Virtual (2023), https://openreview.net/forum?id=YOUR_UNIQUE_ID
23. Wu, J., Cao, M., Cheung, J.C.K., Hamilton, W.L.: Temp: Temporal message passing for temporal knowledge graph completion. arXiv preprint arXiv:2010.03526 (2020), https://github.com/JiapengWu/TeMP
24. Xu, J., Zhu, Y., Zhong, R., Lin, Z., Xu, J., Jiang, H., Huang, J., Li, H., Lin, T.: Deepcropmapping: A multi-temporal deep learning approach with improved spatial generalizability for dynamic corn and soybean mapping. Remote Sens. Environ. **247**, 111946 (2020)
25. Zeng, A., Chen, M., Zhang, L., Xu, Q.: Are transformers effective for time series forecasting? In: Proceedings of the AAAI conference on artificial intelligence. vol. 37, pp. 11121–11128 (2023)

26. Zhang, T., Zhang, Y., Cao, W., Bian, J., Yi, X., Zheng, S., Li, J.: Less is more: Fast multivariate time series forecasting with light sampling-oriented mlp structures. arXiv preprint arXiv:2207.01186 (2022)
27. Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., Zhang, W.: Informer: Beyond efficient transformer for long sequence time-series forecasting. In: Proceedings of the AAAI conference on artificial intelligence. vol. 35, pp. 11106–11115 (2021)

Addressing Multi-Label Learning with Missing Labels via Feature Relevance guided Scaled Model Coefficients

Sanjay Kumar[1,2](✉) and Reshma Rastogi[1]

[1] Machine Learning and Statistical Inference (MLSI) Lab, Department of Computer Science, South Asian University, New Delhi, India
reshma.khemchandani@sau.ac.in

[2] Deshbandhu College, University of Delhi, New Delhi, India
skumar5@db.du.ac.in

Abstract. Feature relevance is pivotal in multi-label learning tasks, as certain features have a greater impact on labels than others. Given the high dimensionality of multi-label datasets, efficiently identifying and utilizing feature relevance in model coefficients determination is essential. This article introduces a multi-label learning with missing labels approach using a modified $l_{2,1}$-norm regularization to scale model coefficients based on feature relevance and ensure global sparsity similar to the Frobenius norm. The approach also incorporates label correlations learning for missing label recovery and expresses model coefficients through learned correlation decomposition. A squared Hinge loss function, known for its robust classification properties, is employed as an empirical loss measure. Experimental validation against six existing multi-label learning methods across six datasets and five metrics demonstrates the efficacy of our unified approach in both recovering missing labels and predicting new instance labels.

Keywords: Multi-label learning · Missing labels · Feature relevance · Model coefficients scaling · squared $l_{2,1}$-norm · Auxiliary label correlations

1 Introduction

In real-world scenarios, data instances often have multiple labels. For example, a tweet can express multiple emotions [1], a satellite image can show different weather conditions and terrain features [6]. Such classification problems extend the binary and multi-class classification problems and are referred as multi-label learning (MLL) problems. A common approach to MLL is binary relevance, which treats each label independently but ignores useful label correlations. Extensions to this approach exploit first-order, second-order, or high-order label correlations. MLL methods are classified as either Problem Transformation,

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 206–222, 2025.
https://doi.org/10.1007/978-3-031-78383-8_14

which converts MLL into binary tasks, or Algorithm Adaptation, which modifies existing algorithms [11].

Typically, MLL assumes complete label information, which is rare [19]. Capturing label information is expensive and error-prone, often resulting in ambiguities and partially labeled training data. Multi-label learning with missing labels is challenging and relatively under-researched. Since most multi-label methods assume complete label information and don't account for missing labels, their models are often suboptimal. The absence of complete label information results in incorrect label correlation, which are otherwise considered essential prior knowledge for multi-label learning. Some methods incorporate the recovery of missing labels into the model-building process, improving the accuracy of label correlations. This improved label correlation structure can then guide the computation of model coefficients, enhancing the prediction of label vectors for new data instances [8,12].

Identifying the correlation between input features and output labels is crucial for multi-label learning. Several feature selection methods [10] are designed for MLL, focusing on identifying or ranking relevant features for algorithm input [23]. However, these methods often treat feature selection separately, leaving room to explore unified approaches that integrate feature selection and relevance identification within multi-label learning with missing labels [5].

While several methods address multi-label learning in various scenarios, approaches for handling missing labels constitute only a small proportion of the research. This paper tackles missing labels by exploring feature relevance and learning label correlations. Our model optimizes weights using squared $l_{2,1}$-norm regularization, preventing overfitting. It simultaneously recovers missing labels and trains a classifier using squared Hinge loss. Empirical evaluations demonstrate competitive performance against existing methods.

The structure of the paper is as follows: Section 2 reviews related work, Section 3 outlines the proposed model, Section 4 presents empirical results, and Section 5 provides the conclusion.

2 Related Work

In multi-label learning, datasets are often sparse and high-dimensional, and labels can be ambiguous. Due to the complexity and ambiguity in the output label space, labeling is a challenging task and datasets frequently have only partial label information.

Several approaches address this issue: pre-processing methods, transductive methods, and synchronized methods. Pre-processing methods first reconstruct missing labels and then train a classifier with the complete label set [19,20]. Transductive methods aim to recover unobserved labels only in the training data, often using matrix completion [7]. Synchronized methods simultaneously train the classifier and recover missing labels [9,14].

In the absence of complete labels, identifying correlations between features and labels becomes crucial. Some methods use label manifolds and feature structures to restore missing labels and build the classifier [17]. Techniques like label-specific features are also popular for identifying relevant features in multi-label learning [13].

Huang et al. [8] propose learning label correlations and constructing a supplementary label matrix to handle missing labels, aiding both label recovery and classifier training. Kumar et al. [11] suggest a low-rank approximation of the label space with auxiliary correlations, using squared Hinge-loss for empirical loss, which offers good classification properties [21]. Huang et al. [9] discuss embedding label correlations into regression coefficients.

Model coefficients are regularized with different norms to achieve sparsity and additional objectives, such as extracting label-specific features. Chen et al. [2] propose a squared $l_{2,1}$-norm approach for regression coefficients with nice properties such as identifying the feature importance in least square problems. We extend these approaches to multi-label learning by constructing model coefficients and learning auxiliary label correlations bidirectionally. The proposed method does a **f**eature relevance based **w**eights scaling of model coefficients for **m**ulti-label learning with **m**issing **l**abels (FWMML).

3 Proposed Model

Let $\{(x_p, y_p)\}_{p=1}^{n}$ be a multi-label classification dataset with n samples. Each data instance $x_p \in \mathbb{R}^d$ has a label vector $y_p \in \{-1, 0, 1\}^l$, where l is the number of labels. The data matrix $X \in \mathbb{R}^{n \times d}$ contains d features, and the label matrix $Y \in \{-1, 0, 1\}^{n \times l}$ indicates label presence (1), absence (−1), or unobserved (0). The goal is to train a classifier from $\{X, Y\}$ to predict a label vector $y \in \{-1, 1\}^l$ for a new instance $x \in \mathbb{R}^d$. Using these notations, we will develop a model for multi-label learning with missing labels.

3.1 Least Squared Hinge Loss for classification

We adopt a squared Hinge loss in our formulation, known for its effectiveness in measuring empirical risk in classification tasks, while preserving its favorable classification properties [11]. Frobenius norm regularization is applied to the model coefficients. Our base multi-label optimization problem can be expressed as:

$$\min_{W} \ \frac{1}{2} \| (|E - (Y) \circ (XW)|_{+})^2 \|_1 + \lambda_W \|W\|_F^2 \tag{1}$$

where $\|.\|_1$ represents the l_1-norm, $|\alpha|_{+} = max(0, \alpha)$, W is the model coefficient matrix, XW represents the prediction score, $E \in \mathbb{R}^{n \times l}$ is the matrix of ones and $\circ$ is the notation for element-wise Hadamard product of matrices. $\|.\|_F^2$ is regularizer for model coefficients.

3.2 Feature Importance Based Model Coefficient Scaling

Features in datasets vary in their impact on label assignments, with some having greater influence than others, especially in high-dimensional and sparse multi-label datasets. Identifying these influential features is crucial. In the following subsection, we introduce a method for scaling model coefficients based on feature importance, supported by theoretical proof.

To quantify the importance of d features $f_1, f_2, ..., f_d$, we introduce relevance weights ψ, where ψ_k denotes the relevance weight for k-th feature. Scaling model coefficients using ψ based on feature relevance, we can express (1) as:

$$\begin{aligned} \min_{W} \ & \frac{1}{2}\|(|E - (Y) \circ (X\Psi W)|_+)^2\|_1 + \lambda_W \|W\|_F^2 \\ & s.t.\ \mathbf{1}^T\psi = 1 \end{aligned} \tag{2}$$

where $\Psi \in \mathbb{R}^{d\times d}$ is a diagonal matrix with $\Psi_{kk} = \psi_k^{\frac{1}{2}}$ as the k-th diagonal element. The problem above leverages feature relevance to adjust model coefficients effectively. Next, we'll demonstrate through the following theorem that explicitly tuning coefficients based on feature relevance in a hinge loss formulation for multi-label learning can be achieved using the squared $l_{2,1}$-norm. The $l_{2,1}$-norm of a matrix A is defined as:

$$\|A\|_{2,1} = \sum_{j=1}^{d}\left(\sum_{k=1}^{l} a_{jk}^2\right)^{\frac{1}{2}},$$

where a_{jk} is the element at the j-th row and k-th column of matrix A.

Theorem 1.

$$\min_{W} \ \frac{1}{2}\|(|E - (Y) \circ (XW)|_+)^2\|_1 + \lambda_W \|W\|_{2,1}^2 \tag{3}$$

Formulation above is an equivalent expression for problem given by (2) where ψ_k is calculted as:

$$\psi_k = \frac{\|w^k\|_2}{\sum_{k=1}^{d}\|w^k\|_2} \tag{4}$$

Proof. Substituting $\overline{W} = \Psi W$, we can rewrite (2) as:

$$\begin{aligned} \min_{\overline{W},\psi} \ & \frac{1}{2}\|(|E - (Y) \circ (X\overline{W})|_+)^2\|_1 + \lambda_W \sum_{k=1}^{d} \frac{\|\overline{w}^k\|_2^2}{\psi_k} \\ & s.t.\ \mathbf{1}^T\psi = 1 \end{aligned} \tag{5}$$

For optimal value of ψ, fixing $\overline{W}$, the optimization problem becomes

$$\min_{\psi, \mathbf{1}^T\psi=1} \ \sum_{k=1}^{d} \frac{\|\overline{w}^k\|_2^2}{\psi_k} \tag{6}$$

The optimal solution to (6) is given by:

$$\psi_k = \frac{\|\overline{w}^k\|_2}{\sum_{\bar{k}=1}^{d} \|\overline{w}^{\bar{k}}\|_2} \tag{7}$$

above is equivalent to writing (6) as:

$$\min_{\psi, 1^T\psi=1} \|\overline{W}\|_{2,1}^2 \tag{8}$$

Using the deduction above, we can write (5) as:

$$\min_{\overline{W}} \frac{1}{2}\|(|E-(Y)\circ(X\overline{W})|_+)^2\|_1 + \lambda_W \|\overline{W}\|_{2,1}^2 \tag{9}$$

By integrating the squared $l_{2,1}$-norm into our squared Hinge loss formulation, we are able to achieve feature-based scaling of model coefficients and regularization akin to the Frobenius norm. To simplify the notation, replacing $\overline{W}$ with W, we can rewrite (9) as:

$$\min_{W} \frac{1}{2}\|(|E-(Y)\circ(XW)|_+)^2\|_1 + \lambda_W \|W\|_{2,1}^2 \tag{10}$$

3.3 Auxiliary Label Learning for Missing Label Recovery

Label correlations encode valuable associations between labels, crucial for recovering missing labels. By enhancing the incomplete label matrix with learned correlations and recovering an auxiliary label matrix, we can strengthen model training to improve classification accuracy.

Let $R \in \mathbb{R}^{l\times l}$ denotes the matrix containing learnt label correlation information where r_{mn} in R denotes the correlation between the m-th and n-th labels. Incorporating R to learn label correlations and recover missing labels, we can express the optimization problem as:

$$\begin{aligned}\min_{W,R}\ & \frac{1}{2}||(|E-(YR)\circ(XW)|_+)^2||_1 + \frac{\lambda_C}{2}\|YR-Y\|_F^2 \\ & + \lambda_W\|W\|_{2,1}^2 + \lambda_R\|R\|_F^2\end{aligned} \tag{11}$$

As label correlation matrix R is learnt, it also assists with learning model coefficients W. For regularizing learnt label correlations R, we take the Frobenius norm.

Association between model coefficient matrix W and label correlation matrix R is expressible in several ways. In this discussion, we also express the model coefficients as a decomposition of the label correlations R [9]. Rewriting the optimization problem to reflect the label correlation decomposition, we have:

$$\begin{aligned}\min_{W,R}\ & \frac{1}{2}||(|E-(YR)\circ(XW)|_+)^2||_1 + \frac{\lambda_C}{2}\|YR-Y\|_F^2 \\ & + \frac{\lambda_E}{2}\|R-W^TW\|_F^2 + \lambda_W\|W\|_{2,1}^2 + \lambda_R\|R\|_F^2\end{aligned} \tag{12}$$

where λ_W, λ_C, λ_E, λ_W, and λ_R are tradeoff parameters. Once label correlations R are learnt, we can interpret YR as auxiliary label matrix which will contain information not only for observed labels but also for unobserved labels.

3.4 Instance Similarity

According to the manifold smoothness assumption, instances that are close to each other are likely to share similar label vectors in a multi-label setting. If the proximity between two instances x_u and x_v is measured by $p_{uv} = exp^{-\|x_u - x_v\|_2^2}$, and the predicted label vectors h_u and h_v are expected to be similar, we can incorporate this assumption into the optimization formulation by introducing a regularizer that promotes smoothness:

$$\begin{aligned} R_I &= \frac{1}{2}\sum_{u,v=1}^{n} p_{uv}\|h_u - h_v\|_2^2 \\ &= trace((XW)^T L_I XW) \end{aligned}$$

where XW denotes the prediction matrix for training instances in X, and $L_I \in \mathbb{R}^{n\times n}$ denotes the Laplacian for instance similarity.

Adding the above smoothness term for instance similarity R_I along with tradeoff parameter λ_5, the optimization problem can be written as:

$$\begin{aligned} \min_{W,R}\ & \frac{1}{2}\|(|E - (YR)\circ(XW)|_+)^2\|_1 + \frac{\lambda_C}{2}\|YR - Y\|_F^2 \\ & + \frac{\lambda_E}{2}\|R - W^T W\|_F^2 + \frac{\lambda_W}{2}\|W\|_{2,1}^2 + \frac{\lambda_R}{2}\|R\|_F^2 \\ & + \frac{\lambda_I}{2} trace((XW)^T L_I XW) \end{aligned} \tag{13}$$

Also note if two labels i and j are identified as strongly correlated due to large r_{ij} entry in the label correlation matrix R, corresponding columns in the regression coefficient matrix W will also be similar. If w^i and w^j are columns in W, we can capture this relationship as:

$$\begin{aligned} \frac{1}{2}\sum_{i,j=1}^{l} r_{ij}||w^i - w^j||_2^2 &= \frac{1}{2}\left(\sum_{i,j=1}^{l} r_{ij}(\|w^i\|_2^2 + \|w^j\|_2^2) - 2\sum_{i,j=1}^{l} r_{ij}\langle w^i, w^j\rangle\right) \\ &= trace(WD_R W^T) - trace(WRW^T) \\ &= trace(WLW^T) \end{aligned}$$

where L represents the graph Laplacian for learnt label correlation matrix R.

Incorporating above in model formulation, the optimization problem can be written as:

$$\begin{aligned}\min_{W,R}\ &\frac{1}{2}||(|E-(YR)\circ(XW)|_{+})^2||_1+\frac{\lambda_C}{2}\|YR-Y\|_F^2\\ &+\frac{\lambda_E}{2}\|R-W^TW\|_F^2+\frac{\lambda_W}{2}\|W\|_{2,1}^2+\frac{\lambda_R}{2}\|R\|_F^2\\ &+\frac{\lambda_I}{2}trace((XW)^TL_IXW)+\lambda_L trace(WLW^T)\end{aligned}\tag{14}$$

where λ_L is the tradeoff parameter for final term. Finally, we define matrix $\Omega\in\mathbb{R}$ such that $\Omega_{i,j}=1$ when corresponding label information Y_{ij} is observed and $\Omega_{i,j}=0$ if label is unobserved. Incorporating Ω in optimization to account for loss evaluation only for observed labels and renaming the λ_s to simplify the notation, we have the final optimization problem as:

$$\begin{aligned}\min_{W,R}\ &\frac{1}{2}||\Omega\circ(|E-(YR)\circ(XW)|_{+})^2||_1+\frac{\lambda_1}{2}\|\Omega\circ(YR-Y)\|_F^2\\ &+\frac{\lambda_2}{2}\|R-W^TW\|_F^2+\frac{\lambda_3}{2}\|W\|_{2,1}^2+\frac{\lambda_4}{2}\|R\|_F^2\\ &+\frac{\lambda_5}{2}trace((XW)^TL_IXW)+\lambda_6 trace(WLW^T)\end{aligned}\tag{15}$$

3.5 Model Optimization

The optimization problem in (15) is composed of two variables: the regression coefficient matrix W and the learnt label correlations R. Let's denote the final objective for optimization problem using $F(\Theta)$ where $\Theta=\{W,R\}$ is the set of function parameters. The objective function can be optimized by iteratively minimizing over W and R alternately using gradient descent.

Updating W: Considering R as constant, gradient of $F(\Theta)$ w.r.t W is given by:

$$\begin{aligned}\nabla_W F(\Theta)=&X^T(|E-(YR)\circ(XW)|_{+}\circ(-YR))\\ &+2\lambda_2(-W)(R-W^TW)+\lambda_3QW\\ &+\lambda_5X^TL_IXW+\lambda_6W(L+L^T)\end{aligned}\tag{16}$$

where $Q\in\mathbb{R}^{d\times d}$ such that

$$q_{ii}=\frac{\sum_{k=1}^{d}\sqrt{\|w^k\|_2^2+\epsilon}}{\sqrt{\|w^i\|_2^2+\epsilon}}$$

ϵ is a small residual constant.

Once the derivative is available, W can be updated using gradient descent with learning rate η_1 as:

$$W=W-\eta_1\nabla W$$

Updating R: Fixing W, gradient of $F(\Theta)$ w.r.t R is given by:

$$\begin{aligned}\nabla_R F(\Theta) =& Y^T(|E-(YR)\circ(XW)|_+\circ(-XW)) \\ &+\lambda_1 Y^T(YR-Y)+\lambda_2(R-W^TW)+\lambda_4 R\end{aligned} \tag{17}$$

Using the derivative above, update rule for R can be written using gradient descent with learning rate η_2 as:

$$R = R - \eta_2 \nabla R$$

Using the aforementioned mathematical derivations, we formalize the optimization steps based on gradient descent in Algorithm 1. The major computation steps in Algorithm 1 are gradient calculation for W and R, the overall complexity is of the order of $O(nl^2+ndl+d^2l)$ suitable for small to medium size datasets, where l denotes total number of labels, d denotes the input space dimension and n is the total number of instances.

Algorithm 1 Optimization Steps for FWMML

```
Require: Training instance matrix $X \in \mathbb{R}^{n\times d}$, class labels $Y \in \{-1,0,1\}^{n\times l}$, regularization parameters $\lambda_1$, $\lambda_2$, $\lambda_3$, $\lambda_4$, $\lambda_5$, and $\lambda_6$, learning rate $\eta_1$ and $\eta_2$
Ensure: W, R
Initialisation: $W_0 \in \mathbb{R}^{d\times l}$, and $R_0 \in \mathbb{R}^{l\times l}$ randomly, t=0.
REPEAT until convergence
    Compute $\nabla_W F(W_t, R_t)$ according to equation (16).
    Update $W$ as
        $W_{t+1} = W_t - \eta_1 \nabla_W F(W_t, R_t)/\|\nabla_W F(W_t, R_t)\|$
    Compute $\nabla_R F(W_{t+1}, R_t)$ according to equation (17).
    Update $R$ as
        $R_{t+1} = R_t - \eta_2 \nabla_R F(W_{t+1}, R_t)/\|\nabla_R F(W_{t+1}, R_t)\|$
    $W_t = W_{t+1}$; $R_t = R_{t+1}$; $t = t+1$
RETURN $W, R$
```

4 Experiments and Discussion

For performance evaluation and comparison with other state-of-the-art algorithms, we use six multi-label datasets collected from Mulan[1] [18] and KDIS[2]. The datasets represent a mix of varying characteristics. Dataset characteristics are listed in Table 1.

[1] http://mulan.sourceforge.net/datasets-mlc.html

[2] http://www.uco.es/kdis/mllresources/

Table 1. Multi-label datasets Characteristics.

Dataset	Instances	Features	Labels	Lcard	avgIR	Domain
cal500	502	68	174	26.044	20.578	music
Image	2000	294	5	1.236	1.193	image
yeast	2417	103	14	4.237	7.197	biology
philosophy	3971	842	233	2.272	68.753	text
stackex_chemistry	6961	540	175	2.109	56.878	text
delicious	16105	500	983	19.02	71.134	text

4.1 Multi-label Metrics

Five multi-label metrics [22] are used for measuring the performance of proposed method against other state-of-the-art algorithms for multi-label learning with missing labels. Let $M = \{(x_k, Y_k)\}_{k=1}^{n}$ be the dataset for testing performance, where Y_k denotes the ground truth label vector for instance x_k. Further, let $h(x_k)$ represents the predicted label vector for k-th instance and $f(x_k, y)$ the confidence value that y is the vector of predicted labels for x_k. Definition of the multi-label metrics used for evaluation are as follows:

1. Hamming Loss(HL): Hamming loss determines the fraction of incorrectly classified instance-label pairs..

$$\text{HL} = \frac{1}{n}\sum_{k=1}^{n}\frac{1}{l}|h(x_k)\Delta Y_k|,$$

   where Δ denotes the symmetric difference, n denotes the number of instances, and l indicates the number of labels.

2. Average Precision(AP): Average precision assess' the average fraction of positive labels that are higher than a particular label.

$$\text{AP} = \frac{1}{n}\sum_{k=1}^{n}\frac{1}{|Y_k|}\sum_{y\in Y_k}\frac{|\{y'|rank_{f(x_k,y')} \le rank_{f(x_k,y)}, y' \in Y_k\}|}{rank_{f(x_k,y)}}.$$

3. Ranking Loss (RL): Ranking loss determines the fraction of negative labels ranked higher than that of positive labels.

$$\text{RL} = \frac{1}{n}\sum_{k=1}^{p}\frac{|\{(y', y'')|f(x_k, y') \le l_k\}|}{|Y_k||\bar{Y_k}|}$$

   where $l_k = f(x_k, y'')|(y', y'') \in Y_k \times \bar{Y_k})$.

4. Coverage (Cov): Coverge indicates the number of more labels on average that should be included to cover all relevant labels.

$$\text{Cov} = \frac{1}{n}\sum_{i=1}^{p}[[\max rank(x_i, j) - 1|j \in Y_i^{+}]]$$

5. Average Area under the ROC Curve(AUC): AUC calculates the average fraction of the ranked positive instances of all labels which are higher than negative instances.

$$\text{AUC} = \frac{1}{l}\sum_{k=1}^{l}\frac{|\{(x,x')|f(x,y_k) \geq f(x',y_k),(x,x') \in I_k \times \bar{I_k}\}|}{|I_k||\bar{I_k}|}$$

where I_k and $\bar{I_k}$ represent positive and negative instances of the kth class label.

4.2 Baselines

We compare our proposed method with six state-of-the-art algorithms, each selected with hyperparameters as per their respective literature. Performance evaluation is conducted using 5-fold cross-validation averaged over four runs across various multi-label datasets.

- LSLC [3] recovers missing labels by learning positive and negative label correlations along with learning label specific features for model training. The hyperparameters are searched in the range $\{2^{-10}, 2^{-9}, \ldots, 2^1\}$.
- MLLCRS [16] considers label-specific features along with utilizing the structural property of data and pairwise label correlation. The hyperparameters are searched in the range $\{10^{-5}, 10^{-4}, \ldots, 10^3\}$.
- GLOCAL [24] exploits local and global label correlations by learning latent label representations and optimizing label manifold. Hyperparameter λ is chosen as 1 and $\lambda_1 \ldots \lambda_5$ are searched in the range $\{10^{-5}, 10^{-4}, \ldots, 10^2\}$. Parameter k is searched in $\{5, 10, 15, 20\}$.
- LSML [8] learns label specific features and prepares a supplementary matrix augmented from incomplete label matrix learning high order label correlations for missing label recovery and training multi-label classifier. The search range for hyperparameters is $\{10^{-5}, 2^{-4}, \ldots, 10^3\}$.
- CIMML [15] exploits auxiliary label correlations and adjusts label weights based on the proportion of missing labels to train a multi-label classifier. Hyperparameters are searched in $\{10^{-5}, 10^{-3}, \ldots, 10^3\}$.
- DM2L [14] models both local and global low rank structures and discriminates labels by expanding globally and shrinking locally. DM2L-linear is used for comparison in this paper. The hyperparameter λ_d for DM2L is selected from $\{10^{-5}, 10^{-4}, \ldots, 10^5\}$.
- FWMML[3] is the approach discussed in this paper. It recovers missing labels and trains the classifier by imposing squared $l_{2,1}$-norm constraint on the regression coefficients which results in adjusted weights based on features relevance. The hyperparameters λ_is are searched in $\{10^{-8}, 10^{-7}, \ldots, 10^4\}$.

[3] https://github.com/sanjayksau/fwmml/

4.3 Empirical Results and Discussion

We compare FWMML with six leading algorithms using six benchmark multi-label datasets, assessed via 5-fold cross-validation. Missing labels are simulated at rates of 30%, 50%, and 70% in the training set. Performance metrics in Tables 2 to 4 include hamming loss, ranking loss, coverage (lower is better), and average precision, AUC (higher is better). The best-performing algorithm in each row is highlighted in bold. Results demonstrate FWMML's competitive performance across all metrics and missing label rates.

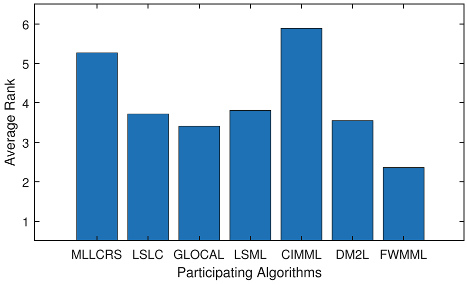


Fig. 1. Consolidated average rank of participating algorithms across all metrics.

For statistical comparison of multiple classifiers, we use the Friedman test [4] with seven algorithms ($k = 7$) and eighteen ($N = 6 \times 3$) data points. Each dataset contributes three points corresponding to the missing label rates. Table 5 shows the critical values of each evaluation metric and Friedman statistics F_f. The critical value $F(6, 17)$ for $\alpha = 0.05$ is 2.6987, allowing us to reject the null hypothesis of similar performance across all metrics except Hamming loss.

We further analyze pairwise comparisons using the Nemenyi test [4]. For seven classifiers, the critical value, $\alpha_{0.5}$ is 2.949, and the CD is 2.1235. Significant differences exist between the best and worst-performing algorithms across all metrics except Hamming loss, as shown in Table 6 and Figure 2. Figures 1 shows the consolidated avg. rank across all metrics. Overall, FWMML demonstrates competitive and often superior performance.

4.4 $||\boldsymbol{W}||_{2,1}$-norm vs $||\boldsymbol{W}||_F$-Norm Based Performance Comparison

To empirically study the impact of $||W||_{2,1}$ and $||W||_F$-norm, we compare performance medical dataset using three multi-label metrics: hamming loss, average precision, and ranking loss. Equation 3 is used for $||W||_{2,1}$-norm, and then replaced with $||W||_F$-norm for Frobenius norm performance. Figure 3 shows that at a missing label rate of 0.3, there is minimal difference in results. However, as the missing label rate increases, $||W||_{2,1}$-norm demonstrates superior performance, highlighting the advantages of using squared $l_{2,1}$-norm over Frobenius norm.

Table 2. Performance results for competing algorithm when missing label rate is 0.3.

Datasets	Metric	MLLCRS	LSLC	GLOCAL	LSML	CIMML	DM2L	FWMML
cal500	HL	0.1434	0.1375	0.1373	0.1374	0.1389	**0.1371**	**0.1371**
	AP	0.4952	0.5001	0.5001	0.5005	0.2561	0.4986	**0.5011**
	RL	0.1853	0.1804	0.1800	0.1798	0.8321	0.1805	**0.1796**
	Cov	0.7711	0.7496	0.7450	0.7478	0.8720	**0.7437**	0.7467
	AUC	0.8112	0.8160	0.8162	0.8168	0.6874	0.8161	**0.8170**
chemistry	HL	**0.0115**	0.0115	0.0115	0.0115	0.0121	0.0120	0.0120
	AP	0.3982	**0.4066**	0.3577	0.3980	0.2356	0.3258	0.2615
	RL	0.1835	0.1705	0.1863	0.1749	0.1767	**0.1234**	0.1817
	COV	0.3066	0.2903	0.3103	0.2944	0.2790	**0.2769**	0.2903
	AUC	0.8058	0.8179	0.8039	0.8151	0.8146	0.8186	**0.8192**
yeast	HL	0.2691	0.2048	0.2030	0.2039	0.2319	0.2016	**0.2011**
	AP	0.6500	0.7530	**0.7559**	0.7545	0.7098	0.7493	0.7539
	RL	0.2691	0.1811	0.1775	0.1811	0.2062	0.1789	**0.1727**
	COV	0.5457	0.4754	0.4712	0.4779	0.4787	0.4726	**0.4564**
	AUC	0.7130	0.8062	0.8092	0.8052	0.7830	0.8144	**0.8145**
image	HL	0.2095	0.2097	0.2115	**0.2093**	0.2472	0.2124	0.2127
	AP	0.7304	0.7168	0.7181	0.7169	0.5090	0.6766	**0.7384**
	RL	**0.2137**	0.2280	0.2295	0.2302	0.4662	0.2714	0.2313
	COV	**0.2232**	0.2333	0.2354	0.2358	0.4281	0.2692	0.2271
	AUC	0.7467	0.7337	0.7331	0.7318	0.5124	0.6937	**0.7514**
philosophy	HL	0.0088	0.0088	0.0091	**0.0087**	0.0094	0.0096	0.0090
	AP	0.4791	0.4836	0.4192	0.4817	0.1287	0.4058	**0.4966**
	RL	0.1460	0.1428	0.1194	0.1455	0.8935	0.1023	**0.0981**
	Cov	0.2779	0.2710	0.2215	0.2797	0.6175	**0.1943**	0.1950
	AUC	0.8359	0.8399	0.8653	0.8360	0.5200	0.8835	**0.8869**
delicious	HL	**0.0181**	**0.0181**	0.0186	**0.0181**	0.0186	0.0183	0.0182
	AP	0.3723	**0.3724**	0.3409	0.3723	0.1120	0.3661	**0.3740**
	RL	0.1361	0.1348	0.1300	0.1344	0.9082	0.1288	**0.1222**
	Cov	0.6547	0.6507	**0.5061**	0.6470	0.9301	0.5409	0.6084
	AUC	0.8623	0.8636	0.8902	0.8640	0.5138	**0.8903**	0.8766

Table 3. Performance results for competing algorithm when missing label rate is 0.5.

Datasets	Metric	MLLCRS	LSLC	GLOCAL	LSML	CIMML	DM2L	FWMML
cal500	HL	0.1440	**0.1371**	**0.1371**	0.1374	0.1418	0.1374	0.1373
	AP	0.4949	0.4989	**0.5003**	0.4987	0.2140	0.4973	0.4994
	RL	0.1860	0.1811	**0.1798**	0.1816	0.8963	0.1815	0.1808
	Cov	0.7767	0.7542	**0.7431**	0.7546	0.8721	0.7492	0.7492
	AUC	0.8104	0.8153	0.8146	0.8152	0.6697	0.8150	**0.8157**
chemistry	HL	0.0122	0.0123	**0.0118**	0.0123	0.0121	0.0120	0.0120
	AP	0.3181	0.3289	0.3075	**0.3348**	0.2356	0.3124	0.2812
	RL	0.2361	0.2186	0.2331	0.2176	0.1766	**0.1453**	0.1732
	COV	0.3728	0.3526	0.3742	0.3487	0.2791	0.2788	**0.2772**
	AUC	0.7532	0.7691	0.7569	0.7707	0.8147	0.8164	**0.8178**
yeast	HL	0.2765	0.2089	0.2079	0.2099	0.2319	**0.2033**	0.2046
	AP	0.6323	0.7426	0.7460	0.7425	0.7087	0.7455	**0.7471**
	RL	0.2923	0.1904	0.1858	0.1887	0.2067	0.1794	**0.1781**
	COV	0.5869	0.4903	0.4835	0.4901	0.4790	0.4771	**0.4636**
	AUC	0.6890	0.7962	0.8015	0.7982	0.7825	**0.8111**	0.8095
image	HL	0.2082	0.2039	**0.1999**	0.2040	0.2472	0.2269	0.2176
	AP	0.7541	0.7559	0.7549	**0.7561**	0.5120	0.6636	0.7327
	RL	0.2025	**0.1965**	0.1967	0.1968	0.4625	0.2754	0.2238
	COV	0.2161	**0.2104**	0.2112	0.2111	0.4200	0.2711	0.2229
	AUC	0.7553	**0.7634**	0.7620	0.7615	0.5221	0.6894	0.7574
philosophy	HL	**0.0088**	**0.0088**	0.0091	0.0089	0.0095	0.0097	0.0090
	AP	0.4482	0.4490	0.4247	0.4508	0.1039	0.3611	**0.4880**
	RL	0.1677	0.1690	0.1197	0.1671	0.9192	0.1101	**0.1095**
	Cov	0.3101	0.3092	0.2206	0.3075	0.6264	**0.2058**	0.2163
	AUC	0.8120	0.8120	0.8658	0.8135	0.5092	0.8739	**0.8740**
delicious	HL	**0.0181**	**0.0181**	0.0186	**0.0181**	0.0188	0.0185	0.0182
	AP	0.3650	0.3652	0.3408	0.3651	0.0850	0.3543	**0.3683**
	RL	0.1489	0.1471	0.1103	0.1474	0.9409	**0.1085**	0.1336
	Cov	0.6864	0.6830	**0.5070**	0.6814	0.9344	0.5330	0.6481
	AUC	0.8494	0.8512	0.8900	0.8511	0.4943	**0.8908**	0.8652

Table 4. Performance results for competing algorithm when missing label rate is 0.7.

Datasets	Metric	MLLCRS	LSLC	GLOCAL	LSML	CIMML	DM2L	FWMML
cal500	HL	0.1473	0.1371	0.1378	0.1372	0.1460	**0.1370**	0.1372
	AP	0.4908	0.4972	0.4989	**0.4991**	0.1545	0.4965	0.4986
	RL	0.1891	0.1840	0.1821	0.1822	0.9667	0.1823	**0.1820**
	Cov	0.7825	0.7632	0.7603	0.7604	0.8722	0.7608	**0.7602**
	AUC	0.8073	0.8126	**0.8150**	0.8143	0.6452	0.8142	0.8135
chemistry	HL	0.0226	0.0227	0.0124	0.0231	0.0121	0.0121	**0.0120**
	AP	0.2272	0.2514	0.2550	0.2528	0.2356	0.3055	**0.3233**
	RL	0.2987	0.2717	0.2778	0.2710	0.1764	0.1484	**0.1447**
	COV	0.4471	0.4131	0.4273	0.4126	0.2786	0.2898	**0.2451**
	AUC	0.6914	0.7178	0.7101	0.7190	0.8149	0.8109	**0.8462**
yeast	HL	0.3043	0.2253	0.2172	0.2264	0.3008	0.2062	**0.2062**
	AP	0.5583	0.7232	0.7315	0.7210	0.7097	0.7389	**0.7447**
	RL	0.3885	0.2073	0.1978	0.2067	0.2063	0.2099	**0.1798**
	COV	0.6832	0.5147	0.5024	0.5163	0.4789	0.4899	**0.4697**
	AUC	0.5936	0.7792	0.7890	0.7799	0.7828	0.8046	**0.8079**
image	HL	0.2071	**0.2059**	0.2111	0.2136	0.2472	0.2401	0.2182
	AP	**0.7445**	0.7211	0.7096	0.7103	0.5125	0.6331	0.7197
	RL	**0.2165**	0.2340	0.2465	0.2448	0.4723	0.3575	0.2515
	COV	0.2308	0.2391	0.2483	0.2470	0.4248	0.3370	**0.2308**
	AUC	**0.7397**	0.7275	0.7153	0.7179	0.5121	0.6136	0.7198
philosophy	HL	0.0098	0.0096	0.0093	0.0099	0.0098	0.0097	**0.0091**
	AP	0.3764	0.3864	0.4197	0.3820	0.0410	0.3516	**0.4622**
	RL	0.2111	0.2081	0.1233	0.2013	0.9830	**0.1124**	0.1218
	Cov	0.3655	0.3584	0.2296	0.3520	0.6458	**0.2104**	0.2357
	AUC	0.7678	0.7710	0.8612	0.7776	0.4855	**0.8717**	0.8603
delicious	HL	0.0182	0.0182	0.0186	0.0183	0.0193	0.0188	**0.0182**
	AP	0.3411	0.3413	0.3388	0.3417	0.0444	0.3265	**0.3521**
	RL	0.1796	0.1769	0.1612	0.1765	0.9850	0.1629	**0.1593**
	Cov	0.7431	0.7385	**0.5125**	0.7370	0.9401	0.5419	0.7065
	AUC	0.8182	0.8213	**0.8893**	0.8218	0.4720	0.8870	0.8391

Table 5. Summary of Friedman statistic F_F. Compared algorithms $k = 7$. Number of datasets, $N = 6 \times 3$. $q_\alpha = 2.949$

Metric	F_F	Critical Value ($\alpha = 0.05$)
Hamming Loss	2.2120	2.6987
Average Precision	17.7312	
Ranking Loss	11.6875	
Coverage	10.9391	
AUC	15.6988	

Table 6. Metric-wise average rank of participating algorithms.

	MLLCRS	LSLC	GLOCAL	LSML	CIMML	DM2L	FWMML
HL	4.97	**3.08**	3.69	3.86	4.86	4.25	3.28
AP	4.56	3.03	3.58	3.00	6.78	4.89	**2.17**
RL	5.56	4.00	3.17	3.89	6.06	3.17	**2.17**
Cov	5.56	4.33	3.22	4.33	5.67	2.67	**2.22**
AUC	5.69	4.14	3.39	3.94	6.11	2.78	**1.94**

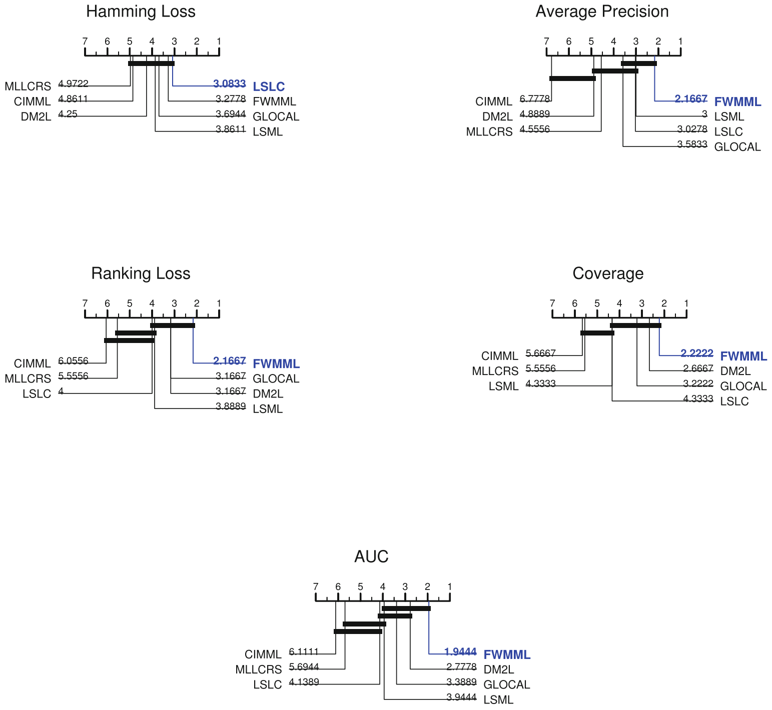


Fig. 2. FWMML comparison with competing algorithms using Nemenyi test, CD=2.123.

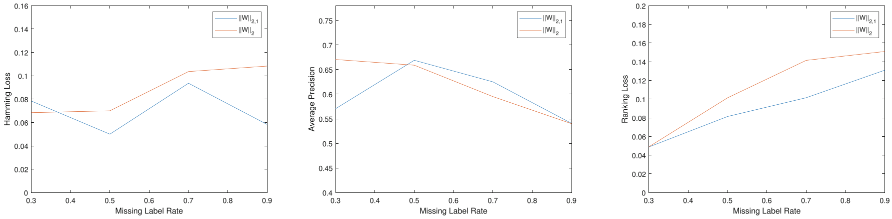


Fig. 3. Result comparison for $||W||_{2,1}$ vs $||W||_2$ norm using equation (3).

5 Conclusion

In multi-label learning tasks, labels are linked to a subset of features, with varying importance. Unlike other approaches that rely on auxiliary label correlations for missing labels, our method uses squared Hinge loss and squared $l_{2,1}$-norm regularization ensuring good classification characteristics, sparsity, and feature relevance-based scaling. We also decompose model coefficients into label correlations. This unified approach handles missing labels and predicts new instances, validated by experiments across six datasets.

References

1. Ameer, I., Sidorov, G., Gomez-Adorno, H., Nawab, R.M.A.: Multi-label emotion classification on code-mixed text: Data and methods. IEEE Access **10**, 8779–8789 (2022)
2. Chen, X., Yuan, G., Nie, F., Huang, J.Z.: Semi-supervised feature selection via rescaled linear regression. In: IJCAI. vol. 2017, pp. 1525–1531 (2017)
3. Cheng, Z., Zeng, Z.: Joint label-specific features and label correlation for multi-label learning with missing label. Appl. Intell. **50**(11), 4029–4049 (2020)
4. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. The Journal of Machine Learning Research **7**, 1–30 (2006)
5. Fan, Y., Liu, J., Tang, J., Liu, P., Lin, Y., Du, Y.: Learning correlation information for multi-label feature selection. Pattern Recogn. **145**, 109899 (2024)
6. Gardner, D., Nichols, D.: Multi-label classification of satellite images with deep learning (2017)
7. Goldberg, A., Recht, B., Xu, J., Nowak, R., Zhu, J.: Transduction with matrix completion: Three birds with one stone. Advances in neural information processing systems **23** (2010)
8. Huang, J., Qin, F., Zheng, X., Cheng, Z., Yuan, Z., Zhang, W., Huang, Q.: Improving multi-label classification with missing labels by learning label-specific features. Inf. Sci. **492**, 124–146 (2019)
9. Huang, J., Xu, Q., Qu, X., Lin, Y., Zheng, X.: Improving multi-label learning by correlation embedding. Appl. Sci. **11**(24), 12145 (2021)
10. Kumar, S., Ahmadi, N., Rastogi, R.: Multi-label learning with missing labels using sparse global structure for label-specific features. Applied Intelligence pp. 1–16 (1–16)
11. Kumar, S., Rastogi, R.: Low rank label subspace transformation for multi-label learning with missing labels. Inf. Sci. **596**, 53–72 (2022)
12. Kumar, S., Rastogi, R.: Auxiliary label embedding for multi-label learning with missing labels. In: Computer Vision and Machine Intelligence: Proceedings of CVMI 2022, pp. 525–537. Springer (2023)
13. Ma, J., Chow, T.W.: Label-specific feature selection and two-level label recovery for multi-label classification with missing labels. Neural Netw. **118**, 110–126 (2019)
14. Ma, Z., Chen, S.: Expand globally, shrink locally: Discriminant multi-label learning with missing labels. Pattern Recogn. **111**, 107675 (2021)
15. Rastogi, R., Kumar, S.: Discriminatory label-specific weights for multi-label learning with missing labels. Neural Processing Letters (2022)
16. Rastogi, R., Mortaza, S.: Multi-label classification with missing labels using label correlation and robust structural learning. Knowl.-Based Syst. **229**, 107336 (2021)

17. Tan, A., Ji, X., Liang, J., Tao, Y., Wu, W.Z., Pedrycz, W.: Weak multi-label learning with missing labels via instance granular discrimination. Inf. Sci. **594**, 200–216 (2022)
18. Tsoumakas, G., Spyromitros-Xioufis, E., Vilcek, J., Vlahavas, I.: Mulan: A java library for multi-label learning. The Journal of Machine Learning Research **12**, 2411–2414 (2011)
19. Wu, B., Liu, Z., Wang, S., Hu, B.G., Ji, Q.: Multi-label learning with missing labels. In: 2014 22nd International Conference on Pattern Recognition. pp. 1964–1968. IEEE (2014)
20. Wu, B., Lyu, S., Ghanem, B.: Ml-mg: Multi-label learning with missing labels using a mixed graph. In: Proceedings of the IEEE international conference on computer vision. pp. 4157–4165 (2015)
21. Wu, G., Tian, Y., Liu, D.: Cost-sensitive multi-label learning with positive and negative label pairwise correlations. Neural Netw. **108**, 411–423 (2018)
22. Wu, G., Tian, Y., Zhang, C.: A unified framework implementing linear binary relevance for multi-label learning. Neurocomputing **289**, 86–100 (2018)
23. Zhang, C., Li, Z.: Multi-label learning with label-specific features via weighting and label entropy guided clustering ensemble. Neurocomputing **419**, 59–69 (2021)
24. Zhu, Y., Kwok, J.T., Zhou, Z.H.: Multi-label learning with global and local label correlation. IEEE Trans. Knowl. Data Eng. **30**(6), 1081–1094 (2017)

Hypergraph Regularized Semi-supervised Least Squares Twin Support Vector Machine for Multilabel Classification

Reshma Rastogi(✉) and Dev Nirwal

Machine Learning and Statistical Inference (MLSI) Lab, Department of Computer Science South Asian University, Delhi, India
reshma.khemchandani@sau.ac.in, devnirwal@students.sau.ac.in

Abstract. In a multi-label learning problem, each instance is associated with multiple labels simultaneously. However, problem becomes more complicated when labels are missing. Many multi-label based real-life applications, such as medical diagnosis, protein function prediction, image annotations are framed as networks which are used to model interactions between complex entities. Although, in many scenarios these interactions may not be pairwise, rather should described as higher-order interactions. For such a scenario Hypergraphs are preferred rather than simple Laplacian. Multilabel twin support vector machines (MLTSVM) has become popular due its performance for multilabel classification. In this paper, to deal with missing label scenario, we propose a semi-supervised framework termed as Hypergraph Least Squares Twin Support Vector Machine for Multi-label Learning (HMLLSTSVM) wherein we have used Hypergraph Laplacian to train our classifier utilizing both labeled and unlabeled samples. We incorporate the idea of Hypergraph along with least squares loss function into MLTSVM, which improves the efficacy in terms of classification accuracy and speed of our proposed model. Taking motivation from KNN-based Least Squares Twin Support Vector Machine (KNNLSTSVM), we have incorporated the intrinsic similarity information among the samples in our proposed model's objective function, which makes our classifier HMLLSTSVM less sensitive to outliers. Experimental results on benchmark multilabel datasets proves the efficacy of the classifier.

Keywords: Semi-supervised Learning · Hypergraph · Laplacian matrix · Multilabel Learning

1 Introduction

Researchers tried to explore Twin Support Vector Machine (TSVM) [9] in different research directions. Authors in ([22]) proposed the Structural Least Squares Twin Support Vector Machine (SLSTSVM), which is an amalgamation of the Structural Twin Support Vector Machine (STSVM) [17] and the Least Squares

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 223–237, 2025.
https://doi.org/10.1007/978-3-031-78383-8_15

Twin Support Vector Machine (LSTSVM) [12]. SLSTSVM ([22]) has the upper hand over other algorithms as it takes advantage of the structural information of the data, which boosts the generalization capability of the model. In addition to this, it also owns less computational complexity as LSTSVM based model finds resulting hyperplanes by solving two systems of linear equations rather than one large quadratic programming problem (QPP) or two QPPs. Researchers across the globe utilized the variants of Twin Support Vector Machine in the domain of Human Activity Recognition, such as Robust Least Squares Twin Support Vector Machine ([10]) and Robust Parametric Twin Support Vector Machine ([11]).

It may be hard to obtain labeled data because it requires subject matter experts and many experiments. Semi-Supervised Learning (SSL) utilizes both labeled and unlabeled data simultaneously to train a classifier. A pairwise association between the samples is assumed in the graph-regularized manifold learning algorithm LapSVM, which however doesn't seem to hold in practice. A simple graph can't express multivariate and higher-order relationships between the samples. We use a Hypergraph rather than a straightforward Graph Laplacian to address this problem. The use cases of such algorithms exist when dealing with large volumes of unlabeled data.

We summarize our contributions to this paper in the following manner:

- We propose improve multi-label semi-supervised hypergraph based prediction model termed as HMLLSTSVM.
- We have also shown the efficacy of the proposed algorithm over existing benchmark datasets.

2 Related Work

In the past decade, TSVM ([9]) has gained popularity among many machine learning researchers. Unlike Support Vector Machines (SVM) ([5]), TSVM generates two non-parallel hyperplanes in such a manner that each hyperplane is in closer proximity to one of the two classes and is far away from the other. Instead of solving a single large-sized Quadratic Programming Problem (QPP) as in SVM, a pair of smaller sized QPP's has been solved in TSVM, making the computational speed of TSVM approximately four times faster than the classic SVM. From the perspective of upgrading the training speed, sparsity, and generalization ability of TSVM, TSVM is still a hot topic of research.

TSVM was extended to least squares TSVM ([12]), in which the solution of the modified primal problem follows directly from solving two systems of linear equations as against solving two QPPs and two systems of linear equations in TSVM. Hence, the Least Squares Twin Support Vector Machine outshines TSVM in terms of training time complexity.

Researchers further enhanced Least squares TSVM to KNN-based Least Squares Twin Support Vector Machine (KNNLSTSVM) ([15]) discovering the intrinsic similarity information among the samples that may be beneficial for improving classification performance. KNNLSTSVM not only preserves the advantage of Least Squares TSVM which is a fast and straightforward algorithm. It also incorporates the similarity information among the samples in the objective function of Least Squares TSVM to improve the generalization capability of the model. Therefore, KNNLSTSVM is less sensitive to outliers in comparison to the Least Squares TSVM.

A beneficial progress of TSVM is MLTSVM for solving multi-label classification problems. The MLTSVM algorithm [3] captures the multi-label information ingrained in data via multiple non-parallel hyperplanes. In the training phase, one hyperplane is learned for each label with the help of binary relevance strategy. To enhance the generalization ability of the model in multi-label learning, intrinsic similarity information among the samples has been adopted by the researchers. Hence, researchers [4] proposed KNN-based Multi-Label Twin Support Vector Machine (KNNMLTSVM).

Recently, Structural Least Squares Twin Support Vector Machine for Multi-label Learning (ML-SLSTSVM) [1] model was introduced, which incorporated the cluster-based structural information of a particular class in the optimization problem of Least Squares Multi-Label Twin Support Vector Machine. But ML-SLSTSVM model considered only the partial information among the samples i.e they considered variance co-variance relationship of features. Although structural information also considers the relationship among inter and intra class samples. Some other recent approaches of multilabel learning are discussed by authors in [13], [18],[21],[14],[20]

The concept of the TSVM was extended in a semi-supervised setting by [16], termed as Laplacian-Twin Support Vector Machine. The researchers used graph Laplacian to exploit labeled and unlabeled information and solved pair of quadratic programming problem to build a nonparallel plane classifier. Further, [2] proposed a least squares version of the LapTSVM, called the Laplacian-Least Squares Twin Support Vector Machine (LapLSTSVM) model to lessen the computational cost of LapTSVM.

3 Proposed Model: Hypergraph based Least Squares Twin Support Vector Machine (HMLLSTSVM) for Multi Label Learning

Problem of semi-supervised binary classification considers a set $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_l, y_l), x_{l+1}, \ldots, x_n\}$, where $X_l = \{x_i : i = 1, 2, \ldots, l\}$ symbolizes the l labeled data points in n dimension corresponding to class labels $Y_l = \{y_i \in [-1, 1] : i = 1, 2, ..., l\}$ and $X_u = \{x_i : i = l{+}1, l{+}2, ..., m\}$ symbolizes the unlabeled data. Hence, $X = X_l \cup X_u$. Based on the structural information of data and label dimensionality, the proposed model finds l hyperplanes.

3.1 Local geometric information

To extract local geometric information, k-nearest neighbor graph is a mostly used technique ([7]). Here, intra-class and inter-class graphs with weights $W_{wc,ij}$

(W_{wc} represents weights within the class), and $W_{bc,il}$ (W_{bc} represents weights between the classes) respectively are considered for each class r, $r = 1, 2, \ldots, l$, where x_i and x_j is associated to class r, while x_l does not belong to class r.

$$W_{wc,ij} = \begin{cases} 1, \text{ if } x_i \text{ is k-nearest neighbors of } x_j \\ \text{or } x_j \text{ is k-nearest neighbors of } x_i \\ 0, \text{ otherwise} \end{cases}$$

$$W_{bc,il} = \begin{cases} 1, \text{ if } x_l \text{ is k-nearest neighbors of } x_i \\ \text{or } x_i \text{ is k-nearest neighbors of } x_l \\ 0, \text{ otherwise} \end{cases}$$

The inter-class weighting matrix W_{bc} is redefined with f_{rl} as:

$$f_{rl} = \begin{cases} 1, \text{ if } \exists\, i,\ W_{bc,il} \neq 0 \\ 0, \text{ otherwise} \end{cases} \tag{1}$$

3.2 Structural information of data

Using Ward's linkage clustering method [7], structural information of the samples is extracted. The mean and covariance matrix of each sub-cluster n_r of class r is given by:

$$u_{C_i} = \tfrac{1}{|C_i|} \sum_{x_j \in C_i} x_j \tag{2}$$

$$\Sigma_{C_i} = \frac{1}{|C_i|} \sum_{x_j \in C_i} (x_j - u_{C_i})(x_j - u_{C_i})^T \tag{3}$$

here C_i, $i = 1, 2, ...n_r$ represents the cluster of samples with label r, n_r indicates the number of clusters, and u_{C_i} and Σ_{C_i} are the mean vector and covariance matrix of a particular cluster C_i. Further, we calculate $\Sigma_r = \Sigma_{C_1} + \Sigma_{C_2} + ... + \Sigma_{C_{n_r}}$, which is the total covariance matrix corresponding to the cluster of samples with label r. This covariance matrix Σ_r comprises of the structural information of class r.

3.3 Hypergraph construction

Given a graph $G = (V, E)$, $V = \{v_1, v_2, ..., v_n\}$ and $E = \{e_1, e_2, ..., e_n\}$ are the vertex and hyper-edges set. We can define a vertex-edge matrix H such that

$$H = h(v, e) = \begin{cases} 1, & v \in e \\ 0, & v \notin e \end{cases} \tag{4}$$

The degree of hyperedge $\delta(e)$ is defined as the number of vertices a hyperedge contains.

$$\delta(e) = \sum_{v \in e} h(v, e)$$

$$d(v) = \sum_{v \in e, e \in E} w(e) = \sum_{e \in E} w(e) h(v, e)$$

The following formula can define the hyperedge weight w(e):

$$w(e) = \frac{1}{\delta(e)(\delta(e)-1)} \sum_{\{v_i, v_j\} \in e} exp \left(- \frac{||x_i - x_j||^2}{\mu} \right)$$

Similar to a simple graph, the Hypergraph's Laplacian matrix can be defined as [24]:

$$HL_a = D_v - HWD_e^{-1}H^T \tag{5}$$

where D_v, D_e, W are the diagonal matrices composed of $d(v), \delta(e)$ and $w(e)$ respectively.

According to Zhou's ([24]), the Laplacian regularized Hypergraph matrix is:

$$HL_a = I - D_v^{-\frac{1}{2}} HWD_e^{-1}H^T D_v^{-\frac{1}{2}} \tag{6}$$

3.4 Formulation of the proposed model HMLLSTSVM:

Working on the lines of LapLSTSVM, we propose a model for multi-label learning which considers geometrical and structural information along with some percentage of known label set of samples. Further, the proposed model is in the spirit of least squares TSVM, which makes it time efficient as well.

$$\begin{aligned} \min_{(w_r, b_r, \xi_j)} \ & \tfrac{1}{2} \sum_{i \in I_r} d_{ir} \left({w_r}^T x_i + b_r \right)^2 + \tfrac{c_{1r}}{2} \textstyle\sum_{j \in I'r} ||\xi_j||^2 + \tfrac{c_{2r}}{2} \sum_{j \in I'_r} {w_r}^T \Sigma_r w_r \\ & + \tfrac{\lambda_r}{2} \left(||w_r||_2 + {b_r}^2 \right) + \tfrac{1 - \lambda_r}{2} \left(X w_r + e b_r \right)^T HL_a \left(X w_r + e b_r \right) \end{aligned}$$

subject to,

$$-f_{rj} \left({w_r}^T x_j + b_r \right) + \xi_j = f_{rj}, \quad j \in I'_r \tag{7}$$

where $\xi = (\xi_1, \xi_2, \ldots, \xi_s)$, s is number of samples in I'_r, $d_{ir} = \sum_{j \in I_r} W_{wc,ij}$, is an element of the intra-class weighting matrix for the samples in the r^{th} label set. The parameters w_r and b_r defines the hyperplane of the r^{th} label , c_{1r} and c_{2r} are the penalty parameters and λ_r is the regularization parameter and Σ_r denotes the covariance matrix of the samples belonging to the r^{th} label. Here, f_{rj} reflects rj^{th} entry of inter-class weighting matrix, λ_1 and λ_2 are the regularization parameters and also, λ_1,λ_2 $\in$ $(0, 1]$. HL_a is the corresponding Hypergraph Laplacian matrix.

After substituting value of ξ_j from the equality constraint (7) in the objective function of the aforementioned problem we obtain the following unconstrained optimization problem:

$$L = \frac{1}{2}\sum_{i\in I_r} d_{ir}||w_r^T x_i + b_r||^2 + \frac{c_{1r}}{2}\sum_{j\in I'r} ||f_{rj}\left(1 + w_r^T x_j + b_r\right)||^2 + \frac{c_{2r}}{2} w_r^T \Sigma_r w_r$$
$$+\frac{\lambda_r}{2}\left(||w_r||^2 + b_r^2\right) + \frac{1-\lambda_r}{2}\left(Xw_r + eb_r\right)^T HL_a\left(Xw_r + eb_r\right) \quad (8)$$

After computing the derivatives of the aforementioned Lagrangian w.r.t. w_r and b_r, and equating them equal to zero, we obtain the following equations:

$$\frac{\partial L}{\partial w_r} = 0 \Rightarrow \sum_{i\in I_r} d_{ir} x_i\left(w_r^T x_i + b_r\right) + c_{1r}\sum_{j\in I'r} f_{rj} x_j\left(1 + w_r^T x_j + b_r\right)$$
$$+c_{2r}\Sigma_r w_r + \lambda_r w_r + \left(1-\lambda_r\right) X^T HL_a\left(Xw_r + eb_r\right) = 0. \quad (9)$$

$$\frac{\partial L}{\partial b_r} = 0 \Rightarrow \sum_{i\in I_r} d_{ir}\left(w_r^T x_i + b_r\right) + c_{1r}\sum_{j\in I'r} f_{rj}\left(1 + w_r^T x_j + b_r\right) + \lambda_r b_r$$
$$+\left(1-\lambda_r\right) X^T HL_a\left(Xw_r + eb_r\right) = 0. \quad (10)$$

After combining (9) and (10), we obtain:

$$D_r\sum_{i\in I_r}\begin{bmatrix} x_i \\ 1\end{bmatrix}\begin{bmatrix} x_i & 1\end{bmatrix}\begin{bmatrix} w_r \\ b_r\end{bmatrix} + c_{1r}F_r\sum_{j\in I'_r}\begin{bmatrix} x_j \\ 1\end{bmatrix}\begin{bmatrix} x_j & 1\end{bmatrix}\begin{bmatrix} w_r \\ b_r\end{bmatrix} + c_{1r}F_r\sum_{j\in I'_r}\begin{bmatrix} x_j \\ 1\end{bmatrix}$$
$$+c_{1r}F_r\sum_{j\in I'_r}\begin{bmatrix} x_j \\ 1\end{bmatrix} + \lambda_r\begin{bmatrix} w_r \\ b_r\end{bmatrix} + c_{2r}\begin{bmatrix} \Sigma_r & 0 \\ 0 & 0\end{bmatrix}\begin{bmatrix} w_r \\ b_r\end{bmatrix} + (1-\lambda_r)H_r HL_a H_r^T\begin{bmatrix} w_r \\ b_r\end{bmatrix} = 0$$

where, $D_r = \text{diag}(d_{1r}, \ldots, d_{N_r})$, N=$|I_r|$ and $F_r = \text{diag}(f_{r1}, f_{r2}, \ldots, f_{rM})$, M $=|I'_r|$, $\theta_r = [w_r \quad b_r]^T$.

$$D_r\sum_{i\in I_r} z_i {z_i}^T\theta_r + c_{1r}F_r\sum_{j\in I'r} z_j {z_j}^T\theta_r + c_{1r}F_r\sum_{j\in I'r} z_j +$$
$$\lambda_r\theta_r + c_{2r}J_r\theta_r + (1-\lambda_r)H_r HL_a H_r^T\theta_r = 0 \quad (11)$$

On rearranging the terms in equation (11), we get

$$\left(D_r\sum_{i\in I_r} z_i {z_i}^T + c_{1r}F_r\sum_{j\in I'r} z_j {z_j}^T + \lambda_r I + c_{2r}J_r + (1-\lambda_r)H_r HL_a H_r^T\right)\theta_r$$
$$= c_{1r}F_r\sum_{j\in I'r} z_j \quad (12)$$

Hence, after deriving the dual by applying Kuhn Tucker conditions, we determine the equation of r^{th} hyperplane as :

$$\begin{aligned}\theta_r = c_{1r}({H_r}^T D_r H_r + c_{1r} {G_r}^T F_r G_r + \lambda_r I + c_{2r} J_r + \\ (1-\lambda_r) H_r^T HL_a H_r)^{-1} {G_r}^T F_r e\end{aligned} \tag{13}$$

where e is vector of ones of appropriate dimension $[w_r \quad b_r]^T$ gives the rth hyperplane, H_r= $[X_{Ir}, e_{1r}]$, $G_r = [X_{I'r}, e_{2r}]$, e_{1r} and e_{2r} are the vector of ones of suitable dimensions, $D_r = \text{diag}(d_{1r}, d_{2r}, \ldots, d_{Nr})$, N=$|I_r|$. $J_r = \begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix}$. c_{1r} is the pre-specified penalty parameter. $F_r = \text{diag}(f_{r1}, f_{r2}, \ldots, f_{rM})$, $M = |I_r'|$.

It can be seen from our formulation of the proposed model that when the regularization parameter $\lambda_r = 1$, KNN-Based Multi-Label Least Squares Twin Support Vector Machine (KNNMLLSTSVM) is the limiting case of HMLLSTSVM. Also, unlike Multi-Label Structural Least Squares Twin Support Vector Machine ([1]), our proposed model HMLLSTSVM considers both local geometric information and structural information among the samples. Hence, HMLLSTSVM is less sensitive to outliers.

3.5 Algorithm

Inputs: A training dataset $\mathcal{D}$ and a testing dataset $\mathcal{D}^*$=$\{(X_i, Y_i)$, where $\{i = 1, 2, \ldots, n\}$, $Y_i \in \{1, -1\}$ for multi-label classification.
Output: A relevant set of labels for each test sample.

1. **For** each label Y_r in the label matrix Y, repeat steps 2 to 5
2. Create a set of instances for each column Y_r.
3. Extract local geometric and structural information from the input samples for each column Y_r using equations (1), (1), (1), (2).
4. Compute the Hypergraph Laplacian matrix HL_a using $HL_a = I - D_v^{-\frac{1}{2}} HW D_e^{-1} H^T D_v^{-\frac{1}{2}}$, where D_v is the degree matrix and W is the weighted adjacency matrix.
5. Utilizing the structural and geometric information ingrained in the data, determine the hyperplane for every label.
6. **For** each test sample t_i, repeat steps 7 and 8
7. Calculate the distances from the hyperplane of each label using the equation

$$d_r = \frac{|w_r^T t_i + b_r|}{||w_r||_2}, r = 1, 2, \ldots, L$$

8. Find a relevant set of labels for a test point t_i on the basis of appropriate threshold value, which is $\delta = min_{r=1,\ldots,L} \frac{1}{||w_r||}$. For each label Y_r, if the distance between the test point t_i and the hyperplane is less than δ, assign the test point t_i to the r^{th} label.

3.6 Non-Linear version of HMLLSTSVM

In real life classification problems, linear classifiers do not work very well. Therefore, we extend the linear HMLLSTSVM to its non-linear version. We can determine non-linear MLLSTSVM by using kernel generated hyperlanes. The kernel takes its input vectors in the original space, and gives the dot product of vectors in the feature space. Formally, let we have data $X_1, X_2 \in X$, and a map $\phi : X \to \mathbb{R}^N$ then $K(X_1, X_2) =< \phi(X_1), \phi(X_2) >$ is a kernel function, $<, >$ is a inner product and u_r and b_r define the r^{th} hyper-surface parameter. Thus, the optimization problem of non-linear HMLLSTSVM is as follows:

$$\begin{aligned} \min_{(u_r, b_r, \xi_j)} \quad & \tfrac{1}{2}\textstyle\sum_{i\in I_r} d_{ir}\left({u_r}^T < \phi(X), \phi(x_i) > +b_r\right)^2 + \\ & c_{1r}\textstyle\sum_{j\in I'r} \|\xi_j\|^2 + \tfrac{1}{2}\lambda_r\left(\|u_r\|_2 + {b_r}^2\right) + \tfrac{c_{2r}}{2}\displaystyle\sum_{j\in I'_r} {u_r}^T \Sigma_r^\phi u_r \\ & + \tfrac{1-\lambda_r}{2}\left(Xw_r + eb_r\right)^T HL_a \left(Xu_r + eb_r\right) \end{aligned}$$

subject to,

$$-f_{rj}\left({u_r}^T x_j + b_r\right) + \xi_j = f_{rj}, \quad j \in I'_r. \tag{14}$$

where $\xi = (\xi_1, \xi_2, \ldots, \xi_s)$, s is number of samples in I'_r, Σ^ϕ is the covariance matrix obtained by the kernel Ward's linkage method in the kernel space of samples. After solving the optimizing problem, the parameters u_r and b_r are obtained as shown in the following formula:

$$\begin{aligned} \left[{u_r}^T b_r\right]^T = c_{1r}({H_r}^{\phi T} D_r {H_r}^\phi + c_{1r} {G_r}^{\phi T} F_r G_r^\phi + \lambda_r I + c_{2r} {J_r}^\phi + \\ (1-\lambda_r){H_r}^{\phi T} HL_a {H_r}^\phi)^{-1} {G_r}^\phi F_r e \end{aligned} \tag{15}$$

where $H_r^\phi = [< \phi(X_{Ir}), \phi(X) >, e_{1r}]$, $G_r^\phi = [< \phi(X_{I'r}), \phi(X) >, e_{2r}]$. $J_r^\phi = \begin{bmatrix} \Sigma_r^\phi & 0 \\ 0 & 0 \end{bmatrix}$, c_r is the penalty factor, and F_r and e_{2r} and e_{1r} are similar to the linear case.

4 Experiments

The experiments are performed using five-fold cross validation in MATLAB version 2020a under Microsoft Windows environment on a machine with 3.40 GHz CPU and 16 GB RAM on six well known multi-label datasets, which are taken from the site http://mulan.sourceforge.net/datasets-mlc.html.

Since, our proposed model is based on multi-label twin support vector machine. So we made a comparison among the algorithms , which are based on multi-label twin support vector machine.

4.1 Compared Algorithms

1. **Multi-label Twin Support Vector Machine (MLTSVM)** ([3]): MLTSVM is a extension of TSVM in multilabel domain. It discovers multiple non-parallel hyperplanes to procure multi-label information embedded in the data.
2. **KNN-based Multi-Label Twin Support Vector Machine (KNN-MLTSVM)** ([15]): KNNMLTSVM is an enhanced version of MLTSVM to tackle multi-label classification. It determines multiple non-parallel hyperplanes, which contain the structural as well as local geometric information ingrained in the data.
3. **Multi-label learning with label-specific features(LIFT)** [23] To discover label-specific features, it employs a clustering approach. As the basic learner in this case, they employed linear kernel SVM. $r = 0.1$.
4. **Learning Label Specific Features(LLSF)** [8] In order to train the multi-label classification model, it makes use of the idea of learning label-specific characteristics. The tuning for the parameter *lambda* is $2 - 10, ..., 210$.
5. **KNN-based Multi-Label Least Squares Laplacian Twin Support Vector Machine**: It is an improved version of KNNMLTSVM. It is computationally fast, as it solves a system of linear equations rather than QPPs. Thus, it considers square loss function instead of hinge loss function. we have use LAP as accronym while reporting results.
6. **Discriminatory Label-specific Weights for Multi-label Learning with Missing Labels**[19] (CIMML): Multi-label learning algorithms that handle the datasets class imbalance issue by assigning favorable weights for positive labels to compensate for their scarce occurrence.
7. **Hypergraph based Least Squares Twin Support Vector Machine (HMLLSTSVM) (Proposed)**: This our proposed semi-supervised algorithm as it learns from both labeled as well as unlabeled information using Hypergraph, we have use HYP as accronym while reporting results.

4.2 Parameter Selection

We used the RBF kernel, and the value of kernel parameter and the penalty factor is tuned for the set $\{2^{-4}, 2^{-3}, 2^{-2}, 2^{-1}, 2^{0}, 2^{1}, 2^{2}, 2^{3}, 2^{4}\}$, and $\{2^{-5}, \ldots, 2^{5}\}$ respectively. The value of k (number of neighbors) is tuned for the set $\{2, 3, 4, 5, 6, 8, 10\}$. To be specific, we set k=4, 6, 8, 2, 4, 8 for emotions, flags, yeast, and image respectively. The regularizer for variance-covariance is tuned for the set $\{2^{-5}, \ldots, 2^{5}\}$. The regularizer for Hypergraph Laplacian, which is λ is kept low, and is set to e^{-4}. Once, the optimal set of parameters is obtained, then it is used to learn the final decision function.

4.3 Datasets

In our experiments, we consider four UCI multi-label datasets shown in the Table 1 below. The datasets such as **Image** and **flags** are from the image domain.

Emotions dataset belongs to the music domain. **Yeast** dataset is associated with a biological field, where each instance represents a yeast gene, and labels denote the functional group of the corresponding yeast gene. The number of instances, the number of features, the number of class labels, and the cardinality of each dataset is observed in Table 1.

Table 1. Particulars of the Benchmark multi-label datasets

Dataset	Instances	Features	Labels	Cardinality
Emotions	593	72	6	1.87
Flags	194	19	7	3.392
Image	2000	294	5	1.24
Yeast	2417	103	14	4.24

Table 2. Results for Multi-Label classification

Dataset	Metric	MLTSVM	CIMML	MLTSVMKNN	LLSF	LIFT	LAP	HYP
Flags	HL(↓)	0.3888	0.4235	0.3991	0.3532	0.4017	0.3165	**0.3098**
	F1(↑)	0.7048	0.6546	0.7007	0.6705	0.5969	0.7369	**0.7407**
	AP(↑)	0.8385	0.6174	0.8458	0.9382	**0.9742**	0.8257	0.8090
	RL(↓)	0.5873	0.4941	0.5894	0.6028	**0.4792**	0.5934	0.6007
	AUC(↑)	0.5362	0.4979	0.5329	0.5581	**0.6018**	0.5559	0.5590
Emotions	HL(↓)	0.3454	0.2569	0.3772	**0.2175**	0.3394	0.4194	0.3384
	F1(↑)	0.6267	0.4757	0.6096	0.6218	0.4588	0.5882	**0.6277**
	AP(↑)	0.5184	0.7051	0.5563	0.7622	**0.9157**	0.6179	0.5174
	RL(↓)	0.5283	**0.27**	0.5072	0.6095	0.3227	0.4847	0.5343
	AUC(↑)	0.7097	0.7023	0.6907	**0.7179**	0.6051	0.6601	0.7105
Image	HL(↓)	0.4947	0.3525	0.5321	**0.2821**	0.2952	0.5295	0.4102
	F1(↑)	0.4814	0.3349	0.4703	0.2649	0.2659	0.4622	**0.5129**
	AP(↑)	0.7075	0.5885	0.7339	0.8690	**0.8840**	0.7627	0.6249
	RL(↓)	0.3851	0.41	0.3641	0.3767	0.2664	0.3647	**0.4247**
	AUC(↑)	0.5862	0.5547	0.5703	0.5352	0.5417	0.5697	**0.6536**
Yeast	HL(↓)	0.2485	0.2903	0.2594	0.2410	0.2484	0.2581	**0.2005**
	F1(↑)	0.666	0.5279	0.6607	0.5251	0.660	0.664	**0.6796**
	AP(↑)	0.7497	0.6446	0.7525	**0.9582**	0.9330	0.7310	0.7439
	RL(↓)	0.3882	**0.2479**	0.3842	0.388	0.222	0.3911	0.4194
	AUC(↑)	0.5986	0.5547	0.5962	0.5389	**0.6973**	0.6043	0.6068

4.4 Evaluation Metrics

Given a test dataset $\mathrm{T}_s = \{x_i, y_i\}_{i=1}^{N_t}$ where $y_i \in \{-1, 1\}^m$. Let N_t, m, y_i, $\hat{y}_i$ denote, respectively, the number of test data, the number of labels, the set of labels relevant to the i^{th} instance and the set of labels that are irrelevant to it. In addition, the function $f_y(x)$ is a real-valued function ($f : \mathbf{X} \times \mathbf{Y} \rightarrow \mathbb{R}$) that returns the confidence of being proper label of x and $\mathrm{rank}_f(x, y)$ returns the rank of y in $\mathbf{Y}$ based on the descending order induced from $f_y(x)$ and $h(\cdot)$ be the learned multi-label classifier.

We have used the following evaluation criteria to compare the performance of different algorithms

Table 3. Results for Multi-Label classification for different percentage of the labelled samples

Metric	LAP	HYP	LAP	HYP	LAP	HYP	LAP	HYP	LAP	HYP
labelled	90%	90%	80%	80%	70%	70%	60%	60%	50%	50%
Flags:										
HL(↓)	0.3055	**0.2996**	0.3047	**0.3018**	0.3040	**0.3017**	**0.3092**	0.3129	**0.3196**	0.3248
F1(↑)	0.7433	**0.7461**	0.7428	**0.7444**	0.7420	**0.7420**	**0.7382**	0.7357	0.7334	**0.7315**
AP(↑)	**0.7940**	0.7891	**0.7946**	0.7823	**0.7914**	0.7791	0.8083	**0.8088**	**0.8118**	0.7962
RL(↓)	**0.6015**	0.6042	**0.6018**	0.6039	**0.5992**	0.6062	0.5979	**0.5958**	**0.5962**	0.5981
AUC(↑)	0.5692	**0.5736**	0.5691	**0.5725**	0.5716	**0.5733**	**0.5622**	0.5606	0.5540	**0.5610**
Emotions:										
HL(↓)	0.3485	**0.2923**	0.3499	**0.3066**	0.3738	**0.3342**	0.3696	**0.3488**	0.3848	**0.3679**
F1(↑)	0.6245	**0.6542**	0.6243	**0.6446**	0.6096	**0.6297**	0.6122	**0.6184**	0.5995	**0.6084**
AP(↑)	**0.5246**	0.4590	**0.5247**	0.4749	0.5575	0.5056	**0.5516**	0.5223	**0.5818**	0.5500
RL(↓)	**0.5269**	0.5585	**0.5258**	0.5501	**0.5072**	0.5253	**0.5055**	0.5202	**0.5016**	0.5107
AUC(↑)	0.7033	**0.7398**	0.7042	**0.7326**	0.6904	**0.7174**	0.6968	**0.7076**	0.6816	**0.6977**
Image:										
HL(↓)	0.4181	**0.2865**	0.3975	**0.3296**	0.4650	**0.4201**	**0.4702**	0.4826	**0.5030**	0.5584
F1(↑)	0.5189	**0.5921**	0.5156	**0.5486**	0.4370	**0.4780**	0.4045	**0.4473**	0.3632	**0.4289**
AP(↑)	**0.6025**	0.4584	**0.5972**	0.5252	**0.7546**	0.6555	**0.8250**	0.7221	**0.9223**	0.7881
RL(↓)	**0.4117**	0.4876	**0.4172**	0.4378	**0.3391**	0.3746	**0.3386**	0.3369	**0.2960**	0.3108
AUC(↑)	0.6538	**0.7485**	0.6858	**0.7249**	0.5794	**0.6679**	0.5685	**0.6592**	0.5338	**0.5967**
Yeast:										
HL(↓)	0.2571	**0.2029**	0.2656	**0.2061**	0.2741	**0.2384**	0.2706	**0.2209**	0.2741	**0.2384**
F1(↑)	0.6636	**0.6781**	0.6580	**0.6791**	0.6509	**0.6662**	0.6533	**0.6746**	0.6509	**0.6662**
AP(↑)	0.7362	**0.7478**	0.7453	**0.7463**	**0.7698**	0.7427	**0.7606**	0.7423	**0.7698**	0.7427
RL(↓)	**0.3896**	0.4158	**0.3841**	0.4126	**0.3777**	0.3932	**0.3790**	0.4024	**0.3777**	0.3932
AUC(↑)	0.6019	**0.6043**	0.5976	**0.6054**	0.5891	**0.5995**	0.5920	**0.6030**	0.5891	**0.5995**

1. **Hamming Loss** (HL): This metric indicates the fraction of labels that are incorrectly classified to the total number of labels.

$$HL = \frac{1}{N_t \times m} \sum_{i=1}^{N_t} \sum_{j=1}^{m} [h_j(x_i) \neq y_{ij}] \tag{16}$$

2. **F1 Score (F1)**
   This measure is the harmonic mean between recall and precision as defined in Eq. (17):

$$F1 = \frac{1}{m'} \sum_{i=1}^{m'} \frac{2 \times |h(x_i) \cap Y_i|}{|h(x_i)| + |Y_i|} \tag{17}$$

3. **Average Precision** (AP): Average precision evaluates the average fraction of relevant labels ranked higher than a particular label y ∈ $\mathbf{y}_i$.

$$AP = \frac{1}{N_t}\sum_{i=1}^{N_t}\frac{1}{y_i}\sum_{y\in\mathbf{y}_i}\frac{\{(y' \in \mathbf{y}_i|rank_f(x_i,y') \leq rank_f(x_i,y)}{rank_f(x_i,y)} \tag{18}$$

4. **Coverage (AUC)** : Coverage is described as the distance to cover all possible labels assigned to a sample x. It is loosely linked to precision at the level of perfect recall. The smaller the value of coverage is, the better the performance is.

$$Coverage = \frac{1}{N_t}\sum_{i=1}^{N_t} maxrank_f(x_i,y) - 1 \tag{19}$$

5. **Ranking Loss** (RL): This criterion is utilized for ranking-based algorithms and measures the average fraction of label pairs that are reversely ordered. For example, an irrelevant label is ranked higher than a relevant label.

$$RL = \frac{1}{N_t}\sum_{i=1}^{N_t}(\frac{1}{|y_i|,|\overline{y}_i|}|\{(y'y")|f_{y'\in \mathrm{y}_i}(x_i) \leq f_{y"\in\overline{y}_i}(x_i)\}|) \tag{20}$$

4.5 Results and Discussion

We investigate the classification performances of proposed algorithm in supervised framework and compared it with aforementioned algorithms in Table 2, further we have also compared performance with Laplacian version of our algorithm under 50%, 60%, 70%, 80%, 90% labelled data in the training dataset. Figure 1 reflects the overall rank of the compared algorithm along with the proposed algorithm. From the figure we can conclude that the overall rank of the proposed algorithm is higher when compared with other comparing approaches.

In the semi-supervised setting, as shown in Table 2 and Table 3, our proposed model performs up to the mark with less percentage of labeled samples.

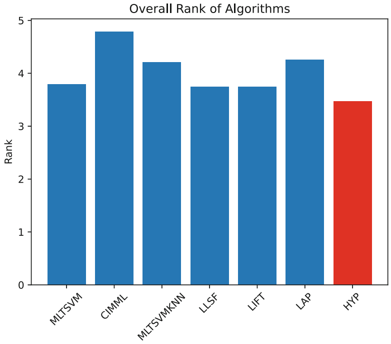
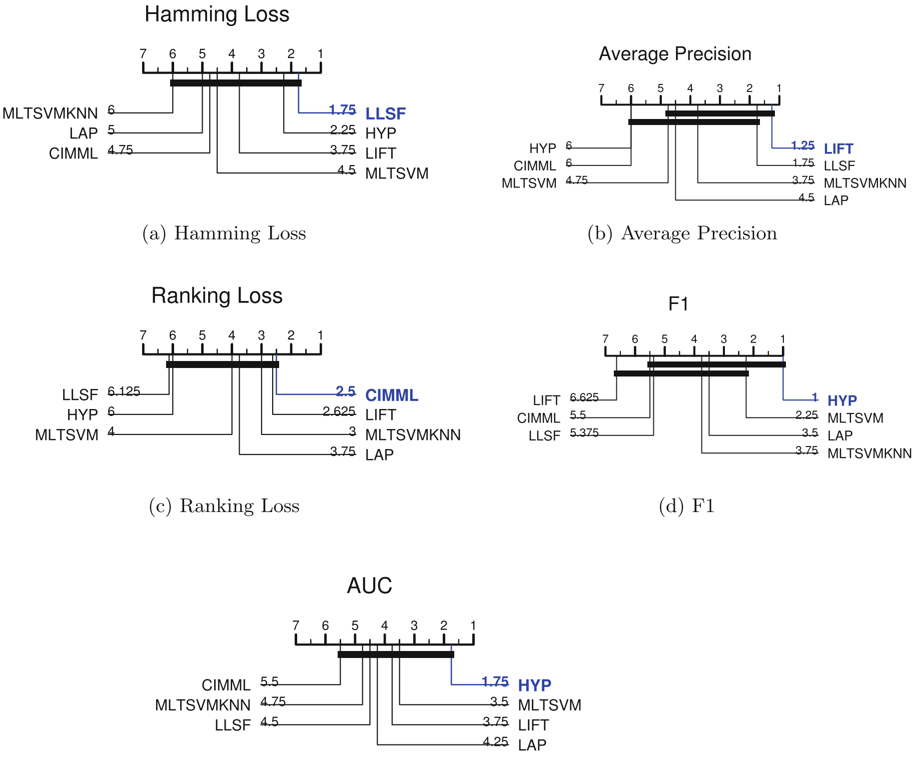


Fig. 1. Consolidated average rank of participating algorithms across all metrics.

We further analyze pairwise comparisons using the Nemenyi test [6]. For six classifiers, the critical value, $\alpha_{0.1}$ is 2.589, and the CD is 2.32. Significant differences exist between the best and worst-performing algorithms across all metrics Figure 2. Overall, HMLLSTSVM demonstrates competitive and often superior performance.



(a) Hamming Loss

(b) Average Precision

(c) Ranking Loss

(d) F1

(e) AUC

Fig. 2. HMLLSTSVM comparison with competing algorithms using Nemenyi test, CD=2.32.

5 Conclusion

This paper presents a Hypergraph Least Squares Twin Support Vector Machine (HMLLSTSVM) classifier for Multi-Label learning. Taking motivation from the KNNLSTSVM, we introduced HMLLSTSVM, wherein the weight is associated to each sample considering the distance from the neighbors, and with the help of Hypergraph, our model exploits information from both labeled and unlabeled samples incorporating higher order relationship present in dataset. Further, solving system of linear equations solution gives an edge above QPPs as solved in

TSVM. Thus, instead of the hinge loss function, the square loss function is considered in multi-label learning which also addresses the issue of outlier sensitivity and noise tolerance. Moreover, we have incorporated the intrinsic similarity information among the data, using cluster-based structural and local geometric information among the samples. Experimental results obtained from different multi-label datasets and various performance measures show good performances of the linear and non-linear models. As a line of future research, we would like to discover ways to leverage the multi-label problem's peculiar properties by incorporating the label correlation while building the model.

Acknowledgments. Authors acknowledge sincere efforts of Saloni Kwatra in preparing related work of the initial draft of the manuscript during her stay in South Asian University as a Master's student.

References

1. Azad-Manjiri, M., Amiri, A., Sedghpour, A.S.: Ml-slstsvm: A new structural least square twin support vector machine for multi-label learning. Pattern Anal. Appl. **23**(1), 295–308 (2020)
2. Chen, W.J., Shao, Y.H., Deng, N.Y., Feng, Z.L.: Laplacian least squares twin support vector machine for semi-supervised classification. Neurocomputing **145**, 465–476 (2014)
3. Chen, W.J., Shao, Y.H., Li, C.N., Deng, N.Y.: Mltsvm: a novel twin support vector machine to multi-label learning. Pattern Recogn. **52**, 61–74 (2016)
4. Cheng, B., Liu, M., Zhang, D., Shen, D., Initiative, A.D.N.: Robust multi-label transfer feature learning for early diagnosis of alzheimer's disease. Brain Imaging Behav. **13**(1), 138–153 (2019)
5. Cortes, C., Vapnik, V.: Support-vector networks. Machine learning **20**(3), 273–297 (1995)
6. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research **7**(Jan), 1–30 (2006)
7. Hanifelou, Z., Adibi, P., Monadjemi, S.A., Karshenas, H.: Knn-based multi-label twin support vector machine with priority of labels. Neurocomputing **322**, 177–186 (2018)
8. Huang, J., Li, G., Huang, Q., Wu, X.: Learning label-specific features and class-dependent labels for multi-label classification. IEEE Trans. Knowl. Data Eng. **28**(12), 3309–3323 (2016)
9. Jayadeva, Khemchandani, R., Chandra, S.: Twin support vector machines for pattern classification. IEEE Transactions on Pattern Analysis and Machine Intelligence **29**(5), 905–910 (2007)
10. Khemchandani, R., Sharma, S.: Robust least squares twin support vector machine for human activity recognition. Appl. Soft Comput. **47**, 33–46 (2016)
11. Khemchandani, R., Sharma, S.: Robust parametric twin support vector machine and its application in human activity recognition. In: Proceedings of International Conference on Computer Vision and Image Processing. pp. 193–203. Springer (2017)
12. Kumar, M.A., Gopal, M.: Least squares twin support vector machines for pattern classification. Expert Syst. Appl. **36**(4), 7535–7543 (2009)

13. Kumar, S., Ahmadi, N., Rastogi, R.: Multi-label learning with missing labels using sparse global structure for label-specific features. Appl. Intell. **53**(15), 18155–18170 (2023)
14. Kumar, S., Rastogi, R.: Low rank label subspace transformation for multi-label learning with missing labels. Inf. Sci. **596**, 53–72 (2022)
15. Mir, A., Nasiri, J.A.: Knn-based least squares twin support vector machine for pattern classification. Appl. Intell. **48**(12), 4551–4564 (2018)
16. Qi, Z., Tian, Y., Shi, Y.: Laplacian twin support vector machine for semi-supervised classification. Neural Netw. **35**, 46–53 (2012)
17. Qi, Z., Tian, Y., Shi, Y.: Structural twin support vector machine for classification. Knowl.-Based Syst. **43**, 74–81 (2013)
18. Rastogi, R., Jain, S.: Multi-label learning via minimax probability machine. Int. J. Approximate Reasoning **145**, 1–17 (2022)
19. Rastogi, R., Kumar, S.: Discriminatory label-specific weights for multi-label learning with missing labels. Neural Process. Lett. **55**(2), 1397–1431 (2023)
20. Rastogi, R., Mortaza, S.: Multi-label classification with missing labels using label correlation and robust structural learning. Knowl.-Based Syst. **229**, 107336 (2021)
21. Rastogi, R., Mortaza, S.: Imbalance multi-label data learning with label specific features. Neurocomputing **513**, 395–408 (2022)
22. Xu, Y., Pan, X., Zhou, Z., Yang, Z., Zhang, Y.: Structural least square twin support vector machine for classification. Appl. Intell. **42**(3), 527–536 (2015)
23. Zhang, M.L., Wu, L.: Lift: Multi-label learning with label-specific features. IEEE Trans. Pattern Anal. Mach. Intell. **37**(1), 107–120 (2014)
24. Zhou, D., Huang, J., Schölkopf, B.: Learning with hypergraphs: Clustering, classification, and embedding. Adv. Neural. Inf. Process. Syst. **19**(1), 1601–1608 (2006)

Transformer Model for Multivariate Time Series Classification: A Case Study of Solar Flare Prediction

Khaznah Alshammari[1,2](✉), Shah Muhammad Hamdi[3], and Soukaina Filali Boubrahimi[3]

[1] New Mexico State University, Las Cruces, NM 88001, USA
kalshamm@nmsu.edu
[2] Northern Border University, Rafha, Saudi Arabia
[3] Utah State University, Logan, UT 84322, USA
{s.hamdi,soukaina.boubrahimi}@usu.edu

Abstract. Classifying solar flares is essential for understanding their impact on space weather forecasting. We propose a novel approach using a multi-head attention and transformer mechanism to classify multivariate time series (MVTS) instances of photospheric magnetic field parameters of the flaring events in the solar active regions. Attention mechanisms and transformer architectures capture complex temporal dependencies and interactions among features in multivariate time series data. Our model simultaneously attends to relevant features and learns their dependencies, enabling accurate classification of solar flare events. We evaluated our approach on *SWAN-SF*, the largest MVTS dataset for predicting solar flares, and compared its performance against several state-of-the-art methods. The experimental results demonstrate that our approach achieves superior classification performance, even when dealing with a highly imbalanced dataset characterized by the scarcity of major flaring events. These findings highlight the effectiveness of attention mechanisms and transformer models in learning discriminatory features from MVTS-based space weather data.

Keywords: Solar flares · Multivariate time series · Attention-based framework · Classification · Space weather forecasting

1 Introduction

A solar flare is an intense, localized eruption of electromagnetic radiation in the Sun's atmosphere. Flares occur in active regions and are often accompanied by coronal mass ejections, solar particle events, and other solar phenomena. The occurrence of solar flares varies with the 11-year solar cycle. Solar flares tend to be more frequent and intense during periods of high solar activity, which coincide with the solar maximum phase of the solar cycle. Solar flares result from the abrupt release of accumulated magnetic energy within the Sun's atmosphere. This energy can be stored in twisted magnetic fields above sunspots or

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 238–254, 2025.
https://doi.org/10.1007/978-3-031-78383-8_16

in other active regions. When magnetic energy is released, it heats the surrounding plasma to millions of degrees Celsius and accelerates particles to near the speed of light. Solar flares can produce a wide range of electromagnetic radiation, from radio waves to X-rays and gamma rays. Solar flares can have a significant impact on Earth and the space environment. They can cause radio blackouts, damage power grids, and disrupt satellite communications. Solar flares can also produce high-energy particles that can pose a hazard to astronauts and aircraft crews [14,18,19]. Effective predictions of solar flares are facilitated by employing time series modeling on magnetic field data collected by The Solar Dynamics Observatory's Helioseismic Magnetic Imager (HMI). Consequently, spatiotemporal magnetic field data is mapped into multiple instances of Multivariate Time Series (MVTS) [7]. Each MVTS instance contains solar magnetic field parameters such as flux, current, helicity, and Lorentz force. The time series associated with these parameters are derived from two distinct time windows: the observation window, which encompasses the recording of magnetic field parameter values, and the subsequent prediction window, corresponding to the period when peak X-ray flux was observed. The instances are then labeled into six classes: Q, A, B, C, M, and X. "Q" represents flare quiet active regions, while the other labels represent flaring events with increasing intensity. Notably, X and M-class flares denote the most intense flaring events. Recent advances in Multivariate Time Series (MVTS) models have demonstrated their superior effectiveness in predicting solar flaring activities when compared to earlier models that relied on single timestamps for magnetic field vector classification [7]. MVTS-based models for flare prediction can be broadly categorized into two main groups. The first category is the statistical feature-based method [16]. In this approach, low-dimensional representations of MVTS instances are computed by aggregating summary statistics from the individual univariate time series components. Subsequently, traditional classifiers such as k-nearest Neighbors (kNN) and Support Vector Machines (SVM) are trained using these labeled MVTS representations. The second category comprises end-to-end deep learning-based methods [24], which utilize Recurrent Neural Network (RNN) or Long Short-Term Memory (LSTM) based deep sequence models. These models learn by sequentially inputting vectors representing magnetic field parameters into the cells of the sequence model. The cell weights are optimized through backpropagation based on gradient descent. However, a limitation of these models is that they can only leverage the temporal dimension of the MVTS instances, resulting in suboptimal classification performance due to their limited ability to exploit the underlying patterns within the data. Vaswani et al. proposed the Transformer model, a neural network architecture based solely on self-attention mechanisms, to address the limitations of previous models [29]. The introduction of the transformer model marked a significant breakthrough in the field of natural language processing (NLP) and served as the cornerstone for numerous subsequent advancements, including cutting-edge language models such as BERT [13] and GPT [32]. One of its primary advantages lies in its efficiency in capturing long-range dependencies in data, all while allowing for parallel processing. This leads to faster training and inference times in comparison to previous models. The effectiveness of the transformer model, makes it a powerful choice for MVTS classification, leveraging its ability to capture long-range dependencies and handle multi-variable, temporal data effectively [29]. In our study, we aim to explore an alternative approach using attention and transformer techniques. By harnessing the power of self-attention mechanisms in transformers, our objective is to capture the extended temporal relationships among magnetic field parameters within the MVTS data, improving solar flare classification performance and deepening our understanding of these events. In this paper, we propose an attention-based model for the MVTS classification by leveraging the self-attention mechanisms to improve the MVTS classification performance. Our experimental results of our model demonstrate a test score of 70% on the solar flare MVTS dataset when using the proposed attention-based model, outperforming the baselines by more than 10%.

2 Related Work

Historically, systems for predicting solar flares heavily relied on human expertise and manual inputs. One early system known as THEO, implemented by the Space Environment Center (SEC) of NOAA back in 1987, required human intervention to input sunspot characteristics to categorize flare classes [22]. However, as recent NASA missions have generated a wealth of magnetic field data, the focus has shifted towards data-driven approaches, moving away from purely theoretical models. These data-driven approaches can be broadly categorized into linear and nonlinear statistical models, depending on the nature of the dataset used. These models can further be divided into line-of-sight magnetogram-based and full-disk photospheric vector magnetogram-based models. Line-of-sight magnetogram data captures only the component of the magnetic field along the line of sight, while full-disk photospheric vector magnetic field data provides a more comprehensive magnetic field state of solar active regions. Linear statistical models aimed to identify highly correlated magnetic field features associated with flare occurrences. For instance, Cui et al. [11] used line-of-sight magnetogram data to establish correlation-based statistical relationships between magnetic field parameters and flare events. Even before the launch of the Solar Dynamics Observatory (SDO), Leka et al. [20] utilized linear discriminant analysis (LDA) to classify flaring events using vector magnetogram data from the Mees Solar Observatory. In contrast, nonlinear statistical models employed a range of machine learning classifiers such as logistic regression, decision trees, neural networks, support vector machines (SVM), and more. For example, Song et al. [28] and Yu et al. [31] applied various classifiers to line-of-sight magnetogram-based datasets. Bobra et al. [9] utilized SVM with SDO-based vector magnetogram data for flare classification, while Nishizuka et al. [26] compared the performance of k-Nearest Neighbors (kNN), SVM, and Extremely Randomized Tree (ERT) on both line-of-sight and vector magnetogram data. Furthermore, Convolutional Neural Networks (ConvNets) have been employed for solar flare prediction using

SDO AIA/HMI images [21,33]. Recently, Angryk et al. introduced a novel approach to solar flare prediction based on temporal windows, which represents an extension beyond the earlier single timestamp-based models [7]. In their method, they employed a Multivariate Time Series (MVTS) dataset consisting of active regions, which recorded magnetic field data over a predefined observation period at a consistent sampling rate. Each instance in this dataset was labeled based on the flare classes that occurred after a specified prediction time. Other efforts, such as Hamdi et al. [17] and Muzaheed et al. [24], utilized statistical summarization, decision trees, and Long Short-Term Memory (LSTM)-based deep sequence modeling for flare prediction. Furthermore, Alshammari et al. [6] addressed the task of forecasting future values of magnetic field parameters within the MVTS representations. This involved predicting forthcoming values based on past data in the MVTS dataset. The transformer model, introduced by Vaswani et al.[29], offers several strengths for MVTS classification, including long-range dependency modeling. The transformer model can capture long-range dependencies in the data, which is effective for MVTS classification. Parallel computation of the transformer model makes it efficient for training and inference on large MVTS datasets. Transformer models can support contextual learning, enabling them to discern contextual relationships between magnetic field parameters without the need for explicit sequential processing. This is important for the classification of MVTS instances, as the context of a particular time step can be informative for predicting the occurrence of a solar flare.

3 Methodology

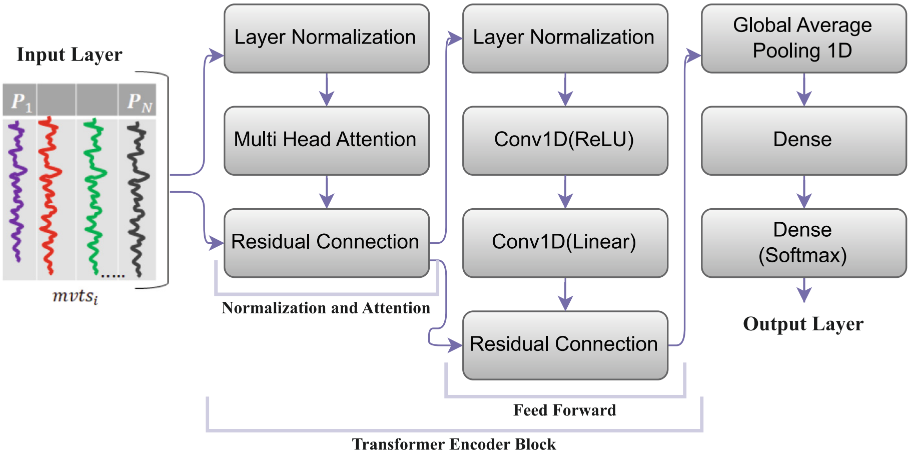


Fig. 1. Transformer Model for MVTS Classification

3.1 Notations

The solar event instance i is represented by an MVTS instance $mvts_i$. The MVTS instance $mvts_i \in \mathbb{R}^{T \times N}$ is a collection of individual time series of N magnetic field parameters, where each time series contains periodic observation values of the corresponding parameter for an observation period T. The MVTS instance can be expanded as $mvts_i = \{v_{t_1}, v_{t_2}, ., ., ., v_{t_T}\}$, where $v_{t_i} \in \mathbb{R}^N$ represents a timestamp vector.

3.2 Data Preprocessing and Normalization

The magnetic field parameter values are recorded in different scales, so we perform z-score normalization of each individual time series of each MVTS instance. At $mvts_i$, parameter-based individual time series are denoted by $P_1, P_2, \ldots, P_N$. For each individual time series P_j, we perform z-normalization as follows.

$$x_k^{(j)} = \frac{x_k^{(j)} - \mu^{(j)}}{\sigma^{(j)}} \tag{1}$$

Here, $x_k^{(j)}$ is the k-th value of the time series P_j, where $1 \leq k \leq T$, $\mu^{(j)}$ is the mean of time series P_j, and $\sigma^{(j)}$ is the standard deviation of the time series P_j. We apply the z-normalization for each partition individually. When partition i is used for training and partition j is used for test, we perform above z-normalization independently in the MVTS instances of partition i and j.

3.3 Transformer Model for MVTS Classification

In this work, we harness an attention-based model (transformer) to enhance the classification performance in an MVTS-based solar flare dataset. Within our model, we have designed the transformer encoder block. The foundation for this approach is rooted in the work of Vaswani et al. [29], where they introduced the transformer. The transformer architecture comprises both an encoder and a decoder, each comprising multiple layers that integrate self-attention and feed-forward neural networks. In our specific application, we primarily focus on the encoder component. This encoder is responsible for processing the input sequence, which, in the context of our study, corresponds to the solar flare data. The encoder structure consists of a stack of identical layers, with each layer housing two sub-layers:

- Self-attention layer: This layer plays a pivotal role by enabling each timestamp within the input time series to attentively consider all other timestamps within the same sequence. This mechanism empowers the layer to capture intricate temporal dependencies between individual timestamps and generate context-aware representations for each timestamp.

- Feed-forward neural network layer: Following the self-attention mechanism, a feed-forward neural network layer is independently applied to each timestamp representation. This layer introduces non-linearity into the model, allowing it to incorporate additional information and enhance its overall performance.

In this model, we incorporate the transformer encoder block and leverage the advantages of the multi-head attention architecture, a critical component of the transformer model. This architecture empowers the model to simultaneously focus on various segments of the input sequence, thereby enhancing its capacity to capture intricate temporal dependencies and extract pertinent features. By employing multiple attention heads, our model can acquire diverse representations and attend to distinct aspects of the input data concurrently. In the context of classifying MVTS data, multi-head attention offers several significant benefits:

- Enhanced Representational Capacity: Multi-head attention permits the model to attend to different portions of the input sequence in parallel, facilitating the capture of both local and global dependencies effectively. This capability empowers the model to discern complex patterns within the time series data, ultimately leading to improved classification performance.
- Robustness to Variable-Length Sequences: MVTS data frequently comprises sequences of varying lengths. Multi-head attention adeptly manages sequences of different lengths by assigning varying attention weights to different segments of the input. This adaptability enables the model to accommodate sequences with differing lengths without compromising its classification accuracy.

The key equations governing the multi-head attention mechanism are presented and explained in [29]. Our model, illustrated in Figure 1 is described in algorithms 1 and 2. Algorithm 1 operates as follows:

Algorithm 1. MVTS Transformer Encoder

```
function TRANSFORMER_ENCODER( $inputs$, $head\_size$, $num\_heads$, $ff\_dim$)
    $x \leftarrow$ LayerNormalization($inputs, \epsilon = 1e - 6$)
    $x \leftarrow$ MultiHeadAttention($x, x, key\_dim = head\_size, num\_heads = num\_heads$)
    $res \leftarrow x + inputs$
    $x \leftarrow$ LayerNormalization($res, \epsilon = 1e - 6$)
    $x \leftarrow$ Conv1D($x, filters = ff\_dim, kernel\_size = 1, activation = "relu"$)
    $x \leftarrow$ Conv1D($x, filters = inputs.shape[-1], kernel\_size = 1$)
    return $x + res$
end function
```

1. Layer Normalization: The tensor representation of MVTS instances is first normalized along each feature dimension by passing it through a layer normalization layer.

2. Self-Attention: The normalized tensor is then fed into a multi-head attention layer, where a self-attention mechanism is applied. Each attention head attends to different parts of the input sequence and learns to capture distinct relationships between time steps. The output of the attention layer retains the same shape as the input.
3. Residual Connection: The output of the multi-head attention layer is element-wise added to the original input tensor (inputs). This residual connection facilitates the direct flow of gradients from the input to the output, easing the learning process for the model.
4. Feed-forward layer: The result of the residual connection is passed through another layer normalization layer.
5. Convolutional Layer: A 1D convolutional layer with ff_dim filters and kernel size 1 is applied to the normalized tensor. This layer acts as a feed-forward neural network layer, applying non-linear transformations independently to each position in the sequence.
6. Second Convolutional Layer: Another 1D convolutional layer with inputs of shape[-1] filters and kernel size 1 is applied to the result obtained from the previous layer.
7. Residual Connection: The output of the second convolutional layer is element-wise added to the result obtained from the first residual connection layer.
8. Final Output: The sum of the previous residual connection and the original input tensor (inputs) is returned as the final output.

Algorithm 2. Build MVTS Transformer(Attention) Model

```
function BUILD_TRANSFORMER_MODEL($input\_shape$, $head\_size$, $num\_heads$, $ff\_dim$, $num\_transformer\_blocks$, $mlp\_units$)
    $n\_classes \leftarrow$ LENGTH($unique\_y\_train$)
    $inputs \leftarrow$ INPUT($shape = input\_shape$)
    $x \leftarrow inputs$
    for $i \leftarrow 1$ to $num\_transformer\_blocks$ do
        $x \leftarrow$ TRANSFORMER_ENCODER( $x$, $head\_size$, $num\_heads$, $ff\_dim$ )
    end for
    $x \leftarrow$ GLOBALAVERAGEPOOLING1D( $x$, $data\_format = "channels\_first"$ )
    for $dim$ in $mlp\_units$ do
        $x \leftarrow$ DENSE($x, dim, activation = "relu"$)
    end for
    $outputs \leftarrow$ DENSE($x, n\_classes, activation = "softmax"$)
    return MODEL($inputs, outputs$)
end function
```

Algorithm 2 incorporates several parameters, each described as follows: $input\_shape$ specifies the shape of the input data, $head\_size$ determines the size of each attention head in the transformer, num_heads denotes the number of attention heads in the transformer, ff_dim represents the dimension of the feed-forward network in the transformer, $num\_transformer\_blocks$ indicates the number of transformer blocks to be stacked, and mlp_units is a list of integers specifying the number of units in each MLP layer. Within the algorithm, it first determines the number of classes ($n\_classes$) based on the unique labels present in the training data. It then defines the input layer and sets it as the current layer, denoted as x. The algorithm proceeds by applying the transformer encoder block through the $transformer\_encoder$ function. After the transformer encoder blocks, a global average pooling layer is applied to reduce the spatial dimensions of the data. Subsequently, a series of MLP layers are implemented as specified by the mlp_units parameter, with each layer employing $ReLU$ activation. Finally, an output layer is added with $n\_classes$ units and a $softmax$ activation function for classification.

4 Experiments

In this section, we present our experimental findings, where we compare the performance of our model with five other MVTS-based flare prediction baselines using a benchmark dataset. We implemented our attention-based MVTS classifier using TensorFlow on the A100 Nvidia GPU. The hyper-parameters were found by random hyper-parameter search, and set as $head\_size$=256, num_heads=4, ff_dim=4, $num\_transformer\_blocks$=10, mlp_units= 64. The source code of our model, along with the experimental dataset, is available in our GitHub repository.[1]

4.1 Evaluation Metrics

We used True Skill Statistic (TSS) as a performance measure for our experiments. The True Skill Statistic (TSS) takes into account both the hits and false alarms in the prediction. It is calculated as

$$TSS = \frac{TP}{TP + FN} - \frac{FP}{FP + TN}$$

where TP is the number of true positives (correct predictions of flares), FN is the number of false negatives (missed predictions of flares), FP is the number of false positives (incorrect predictions of flares), and TN is the number of true negatives (correctly predicted non-flares). The TSS ranges from −1 to 1, where a value of 1 represents a perfect prediction, a value of 0 represents a random prediction, and −1 indicates that the model is wrong in all of its predictions [8]. We use TSS as a performance metric because it has been used frequently to report the performance of machine learning models for the prediction of rare events, e.g., solar flares [1,7,17]. TSS can accurately measure the model's ability to distinguish between the classes, regardless of how common or rare they are. TSS is widely used in machine learning and statistical modeling, especially for tasks

[1] https://github.com/Kalshammari/Transformer-Model.git.

such as binary classification [15]. One advantage of using TSS as a performance metric in datasets with high-class imbalance is that it takes into account both the true positive rate and the true negative rate of the classifier, which is important when the classes are imbalanced [15]. TSS can also be used to compare models with different thresholds for presence-absence predictions [3].

4.2 SWAN-SF Benchmark Data Set

As the benchmark dataset of our experiments, we used the MVTS-based solar flare prediction dataset SWAN-SF published in [7]. The SWAN-SF benchmark dataset is a collection of multivariate time series (MVTS) data instances that facilitate unbiased flare forecasting. The MVTS instances of the SWAN-SF benchmark dataset are labeled by five different flare classes, namely, GOES X, M, C, and B, and a non-flaring class denoted by Q. Class Q includes flare-quiet events and GOES A-class events. The dataset comprises five temporally segmented partitions and is designed in a way that each partition includes approximately the same number of X- and M-class flares. Table 1 shows the partition-wise label statistics of the SWAN-SF dataset. The dataset contains various time series parameters derived from solar photospheric magnetograms along with NOAA's flare history of active regions. The magnetograms and their metadata are obtained from the Spaceweather HMI Active Region Patch (SHARP) data product. The magnetic field parameters are physics-based and were recalculated and enhanced for validation purposes. Each MVTS instance in the dataset is made up of a 24-time series of active region magnetic field parameters (the full list can be found in [4,9]. The time series instances are recorded at 12-minute intervals for a total duration of 12 hours (that results in 60-time steps). In this paper, $T = 60$ is used to denote the number of observation time steps, and $N = 24$ to denote the number of magnetic field parameters. In this study we use all 24 magnetic field parameters. In our experiments of feature selection from MVTS data, we conduct the binary classification between flaring and non-flaring AR, where we consider flaring AR (class X and M) to be in a positive class and non-flaring Active Regions (class Q) to be in the negative class. We removed the B and C class events since the opposite event classes (X + M vs Q) help in contrastive learning. The removal of B and C class flares for maximizing flare prediction performance was also suggested by the experimental findings of multiple previous studies [2,5,9,17,27].

4.3 Baseline Models

We evaluated our model with five other baselines.

1. **Long Short-Term Memory (LSTM)** The LSTM-based approach was proposed by Muzaheed et. al. [24]. Each MVTS instance was considered as a T -length sequence of $x^{<t>} \in \mathbb{R}^N$ timestamp vectors. After sequentially feeding the LSTM model with each timestamp vector, the last hidden representation was considered as the MVTS representation. As suggested by the paper, we

set the number of cell state and hidden state dimensions to 64, the number of training epochs to 500, and the learning rate in stochastic gradient descent to 0.01.

2. **Support Vector Machine (SVM)** SVM is known for its ability to handle linear and non-linear data effectively, making it a versatile choice for various applications. It employs support vectors, which are data points closest to the decision boundary, to determine the orientation and placement of the hyperplane. This approach allows SVM to excel in complex and high-dimensional datasets[10].
3. **Canonical Interval Forest (CIF)** The time series forest (TSF) classifier, known for its high performance, quick training, and prediction, is commonly regarded as a powerful interval method proposed by [23].
4. **Multiple Representations SEQuence Learner (MRSEQL)** MRSEQL, proposed by [25], is a robust univariate time series classifier that trains on features derived from multiple symbolic representations of time series. These representations include Symbol Aggregation Approximation (SAX) and Symbol Fourier Approximation (SFA), which are used with linear classification models (logistic regression).
5. **MINImally RandOm Convolutional KErnels Transform (MINIROCKET)** MINIROCKET is a fast and accurate algorithm for time series classification. It is a (nearly) deterministic reformulation of the ROCKET algorithm, which is a state-of-the-art algorithm for time series classification [12].

4.4 Train/validation/test splitting method

The SWAN-SF dataset has a temporal coherence property that measures how stable and consistent the magnetic field structures of a solar active region are over time. It poses a challenge for predicting rare events such as solar flares using time series data. It requires that the predictions for a given time point are in agreement with past and future predictions. If temporal coherence is ignored, the model performance may be artificially inflated. This problem stems from the data collection method and affects the data splitting into training, validation, and testing sets. To address the issue of temporal coherence, we use different time-segmented partitions of the dataset for training and testing samples. This is the reason why the SWAN-SF dataset has multiple non-overlapping partitions. Table 1 shows each partition statistics. By using different partitions of SWAN-SF for training and testing, we avoid testing the model on time series that are partly identical to those used for training [1]. In this study, we use the following settings: partition 1 for training and validation and partition 2 for testing, partition 2 for training and validation and partition 3 for testing, partition 3 for training and validation and partition 4 for testing, partition 4 for training and validation and partition 5 for testing, and partition 5 for training and validation and partition 1 for testing.

Table 1. Event type statistics of each partition of the SWAN-SF dataset

Flare Type	**Partitions**				
	P1	P2	P3	P4	P5
Q	60,130	73,368	34,762	43,294	62,688
B	5,692	4,978	685	846	5,924
C	6,416	8,810	5,639	5,956	5,763
M	1,089	1,329	1,288	1,012	971
X	165	72	136	153	19
sum	**73,492**	**88,557**	**42,510**	**51,261**	**75,365**

4.5 Binary classification performance

Binary classification plays a significant role in distinguishing major flaring events from minor flaring events or flare quiet events. In this experiment, we focus on classifying X and M class MVTS instances as flaring events, while considering all other instances (Q) as non-flaring events. Figure 2 depicts the binary classification performances of all models. The results demonstrate that the transformer-based MVTS model outperforms all other baseline models, and achieves an average improvement of approximately 8% to 20% compared to the second-best performing MINIROCKET algorithm. These findings highlight the superior performance of our model in binary classification. This consistency reinforces the efficacy and reliability of our Transformer-based model in accurately predicting flaring events.

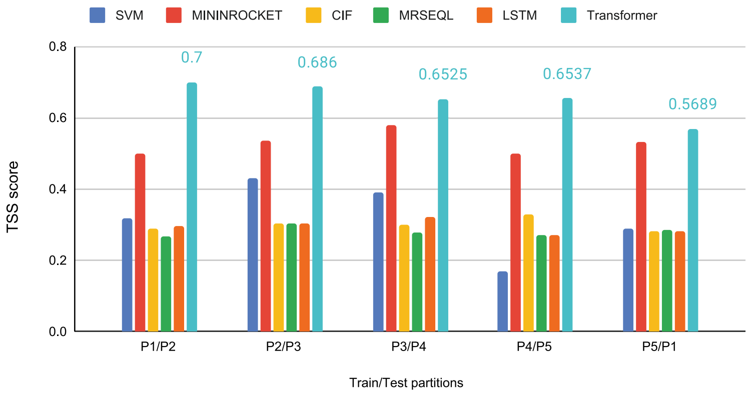


Fig. 2. Binary classification performance of all baselines compared to the transformer model.

4.6 Ablation Study of the Transformer-base MVTS Classification Model

To get a better understanding of the effectiveness of the different layers in our model, we conducted several experiments to evaluate the significance of various layers. First, we evaluated the importance of the self-attention mechanism by removing it from the model architecture and comparing the results. The removal of the attention mechanism (when partition 1 was used for training and partition 2 for testing) led to a noticeable drop in the TSS score, from 70% to 54%. This outcome highlights the significant role played by the multi-head attention layer in capturing relevant patterns and relationships within the MVTS data. Second, we examined the impact of layer normalization by removing it from the model. This resulted in a decrease in TSS from 70% to 44%. This finding underscores the importance of layer normalization in maintaining the model's performance and stability. Finally, we investigated the effect of the 1D convolutional layers. When these layers were removed from the model, there was a significant drop in TSS from 70% to 51%. This result demonstrates the crucial role played by the 1D convolutional layers in capturing important temporal features and contributing to the overall performance of the model. The ablation study provided valuable insights into the contributions of different layers in our model. The significant decrease in TSS upon removing the attention mechanism, layer normalization, and 1D convolutional layers highlights their importance in capturing relevant patterns, maintaining stability, and extracting essential temporal features. These findings underscore the effectiveness and significance of each layer in our model architecture.

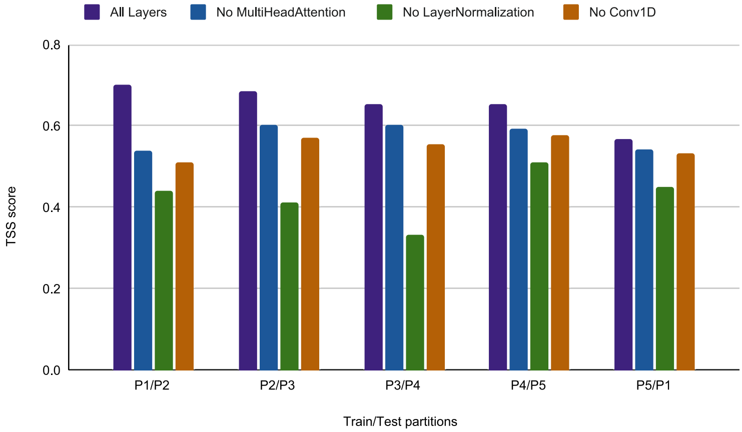


Fig. 3. Ablation Study: The Contributions of Model Components in MVTS Classification of Solar Flares.

Table 2. Experimental Results (TSS scores) for Different Hyperparameters Values on (Train/Test) Partitions.

Head Size	**Num Heads**	**FF Dim**	**Num of Transformer Blocks**	**MLP Units**	**TSS (P1/P2)**	**TSS (P2/P3)**	**TSS (P4/P5)**	**TSS (P5/P1)**
256	4	4	10	64	**0.70**	**0.69**	**0.58**	0.57
512	4	4	10	64	0.62	0.68	0.52	0.63
512	8	8	20	128	0.68	0.68	0.57	**0.64**
128	2	2	5	32	0.69	0.65	0.47	0.29
256	2	2	5	32	0.58	0.65	0.41	0.45
256	4	4	10	128	0.58	0.60	0.32	0.46

4.7 The Impact of Different Hyperparameters Values on The Model Performance

Table 2 presents the performance of various Transformer model configurations, highlighting key hyperparameters such as head size, number of heads, feed-forward dimensions, number of Transformer blocks, and MLP units. The Total Sum of Squares (TSS) scores across different train/test partitions (P1/P2, P2/P3, P4/P5, and P5/P1) demonstrate the model's effectiveness in capturing data variance. For the first row in the table, the model configuration includes an attention head size of 256, 4 attention heads, a feed-forward dimension of 4, and 10 Transformer blocks. Additionally, the MLP (Multi-Layer Perceptron) units are set to 64. The computational complexity for this configuration is approximately $O(10 \cdot (n^2 \cdot 256 + n \cdot 256^2))$, where n represents the sequence length. This complexity estimate indicates how the computational cost grows with the sequence length (n) and model size (256), helping to understand the resource requirements for this specific model setup.

5 Discussion

We acknowledge that the application of the vanilla transformer architecture is not novel in a methodological sense, we believe that the contribution of our study lies in its specific adaptation to the solar flare prediction domain. The utilization of self-attention mechanisms within the transformer framework, tailored to the characteristics of solar flare MVTS datasets, addresses unique challenges in time series classification. Our primary focus was to explore the effectiveness of self-attention mechanisms in capturing long-range dependencies and intricate patterns inherent in solar flare data. We believe that the context-specific adaptation of the transformer architecture contributes valuable insights to the solar flare prediction community. The impact of this study in the field of space weather forecasting is significant. Accurate prediction of solar flares is important for mitigating the adverse effects of space weather on satellite communications, power grids, and other critical infrastructure. By leveraging the advanced capabilities of Transformer models, this research provides a robust framework for enhancing the prediction accuracy of solar flare events. The use of self-attention mechanisms enables the model to capture intricate temporal dependencies and interactions among multiple magnetic field parameters, which are essential for understanding the complex dynamics of solar flares. The proposed model's ability to handle large-scale multivariate time series data and its applicability to real-world scenarios make it a practical tool for operational space weather forecasting. By addressing the limitations of previous models and demonstrating superior performance, this research contributes to the development of more reliable and effective space weather prediction systems. The Transformer model is important, particularly in the context of solar flare prediction. The self-attention mechanism used in the Transformer model allows it to focus on different parts of the input sequence, providing insights into which features and time steps are most influential in making predictions. This capability can help identify key magnetic field parameters and their interactions that contribute to the occurrence of solar flares. Furthermore, by analyzing the attention weights, researchers can gain a better understanding of the physical mechanisms underlying solar flare events. The model's ability to capture long-range dependencies and complex temporal relationships in multivariate time series data makes it a powerful tool for studying the dynamics of solar active regions. This can lead to improved predictions and a deeper understanding of the processes that drive solar flare activity, ultimately contributing to advancements in space weather forecasting.

6 Conclusion

In this work, we introduced a transformer-based model for predicting solar flares, employing the self-attention mechanism for the classification of Multivariate Time Series (MVTS) instances. Our study presents an innovative approach that harnesses the capabilities of the transformer model and the self-attention mechanism for MVTS classification. Through an end-to-end learning process, our proposed model effectively captures the temporal relationships inherent within MVTS instances. This includes the recognition of higher-order inter-variable relationships as well as local and global temporal changes. By incorporating attention-based techniques, our experiments conducted on a solar flare prediction dataset showcase the remarkable performance of our model in binary class MVTS classification, achieving an impressive TSS score of 70%. These outcomes underscore the potential of our approach to offer more comprehensive and precise predictions in the realm of solar physics and space weather forecasting. For future research, we intend to apply the Graph Attention Network [30] on the functional network constructed from the time series correlation so that the model can capture both spatial (inter-variable) and temporal dependencies for learning robust representations of the MVTS instances.

Acknowledgements. This project has been supported in part by funding from the Division of Atmospheric and Geospace Sciences within the Directorate for Geosciences, under NSF awards #2301397, #2204363, and #2240022, and by funding from the Office of Advanced Cyberinfrastructure within the Directorate for Computer and Information

Science and Engineering, under NSF award #2305781. The authors acknowledge the use of ChatGPT (GPT-3.5) to rephrase sentences and improve the writing style of the manuscript.

References

1. Ahmadzadeh, A., Aydin, B., Georgoulis, M.K., Kempton, D.J., Mahajan, S.S., Angryk, R.A.: How to train your flare prediction model: Revisiting robust sampling of rare events. The Astrophysical Journal Supplement Series **254**(2), 23 (may 2021) https://doi.org/10.3847/1538-4365/abec88,https://doi.org/10.3847/1538-4365/abec88
2. Ahmadzadeh, A., Aydin, B., Georgoulis, M.K., Kempton, D.J., Mahajan, S.S., Angryk, R.A.: How to train your flare prediction model: Revisiting robust sampling of rare events. Astrophys. J. Suppl. Ser. **254**(2), 23 (2021). https://doi.org/10.3847/1538-4365/abec88
3. Allouche, O., Tsoar, A., Kadmon, R.: Assessing the accuracy of species distribution models: prevalence, kappa and the true skill statistic (tss). J. Appl. Ecol. **43**(6), 1223–1232 (2006). https://doi.org/10.1111/j.1365-2664.2006.01214.x
4. Alshammari, K., Hamdi, S.M., Boubrahimi, S.F.: Feature selection from multivariate time series data: A case study of solar flare prediction. In: IEEE International Conference on Big Data, Big Data 2022, Osaka, Japan, December 17-20, 2022. pp. 4796–4801. IEEE (2022).https://doi.org/10.1109/BIGDATA55660.2022.10020669,https://doi.org/10.1109/BigData55660.2022.10020669
5. Alshammari, K., Hamdi, S.M., Boubrahimi, S.F.: Identifying flare-indicative photospheric magnetic field parameters from multivariate time-series data of solar active regions. Astrophys. J. Suppl. Ser. **271**(2), 39 (2024)
6. Alshammari, K., Hamdi, S.M., Muzaheed, A.A.M., Boubrahimi, S.F.: Forecasting multivariate time series of the magnetic field parameters of the solar events. CIKM workshop for Applied Machine Learning Methods for Time Series Forecasting (AMLTS) (2022)
7. Angryk, R.A., Martens, P.C., Aydin, B., Kempton, D., Mahajan, S.S., Basodi, S., Ahmadzadeh, A., Cai, X., Filali Boubrahimi, S., Hamdi, S.M., et al.: Multivariate time series dataset for space weather data analytics. Scientific data **7**(1), 1–13 (2020)
8. Bloomfield, D.S., Higgins, P.A., McAteer, R.T.J., Gallagher, P.T.: Toward reliable benchmarking of solar flare forecasting methods. The Astrophysical Journal Letters **747**(2), L41 (feb 2012).https://doi.org/10.1088/2041-8205/747/2/L41,https://doi.org/10.1088/2041-8205/747/2/L41
9. Bobra, M.G., Couvidat, S.: Solar flare prediction using sdo/hmi vector magnetic field data with a machine-learning algorithm. Astrophys J **798**(2), 135 (2015)
10. Cortes, C., Vapnik, V.: Support-vector networks. Machine learning **20**(3), 273–297 (1995)
11. Cui, Y., Li, R., Zhang, L., He, Y., Wang, H.: Correlation between solar flare productivity and photospheric magnetic field properties. Sol. Phys. **237**(1), 45–59 (2006)
12. Dempster, A., Schmidt, D.F., Webb, G.I.: MiniRocket: A very fast (almost) deterministic transform for time series classification. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. pp. 248–257. ACM, New York (2021)

13. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding (2019)
14. Eastwood, J., Biffis, E., Hapgood, M., Green, L., Bisi, M., Bentley, R., Wicks, R., McKinnell, L.A., Gibbs, M., Burnett, C.: The economic impact of space weather: Where do we stand?: The economic impact of space weather. Risk Analysis **37** (02 2017) https://doi.org/10.1111/risa.12765
15. Gao, J., Han, Y., Mao, Y.: A novel evaluation metric for imbalanced classification based on gini coefficient and tss. IEEE Access **8**, 80268–80280 (2020) https://doi.org/10.1109/ACCESS.2020.2996775
16. Hamdi, S.M., Aydin, B., Boubrahimi, S.F., Angryk, R., Krishnamurthy, L.C., Morris, R.: Biomarker detection from fmri-based complete functional connectivity networks. In: 2018 IEEE First International Conference on Artificial Intelligence and Knowledge Engineering (AIKE). pp. 17–24. IEEE (2018)
17. Hamdi, S.M., Kempton, D., Ma, R., Boubrahimi, S.F., Angryk, R.A.: A time series classification-based approach for solar flare prediction. In: 2017 IEEE International Conference on Big Data (Big Data). pp. 2543–2551 (2017).https://doi.org/10.1109/BigData.2017.8258213
18. Hosseinzadeh, P., Boubrahimi, S.F., Hamdi, S.M.: Improving solar energetic particle event prediction through multivariate time series data augmentation. Astrophys. J. Suppl. Ser. **270**(2), 31 (2024)
19. Hosseinzadeh, P., Filali Boubrahimi, S., Hamdi, S.M.: Toward enhanced prediction of high-impact solar energetic particle events using multimodal time series data fusion models. Space Weather **22**(6), e2024SW003982 (2024) https://doi.org/10.1029/2024SW003982, https://doi.org/10.1029/2024SW003982, e2024SW003982 2024SW003982
20. Leka, K., Barnes, G.: Photospheric magnetic field properties of flaring versus flare-quiet active regions. ii. discriminant analysis. The Astrophysical Journal **595**(2), 1296 (2003)
21. Li, X., Zheng, Y., Wang, X., Wang, L.: Predicting solar flares using a novel deep convolutional neural network. Astrophys J **891**(1), 10 (2020)
22. McIntosh, P.S.: The classification of sunspot groups. Sol. Phys. **125**(2), 251–267 (1990)
23. Middlehurst, M., Large, J., Bagnall, A.J.: The canonical interval forest (CIF) classifier for time series classification. CoRR **abs/2008.09172** (2020), https://arxiv.org/abs/2008.09172
24. Muzaheed, A.A.M., Hamdi, S.M., Boubrahimi, S.F.: Sequence model-based end-to-end solar flare classification from multivariate time series data. In: 20th IEEE International Conference on Machine Learning and Applications, ICMLA 2021, Pasadena, CA, USA, December 13-16, 2021. pp. 435–440. IEEE (2021)https://doi.org/10.1109/ICMLA52953.2021.00074,https://doi.org/10.1109/ICMLA52953.2021.00074
25. Nguyen, T.L., Gsponer, S., Ilie, I., O'Reilly, M., Ifrim, G.: Interpretable time series classification using linear models and multi-resolution multi-domain symbolic representations. CoRR **abs/2006.01667** (2020), https://arxiv.org/abs/2006.01667
26. Nishizuka, N., Sugiura, K., Kubo, Y., Den, M., Watari, S., Ishii, M.: Solar flare prediction model with three machine-learning algorithms using ultraviolet brightening and vector magnetograms. Astrophys. J. **835**(2), 156 (2017)
27. Saini, K., Alshammari, K., Hamdi, S.M., Filali Boubrahimi, S.: Classification of major solar flares from extremely imbalanced multivariate time series data using minimally random convolutional kernel transform. Universe **10**(6) (2024) https://doi.org/10.3390/universe10060234, https://www.mdpi.com/2218-1997/10/6/234

28. Song, H., Tan, C., Jing, J., Wang, H., Yurchyshyn, V., Abramenko, V.: Statistical assessment of photospheric magnetic features in imminent solar flare predictions. Sol. Phys. **254**(1), 101–125 (2009)
29. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need (2017)
30. Velivckovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. In: Proceedings of the 5th International Conference on Learning Representations (ICLR) (2018)
31. Yu, D., Huang, X., Wang, H., Cui, Y.: Short-term solar flare prediction using a sequential supervised learning method. Sol. Phys. **255**(1), 91–105 (2009)
32. Zheng, X., Zhang, C., Woodland, P.C.: Adapting gpt, gpt-2 and bert language models for speech recognition (2021)
33. Zheng, Y., Li, X., Wang, X.: Solar flare prediction with the hybrid deep convolutional neural network. Astrophys J **885**(1), 73 (2019)

Learning Social and Physical Compliant Multi-modal Futures

Xiaodan Shi[1,3](✉), Haoran Zhang[2,3], Shibasaki Ryosuke[3], and Jinyue Yan[4]

[1] Future Energy Center, Mälardalens University, 72123 Västerås, Sweden
xiaodan.shi@mdu.se

[2] School of Urban Planning and Design, Peking University, No. 2199 Lishui Road, Nanshan District, Shenzhen 518055, Guangdong, China
h.zhang@pku.edu.cn

[3] Center for Spatial Information Science, The University of Tokyo, Kashiwa, Japan
shiba@csis.u-tokyo.ac.jp

[4] Department of Building Environment and Energy Engineering, The Hong Kong Polytechnic University, Hung Hom, Hong Kong SAR, China
j-jerry.yan@polyu.edu.hk

Abstract. Long-term human path forecasting in crowds is critical for autonomous moving platforms (like autonomous driving cars and social robots) to avoid collision and make high-quality planning. It is not easy for prediction systems to successfully model the inherent uncertainty of futures while take into account dynamic social and physical interactions in a highly interactive circumstance. Towards these goals, we develop a unifying model to predict socially and physically acceptable multi-modes of future trajectories. The modes of trajectories condition on past observation, which are not explicit labels but indicate walking patterns (such as walking straight, turning left/right) and interacting patterns (such as aggressive, mild). Our model contains an encoder and a decoder, which leverages two source of information, all past path of agents in a shared scenario and scene context to jointly learn the representations of social and physical interactions, where the former can efficiently scale to any number of agents and the latter helps the model learn the traversable parts of the scenario. Extensive experiments over several trajectory prediction benchmarks demonstrate that our method is able to capture the multi-modality of human motion and forecast the distributions of plausible futures in complex scenarios.

Keywords: Trajectory Prediction · Multi-Modal Prediction · LSTM

1 Introduction

Forecasting long-term future trajectories of dynamic pedestrians through crowded scenarios is of major importance for autonomous driving, social robots navigation and surveillance systems [1–5]. In autonomous driving and social robot navigation, autonomous driving cars and social robots share the same

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 255–269, 2025.
https://doi.org/10.1007/978-3-031-78383-8_17

ecosystem with humans. They adjust their path by anticipating human movement, specifically, avoid collision or keep safe distance with other people. Predicting long-term trajectories in crowds is still a challenging topic due to the following properties of human motion.

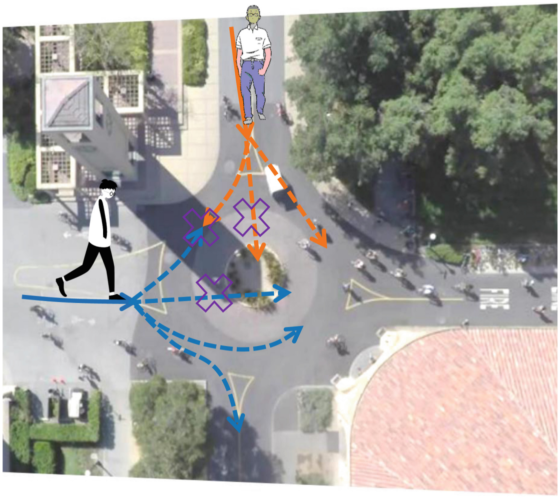


Fig. 1. Our goal is to predict multi-modal futures that are social and physical acceptable.

1. Multi-modes of future trajectories. Given the observation of motion sequence, multiple future trajectory sequences are acceptable. It is not rational to forecast a single path especially for the task of long-term prediction.
2. Social interactions. Interactions between people happens frequently. Although humans can intuitively know how to interact with other people in crowds, it is not easy for machines to learn those rules due to complexity and dynamics of social interactions.
3. Scene context. Pedestrians motion should also obey physical constraints. Pedestrians walk on feasible terrains such as sidewalk or grass and avoid static obstacles such as roadblocks. Instead of encoding image of scenario into prediction model, physical constraints can also be learned from trajectories.

Since the success of recurrent neural network (RNN) on sequence modeling, the existing state-of-art research focus on inventing RNN-based models for addressing the above problems and predicting long-term trajectories. However, they often only take into account one or two of the aforementioned features. Social LSTM[6] pooling latent states coming from LSTMs of spatially proximal trajectories to model social interactions, is an critical development for real-world path forecasting. Social GAN and Social BiGAT investigate the uncertainty of futures while considering multi-agents' interactions by proposing GAN-based (RNN-based generator and RNN-based discriminator) encoder-decoder architectures with social mechanism, but they ignore the influence of static scene on trajectories. The research[7] represents both trajectories and the scene as images and encodes them via CNNs, but it considers agents independently from each other. Sophie[8] forecasts social and physical compliant multiple futures by proposing GAN-based encoder-decoder model which blends a social attention mechanism with a physical attention mechanism. But It falls short of learning the truly multi-modality of futures[9].

To address the above limitations, we develop an unifying model which is able to forecast social and physical compliant multi-modal future trajectory sequences jointly for all agents in a shared scenario. The proposed model is an encoder-decoder model which jointly integrates past trajectories, dynamic people-people and people-space interactions and learns multiple meaningful modes of trajectories. The modes of trajectories conditions on past observation, which are not explicit labels but indicate walking patterns (such as walking straight, turning left/right) and interacting patterns (such as aggressive, mild). The decoder conditions on the predicted mode and generate the sequence of future trajectories. Our model takes on point-of-view images of agents and learns an abstract, compressed representation of historical scene images. Besides, we utilize a spatio-temporal graph representation to naturally model social interactions and ego-trajectories. We leverage relative motion between people while consider time dependencies of long-term social interactions. Instead of setting a certain neighborhood size or certain number of neighbors, we assume all people in a shared environment interacting and pool relative latent states between people through an attention mechanism. We test the model using classic trajectory prediction benchmarks and the experiments show promising results.

2 Related Works

2.1 Social interactions for trajectory prediction

Social LSTM introducing Social Pooling to learn a global feature of all nearby neighbors around an agent which is meant to represent common sense rules and social conventions, is a tipping point for data-driven long-term trajectory prediction. Many research follow the way of Social LSTM[6] but with improvements. Attention mechanism is introduced to learn neighbors' weights on agent[8,10,11]. Fernando et al. extended the classic model to incorporate both soft attention as well hard attention where the former is for handling longer trajectories and the latter is used for modeling interacting people[11]. Instead of directly modeling hidden states of neighbors' motion, some research pool relative motion between agent and neighbors to model interactions. SR-LSTM proposed a state refinement module for LSTM, which extracting social effects of neighbors by embedding and aggregating the relative spatial location between agent and neighbors[10].

2.2 Graph for trajectory prediction

Graph representation, specifically spatio-temporal graph (ST-graph) is recently applied to illustrate human motion and their interactions[2,12,13]. ST-graph for human motion representation is unrolled through time and characterized with elements points, spatial-edges and temporal-edges. ST-graph provide a more direct and natural way to model interactions for trajectory prediction. Structure-RNN[14] combining high-level spatio-termporal graphs with sequence modeling success of RNN made significant improvements on problem of human motion modeling. Some research follow this direction. Social-BiGAT introduced a flexible graph attention network to model social interactions between pedestrians in a scene. It assumes all people in a scene interacting instead of setting a local neighborhood[2]. Social-STGCNN utilized spatio-temporal graph representation and proposed a weighted adjacency matrix to meansure the influence between pedestrians[13].

2.3 Multi-modality of trajectory prediction

Human motions under crowded scenarios imply a multiplicity of modes. To capture the uncertainty of future path, some research apply generative adversarial network (GAN) to generate multiple possible paths. Gupta A. et al. proposed Social GAN which contains RNN based encoder-decoder generator and RNN-based decoder discriminator[15]. Social GAN integrates all the interactions involved in the scenarios and encourages the generative network to spread its distribution and cover the space of possible paths by introducing a variety loss. Sadeghian A. et al proposed Sophie, an attentive GAN to jointly model static human-space, and dynamic human-human interactions by blending a social attention mechanism with a physical attention that helps the model to learn where to look in a large scene and to extract the most salient parts of the image relevant to the path[8]. Some research apply Mixture Density Network (MDN) to map the distribution of future trajectories[12,16]?. The article[16], based on MDN, proposed a two stage strategy that first predicted several samples of future with Winner-Takes-All loss and then iteratively grouped the samples to multiple modes.

2.4 Scene context for trajectory prediction

Static scene context containing walkable area such as roads, side-walks, crosswalks and unwalkable area such as buildings, obstacles, which are useful for trajectory prediction. Most of the existing research encode the scene image to get a representation of scene context through Convolutional Neural Network (CNN) and then concatenate it with past trajectory to predict futures. The search extracts features of top-view scene images through CNN and then applies a visual attention model to learn most salient parts of scene for trajectories. The research[7] represent the past trajectories of agents using binary 2-D grids, and the underlying scene as a RGB birds-eye view images. The research[8] encodes the scene image and binary image using ResNet and U-Net respectively, and generate trajectory forecasts using a Convolutional LSTM decoder. The research[17] first train a Variational Autoencoder and then use the encoder part to encode the scene context for various tasks such as prediction. Similar to , our model compress the scene images into an abstract, compressed representation and integrate it with other features for long-term trajectory prediction.

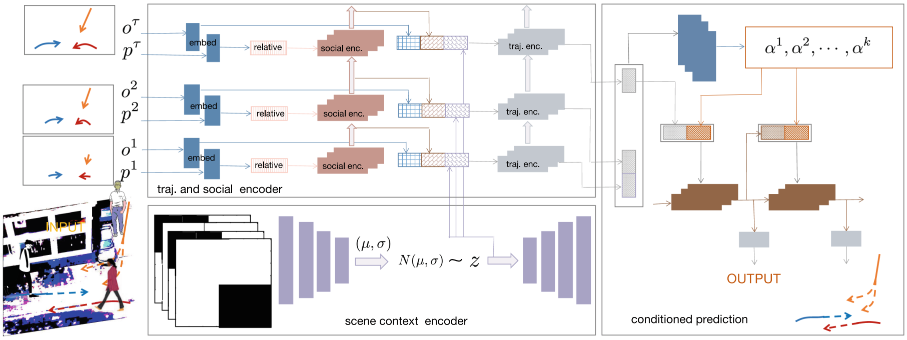


Fig. 2. Architecture of our model.

3 Problem Formulation

We assume that each scenario has been pre-processed to get 2D spatial coordinates $(x_i^t, y_i^t) \in \mathbf{R}^2$ and 2D walking speed $(u_i^t, v_i^t) \in \mathbf{R}^2$ of all people, scenes where people are looking at m_i^t at all time instances. There are N agents in a scenario. The observation of agent i is past trajectories represented as: $X_i^{1:\tau} = \{(x_i^t, y_i^t, u_i^t, v_i^t) | t = 1, 2, \cdots, \tau\}$, past scenes $I_i^{1:\tau} = \{m_i^t | t = 1, 2, \cdots, \tau\}$ while the future trajectories is $Y_i^{\tau:T} = \{(x_i^t, y_i^t) | t = \tau + 1, \cdots, T\}$.

Our goal is to learn the distribution of futures $p(Y_i^{\tau:T})$. To generate the distribution of future trajectories, we jointly model multiple ego-trajectories and their interactions with f. Therefore, the distribution is denoted as:

$$p(Y_i^{\tau:T}) = f(X_i^{1:\tau}, I_i^{1:\tau}, X_{1:N\backslash i}^{1:\tau}, I_{1:N\backslash i}^{1:\tau}; w^*), \tag{1}$$

where w^* are the parameters of the model we aim to learn. We denote the predicted future paths as $\hat{Y}^{\tau:T}$ whose distributions are learned from our model.

4 Methodology

4.1 Overall Architecture

Fig. 2 illustrates our encoder-decoder model. We leverage three sets of LSTMs to encode motion sequence, social interactions and to construct decoder. We utilizes a spatial-temporal graph to represent human motion and their interactions.

At any time instance, point elements of a graph are people characterized with real-world location and velocity while lines between two points are spatial edges represent their current interaction. In encoder, at any time instance, the location and the offset of all people are firstly embeded. Then agent and neighbors are connected to generate social features. The point-of-view images are encoded to get an abstract representation. Our model encode those important information and forecast discrete latent variables that indicate semantically meaningful modes of trajectories. Based on the latent variables, our decoder directly forecast sequence of futures.

4.2 Encoder

Embedding trajectories. As mentioned in Section 3, the agent i at time instance t is characterized with location (x_i^t, y_i^t) and velocity (u_i^t, v_i^t). We embed them respectively.

$$\begin{aligned}\tilde{f}_i^t &= \phi_1((x_i^t, y_i^t); \omega_1^*)\\ f_i^t &= \phi_2((u_i^t, v_i^t); \omega_2^*)\end{aligned} \tag{2}$$

where $\phi_1(\cdot)$ and $\phi_2(\cdot)$ are fully connected layers with ReLU non-linearity, ω_1^* and ω_2^* are the embedding weights. Trajectories' offset between two adjacent time instants are important features for trajectory prediction, which can stabilize the training the process and improve the model performance.

The hidden states from single person's LSTM at time instance t-1 are represented as h_i^{t-1}. h_i^{t-1} and $\tilde{f}_i^t$ are then utilized to construct spatial edges between agent and neighbors as $\mathcal{E}_j^t = \{[h_j^{t-1}, \tilde{f}_j^t] \odot [h_i^{t-1}, \tilde{f}_i^t] | j = 1, 2, \cdots, N \backslash i\}$. We assume all people in a shared scenario are allowed to interact. Most of the existing research construct spatial edges by embedding relative location between people. Different from those research, we not only take use of the location but also the latent states of sigle person's LSTM. We utilize an element wise operation instead of only embedding the relative current movement, which encourages the model to better capture the dynamic connection between agent and neighbor.

Social interactions.

$$\eta_j^t = \psi_{e_1}(\eta_j^{t-1}, \mathcal{E}_j^t; w_\eta^*) \tag{3}$$

where $\psi_{e_1}(\cdot)$ is another LSTM for social interactions and its weights w_η^* are shared between all people in a scenario. An attention mechanism is then applied to get the weights of neighbors on agent.

$$w_j^t = \frac{exp(\phi_\eta(\eta_j^t; \omega_\eta^*))}{\sum_{k=1}^{N \backslash i}(exp(\phi_\eta(\eta_k^t; \omega_\eta^*)))} \tag{4}$$

where $\phi_\eta(\cdot)$ is a fully connected layer, ω_η^* are the embedding weights. w_j^t is a variable-length alignment vector, whose size equals the number of neighbors N-1.

$$s_i^t = \sum_{j=1}^{N \backslash i}(w_j^t * \eta_j^t) \tag{5}$$

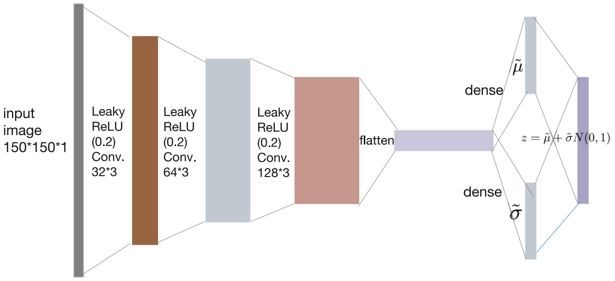

input image 150*150*1
Leaky ReLU (0.2) Conv. 32*3
Leaky ReLU (0.2) Conv. 64*3
Leaky ReLU (0.2) Conv. 128*3
flatten
dense
$\tilde{\mu}$
dense
$\tilde{\sigma}$
$z = \tilde{\mu} + \tilde{\sigma}N(0,1)$

Fig. 3. Scene Context Encoder.

Given w_j^t as weights, the social feature s_i^t is computed as weighted sum over all spatial edges. s_i^t stores information how the agent interact with others. Some previous works use a Euclidean distance-based ordering structure to select a fixed number of neighbors, which is not rational in highly interacted, dynamic scenes. Some research used Max or Average functions to pool neighbors which may lost individual uniqueness. Here, we allow all neighbors to interact and selectively sum their features through an attention mechanism, which fits the realistic circumstance and doesn't lose individual uniqueness.

Scene context. Instead of encoding the whole scene image, at any time instant t, we take point-of-view images of agents as input of scene context encoder. we transform the scene according to the walking direction of agents and get sequences of dynamic scene images. At any time instant, we learn an abstract, compressed representation of each observed scene image.

$$z_i^t = \phi_z(I_i^t; w_z^*) \tag{6}$$

where $\phi_z(\cdot)$ is Variational Autoencoder as depicted in Fig.3, z_i^t is the compressed latent vector of image I_i^t. I_i^t is a binary image where 0 means not walkable area and 1 means walkable area.

Trajectory encoder. We represent social state and scene representation for agent i at time t as s_i^t and z_i^t respectively, and then concatenate them with f_i^t .

$$h_i^t = \psi_{e_2}(h_i^{t-1}, [f_i^t, s_i^t, z_i^t]; w_h^*) \tag{7}$$

where $\psi_{e_2}(\cdot)$ is LSTM and its weights w_h^* are shared between all people in a scenario. By taking social state, scene representation and past trajectories as input, $\psi_{e_2}(\cdot)$ is able to learn:(1) features of past trajectories which indicate walking direction, speed, goal .etc, (2) how pedestrians interact with each other, (3) how pedestrians interact with static scene context. The hidden states h_i^t implies not only the walking patterns of pedestrians but also the futures.

4.3 Conditioned decoder

Latent modes. To capture the multi-modal futures, inspired by [18] , we learn discrete latent variables which indicate meaningful walking patterns of trajectories. We denote the discrete latent variables as α_i which take on k values. α_i doesn't have explicit meanings but implies the various walking patterns: (1)interactions modes, such as aggressive or mild, (2) walking goals, such as walking straight or turning left/right. α_i conditions on the features h_i^t of past trajectories.

$$H_i = \phi_h([h_i^0, h_i^1, \cdots, h_i^\tau]; w_h^*) \tag{8}$$

where $\phi_h(\cdot)$ is fully connected layer, w_h^* are embedding weights, $H_i = \{H_i^g | g = 0, 1, \cdots, k-1\}$.

$$prob_i^g = \frac{exp(H_i^g)}{\sum_{d=1}^{K} exp(H_i^d)} \tag{9}$$

where $prob_i$ sum up to 1.0. For each agent i, $\alpha_i \sim Multinoulli(prob_i)$.

Futures prediction. When predicting multi-steps of futures, our model sample once and get α_i at time τ and take α_i as input at each time instance. To predict multi-modes of futures, we can sample multiple times at τ to get α_i. But for each mode of prediction, α_i is consistent over all time steps.

$$\begin{aligned} fa_i &= \phi_\alpha(\alpha_i; w_\alpha^*) \\ H_i^t &= \psi_d(H_i^{t-1}, [\tilde{H}_i^{t-1}, fa_i]; w_d^*) \end{aligned} \tag{10}$$

where $\phi_\alpha(\cdot)$ is fully connected layer with weights w_α^*, $\psi_d(\cdot)$ is LSTM, its weights w_d^* are shared between all people in a scenario. For the prediction at time instant $\tau + 1$, $\tilde{H}_i^\tau$ equals to h_i^τ. For prediction at the rest time instants, $\tilde{H}_i^t$ equals to H_i^t. The futures $\hat{Y}_i^t$ at time instant t are given by: $\hat{Y}_i^t \sim N^2(\mu_i^t, \sigma_i^t)$. Means $\mu_i^t = (\mu_x, \mu_y)_i^t$ and standard deviation $\sigma_i^t = (\sigma_x, \sigma_y)_i^t$ are obtained by applying fully connected layers $\phi_\mu(\cdot)$ and $\phi_\sigma(\cdot)$ to H_i^t respectively.

4.4 Training

The loss function is usually designed to compute negative log-likelihood of future trajectories.

$$\mathcal{L} = -\sum_{t=\tau}^{T-1} log(\tfrac{1}{N} \sum_{i=1}^{N} p(\hat{Y}^{t+1} | \mu_i^t, \sigma_i^t, \alpha_i)) \tag{11}$$

To train the whole model, we first train the Variational Autoencoder to learn the scene context encoder of scene images in an unsupervised manner. Then we train the whole model end to end by minimizing the loss function.

5 Experiments

In this section, the proposed model is evaluated on two publicly available datasets: UCY[19] and ETH[20]. The two datasets contain 5 sets, which are

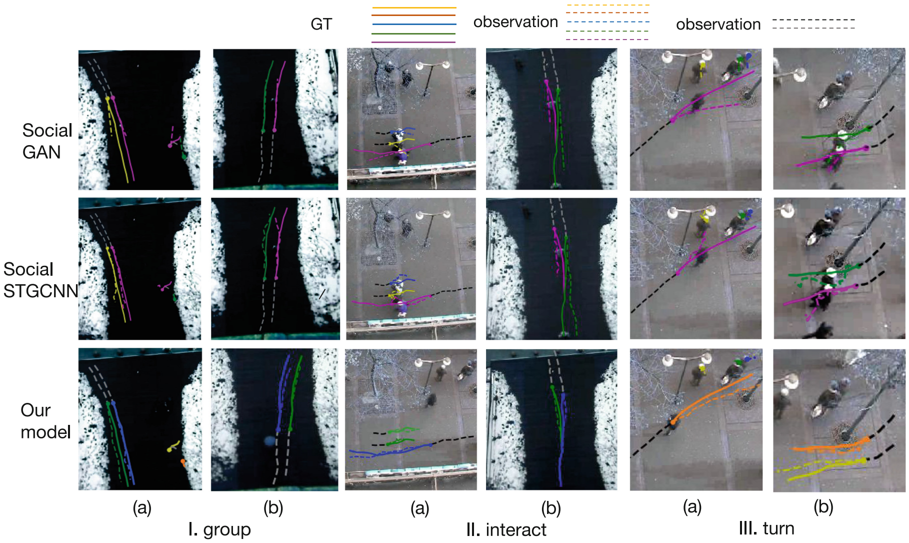


Fig. 4. Comparison between the proposed model and two recent baselines, Social GAN and Socila STGCNN

UCY-zara01, UCY-zara02, UCY-univ, ETH-hotel, ETH-eth in 4 crowded scenarios with totally 1536 trajectories. We firstly preprocess those two datasets by resampling them as 2.5fps and transforming the coordinates of people to world coordinates in meters.

Implementation Details. The experiments are implemented using Pytorch under Ubuntu 16.04 LTS with a GTX 1080 GPU. The size of hidden states of LSTM is set to 128. The embedding layers are composed of a fully connected layer with size 128. The batch size is set to 8 and all the methods are trained for 200 epochs. We clip the gradients of LSTM with a maximum threshold of 10 to stabilize the training process. k of latent modes is set as 5. As mentioned before in 4.4, we first train the Variational Autoencoder using Adam optimizer with learning rate 0.001 to get scene context encoder. We didn't fine-tune scene context encoder when training entire model. For training the entire model, the optimizer Adam with learning rate 0.001 is used.

Evaluation Approach. The proposed model is trained and tested on the two datasets with leave-one-out approach: trained on four sets and tested on the remaining set. We observe the trajectories for 8 timesteps (3.2 sec) and show prediction results for 12 timesteps (4.8 sec). To evaluate the performance, we compare our method with other state-of-the-art models on two generally used metrics.

1. Average displacement error (ADE): average L2 distance over all prediction results and ground truth. ADE measures average error of the predicted trajectory sequence.

2. Final displacement error (FDE): distance between prediction result and ground truth at final timestep. FDE measures the error "destination" of the prediction.

Baselines.The proposed model is compared with the following baselines.

1. Linear. The second order Kalman Filter, which is modeled based on position, velocity, acceleration, is used as the linear method.
2. LSTM. Human motion is modeled without considering human interaction. Offset is used as input[21].
3. Social LSTM. This method models human interactions by pooling hidden states of spatially proximal motion sequences [6].
4. Social GAN. This approach captures the multi-modality of future trajectory prediction, which contains a RNN based encoder-decoder generator and a RNN-based encoder discriminator. We consider one variant of Social GAN: best results of sampling 20 times[15].
5. Sophie. This is a GAN-based model which takes into account both social and physical interactions to make more realistic predictions. We consider one variant of Sophie: best results of sampling 20 times[8].
6. PIF. An end-to-end, multi-task learning system utilizing visual features about human behavioral information and interaction to forecast future trajectories [22].
7. Social BiGAT. This method uses a generator, two discriminators (local discriminator and global discriminator) and a latent noise encoder to construct a reversible mapping between predicted paths and learned latent features of trajectories. We consider one variant of Social BiGAT: best results of sampling 20 times[9].
8. Social STGCNN. The method substitutes aggregation methods by modeling the interactions as a graph.

Ablation study. To explain how our model works, we also represent results of various versions of our models in an ablative setting by $V1$: our model that doesn't consider static scene context, $V2$:our model that doesn't consider social interactions, $V3$: our entire model that generates results from one mode, $V4$:our entire model that generates results from two modes, $V5$:our entire model that generates results from three modes, $V6$:our entire model that generates results from four modes, $V7$:our entire model that generates results from five modes.

5.1 Quantitative Evaluation

ETH and UCY. We compare our model to various baselines in Table 1, reporting the average displacement error (ADE) and final displacement error (FDE) for 12 timesteps of human movements. In general, the linear method performs worse than the other methods because it is limited to modeling the social context or multi-modality of human motion. Social LSTM only achieves an accuracy similar to that of LSTM, although it is trained with synthetic data and then fine-tuned

Table 1. Quantitative results of baselines versus our method across datasets for predicting 12 future timesteps(4.8 sec) given 8 timesteps observations(3.2 sec)(lower is better). The results of Social LSTM, Social GAN are from [15], the results of Sophie, Social BiGAT, Social STGCNN are from [8,9,13] respectively. The results of PIF are from [22].

Method	Note	Evaluation (ADE(m)/FDE(m))					
		ETH-eth	ETH-hotel	UCY-univ	UCY-zara01	UCY-zara02	AVG
Linear	kalman filter	1.65/2.84	0.99/1.70	0.86/1.51	0.83/1.44	0.54/0.96	0.97/1.69
LSTM	offset is input	0.71/1.40	1.15/2.09	0.72/1.49	0.48/0.98	0.38/0.77	0.69/1.35
Social LSTM	social pooling	1.09/2.35	0.79/1.76	0.67/1.40	0.47/1.00	0.56/1.17	0.72/1.54
Sophie	20 samples	0.70/1.43	0.76/1.67	0.54/1.24	**0.30**/0.63	0.38/0.78	0.54/1.15
Social GAN	20 samples	0.72/1.29	0.48/1.01	0.56/1.18	0.34/0.69	0.31/0.65	0.48/0.96
PIF	20 samples	0.73/1.65	0.30/0.59	0.60/1.27	0.38/0.81	0.31/0.68	0.46/1.00
Social BiGAT	20 samples	0.69/1.29	0.49/1.01	0.55/1.32	**0.30**/0.62	0.36/0.75	0.48/1.00
Social STGCNN	20 samples	0.64/1.11	0.49/0.85	**0.44**/0.79	0.34/**0.53**	**0.30/0.48**	0.44/0.75
Our model-V1	without scene	0.79/1.16	0.53/0.95	0.82/1.30	0.54/0.81	0.66/1.21	0.67/1.09
Our model-V2	without social	0.66/1.16	0.56/1.06	0.74/0,95	0.49/0.75	0.61/1.11	0.61/1.00
Our model-V3	1 mode	0.58/1.11	0.30/0.58	0.62/0.78	0.40/0.65	0.51/1.02	0.48/0.83
Our model-V4	2 modes	0.54/1.02	0.28/0.52	0.59/0.74	0.37/0.64	0.48/0.94	0.49/0.68
Our model-V5	3 modes	0.53/0.98	0.24/0.44	0.57/0.75	0.36/0.63	0.47/0.91	0.45/0.77
Our model-V6	4 modes	0.51/0.97	0.23/0.43	0.53/0.72	0.40/0.89	0.46/0.89	0.43/0.78
Our model-V7	5 modes	**0.51/0.95**	**0.23/0.41**	0.50/**0.69**	0.41/0.83	0.44/0.86	**0.42/0.75**

on the benchmarks[15]. LSTM use offset as the input, which stabilizes the learning process and improves the performance. Sophie, Social GAN, Social BiGAT and PIF, Social STGCNN capturing the uncertainty of long-term movement all achieve better results than Social LSTM and basic LSTM.

Our first model that doesn't consider scene context and second model that doesn't consider social interactions, don't obtain good results, which indicate that scene context and social interactions are critical for long-term trajectory prediction. Interestingly, our second model outperforms the first one, which also imply that scene information might be more important than social interactions for long-term prediction. The rest models take into account past trajectories, social interactions, scene context. The model$V7$ that generates results from the entire five modes better than other models, which indicates that our model is able to capture the uncertainty and learn multiple modes of futures. Our entire model achieves good results by comparing with other methods, especially over ETH-eth and ETH-hotel and UCY-univ.

5.2 Qualitative Evaluation

To investigate the ability of our model to forecast more accurate and reasonable futures , we visualize three sets of scenarios from ETH-hotel and ETH-eth and compare the predictions of two state-of-the-art models, Social GAN and Social STGCNN, to that of our model (Fig.4). We plot the best results of 20 samples for Social GAN and Social STGCNN. The results from our model are the best results of 5 modes. In the first set of scenarios group, (a) contains two people walking along the road and two people standing together. Our model predict the future trajectories better than Social GAN and Social STGCNN. The two baselines wrongly forecast the walking speed for both walking group and standing group. In (b), the group are turning a corner. Social STGCNN predict forecasts the group will walking straight. The second set of scenarios show the prediction results when people are interacting each other. II (a) shows three people with different walking speed and directions are interacting with each other. II(b) shows a pedestrian is overtaking another. Our model outperforms the two baselines by better forecasting the waking direction and speed when interaction happens, which indicates that the proposed model is able to better capture the interaction among people. In the third set (a), even we show the best results of 20 samples, Social GAN and Social STGCNN still wrongly predict the walking goals. In (b), two people are turning in an aggressive way. Given the observation, our model predict better results than Social GAN and Social STGCNN by better capturing their walking patterns.

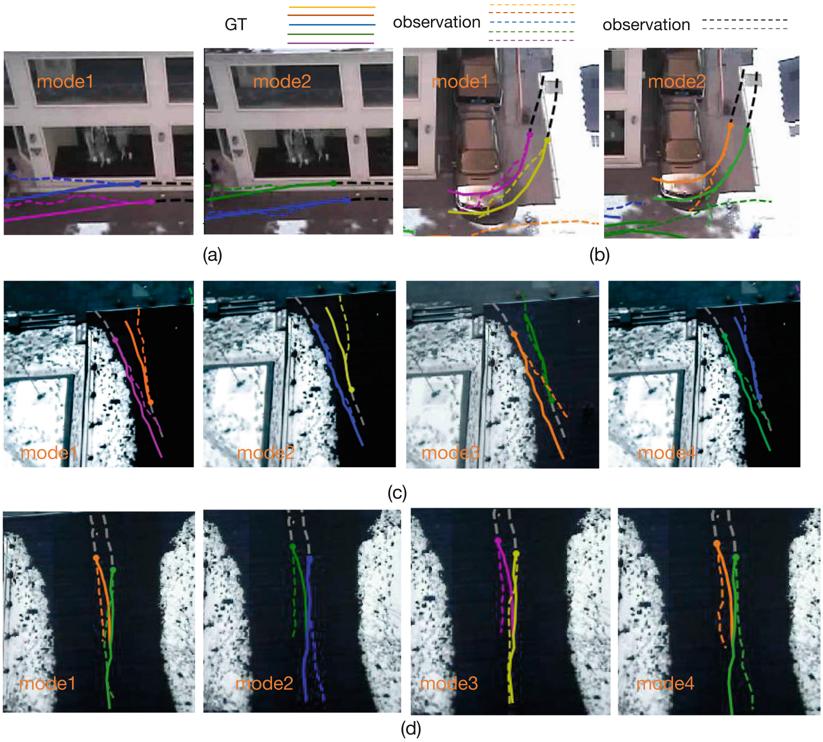


Fig. 5. Multiple socially and physically acceptable futures.

To further illustrate that our model is able to forecast both socially and physically acceptable multi-modal futures, we show four scenarios where multiple futures are forecasted in Fig.5. In (a) , the pedestrians will either walk into the store or walk along the road. In (b), two people are turning a corner. They will behave differently to turn the corner. For (a) and (b), two modes of futures which are physically acceptable are forecasted. In(c) and (d), intense social interactions are happened among people and four modes of possible futures are illustrated. Our model is able to forecast multiple meaningful modes of futures, such as aggressive or mild interactions.

6 Conclusion

We propose a unifying trajectory prediction model, which jointly takes into account multiple interacting movements, dynamic point-of-view scene images, and predicts multi-modal socially and physically acceptable futures. Our model utilizes three set of LSTMs and contains an encoder and a decoder. In encoder, we learn a joint representation of past trajectories, dynamic people-people and people-scene interactions through a social mechanism and scene context encoder. Our social mechanism scales effectively to any number of pedestrians and learns to model dynamic interactions. The scene context encoder learns compressed features via VAE. The latent variables, a fixed-length vector, learns to indicate the walking or interacting patterns of trajectories. The latent variables condition on the joint representation and keep consistent for sequence prediction. Then decoder forecasts multi-modes of future by sampling from the the latent variables. Our model outperforms other state-of-the-art models over a number of publicly available datasets. We also demonstrate that it is able to provide more socially and physically acceptable distributions by qualitatively analyzing the performance of our model under scenarios such as group meeting, collision avoidance comparing to other baselines. Future work will extend our model to forecast futures of multiple objects, such as bicycles, cars, and test the model performance with more benchmarks.

References

1. Kris M Kitani, Brian D Ziebart, James Andrew Bagnell, and Martial Hebert. Activity forecasting. In *European Conference on Computer Vision*, pages 201–214. Springer, 2012
2. Vasiliy Karasev, Alper Ayvaci, Bernd Heisele, and Stefano Soatto. Intent-aware long-term prediction of pedestrian motion. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2543–2549. IEEE, 2016
3. Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. Predicting the next location: A recurrent model with spatial and temporal contexts. In *Thirtieth AAAI conference on artificial intelligence*, 2016
4. Namhoon Lee, Wongun Choi, Paul Vernaza, Christopher B Choy, Philip HS Torr, and Manmohan Chandraker. Desire: Distant future prediction in dynamic scenes with interacting agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 336–345, 2017

5. Hang Su, Jun Zhu, Yinpeng Dong, and Bo Zhang. Forecast the plausible paths in crowd scenes. In *IJCAI*, volume 1, page 2, 2017
6. Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 961–971, 2016
7. Ridel, D., Deo, N., Wolf, D., Trivedi, M.: Scene compliant trajectory forecast with agent-centric spatio-temporal grids. IEEE Robotics and Automation Letters **5**(2), 2816–2823 (2020)
8. Amir Sadeghian, Vineet Kosaraju, Ali Sadeghian, Noriaki Hirose, Hamid Rezatofighi, and Silvio Savarese. Sophie: An attentive gan for predicting paths compliant to social and physical constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1349–1358, 2019
9. Vineet Kosaraju, Amir Sadeghian, Roberto Martín-Martín, Ian Reid, Hamid Rezatofighi, and Silvio Savarese. Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. In *Advances in Neural Information Processing Systems*, pages 137–146, 2019
10. Pu Zhang, Wanli Ouyang, Pengfei Zhang, Jianru Xue, and Nanning Zheng. Sr-lstm: State refinement for lstm towards pedestrian trajectory prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12085–12094, 2019
11. Fernando, T., Denman, S., Sridharan, S., Fookes, C.: Soft+ hardwired attention: An lstm framework for human trajectory prediction and abnormal event detection. Neural Netw. **108**, 466–478 (2018)
12. Xiaodan Shi, Xiaowei Shao, Zipei Fan, Renhe Jiang, Haoran Zhang, Zhiling Guo, Guangming Wu, Wei Yuan, and Ryosuke Shibasaki. Multimodal interaction-aware trajectory prediction in crowded space. In *AAAI*, pages 11982–11989, 2020
13. Abduallah Mohamed, Kun Qian, Mohamed Elhoseiny, and Christian Claudel. Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14424–14432, 2020
14. Ashesh Jain, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 5308–5317, 2016
15. Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2255–2264, 2018
16. Osama Makansi, Eddy Ilg, Ozgun Cicek, and Thomas Brox. Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7144–7153, 2019
17. David Ha and Jürgen Schmidhuber. World models. *arXiv preprint* arXiv:1803.10122, 2018
18. Charlie Tang and Russ R Salakhutdinov. Multiple futures prediction. In *Advances in Neural Information Processing Systems*, pages 15424–15434, 2019
19. Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. Crowds by example. In *Computer graphics forum*, volume 26, pages 655–664. Wiley Online Library, 2007
20. Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th International Conference on Computer Vision*, pages 261–268. IEEE, 2009

21. Stefan Becker, Ronny Hug, Wolfgang Hübner, and Michael Arens. An evaluation of trajectory prediction approaches and notes on the trajnet benchmark. *arXiv preprint* arXiv:1805.07663, 2018
22. Junwei Liang, Lu Jiang, Juan Carlos Niebles, Alexander G Hauptmann, and Li Fei-Fei. Peeking into the future: Predicting future person activities and locations in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5725–5734, 2019

Hierarchical Ensemble of AutoEncoder for Restoration of Images Corrupted by Cumulative Combination of Noise

Sayarnil Ganguly, Sanjana Reddy Katham, Sanyam Agrawal, Soumen Sinha, and Rahul Roy(✉)

Department of Computer Science Engineering, Mahindra University, Hyderabad, Telangana, India
{soumen20ucse179,rahul.roy}@mahindrauniversity.edu.in

Abstract. Denoising images corrupted with multiple types of noise is significantly challenging due to their variable noise distribution. These numerous types of noise have a cumulative effect on the image. Therefore, considering only a single or multiple distribution of noise would be ineffective for restoration. Moreover, estimation of the noise is difficult where there is heterogeneity in the combination of the noise. Hence in this paper, we propose a sequence of autoencoders to progressively restore the noisy images. Each autoencoder specializes in the restoration of images affected by a particular combination of noise. These trained autoencoders are arranged in a state-space graph such that all possible combinations of noises can be screened by the autoencoder or their ensembles. The root of the state space graph is the N^{th} level autoencoder as it is exposed to all kinds of noise. In contrast, the autoencoder at the leaf level is exposed to one type of noise. During restoration of images, the obtained state space tree is searched using a heuristics search algorithm to find the near-optimal ensemble of autoencoders which can denoise the combinations of noises present in an image. Experiments are conducted with diverse combinations of noises to demonstrate the efficacy of the proposed approach. Furthermore, the proposed method is compared with various state-of-the-art approaches. It was observed that the proposed approach had significant efficiency in handling the combination of noise and outperforms various state-of-the-art techniques.

Keywords: Image Denoising · N-Layered autoencoder tree · heuristic search · Image Restoration

1 Introduction

Image denoising remains a fundamental challenge in digital image processing, aimed at removing noise while preserving essential structural details and image

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78383-8_18.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 270–283, 2025.
https://doi.org/10.1007/978-3-031-78383-8_18

quality. Traditional techniques, such as filtering [6], though foundational, often fail to maintain finer image details and usually assume a single noise type, which is rarely the case in practical scenarios. With the evolution of deep learning, more sophisticated denoising methods have emerged. Deep autoencoder-based architectures, for instance, have shown promising results in efficiently restoring images with unknown noise models [19]. However, these often struggle with images corrupted by multiple types of noise or higher noise levels. In response to these challenges, the focus has shifted towards using Generative Adversarial Networks (GANs), which offer robust mechanisms for handling a broader spectrum of noise types and intensities. The use of GANss for image denoising has been explored by many researchers, where a generator creates denoised images to confuse a discriminator into treating them as real, un-corrupted images. Variants of GANs such as Cyclic GAN [8], Contextual GAN [22], and Residual GAN [7] enhance the technique of denoising for better performance. Recent advances have also explored attention mechanisms and visual transformers for image denoising. Wang et al. [18] proposed a channel and space attention neural network for image denoising, focusing on the adaptive processing of noisy areas of an image. In recent times, researchers have begun to develop more versatile models. For example, Cheng et al. [9] combined deep hybrid learning models with innovative optimization algorithms such as the Self-Improved Orca Predation Algorithm which demonstrated enhanced denoising capabilities, particularly in preserving image details and computational efficiency. Furthermore, using sparse representations in deep convolutional networks by Bian et al. [2] has provided significant improvements in standard denoising benchmarks.

Previous methods for reducing noise in images, like deep autoencoders and Generative Adversarial Networks (GANs), have brought some important improvements. However, they often struggle when dealing with images that have different types of noise mixed or very high levels of noise. While deep autoencoders are good at learning from data, they sometimes lose small but important details in images with complex noise patterns. Similarly, GANs face difficulties in distinguishing between various noise types and might not always accurately restore the original quality of highly corrupted images. However, if learn progressively the cumulative distribution of noise with a series of autoencoders, we can overcome the issue of handling multiple types. But, we need an efficient arrangement of these autoencoders such that they can handle arbitrary combinations of noise. Vahdat and Kautz's work [16] on deep hierarchical variational autoencoders showed the potential of layered learning approaches in complex image tasks. This motivated us to explore the autoencoders to learn the cumulative noise models and arrange them in a hierarchy for the restoration of noisy images. Our work introduces a novel hierarchical approach to image denoising using a combination of autoencoders. Our primary contribution is a hierarchical ensemble of autoencoders. The autoencoder ensemble was trained sequentially to target specific noise types and levels, thereby progressively enhancing image quality. By employing a state-space tree representation and a uniform cost search algorithm, we optimize autoencoder combinations based on the heuristic cost function, allowing for a tailored approach to address various noise types within an image. The proposed approach was evaluated on an incremental cumulative combination of single and multiple noise models. The results of the proposed approach were compared with four state-of-the-art approaches using the peak signal-to-noise ratio (PSNR) and structural similarity matrix (SSIM). The results obtained depict the dominance of the proposed approach over other state-of-art approaches.

The rest of the article is organized into the following sections. Section 2 discusses the related work in this area. Following this, in Section 3 we discuss the proposed methodology. In Sections 4 and 5, experimental setup and result analysis are presented. Finally, a conclusion with future perspectives is provided in section 6.

2 Related Work

Autoencoders, a type of neural network, are frequently utilized for image denoising tasks. They work by learning to compress (encode) the input image into a lower-dimensional space and then reconstruct (decode) it back to the original space while attempting to retain the essential features and discard the noise. Vincent et al. [17] introduced the concept of stacked denoising autoencoders, which utilize a deep network with a local denoising criterion to learn useful representations for noise reduction. This approach has set a foundational technique that has been widely adopted in subsequent research. Additionally, the exploration of deep convolutional autoencoders by Gondara [5] has demonstrated their effectiveness in handling complex noise patterns in medical images, providing a robust framework for improving denoising outcomes. The non-linear activation free network (NAFNet) proposed by Chen et al. [4] optimizes for computational efficiency but under performs with varied noise types due to its linear structure. Meanwhile, the Restoration Transformer (Restormer) proposed by Zamir et al. [21] excels in handling global dependencies in images but faces scalability issues with high-resolution or real-time applications due to high computational demands. Most of these models address a single type of noise but Remez et al. [11] made a significant contribution to handling two kinds of noise, namely Gaussian and Poisson noise. The three state-of-the-art algorithms, while advanced, exhibit a lot of limitations in adaptability and efficiency across complex noise conditions which our proposed hierarchical autoencoder approach seeks to address by enhancing flexibility and efficiency in handling diverse noise conditions.

3 Proposed Method

We propose a novel hierarchical approach to denoising images degraded with a cumulative combination of noise using a sequence of autoencoders.

The mathematical model cumulative noise degradation process is defined as follows: let us consider the observed image $I \in R^2$ which is degraded by a cumulative combination of p external noise $\eta_1, \eta_2, \eta_3, ...\eta_p$. Mathematically,

$$I = ((((f \circ \eta_1) \circ \eta_2) \circ \eta_3) ... \circ \eta_p) \tag{1}$$

where f is the unobserved clean image. The operator $\circ$ denotes the noise degradation process.

We develop a hierarchical ensemble of autoencoders, each trained sequentially to target specific combinations of noise types and levels, thereby progressively enhancing image quality. It may be noted here that the autoencoder is trained with a cumulative combination of noise. Thus during denoising, there is a noise requirement for the prior combination and type of noise as every complex noise is a cumulative combination of simple types of noise. For example, the low-illuminated image with jitters can be a combination of Rayleigh and Gaussian noise. we optimize autoencoder combinations based on a cost function by employing a state-space tree representation and a heuristic search algorithm, As during denoising, the set of the autoencoder is searched in the state space, the prior information of the order of the noise is not required for denoising. Our proposed methodology encompasses two main stages: Sequential training of multiple autoencoders on images corrupted with a cumulative noise distribution and utilizing an N-layered tree to determine optimal denoising strategies. Each autoencoder in our sequence is specialized for a specific cumulative combination of noise, enhancing adaptability across diverse noise conditions.

3.1 Encoder-Decoder Architecture

The encoder-decoder model constitutes L layers of encoder and decoder block. The architecture of the proposed method is shown in Figure 1. The encoder block (E) takes an input image I of dimension $C \times H \times W$ where C represents the number of channels, H and W are the height and width of the image respectively. Details of each component are described in the following subsections.

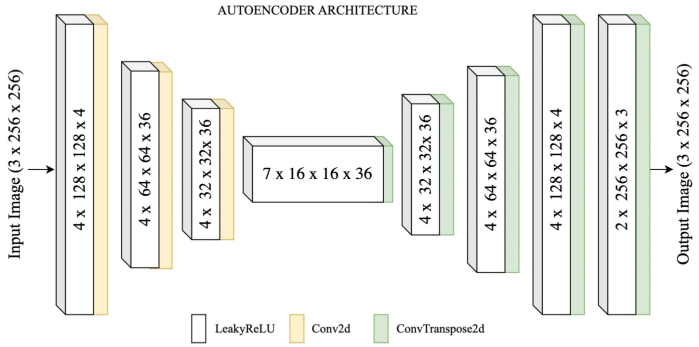


Fig. 1. Autoencoder Architecture

Encoder Design The encoder block consists of convolutional neural network (CNN) layers, enhanced by Leaky ReLU activation functions, to condense noisy input images into a compact latent representation, capturing essential information while removing noise. In each convolutional layer, there are q number of filters of size $k \times k$ that capture essential features in the image. Batch normalization is applied after some convolutional layers to stabilize learning and normalize the features. The LeakyReLU activation function allows for a small, non-zero gradient when the unit is not active, which helps to maintain a flow of gradients during training. The encoder progressively down-samples the image, increasing the number of feature channels while reducing spatial dimensions. Mathematically it can be written as,

$$y_l = \text{LeakyReLU}(\text{Conv2D}(y_{l-1})) \tag{2}$$

where y_{l-1} is the input to the l^{th} layer of the encoder. It may be noted that initially y_l is set to the input image I.

Decoder Design The decoder mirrors the encoder's structure but in reverse, using transposed convolutional layers to upsample the feature maps back to the original image size. The use of BatchNorm2d and LeakyReLU continues in the decoder to maintain consistency with the encoder's processing. The reconstruction starts with the most compressed feature representation and progressively increases the spatial dimensions while reducing the number of feature channels. The final layer typically uses a Sigmoid activation function ($\sigma(\dots)$) to output the pixel values of the reconstructed image, ensuring that they are in the same range as the input image, which is usually between 0 and 1 for normalized image data. The decoder operation is represented as

$$y_{l-1} = \text{LeakyReLU}(\text{Conv2D}^T(y_l)). \tag{3}$$

where y_l is the input to the $l-1^{th}$ layer of the decoder. Here, it is to be noted that the decoder has reverse indexing as the encoder. The final output is obtained as

$$\hat{I} = \text{LeakyReLU}(\text{Conv2D}^T(y_1)). \tag{4}$$

where $\hat{I}$ represent the reconstructed image.

3.2 Training

The training procedure for our autoencoder series follows a sequential methodology where each autoencoder is exposed to a cumulatively increasing noise complexity. This strategy aims to enhance the specialization of each autoencoder in the sequence, by sharing the learned noise from the previous layer. Figure 2 depicts the schema for autoencoder training.

The sequence initiates with a clean image I. An initial noise N_1, such as Gaussian noise, is added to generate a noisy image I'. The first Autoencoder (AE 1) is then trained with I' as the input and I as the target, adapting to

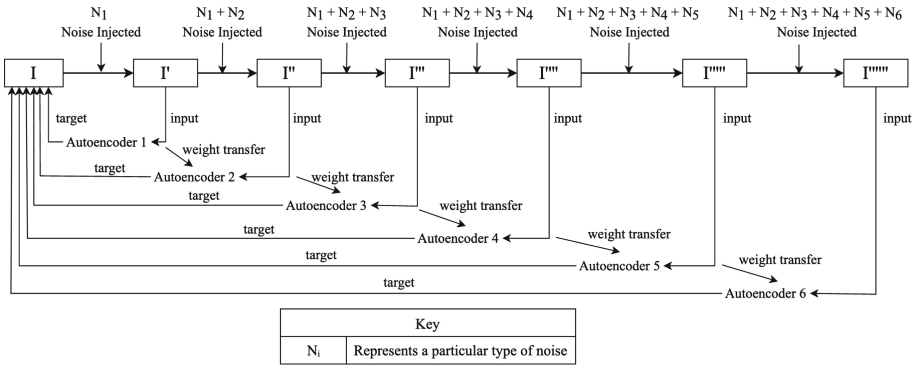


Fig. 2. Schematic representation of the sequential training process with cumulative noise exposure and weight transfer between subsequent autoencoders.

remove the introduced noise. After training, the weights of Autoencoder 1 (AE 1) are used to initialize the weights of Autoencoder 2 (AE 2) a process known as weight transferring. This process is repeated with an additional noise type N_2 added to the image, creating I''. All autoencoders from (AE 1) to (AE 6) have I as their output. This pattern of noise addition and weight transferring continues during the training of the sequence of autoencoders.

As the training process advances, the noise complexity is incremented by introducing additional noise types, such as Poisson noise (N_2) Colored Gaussian noise (N_3), and so on. Each successive autoencoder is trained to denoise a cumulative combination of noises, with AE2 focusing on $N_1 + N_2$ and AE3 on $N_1 + N_2 + N_3$. This cumulative training strategy ensures that each autoencoder develops a specialization for a distinct noise combination, thus progressively improving the system's capability to denoise images afflicted with an increasingly complex mixture of noise types.

$$MSE(I, \hat{I}) = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} (I(i,j) - \hat{I}(i,j))^2. \tag{5}$$

The Mean Squared Error (MSE) (5) is utilized as the loss function to optimize the autoencoders where i , j represent pixel position in image I and $\hat{I}$.

3.3 N-Layered Tree for Optimal Autoencoder Sequence Determination

The N-layered tree structure is pivotal in determining the most efficient combination of autoencoders to address the multiple noise profiles in images. It is configured as a binary tree and systematically navigates through decision points, where each node represents a binary decision to either engage or bypass a specific autoencoder based on the reconstructed image's total variation. Mathematically this is defined as

$$cost = \sum_{i=1}^{M}\sum_{j=1}^{N}(\hat{I} - I)^2 + \parallel \nabla \hat{I} \parallel_2 \tag{6}$$

In Eq. 6, the first part constrains the reconstructed image from deviating from the original image, and the second part represents the total variation which ensures the piecewise smoothness in the reconstructed image.

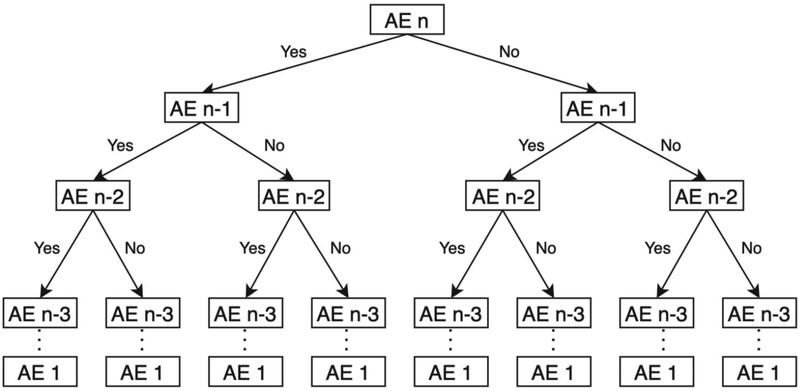


Fig. 3. Schematic of the N-Layered Tree illustrating decision paths for selecting autoencoder sequences.

As evident from Figure 3, the tree's root contains the most comprehensive autoencoder, aimed to tackle all noise types. We begin with the N^{th} Autoencoder at the root of the tree, which branches out into two paths: one utilizing the N^{th} Autoencoder and one without it. It then reaches the $N-1^{th}$ autoencoder, which further branches out to two branches, with one utilizing the $N-1^{th}$ autoencoder and another without it. This process continues recursively until we reach Autoencoder 1 at the leaf level of the tree. A bifurcation at each level, indicated by 'Yes' or 'No,' denotes the decision to incorporate the autoencoder at that particular layer for denoising purposes. Following the uniform cost search approach, the autoencoder tree is traversed from root to leaf (goal node). The effectiveness of each autoencoder and path through the tree is quantitatively assessed by calculating the cost function defined in Equation 6. The reconstructed outputs from autoencoders are propagated down the tree. The autoencoders in the lowest cost path is included in the optimal sequence of autoencoders. This optimal sequence signifies the most effective combination of autoencoders for a given noise profile.

It is to be noted the uniform cost search is guaranteed to give an optimal path to the goal node [13]. Thus the generated sequence of autoencoder will be optimal for a particular cumulative combination of noise profiles.

4 Experimental Setup

4.1 Dateset Description

The images used in the experimentation are taken from the Kaggle house room dataset[1]. The images are corrupted by adding noise generated using various noise functions. This enabled the generation of diverse noise patterns onto a single image. The noise types considered in the experiment include Gaussian noise, commonly found in digital images due to electronic circuit noise[3] and Poisson noise, which arises from the stochastic nature of photon counting[10]. Additionally, our simulations incorporate Gaussian noise, Poisson noise, Coloured Gaussian noise[15], Speckle noise[14] and Salt-pepper noise[1] as well, which affects the overall brightness of an image. We also include Uniform noise[24], altering the pixel intensities evenly and Speckle noise. It may be noted here that we considered simulating our dataset because most of the available standard datasets contain one or a maximum of two types of noise conditions which limits us from rigorously testing the effectiveness of the model when there are multiple source noises (greater than 2) which degrades the quality of image drastically. The dataset is partitioned into training and testing subsets (Table. 1), with 60% of the images earmarked for training purposes, and the remaining 40% reserved for testing.

Table 1. Train Test Split

Image	Percentage	Sample Size
Clean Train	60 percent	3000
Clean Test	40 percent	2000
Noisy Train	60 percent	3000
Noisy Test	40 percent	2000

4.2 Model Parameters

The network has four encoder layers, three decoder layers, and one output layer. The number of filters considered for the encoder block are [128,64, 32, 16] and the filter size was set to 3×3 The learning rate utilized in the training process is set to 0.001. The loss function employed throughout training is the Mean Squared

[1] https://www.kaggle.com/datasets/robinreni/house-rooms-image-dataset/data

Error (MSE) loss. To optimize the model parameters, the Adam optimizer is utilized. The training data is divided into batches of size 16 to facilitate efficient computation. Each autoencoder undergoes training for a total of 100 epochs, ensuring that the model iterates over the entire dataset multiple times. The input considered for experimentation are 3-channel coloured RGB images. It may be noted that the images are of variable size. So, we fix the input image size to 256 × 256.

5 Result Analysis

5.1 Model Comparison

Our comprehensive experimental evaluation compares the Hierarchical Ensemble of Autoencoders (the proposed model) against various state-of-the-art models namely Restormer [20], NAFNet [4], FFDNet [23] and Class aware denoising model [12]. Using a suite of metrics like Mean Absolute Error (MAE), Peak Signal-to-Noise Ratio (PSNR), and Structural Similarity Index Measure (SSIM) we present a rigorous assessment across various noise levels. We would like to highlight that this is an evolving problem where few or no models exist where such multiple noises are handled in one shot. Thus we considered the model that has shown potential in handling two or more kinds of noise.

In Table 2 we showcase results for the accumulation of noise up to 10 times. Looking closely at the table data, we see that the DCHEA model consistently outperforms the other models in both PSNR and SSIM values as the types of noises increase. Our model(DHCEA) starts a PSNR of **37.97** and an SSIM of **0.969** in the first scenario and continues to attain high PSNR/SSIM values till the 10th noise addition. This suggests that DCHEA is quite reliable in reducing noise in images. In comparison, models like NAFNET, FFDNet, and Restormer start with lower PSNR/SSIM scores and decrease further as noise is added. It may be noted that the models are designed to handle a single type of noise and hence there is a significant deterioration in performance as the noise type and level increases.

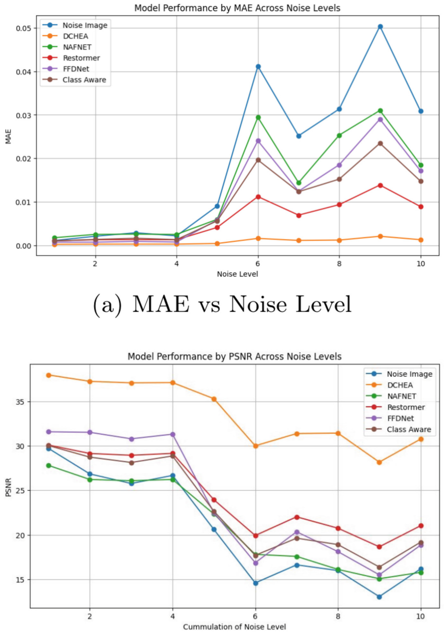
5.2 Ablation study

In this ablation study, the impact of incrementally adding homogeneous noise is quantified using PSNR and SSIM metrics.

For Gaussian noise, as we keep accumulating Gaussian noise with varied standard deviations, there's a notable deterioration in image quality. Image reconstructed using our model DHCEA obtains high PSNR/SSIM scores making it efficient in handling homogeneous noise as well. Similarly, for Poisson noise, the original images experience a significant degradation, with PSNR reducing from 28.95 to 21.62 and SSIM from 0.863 to 0.618 as there is an accumulation of noise. However, the reconstructed images again maintain remarkably high PSNR values, all around 37, and consistently high SSIM values. Results for other noises are included in the supplementary materials.

Table 2. DHCEA Model Comparison with various states of the model given by PSNR/SSIM values. Results for the best performance are highlighted in bold.

Cumulation	Noise Image	**DCHEA**	NAFNET [4]	Restormer [20]	FFDNet [23]	Class Aware [11]
of Noise	PSNR / SSIM	PSNR / SSIM	PSNR / SSIM	PSNR / SSIM	PSNR / SSIM	PSNR / SSIM
1	29.71 / 0.795	**37.97 / 0.969**	27.80 / 0.895	30.09 / 0.923	31.58 / 0.955	30.03 / 0.864
2	26.84 / 0.715	**37.25 / 0.965**	26.22 / 0.851	29.14 / 0.901	31.52 / 0.955	28.73 / 0.794
3	25.77 / 0.702	**37.08 / 0.965**	26.08 / 0.844	28.94 / 0.893	30.79 / 0.948	28.13 / 0.780
4	26.67 / 0.712	**37.11 / 0.965**	26.22 / 0.850	29.15 / 0.901	31.31 / 0.955	28.86 / 0.792
5	20.60 / 0.599	**35.30 / 0.950**	22.34 / 0.769	23.94 / 0.824	22.49 / 0.878	22.65 / 0.682
6	14.59 / 0.342	**30.00 / 0.887**	17.80 / 0.565	19.92 / 0.578	16.86 / 0.535	17.69 / 0.409
7	16.62 / 0.411	**31.38 / 0.920**	17.57 / 0.469	22.02 / 0.663	20.34 / 0.745	19.62 / 0.486
8	15.98 / 0.415	**31.43 / 0.911**	16.11 / 0.455	20.74 / 0.668	18.11 / 0.638	18.90 / 0.486
9	13.05 / 0.268	**28.17 / 0.869**	15.06 / 0.301	18.65 / 0.490	15.50 / 0.437	16.38 / 0.332
10	16.17 / 0.392	**30.76 / 0.905**	15.78 / 0.434	21.02 / 0.616	18.83 / 0.626	19.17 / 0.460



(a) MAE vs Noise Level

(b) PSNR vs Noise Level

Fig. 4. Plots for PSNR and MAE as noise level changes

This ablation study effectively demonstrates the robustness of the reconstruction method used against increasing noise levels for homogeneous noise as well.

Table 3. DHCEA's performance on the cumulation of homogenous noise (Gaussian and Poisson)

Cumulation of	Gaussian Noise		Poisson Noise	
Homogeneous Noise	PSNR / SSIM	**PSNR / SSIM**	PSNR / SSIM	**PSNR / SSIM**
1	35.12 / 0.958	**37.94 / 0.979**	28.95 / 0.863	**37.17 / 0.979**
2	30.60 / 0.901	**37.79 / 0.979**	26.12 / 0.785	**37.16 / 0.979**
3	28.38 / 0.852	**37.94 / 0.980**	24.48 / 0.729	**37.19 / 0.979**
4	27.11 / 0.817	**38.13 / 0.980**	23.31 / 0.685	**37.18 / 0.979**
5	26.04 / 0.781	**38.07 / 0.980**	22.39 / 0.649	**37.19 / 0.979**
6	25.18 / 0.751	**34.30 / 0.963**	21.62 / 0.618	**32.11 / 0.951**

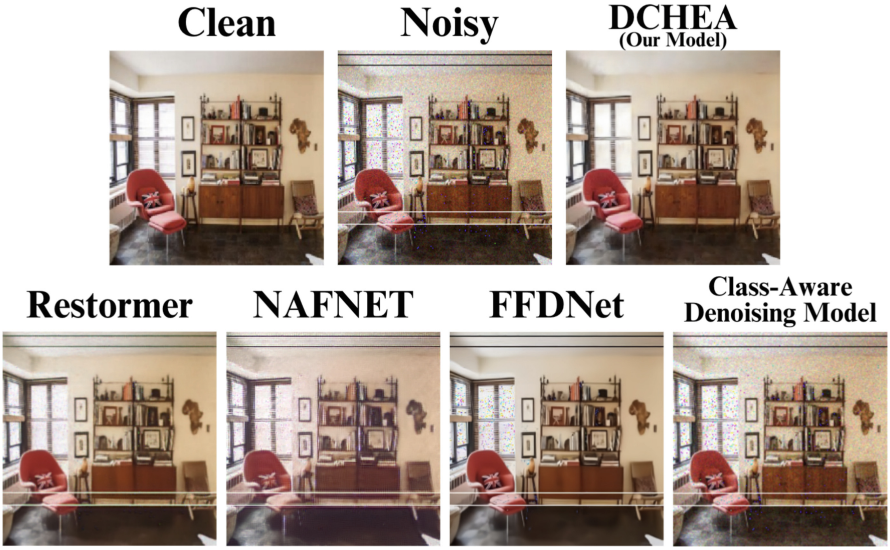


Fig. 5. Comparison across different models for noise level=8

5.3 Qualitative Results

In Figure 5 we show a a sample image result for qualitative comparison of our proposed model with various state-of-the-art models. Results for the remaining noise levels are present in the supplementary materials.

It is evident that the DCHEA model significantly outperforms other denoising models such as Restormer, NAFNET, FFDNet, and Class Aware models in terms of image restoration from noise. DCHEA excels in preserving intricate details within the scene, notably maintaining the textures on the couch and the items on the bookshelf, which appear clearer and more defined compared to results from other models. Additionally, DCHEA demonstrates superior color accuracy; the authenticity of colors in the denoised image, such as those of the books and artwork is noticeably closer to the original clean image. This corroborates the quantitative values obtained in Table 2. The model also stands out in its ability to remove noise effectively without the adverse effect of over-smoothing, which can lead to a washed-out appearance. In contrast, some models like FFDNet and Restormer tend to sacrifice sharpness and detail for noise reduction, often resulting in a blurred or softened image along with loss of textual characteristics, tint effect, and clarity.

5.4 Discussion

It was found from the experiment that DCHEA showed promising results in denoising images that were corrupted by multiple kinds of noise. It is evident from Tables 2 & 3 that the model is robust with the increase in both homogeneous and heterogeneous kinds of noise. This could be attributed to the selection of the set of autoencoders by the uniform cost search mechanism. Moreover, it was observed during testing that the noise order does not affect the performance. However, as the search algorithm is more focused on cost optimization, it does not take into account the number of autoencoders involved in creating the set. Hence, the complexity of the search may increase. Moreover, if the state-space tree of autoencoders is not balanced the scalability can be a limiting factor in the performance of the algorithm. However, it is observed that a combination of a lesser number of autoencoders (6 considered in the experimentation) can handle gracefully different combinations of noise (10 different levels of noise), This can be attributed to the fact that the most general autoencoder is trained with all cumulative combinations of noise. Hence, we do not face the scalability issue with denoising any arbitrary combination of noise. However, rigorous experimentation and validation under different conditions is needed to justify the claim.

6 Conclusion and Future work

In this research, we have successfully introduced the DCHEA model, which demonstrates promising performance in image denoising. Through rigorous comparative analysis, it consistently surpasses various state-of-the-art models. Our implementation of a hierarchical autoencoder structure, complemented by the novel N-layered tree methodology, validates the capability of DCHEA to maintain image structural details and restore piecewise intensity homogeneity under various noise conditions. The findings of this study pave the way for future explorations into complex denoising applications, affirming the significant impact of advanced deep-learning techniques in enhancing image quality. However, rigorous experimentation and validation are needed to analyze the complexity and scalability of the model. Further, we need a better search optimization technique that can provide an optimal set of autoencoders.

Acknowledgements. We would like to express our sincere gratitude to Mahindra University for their financial support to carry out the research work.

References

1. Jamil Azzeh, Bilal Zahran, and Ziad Alqadi. Salt and pepper noise: Effects and removal. *JOIV: International Journal on Informatics Visualization*, 2(4):252–256, 2018
2. Bian, S., He, X., Zhengguang, X., Zhang, L.: Image denoising by deep convolution based on sparse representation. Computers **12**(6), 112 (2023)
3. Charles Boncelet. Image noise models. In Alan C. Bovik, editor, *The essential guide to image processing*, pages 143–167. Elsevier, 2009
4. Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. In *European conference on computer vision*, pages 17–33. Springer, 2022
5. Lovedeep Gondara. Medical image denoising using convolutional denoising autoencoders. *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*, pages 241–246, 2016
6. Charles, C.: Beckner Jr. and Charles L. Matson. Using mean-squared error to assess visual image quality. In: Luk, F.T. (ed.) Advanced Signal Processing Algorithms, Architectures, and Implementations XVI, volume 6313, page 63130E. International Society for Optics and Photonics, SPIE (2006)
7. Dong-Wook Kim, Jae Ryun Chung, and Seung-Won Jung. GRDN:grouped residual dense network for real image denoising and gan-based real-world noise modeling, 2019
8. Li, W., Wang, J.: Residual learning of cycle-gan for seismic data denoising. IEEE Access **9**, 11585–11597 (2021)
9. Kok Cheng Lim and Ali Al-Naji: Image denoising using hybrid deep learning approach and self-improved orca predation algorithm. Technologies **11**(4), 111 (2023)
10. Luisier, F., Blu, T., Unser, M.: Image denoising in mixed poisson-gaussian noise. IEEE Trans. Image Process. **20**(3), 696–708 (2010)
11. Tal Remez, Or Litany, Raja Giryes, and Alex Bronstein. Class-aware fully-convolutional gaussian and poisson denoising. *IEEE Transactions on Image Processing*, PP:1–1, 07 2018
12. Tal Remez, Or Litany, Raja Giryes, and Alex M Bronstein. Class-aware fully convolutional gaussian and poisson denoising. *IEEE Transactions on Image Processing*, 27(11):5707–5722, 2018
13. Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach.* Pearson, 2016
14. Singh, P., Shree, R.: Speckle noise: Modelling and implementation. International Journal of Control Theory and Applications **9**(17), 8717–8727 (2016)
15. Yidi Teng, Shouzhao Sheng, and Yubin Zheng. Nonlinear gaussian filter with multistep colored noise. *Actuators*, 11(4), 2022
16. Vahdat, A., Kautz, J.: Nvae: A deep hierarchical variational autoencoder. Adv. Neural. Inf. Process. Syst. **33**, 19667–19679 (2020)
17. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.-A.: Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. J. Mach. Learn. Res. **11**, 3371–3408 (2010)
18. Wang, Y., Song, X., Chen, K.: Channel and space attention neural network for image denoising. IEEE Signal Process. Lett. **28**, 424–428 (2021)
19. Xiufen Ye, Lin Wang, Huiming Xing, and Le Huang. Denoising hybrid noises in image with stacked autoencoder. In Hou, Y and Zhang, H and Zhou, S, editor, *2015 IEEE International Conference on Information and Automation*, pages 2720–2724. IEEE, 2015

20. Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5728–5739, 2022
21. Syed Waqas Zamir, Aditya Arora, Salman H. Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration. *CoRR*, abs/2111.09881, 2021
22. Yanhong Zeng, Jianlong Fu, Hongyang Chao, and Baining Guo. Aggregated contextual transformations for high-resolution image inpainting. *IEEE Transactions on Visualization and Computer Graphics*, 2022
23. Zhang, K., Zuo, W., Zhang, L.: FFDNet: Toward a Fast and Flexible Solution for CNN-Based Image Denoising. IEEE Trans. Image Process. **27**(9), 4608–4622 (2018)
24. Zhan Zhang and Wenyou Wei. Primal-dual approach for uniform noise removal. In *First International Conference on Information Science and Electronic Technology (ISET 2015)*, pages 103–106. Atlantis Press, 2015

Leveraging Computer Vision for Automatic Modulation Classification: Insights from Spectrum and Constellation Diagram Analysis

Wenjie Zhao and Qiuming Luo(✉)

College of Computer Science and Software Engineering, Shenzhen University,
Shenzhen 518060, China
2210273073@email.szu.edu.cn, lqm@szu.edu.cn

Abstract. Automated modulation recognition is a challenging task in communication systems. Leveraging recent advancements in transfer learning, this paper proposes a novel method for automatic modulation recognition using transferred computer vision models. The method allows fine-tuning of the vision models to recognize modulation signals through spectrum and constellation diagrams. Experiments on the Radioml dataset demonstrate that the proposed method outperforms recent traditional methods by 8.97%, with an average accuracy of 0.5732. An ablation study confirms the effectiveness of using spectrum and constellation diagrams. This study verifies the feasibility of transferring vision models to AMC tasks.

Keywords: Transfer Learning · Automatic Modulation Classification · Deep Learning

1 Introduction

Automatic Modulation Classification (AMC) is a critical yet challenging task in wireless communication that involves automatically identifying the modulation scheme of an unknown signal through its features and estimating the related parameters.

In recent years, deep-learning methodologies have emerged as the dominant approach for AMC. Given the temporal nature of signals, researchers have traditionally resorted to using Long Short-Term Memory (LSTM) networks or Recurrent Neural Networks (RNNs) to address the AMC task, as evidenced in [13]. The advent of Convolutional Neural Networks (CNNs) has subsequently given rise to a plethora of CNN-based AMC methods, which vary in terms of architectural design [11] and preprocessing techniques [12]. Additionally, there have been attempts to perform AMC through the recognition of waveform shapes [6], via adversarial learning [1], and through the application of federated learning [10].

At the same time, neural image classification has experienced a technological surge.Originating with AlexNet [4], the superior accuracy in neural image

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 284–294, 2025.
https://doi.org/10.1007/978-3-031-78383-8_19

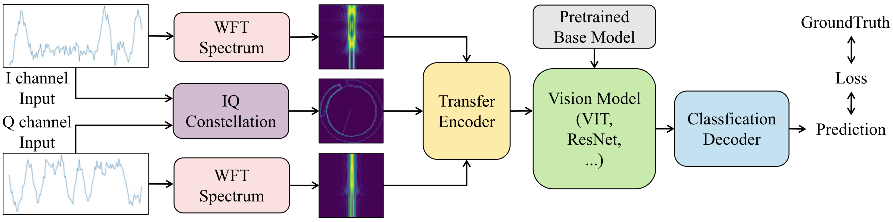


Fig. 1. Complete Architecture of the Proposed Method.

classification was long-dominated by ResNet[3]. Subsequently, the Transformer[9] introduced a revolutionary transformation across the deep learning landscape, leading to Vision Transformer (ViT)[2] surpassing ResNet in performance. Most recently, EfficientNet V2[8] has outperformed all preceding models, securing the top position in neural image classification. Given the impressive accuracy of these foundational models, transfer learning has seen rapid development, effectively leveraging their success for various downstream applications [7].

Inspired by the succession of breakthroughs in the computer vision domain, this paper presents a transfer learning architecture designed to utilize powerful pretrained visual recognition models for the purpose of automatic modulation classification via spectral and constellation diagram analysis. This approach leverages the triumphs of computer vision to mitigate carbon emissions, with the ambition of surpassing traditional state-of-the-art methodologies in the field of automatic modulation classification. First,we introduced a novel preprocessing technique that converts signal sequences into spectral and constellation diagrams to align with vision models' input requirements, thereby improving their interpretive efficacy. Second, we developed a transfer network that adapts any computer vision model for automatic modulation classification through the recognition of spectral and constellation diagrams, enabling accurate classification of modulated signals. Third, the proposed method's feasibility is confirmed and its state-of-the-art performance is demonstrated through comprehensive assessments, with the transferred computer vision models outperforming traditional CNN models in comparative studies. Forth, the ablation study confirms the critical efficacy of spectrum and constellation diagrams as transformative tools for signal inputs, enhancing their conversion into the image domain.

The remainder of this paper is structured as follows: Chapters II and III will delineate the problem statement and methodologies employed in this study, encompassing data preprocessing, model architecture design, and training procedures. Chapter IV will validate the efficacy of our proposed methodology through comparative and ablation studies on the RADIOML dataset, disclosing the empirical findings alongside analytical discourse. Chapter V will critically evaluate the strengths and limitations of the proposed method, followed by a discussion on potential avenues for future research. Conclusively, Chapter VI will encapsulate the paper and proffer potential trajectories for future inquiry along with reflective insights.

2 Problem Satement

In the conventional framework of wireless communication systems, the radio signal transmitted within a real-world context can be accurately modeled as a convolution of the channel impulse response and the normalized radio signal, superimposed with uncorrelated Gaussian noise. Mathematically, this relationship is expressed as:

$$X(n) = H(n) * S(n) + W(n)$$

Here, $H(n)$ denotes the channel impulse response, while $S(n)$ represents the normalized radio signal. The asterisk (*) indicates the convolution operation. $W(n)$ signifies an uncorrelated additive Gaussian noise process, a common element in both simulation and modeling frameworks. The magnitude of the noise is typically quantified by the Signal-to-Noise Ratio (SNR), a metric that delineates the quality of the radio signal. Subsequently, the primary objective of Automatic Modulation Classification is to discern the modulation type of the signal $X(n)$—which is subject to both noise and channel effects—based solely on observed data.

3 Method

To facilitate the recognition of time-domain signals by a vision model, it is imperative to convert one-dimensional (1D) sequence signals into a two-dimensional (2D) image format. We employ the Windowed Fourier Transform (WFT) function to convert the signal sequences into the frequency domain, manifesting as a spectral diagram. Concurrently, we generate the In-phase and Quadrature (IQ) Signal Constellation Diagram to preserve the original signal information within a 2D framework. This concatenation procedure merges the I-channel Spectrum, Q-channel Spectrum, and IQ Constellation Diagram into a single 3-channel image. Given that the majority of computer vision models are trained and optimized for natural image processing, we incorporate a transfer learning layer to bridge the gap between the domains. We initiate our model with pretrained base architectures to optimize efficiency. An extensive evaluation of various computer vision models is conducted to identify the most appropriate architecture for our task. In the final stages, the network processes the input data, decoding the probability of each class label. The loss is computed by comparing the network's predictions against the ground truth labels, thereby facilitating backward propagation and gradient descent optimization. The complete architectural framework is detailed in Figure 1.

3.1 WFT

Consider a time series $X = \{x_1, x_2, \ldots, x_n\}$ of length n. We traverse the time series X with a window of size $\frac{n}{2}$ to generate a window sequence W_k at each position k, repeated $\frac{n}{2}$ times. Subsequently, a fast Fourier transform (FFT) is applied to each W_k to yield a window frequency spectrum sequence F_k of length

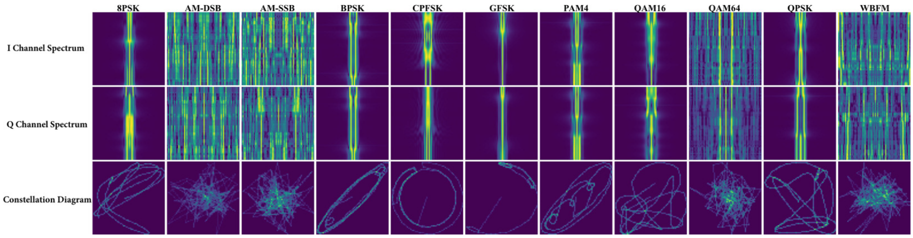


Fig. 2. Partially Preprocessed Spectrum and Constellation Diagram for Signals of Various Modulation Classes

$\frac{n}{2}$. By stacking the rows of the F_k sequences, we assemble a matrix P with dimensions $\frac{n}{2} \times \frac{n}{2}$.

$$W_k = \{x_{k+1}, x_{k+2}, \ldots, x_{k+\frac{n}{2}}\}, \quad k = 1, 2, \ldots, \frac{n}{2} \tag{1}$$

$$F_k = FFT(W_k), \quad P = concat(F_1, F_2, \ldots, F_{\frac{n}{2}}) \tag{2}$$

When n is even, the FFT can traditionally be decomposed into two components, and defined as:

$$X(k) = \sum_{n=0}^{\frac{n}{2}-1} x(n)e^{-j2\pi k\frac{n}{n}} + W_n^k \sum_{n=\frac{n}{2}}^{n-1} x(n)e^{-j2\pi k\frac{n}{n}} \tag{3}$$

Here, W_n^k represents the rotation factor for n points, which depends on both k and n. When k is even, $W_n^k = 1$; for odd k, $W_n^k = -1$. The samples of the output matrix P are displayed in the first and second rows of Figure 2.

3.2 Signal Constellation Diagram

The IQ Signal Constellation Diagram serves as a specialized visualization tool for complex-valued samples within a signal. This diagram arranges the In-phase (I) and Quadrature (Q) components of the signal along the x and y axes of a two-dimensional graphical representation. The I component denotes the in-phase component, while the Q component signifies the quadrature component. Conventionally, the phase of the I component is established at 0 degrees (or 0 radians), with the Q component phased at 90 degrees (or $\frac{\pi}{2}$ radians) in relation to the I component. Within the complex-valued domain, the signal constellation can be articulated as:

$$S(t) = I(t) + jQ(t) \tag{4}$$

We track the trajectory of $S(t)$ within the complex-valued plane, ultimately synthesizing the constellation diagram as depicted in the lowermost row of Figure 2.

3.3 Transfer Layer and Vision Model Finetune

In this study, we introduce a tiny linear transformation as a pre-processing encoder prior to inputting the data into the vision model. This step is designed to bridge the inherent disparity between natural images and spectral or constellation diagrams. By doing so, we aim to enhance the vision model's capacity to effectively adapt and perform in the context of automatic modulation classification tasks. For the purpose of testing, we selected top-performing and widely recognized natural image classification models, namely Vision Transformer (ViT), ResNet-50, and EfficientNet. These models were chosen due to their exceptional performance in the field of computer vision. Additionally, we utilized pre-trained models to maximize computational efficiency. Ultimately, the class predictions were rendered as probability vectors, and the loss was computed by comparing these predictions to the ground truth labels. The Cross-Entropy Loss function was employed as the loss metric for this purpose. The whole processes can be seen in Figure 1.

4 Experiments and results

4.1 Experiment Settings

The RadioML2016.10a dataset [5], derived from Shakespeare's Gutenberg works and the TV series "Serial Episode," is utilized in this paper for evaluating methods and conducting experiments for wireless machine learning tasks.

The dataset is formatted in IQ data format with a sample size of 2×128. It comprises 220,000 samples at a 200kHz sampling rate, featuring offsets and noise. Sampling rate and carrier frequency offsets have standard deviations of 0.01Hz and max offsets of 50Hz and 500Hz, respectively. It includes 8 digital and 3 analog modulation methods(8PSK, BPSK, CPFSK, GFSK, PAM4, 16QAM, 64QAM, and QPSK), with SNR ranging from -20dB to 18dB. It also features delay settings with corresponding amplitudes, and AWGN noise. The channel environment includes selective fading, CFO, and SRO.

In our experimental framework, we divided a comprehensive dataset of 220,000 samples into training and testing subsets. Specifically, we allocated 176,000 samples for training purposes and an additional 44,000 samples for evaluation, maintaining a ratio of one-forth of the train set and the test set. Throughout the training phase, we employed the Adam optimizer, initializing the learning rate at 5e-4 and allowing it to decay to zero over the course of 100 epochs. The batch size was set at 64 samples for each iteration.

4.2 Comparison Experiments

In the interest of fairness, we selected two recent traditional CNN AMC methods, ATLA[1] and MSCN[11], for our baseline comparison. For the vision models, we opted for the recently popular architectures: VIT, Resnet50, and EfficientNet, for our experimental analysis. In terms of vision models, we explored two training

Table 1. Quantitative Comparison Results: Proposed Method vs. Traditional Method

Method	Model	Average Accuracy↑	Maximum Accuracy↑
Traditional	ATLA[1]	0.5260	0.8240
Sequences	MSCN[11]	0.5408	-
Proposed	VIT(B16)	0.5310	0.8178
Method With	VIT(L16)	0.5279	0.8151
Pretrained	EfficientNet(B0)	**0.5732**	**0.8881**
Model	EfficientNet(B7)	0.5703	0.8786
	Resnet50	0.5636	0.8773

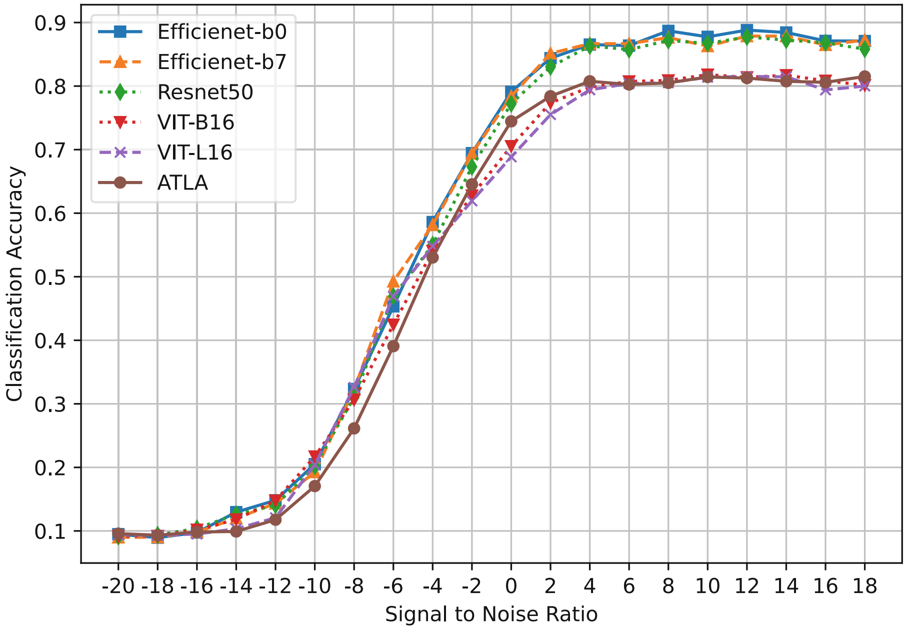


Fig. 3. Average Classification Accuracy of Various Comparison Methods Across Different SNRs.

options: training from scratch and loading pretrained models. As illustrated in Table 1, employing pretrained models significantly reduces training time while yielding superior performance on the dataset. Among the pretrained models, EfficientNet-b0 demonstrated the most outstanding performance within our proposed architecture, achieving an average accuracy of 0.5732. This surpasses the traditional methods ATLA (0.5260) and MSCN (0.5408) by 8.97% and 5.99%, respectively. Moreover, EfficientNet-b0 also exhibits the highest maximum accuracy. Additionally, we note that although Resnet50 does not achieve the highest performance, it still exhibits commendable results on the test dataset. In contrast, VIT demonstrates the least favorable performance when compared to the other pretrained vision models.

Figure 3 illustrates the performance of various methods across different SNRs. It can be discerned that the proposed architectures, when equipped with pretrained ResNet and EfficientNet models, consistently outperform the traditional ATLA method across all SNRs.

4.3 Ablation Experiments

To validate the efficacy of constellations and spectral analysis for recognizing modulation classes, an ablation study was conducted, with results presented in Table 2. The table shows that, among single-modal inputs, constellation diagrams achieve the highest accuracy with a score of 0.4650, indicating their effectiveness in distinguishing modulation classes. The performance with I-channel and Q-channel spectra approaches traditional method benchmarks. When constellation diagrams are included, the performance jumps from 0.5107 to 0.5732, an increase of 12.24%, securing a leading position.

Table 2. Quantitative Results for the Ablation Experiment: Spectrum and Constellation Diagrams

Model	I-channel Spectrum	Q-channel Spectrum	Constellation Diagram	AVG Acc	Max Acc
EfficientNett	✓			0.3977	0.6815
EfficientNet		✓		0.4043	0.6646
EfficientNet			✓	0.4650	0.7746
EfficientNet	✓	✓		0.5107	0.7706
EfficientNet	✓	✓	✓	**0.5732**	**0.8881**

4.4 SNR and Classification Analysis

Figure 4 illustrates the classification accuracy for various modulation classes and signal-to-noise ratios (SNRs) using the proposed method with the optimal performing model: EfficientNet-b0. Observations indicate that, with the exception of WBFM, the proposed method exhibits commendable performance across all other modulation classes. Even at an SNR of -4 dB, the accuracy for the majority of classes remains above 80%, demonstrating the robustness of the proposed method under challenging signal conditions. Figure 5 displays the confusion matrix for the classification results obtained from the dataset using the proposed method with pretrained EfficientNet-b0. When compared to traditional methods, the proposed method demonstrates a notable improvement in distinguishing between QAM16 and QAM64, which are frequently misclassified by

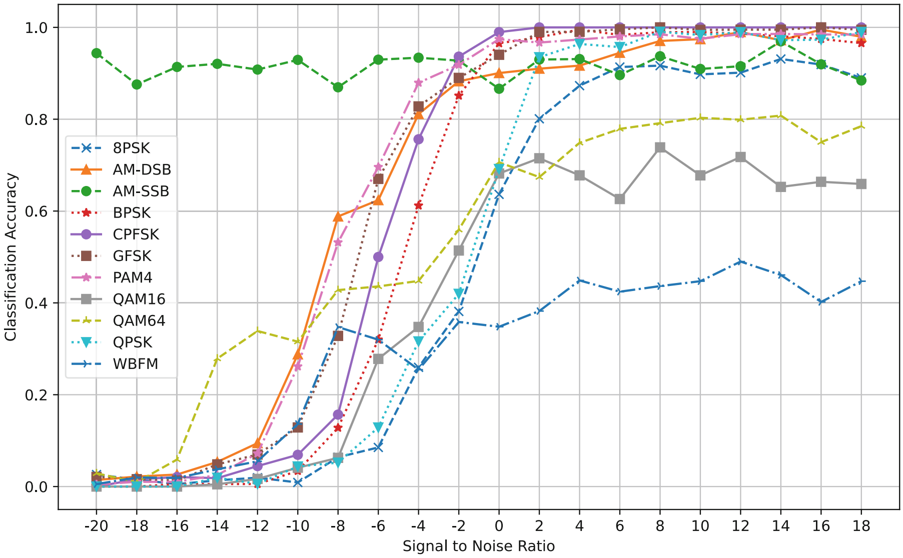


Fig. 4. The Correlation Between Classification Accuracy for Diverse Classes and SNR in the Context of the Proposed Method Using Pretrained EfficientNet-b0.

most methods [1], achieving an overall high accuracy with the majority of the metrics exceeding 90 Despite these advancements, the proposed method continues to face challenges in differentiating between WBFM and AM-DSB. Addressing this issue is a potential direction for future research.

5 Discussion

In this paper, the novel transfer learning architecture is proposed for Automatic Modulation Classification (AMC), which leveraging of pretrained visual recognition models to analyze spectral and constellation diagrams. The preprocessing technique introduced is highlighted as a significant strength due to its alignment with the input requirements of vision models, which enhances interpretive efficacy. Additionally, the development of a transfer network that adapts computer vision models for AMC is recognized as a substantial methodological advancement, offering superior performance compared to traditional CNN-based methods, as supported by comprehensive assessments and comparative studies.The use of pretrained models like EfficientNet and Vision Transformer (ViT) and the preprocessing encoder to bridge the gap between natural images and spectral or constellation diagrams are strengths. The robustness of the dataset and the clarity of the presentation are also praised.

However, the paper's weaknesses are also addressed. The assumption of the universal applicability of spectrum and constellation diagrams, along with the use of pretrained computer vision models, may overlook crucial signal features or

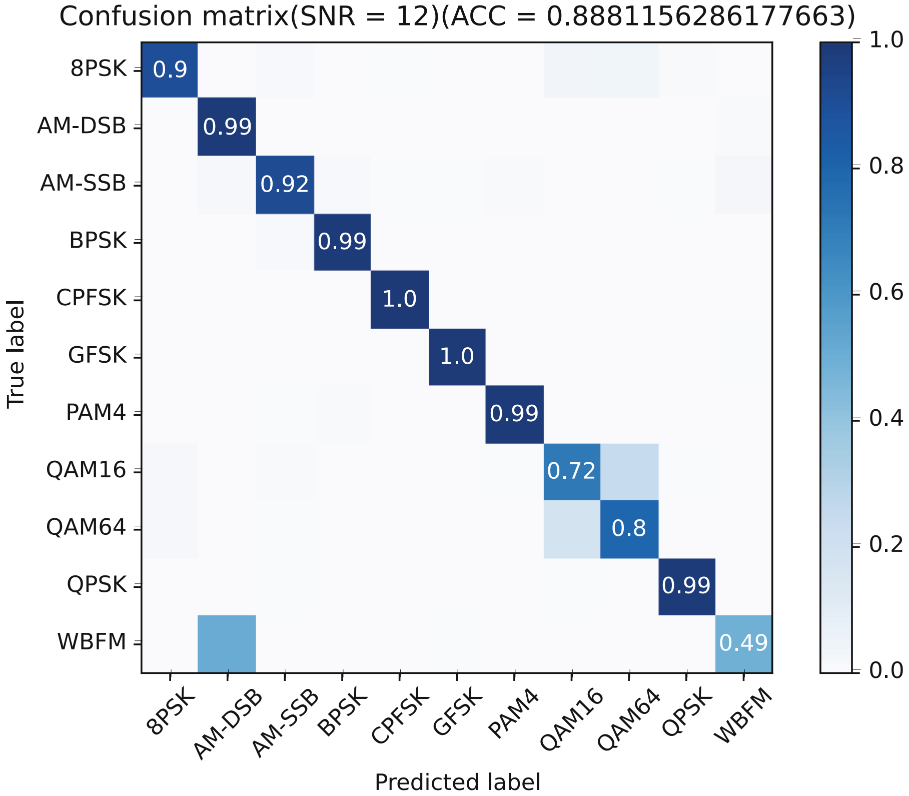


Fig. 5. Confusion Matrix for the Optimal Performance Attained by the Proposed Method Utilizing Pretrained EfficientNet-b0 at an SNR of 12

constraints in different contexts or modulation schemes. The preprocessing technique, while innovative, may introduce biases or information loss, potentially affecting classification performance. By transforming I and Q channel inputs into spectrograms and combining them with IQ constellation diagrams is noted for its leveraging of deep learning's success in image classification. The consistent outperformance of existing approaches across multiple models is seen as a consolidation of the argument. Nevertheless, concerns are raised about the scalability of the models used, given the limited number of modulation methods in the experiments. Besides, the method's specificity and its generalizability to other datasets or real-world scenarios are called into question, along with the need for further insights into model performance under extreme noise conditions.

Additionally, our study yields numerous insights into the domain of wireless communication. The methodology proposed herein paves the way for subsequent investigations. By leveraging advanced large-scale models through visual processing, we can significantly enhance performance in the domain of AMC and related tasks. The application of vision models to the field of signal processing opens up a vast array of research opportunities, which constitutes a promising direction for future work in this area.

6 Conclusion

In this paper, we introduce a transfer learning framework that empowers vision models to perform automatic modulation classification utilizing spectral analysis and constellation diagrams. By leveraging popular pre-trained vision models, we demonstrate the viability of this approach and achieve superior performance compared to traditional methods on the dataset in our experiments. Additionally, an ablation study is conducted to substantiate the efficacy of incorporating spectral and constellation diagram analysis. This paper illustrates that computer vision models are capable of executing automatic modulation classification through the analysis of spectrum and constellation diagrams. Moreover, the findings suggest that there is potential for further advancements and exploration in this domain.

Acknowledgments.. The research was funded by projects granted, from National key research and development program (2021YFB3301500), and Shenzhen Science Technology Foundation (20220811154353002).Corresponding author: Qiuming Luo

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Bu, K., He, Y., Jing, X., Han, J.: Adversarial transfer learning for deep learning based automatic modulation classification. IEEE Signal Process. Lett. **27**, 880–884 (2020). https://doi.org/10.1109/LSP.2020.2991875
2. 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. ArXiv **abs/2010.11929** (2020), https://api.semanticscholar.org/CorpusID:225039882
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 770–778 (2016) https://doi.org/10.1109/CVPR.2016.90
4. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Commun. ACM **60**(6), 84–90 (may 2017) https://doi.org/10.1145/3065386, https://doi.org/10.1145/3065386
5. O'Shea, T., West, N.: Radio machine learning dataset generation with gnu radio. Proceedings of the GNU Radio Conference **1**(1) (2016), https://pubs.gnuradio.org/index.php/grcon/article/view/11
6. Qi, P., Zhou, X., Zheng, S., Li, Z.: Automatic modulation classification based on deep residual networks with multimodal information. IEEE Transactions on Cognitive Communications and Networking **7**(1), 21–33 (2021). https://doi.org/10.1109/TCCN.2020.3023145

7. Tan, B., Zhang, Y., Pan, S., Yang, Q.: Distant domain transfer learning. Proceedings of the AAAI Conference on Artificial Intelligence **31** (2017https://doi.org/10.1609/aaai.v31i1.10826
8. Tan, M., Le, Q.: Efficientnetv2: Smaller models and faster training. In: Meila, M., Zhang, T. (eds.) Proceedings of the 38th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 139, pp. 10096–10106. PMLR (18–24 Jul 2021), https://proceedings.mlr.press/v139/tan21a.html
9. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. p. 6000–6010. NIPS'17, Curran Associates Inc., Red Hook, NY, USA (2017)
10. Wang, Y., Gui, G., Gacanin, H., Adebisi, B., Sari, H., Adachi, F.: Federated learning for automatic modulation classification under class imbalance and varying noise condition. IEEE Transactions on Cognitive Communications and Networking **8**(1), 86–96 (2022). https://doi.org/10.1109/TCCN.2021.3089738
11. Xiao, J., Wang, Y., Zhang, D., Ma, Q., Ding, W.: Multiscale correlation networks based on deep learning for automatic modulation classification. IEEE Signal Process. Lett. **30**, 633–637 (2023). https://doi.org/10.1109/LSP.2023.3275912
12. Zhang, H., Huang, M., Yang, J., Sun, W.: A data preprocessing method for automatic modulation classification based on cnn. IEEE Commun. Lett. **25**(4), 1206–1210 (2021). https://doi.org/10.1109/LCOMM.2020.3044755
13. Zhang, Z., Luo, H., Wang, C., Gan, C., Xiang, Y.: Automatic modulation classification using cnn-lstm based dual-stream structure. IEEE Trans. Veh. Technol. **69**(11), 13521–13531 (2020). https://doi.org/10.1109/TVT.2020.3030018

DG2Net: A MLP-Based Dynamixing Gate and Depthwise Group Norm Network for Classification of Glaucoma

Yu Feng(✉), Cong Wu, and Yuan Zhou

Hubei University of Technology, Wuhan, China
yufeng@hbut.edu.cn

Abstract. The accurate extraction of pertinent information from fundus images is of paramount importance for the diagnosis of glaucoma. For a considerable period, researchers engaged in this field have typically employed the convolutional neural network approach for detection. Despite notable advancements, the inability of convolutional neural networks to capture long-range dependencies in order to make a judgement in the fundus images as a whole represents a current challenge. Conversely, MLP-based models are gaining widespread attention due to their simple network structure and performance that is not weaker than CNNs and transformers. In this paper, we propose a novel MLP-based approach called Dynamixing Gate and Depthwise Group Norm Network (DG2Net), which uses a convolutional neural network to extract local information while using an MLP-like model to obtain global information. Specifically, we propose the Dynamixing Gate MLP for enhancing the extraction of spatial information and the Depthwise Group Norm MLP to enhance the model's ability to extract channel information. Extensive experiments were conducted on two publicly available glaucoma datasets, EyePACS AIROGS-Light and EyePACS-AIROGS-light-V2. The results demonstrated that our method outperforms other existing methods, and that DG2Net is highly competitive for glaucoma detection.

Keywords: Glaucoma diagnosis · MLP · Convolutional neural network

1 Introduction

In recent years, there has been a notable increase in the prevalence of glaucoma, a common eye disease that can result in irreversible blindness in the majority of patients. Statistical data indicates that glaucoma affects approximately 80 million individuals globally in 2022. By 2040, this figure is projected to exceed 100 million. In the early stages of glaucoma, the patient's vision changes are not readily apparent and may be difficult to detect. However, as the disease progresses, the patient experiences a significant loss of vision, which may result in missed opportunities for treatment and ultimately lead to blindness. Glaucoma is a condition that occurs in the optic papilla region and is characterized by an

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 295–308, 2025.
https://doi.org/10.1007/978-3-031-78383-8_20

elevated intraocular pressure, a deformation of the optic nerve head, visual field defects and optic nerve atrophy. Nevertheless, it is important to note that glaucoma can also occur in patients with normal IOP and diabetes [3]. The diagnosis of glaucoma is challenging, and in the past, ophthalmologists made subjective judgements based on their clinical experience, which was time-consuming and labour-intensive. Furthermore, the results were influenced by many subjective factors. To address the limitations of manual diagnosis, deep learning technology and computer-aided diagnosis have been introduced into glaucoma diagnosis [19]. In comparison to traditional detection methods, deep learning techniques permit the rapid and effective extraction of morphological features of the optic nerve papilla, including the optic cup, optic disc, cup-to-disc ratio, and angle of incision, from fundus images. This provides significant convenience for relevant personnel.

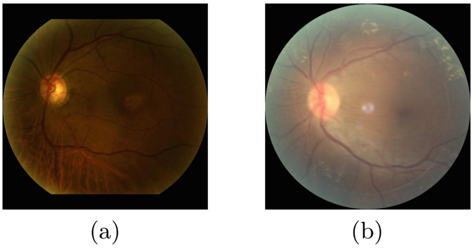


Fig. 1. (a) Refers to referable glaucoma, and (b) refers to non-referable glaucoma

The growing prevalence and scope of deep learning techniques have led to the remarkable success of methods based on convolutional neural networks in medical image classification tasks, largely due to their exceptional local feature extraction capabilities. Many of these techniques have been successfully applied to glaucoma detection tasks. One of the most commonly used methods for diagnosing glaucoma is the fundus image, as illustrated in Fig 1. Nayak et al. [14] proposed a new non-manual feature extraction method called the evolutionary convolutional network (ECNet) for glaucoma detection. The method employs numerous convolutional, compression, rectified linear units (ReLU), and summation layers to extract fundus image features. A real-coded genetic algorithm (RCGA) evolutionary algorithm is also proposed to optimise the weights of different layers. Finally, an accuracy of 97.20 % was achieved using ECNet with SVM. Hung et al. [6] used a pre-trained Efficientnet-b0 model along with other features of the patient, such as age, gender, and high myopia, for binary and tertiary classification of fundus images, respectively. Conversely, Cho et al. [2] developed an integrated system of convolutional neural networks with 56 different features based on InceptionNet to classify the dataset into three categories: normal eye, early glaucoma and advanced glaucoma. This approach achieved an accuracy of 88.1 % and an average area under the receiver operating characteristic of 0.950. Liao et al. [11] proposed a novel approach for aggregating features at different scales to enhance the performance of glaucoma diagnosis, termed M-LAP. Furthermore, by modelling the correspondence between the binary diagnostic information and the spatial pixels, the scheme generated glaucomatous activation, thereby bridging the gap between global semantic diagnosis and precise localisation.D'Souza et al. [5] proposed a parameter-efficient AlterNet-K model based on an alternating design pattern, which combines ResNets and multiple self-attention (MSA). This approach aims to leverage the complementary attributes of these two components in order to improve the generality of the overall model.

Nowadays, networks based on multilayer perceptrons (MLP) have demonstrated superior performance in computer vision tasks, particularly in the extraction of global information, rivaling networks based on the transformer family. Tolstikhin et al. [16] initially proposed a succinct network architecture without convolution and self-attention, MLP-Mixer, which has achieved remarkable success in the field of image classification. Liu, Dai, So and Le [12] proposed gMLP, which enhances the extraction of spatial information using gating operations on MLP-Mixer. Wang, Jiang et al. [18] used a dynamic information fusion technique to dynamically generate mixing matrices, which improves the robustness of the model and reduces the time complexity. Cao et al. [1] enriched the interaction capability of the tokens in three ways, which significantly improves the performance of the MLP-based model in small datasets.

A review of the existing literature revealed that the majority of glaucoma detection methods employ traditional machine learning techniques, in addition to convolutional neural networks. These networks have proven to be a significant advancement over manual diagnosis, yet there remain significant challenges. These include the loss of crucial information during manual cropping of fundus images, data imbalance, and models that are disrupted by noise, rendering them unable to recognise glaucoma in more complex cases [21]. Furthermore, while convolutional neural networks are more adept at detecting changes in the optic papilla region in fundus images, they are constrained in their ability to acquire remote context dependencies [10]. To effectively address the aforementioned challenges, we propose a novel MLP-based network, the Dynamixing Gate and Depthwise Group Norm Network (DG2Net), for the accurate detection of glaucomatous lesions.

In conclusion, the following contributions are made:

(1) A novel model, designated DG2Net, has been developed for the purpose of glaucoma detection. The model employs EfficientNet-b0 as the backbone, which serves to extract local information. This approach avoids the significant computational and parametric quantities associated with the self-attention mechanism, while simultaneously introducing an enhanced multilayer perceptron for the first time, which is capable of extracting global information.

(2) The Dynamixing Gate module is incorporated into the token-mixing component of MLP-Mixer, enabling the dynamic generation of the mixing matrix while utilising spatial gating to enhance the interaction between tokens.

(3) The Depthwise Group Norm module is integrated into the channel-mixing section of MLP-Mixer, facilitating the grouping of channels for mixing, thereby enhancing the competitiveness between channels and facilitating the acquisition of channel features.
(4) A series of experiments were conducted on EyePACS AIROGS-Light and EyePACS-AIROGS-light-V2, and the results demonstrated that our method achieved excellent classification outcomes.

2 Method

2.1 Network Design

In this work, we employ a multi-stage approach to construct the overall architecture of DG2Net, as illustrated in Fig 2 . DG2Net comprises a series of Mobile Inverted Bottleneck Convolution(MBConv) blocks, as depicted in Fig 3, and a series of MLP blocks. In fundus images, the boundary between the diseased and healthy regions of the optic papilla is not as obvious as in other ordinary images, and the texture of the diseased region is relatively fuzzy. Therefore, multiple stages of MBConv blocks are employed to extract the underlying information of the fundus image in order to obtain the relevant feature information of the diseased region. However, in medical images, lesion regions are not always discernible in isolation and must be considered in conjunction with multiple other regions to form an overall assessment. While convolution can effectively extract local features from an image, it is not sufficient for long-range dependencies acquisition. To address this issue, we subsequently devised two parallel enhanced MLP modules for extracting and learning global features in fundus images. The Dynamixing Gate MLP enhances the model's capacity to acquire spatial information, while the Depthwise Group Norm MLP enhances its ability to acquire

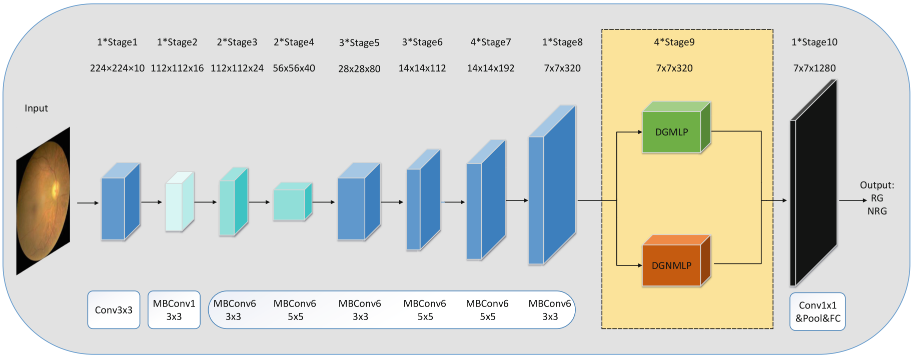


Fig. 2. The network structure of the proposed DG2Net. In front of the stage number represents the layers.

channel information. Finally, the convolution, pooling and fully connected layers are applied, resulting in the final classification result.

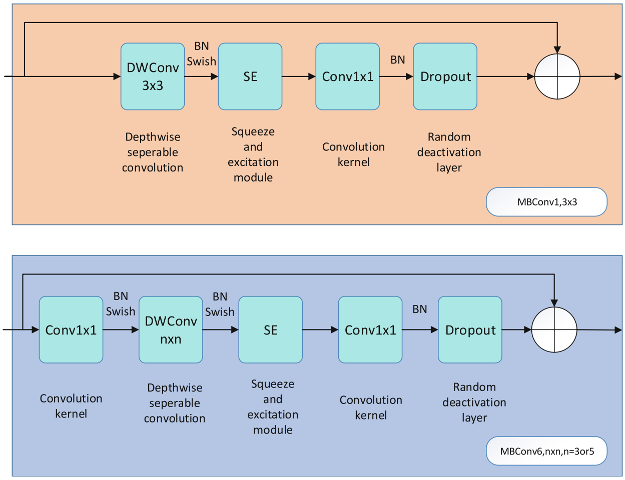

BN
Swish
DWConv 3x3
SE
Conv1x1
BN
Dropout
Depthwise seperable convolution
Squeeze and excitation module
Convolution kernel
Random deactivation layer
MBConv1,3x3
BN
Swish
BN
Swish
Conv1x1
DWConv nxn
SE
Conv1x1
BN
Dropout
Convolution kernel
Depthwise seperable convolution
Squeeze and excitation module
Convolution kernel
Random deactivation layer
MBConv6,nxn,n=3or5

Fig. 3. Structure of MBConv. Above is MBConv1, below is MBConv6, n represents the number of convolutional kernels.

2.2 MLP-Mixer

The MLP-Mixer, a novel and compact structure that eschews convolution and attention, has demonstrated remarkable performance on the ImageNet dataset by employing only linear layers. Its architectural design is illustrated in Fig 4. In order to combine the overall judgement of the fundus image, a network based on MLP-Mixer is introduced to extract global features. Subsequently, the local features are extracted by the convolutional layer. For MLP-Mixer, the input feature map $F_{in} \in R^{C \times H \times W}$ is split into multiple patches, where C is the number of channels, and H and W are the height and width of the input tensor, respectively. At this juncture, all the patches are subjected to the Patch Embedding

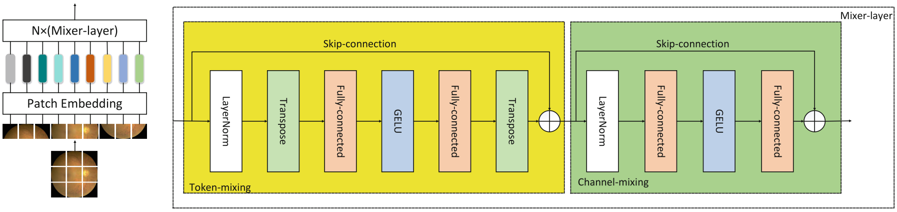


Fig. 4. Structure of MLP-Mixer

operation, which is executed by a fully-connected network to extract the tokens. At this stage, the shape of the feature map is $R^{C\times N}$, where C is the number of channels and N is the patch size. Thereafter, the feature map is fed into N Mixer-layers. The Mixer-layer comprises two stages: token mixing and channel mixing. Token mixing is used to extract feature information among different patches, as shown in the yellow box in Fig 4. Channel mixing is used to extract information among different channels, as shown in the green box in Fig 4. In particular, the feature map $R^{C\times N}$ is subjected to LayerNorm and then Transpose, resulting in $R^{N\times C}$. Two fully connected operations and a GELU activation function are applied to the patch direction, the feature map is Transposed once more to obtain $R^{C\times N}$, which is combined with the skip connection to complete the feature extraction in the patch direction. In a similar manner, the feature map $R^{C\times N}$ is subjected to LayerNorm, two fully connected layers, and a GELU activation function in order to complete the information extraction in the channel direction.

2.3 Dynamixing Gate MLP

The most fundamental MLP-Mixer employs only static fusion between tokens to extract information, and lacks the capacity to adapt to the mixing of the contents of tokens. In contrast, we adopt a novel dynamic fusion of tokens that generates a mixing matrix using the contents of all tokens to be mixed. Furthermore, we integrate a spatial gating unit that interacts across tokens, enables learnable spatial transformations, and learns a single transformation shared across channels via spatial projection, in contrast to classical deep convolution based on channel-specific filters. The Dynamixing Gate MLP further enhances the MLP-Mixer's ability to acquire global information in the token direction, as illustrated in Fig 5. In particular, the input feature map $F_{in} \in R^{C\times H\times W}$ is initially subjected to a Dynamixing Gate operation in the H dimension, followed by a Linear operation in the C dimension, and then a permute operation to obtain $R^{C\times W\times H}$. This is subsequently subjected to a Dynamixing Gate operation in the W dimension, after which a permute operation is performed to obtain the output feature map $F_{out} \in R^{C\times H\times W}$.

Finally, the channel mixing process is initiated. This can be expressed by the following equation:

$$F_{out} = DGop\_W(Linear(DGop\_H(F_{in}))) \tag{1}$$

where F_{in} represents the input feature map, F_{out} represents the output feature map, $DGop\_W(\bullet)$ represents the Dynamixing Gate operation in the W dimension, $Linear(\bullet)$ represents the Linear operation in the C dimension, and $DGop\_H(\bullet)$ represents the Dynamixing Gate operation in the H dimension. In the context of Dynamixing Gate operations, our design principle is to divide a set of input tokens $X \in R^{N\times D}$, where N is the number of tokens and D is the number of feature dimensions, into two branches for input. This is followed by the generation of a dynamic mixing matrix Q in the left branch based on the input,

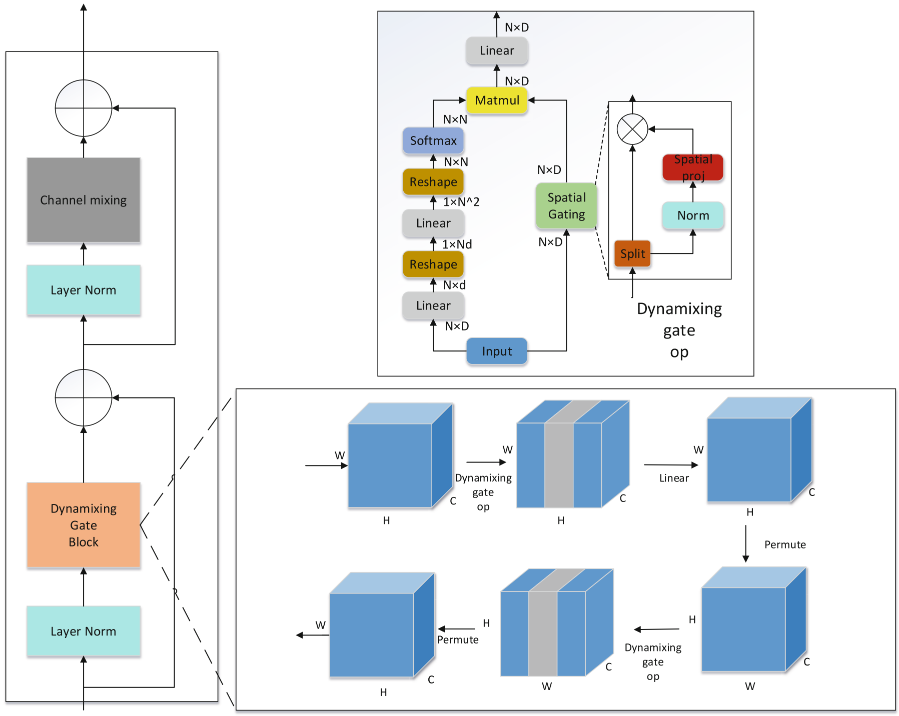


Fig. 5. Structure of Dynamixing Gate MLP

and then the multiplication of the obtained matrix Q with the input tokens X to obtain the output token Y, so $Y = QX$. In order to obtain the dynamic mixing matrix Q, we generate it using a linear function of all the input token features. This is achieved by flattening the input X as a vector and generating the mixing matrix according to the following equation:

$$\hat{X}^{(s)} = XW_d^{(s)} \tag{2}$$

$$Q_{i\cdot}^{(s)} = softmax(flat(\hat{X}^{(s)})W^{(s,i)}) \tag{3}$$

$$Y = [Q^{(0)}X^{(0)}, ..., Q^{(s-1)}X^{(s-1)}]W_o \tag{4}$$

In order to enhance the resilience of the model and reduce the number of parameters, we divide the features into S segments, where $d \ll D$, is a very small number such as 1 or 2, $\hat{X}^{(s)} \in R^{N\times d}$, $W_d^{(s)}$is the reduced dimension matrix, $flat(\hat{X}^{(s)}) \in R^{1\times Nd}$ is the vector resulting from flattening. $W^{(s,i)} \in R^{Nd\times N}$. $Softmax(\bullet)$ is the softmax operation applied to the row vectors, and $Q_{i\cdot}^{(s)}$ is the s-th segment of the row in the mixing matrix contains the mixing weights of the i-th output token. The splicing operation is represented by the symbol $[\bullet, \bullet]$, while the output feature fusion matrix is denoted by $W_o \in R^{D\times D}$. Conversely, in the right branch, the Spatial Gating Unit (SGU) is included, which is designed to enhance the module's capacity to interact with tokens across the network. This is achieved through a straightforward linear projection:

$$f_{W,b}(Z) = WZ + b \tag{5}$$

where $W \in R^{N \times N}$ is of the same length as the sequence and is independent of the input Z representation. Both N and b are token-specific deviations. Subsequently, the input Z is divided into two independent parts *(Z1, Z2)* along the channel dimension. This results in the output $s(\bullet)$ of the spatial gating unit being expressed as:

$$s(Z) = Z_1 \otimes f_{W,b}(Z_2) \tag{6}$$

where $\otimes$ represents the product of elements. The division operation has the effect of greatly improving the stability of the model and ensuring independent processing between tokens during learning. The left and right branches then perform the product operation after their respective operations, and finally, the output of the Dynamixing Gate operation, $F_{out} = Y \bullet s(Z)$, is obtained through the linear layer.

2.4 Depthwise Group Norm MLP

The traditional MLP-Mixer performs a simple two-times fully connected operation and GELU activation function to extract the channel features in the channel direction. In contrast, most of the MLP-like models are improved in token-mixing for enhancing the ability of information interaction between tokens. We propose a brand new MLP-like module, called Depthwise Group Norm MLP, as shown in Fig 6, which greatly improves the ability to obtain information in the channel direction. In particular, the MLP model produces a considerable number of parameters and runs at a relatively slow pace. To address this, we initially incorporated DWConv to enhance the running speed and reduce the number of parameters. Subsequently, after a LayerNorm layer, a fully-connected layer, and a GELU activation function, we introduced Group Normalization. This is achieved by dividing the feature map into G groups in the channel direction, which results in a change in shape from $R^{C \times H \times W}$ to $R^{G \times C//G \times H \times W}$. The mean and variance are then calculated within each group, which strengthens the competitiveness between channels and thus the extraction ability of channel features. Concurrently, given that MLP-like models typically necessitate a substantial number of samples for training, and that medical image data are more challenging to obtain, we have incorporated Group Normalization to enhance the model's stability during training. This has resulted in a faster convergence rate and an improved model's ability to generalize. Furthermore, it has facilitated the model's capacity to learn the general patterns in the data, rather than relying excessively on the features of a few specific samples. This has effectively prevented model overfitting.

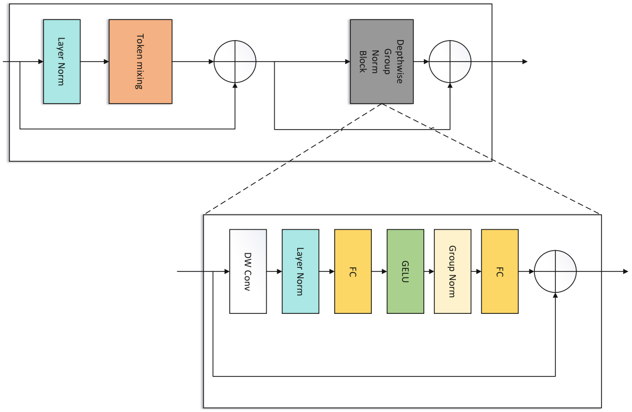


Fig. 6. Structure of Depthwise Group Norm MLP

3 Experiments

3.1 Dataset

In order to assess the efficacy of the proposed methodology, a series of comprehensive experiments were conducted on two publicly available glaucoma datasets: EyePACS AIROGS-Light [7] and EyePACS-AIROGS-light-V2 [8]. The following section provides specific details. The EyePACS AIROGS-Light and EyePACS-AIROGS-light-V2 datasets comprise balanced subsets of standardized fundus images based on the EyePACS AIROGS produced by Riley Kiefer et al. The EyePACS AIROGS-Light dataset contains a total of 6,540 images, which have been divided equally into two categories: referable glaucoma (RG) and non-referable glaucoma (NRG). Each of the aforementioned categories comprises 2,500 images for training, 270 images for validation, and 500 images for testing, respectively. The image size is 256 × 256. The EyePACS-AIROGS-light-V2 dataset comprises a total of 9,540 images, which have been divided into two categories: RG and NRG. Each category contains 4,000 images for training, 385 for validation, and 385 for testing, respectively. The image dimensions are 512 × 512. A series of ablation experiments was conducted to ascertain the relative importance of each module in the proposed architecture. Furthermore, we present the classification results for both test datasets, which are then compared with those obtained by state-of-the-art methods.

3.2 Implementation Details and Evaluation Metrics

All experiments are based on the PyTorch framework and trained on an NVIDIA Tesla M40, a GPU with 24 GB of memory. The input image size of all datasets is 224 × 224. The Adam optimizer is employed as the model's optimizer, with an initial learning rate of 0.0001. CosineAnnealingLR is used as the learning rate adjuster, which automatically adjusts the learning rate of the optimizer according to the training progress. The training batch size was set to 4. In order to ensure the fairness of the experiments, all experiments were conducted without the use of pre-training weights and trained for 300 epochs. The experiments employ five commonly used metrics to evaluate the performance, including accuracy (ACC), F1-score (F1), precision (Prec), area under the curve (AUC), and recall (Recall).

3.3 Loss Function

The cross-entropy value indicates the uncertainty of the classifier's decision. A smaller cross-entropy value indicates a more accurate prediction and a higher model accuracy. Consequently, the cross-entropy loss function is employed as the loss function for the two glaucoma classification datasets, with the objective of measuring the distance between the probability distribution of the model output and the true labels. This allows for the assessment of the model's performance, with a focus on determining the extent to which the model's output aligns with the true labels. By reducing the loss, the performance of the model is enhanced. The specific formula is as follows:

$$Loss = -\sum_{i=1}^{n} p(x_i) \log q(x_i) \tag{7}$$

where x_i represents the actual value of the image category, *p(x)* represents the predicted value of the true distribution of the sample, and *q(x)* represents the distribution predicted by the model.

3.4 Experimental Results and Analysis

In order to validate the effectiveness of our model, we conducted a comparison with several classical and state-of-the-art models, all of which were trained using the same training strategy and experimental configurations. The experimental results on the EyePACS AIROGS-Light and EyePACS-AIROGS-light-V2 datasets are presented in Table 1 and Table 2, respectively. The results of the two experiments are presented in the following table. On the EyePACS AIROGS-Light dataset, our method achieved 91.80% accuracy, 91.83% F1-score, and 91.90% AUC, which were all superior to those of other state-of-the-art methods. The second most effective model is Efficientnet, which is unable to capture long-range dependencies despite its advantage of strong ability to extract local information. In terms of the number of parameters, our network demonstrates superior performance compared to MLP-based, Transformer-based, and select CNN-based networks. However, in the context of lightweight networks, such as Efficientnet, our network exhibits a higher number of parameters. In terms of accuracy, F1-score, and AUC, our method outperforms other methods on the EyePACS-AIROGS-light-V2 dataset. In comparison to MaxVIT, which is the most effective method among the others, our method demonstrated superior performance in terms of ACC, F1, and AUC by 1.04%, 1.11%, and 1.04%, respectively. This is attributed to the following factors: This is because MaxVIT, while having some global feature extraction capability, is limited to mixing information in terms of axiality, and our approach allows information to interact dynamically between tokens, and therefore has a greater ability to acquire feature. In terms of qualitative analysis, we present the confusion matrices of our model on the two datasets, as shown in Fig 7, as well as a plot of the experimental results comparing our model with other models, as shown in Fig 8. The graph demonstrates that our model accurately predicted glaucoma disease and that, in comparison to other methods, our method is more advantageous.

Table 1. Performance comparison on the EyePACS AIROGS-Light dataset(Bold is the best results, underlined is the second best results).

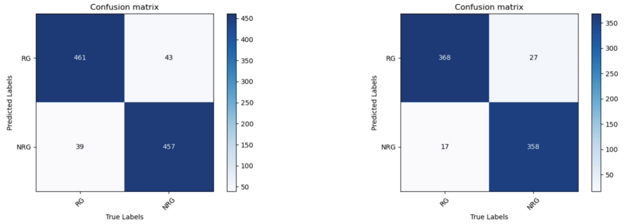
Method	Network	ACC(%)	F1(%)	AUC(%)	Params
CNN-based	Efficientnet [15]	90.80	91.00	90.90	5.3M
	Convnext [13]	85.40	85.40	85.90	28.6M
	Inceptionnext [20]	90.60	90.59	90.80	8.4M
	Alternet-K [5]	90.70	90.70	91.00	1.3M
Transformer-based and MLP-based	Vision transformer [4]	61.20	60.72	61.20	86.0M
	MaxVIT [17]	89.30	89.30	89.80	31.0M
	Dynamixer [18]	78.80	78.79	79.00	26.0M
	StripMLP [1]	90.60	90.60	90.80	24.3M
	RaMLP [9]	86.20	86.20	86.20	25.0M
	SCnet [22]	89.60	89.60	90.20	201.0M
Ours	DG2Net	**91.80**	**91.83**	**91.90**	20.7M

3.5 Ablation Experiment

In order to verify the effectiveness of our proposed module, we conducted ablation experiments on the Eyepacs AIROGS-Light dataset. The results of these experiments are presented in Table 3. In particular, we used the Efficientnet-b0 model as our baseline, upon which we designed our network structure. Firstly, the combination of the baseline and DGMLP resulted in an improvement of 0.3% in ACC, 0.1% in F1 and 0.4% in AUC, respectively. This indicates that the Dynamixing Gate operation effectively promotes the interaction of information between tokens. The combination of the baseline and DGNMLP resulted in an improvement of 0.6%, 0.4% and 0.7% in ACC, F1 and AUC, respectively. This demonstrates that the capacity for channel mixing has been enhanced, leading to more accurate classification. Finally, the combination of both with the baseline resulted in an improvement of 1% for ACC, 0.83% for F1, and 1% for AUC, respectively. This indicates that the method achieves satisfactory results in the glaucoma detection task.

Table 2. Performance comparison on the EyePACS-AIROGS-light-V2 dataset(Bold is the best results, underlined is the second best results).

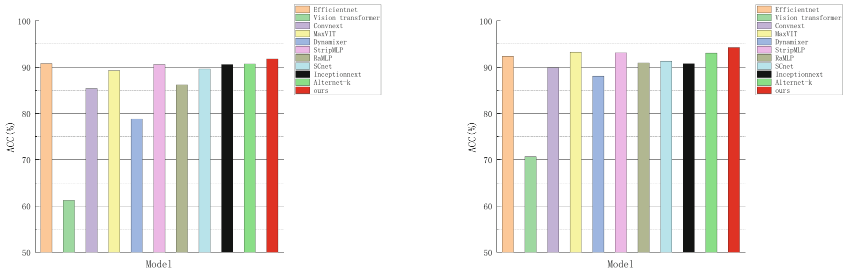
Method	Network	ACC(%)	F1(%)	AUC(%)
CNN-based	Efficientnet [15]	92.34	92.34	92.44
	Convnext [13]	89.87	89.88	89.97
	Inceptionnext [20]	90.78	90.80	90.88
	Alternet-K [5]	93.03	93.12	93.06
Transformer-based and MLP-based	Vision transformer [4]	70.65	70.59	70.65
	MaxVIT [17]	<u>93.25</u>	<u>93.24</u>	<u>93.25</u>
	Dynamixer [18]	88.05	88.15	88.15
	StripMLP [1]	93.12	93.17	93.22
	RaMLP [9]	90.91	90.93	90.94
	SCnet [22]	91.30	91.50	91.25
Ours	DG2Net	**94.29**	**94.35**	**94.29**



(a)

(b)

Fig. 7. (a) Refers to confusion matrix of DG2Net on EyePACS AIROGS-Light, and (b) refers to confusion matrix of DG2Net on EyePACS-AIROGS-light-V2.



(a)

(b)

Fig. 8. (a) is the results of EyePACS AIROGS-Light, and (b) is the results of EyePACS-AIROGS-light-V2.

Table 3. The result of ablation studies on the EyePACS AIROGS-Light dataset. The best results are in bold.

Network	ACC(%)	F1(%)	AUC(%)
Baseline(Efficientnetb0)	90.80	91.00	90.90
Baseline+DGMLP	91.10	91.09	91.30
Baseline+DGNMLP	91.40	91.40	91.60
Baseline+DGMLP+DGNMLP(DG2Net)	**91.80**	**91.83**	**91.90**

4 Conclusion

In this paper, we propose DG2Net, a deep learning model for efficient detection of glaucoma lesions. DG2Net is an MLP-based network architecture comprising multiple MBConv blocks and MLP blocks. To this end, we have introduced the MLP block, which enables the acquisition of global information in the fundus image. Furthermore, we have designed two novel modules, which enhance the token-mixing and channel-mixing abilities of the MLP block, resulting in significant detection outcomes.

The DG2Net model was validated on the EyePACS AIROGS-Light and EyePACS-AIROGS-light-V2 datasets, achieving state-of-the-art performance. In the future, we will continue to refine the DG2Net model in order to make it more generalizable for practical implementation in the future, with the aim of assisting those working in the healthcare industry.

References

1. Cao, G., Luo, S., Huang, W., Lan, X., Jiang, D., Wang, Y., Zhang, J.: Strip-mlp: Efficient token interaction for vision mlp. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1494–1504 (2023)
2. Cho, H., Hwang, Y.H., Chung, J.K., Lee, K.B., Park, J.S., Kim, H.G., Jeong, J.H.: Deep learning ensemble method for classifying glaucoma stages using fundus photographs and convolutional neural networks. Curr. Eye Res. **46**(10), 1516–1524 (2021)
3. Desideri, L.F., Rutigliani, C., Corazza, P., Nastasi, A., Roda, M., Nicolo, M., Traverso, C.E., Vagge, A.: The upcoming role of artificial intelligence (ai) for retinal and glaucomatous diseases. Journal of Optometry **15**, S50–S57 (2022)
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. D'Souza, G., Siddalingaswamy, P., Pandya, M.A.: Alternet-k: a small and compact model for the detection of glaucoma. Biomed. Eng. Lett. **14**(1), 23–33 (2024)
6. Hung, K.H., Kao, Y.C., Tang, Y.H., Chen, Y.T., Wang, C.H., Wang, Y.C., Lee, O.K.S.: Application of a deep learning system in glaucoma screening and further classification with colour fundus photographs: a case control study. BMC Ophthalmol. **22**(1), 483 (2022)

7. Kiefer, R., Abid, M., Ardali, M.R., Steen, J., Amjadian, E.: Automated fundus image standardization using a dynamic global foreground threshold algorithm. In: 2023 8th International Conference on Image, Vision and Computing (ICIVC). pp. 460–465. IEEE (2023)
8. Kiefer, R., Abid, M., Steen, J., Ardali, M.R., Amjadian, E.: A catalog of public glaucoma datasets for machine learning applications: A detailed description and analysis of public glaucoma datasets available to machine learning engineers tackling glaucoma-related problems using retinal fundus images and oct images. In: Proceedings of the 2023 7th International Conference on Information System and Data Mining. pp. 24–31 (2023)
9. Lai, S., Du, X., Guo, J., Zhang, K.: Ramlp: vision mlp via region-aware mixing. In: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence. pp. 999–1007 (2023)
10. Lian, J., Liu, T.: Lesion identification in fundus images via convolutional neural network-vision transformer. Biomed. Signal Process. Control **88**, 105607 (2024)
11. Liao, W., Zou, B., Zhao, R., Chen, Y., He, Z., Zhou, M.: Clinical interpretable deep learning model for glaucoma diagnosis. IEEE J. Biomed. Health Inform. **24**(5), 1405–1412 (2019)
12. Liu, H., Dai, Z., So, D., Le, Q.V.: Pay attention to mlps. Adv. Neural. Inf. Process. Syst. **34**, 9204–9215 (2021)
13. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11976–11986 (2022)
14. Nayak, D.R., Das, D., Majhi, B., Bhandary, S.V., Acharya, U.R.: Ecnet: An evolutionary convolutional network for automated glaucoma detection using fundus images. Biomed. Signal Process. Control **67**, 102559 (2021)
15. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. pp. 6105–6114. PMLR (2019)
16. Tolstikhin, I.O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., et al.: Mlp-mixer: An all-mlp architecture for vision. Adv. Neural. Inf. Process. Syst. **34**, 24261–24272 (2021)
17. Tu, Z., Talebi, H., Zhang, H., Yang, F., Milanfar, P., Bovik, A., Li, Y.: Maxvit: Multi-axis vision transformer. In: European conference on computer vision. pp. 459–479. Springer (2022)
18. Wang, Z., Jiang, W., Zhu, Y.M., Yuan, L., Song, Y., Liu, W.: Dynamixer: a vision mlp architecture with dynamic mixing. In: International conference on machine learning. pp. 22691–22701. PMLR (2022)
19. Wu, C., Li, S., Liu, X., Jiang, F., Shi, B.: Dms-mafm+ efficientnet: a hybrid model for predicting dysthyroid optic neuropathy. Medical & Biological Engineering & Computing **60**(11), 3217–3230 (2022)
20. Yu, W., Zhou, P., Yan, S., Wang, X.: Inceptionnext: When inception meets convnext. arXiv preprint arXiv:2303.16900 (2023)
21. Zedan, M.J., Zulkifley, M.A., Ibrahim, A.A., Moubark, A.M., Kamari, N.A.M., Abdani, S.R.: Automated glaucoma screening and diagnosis based on retinal fundus images using deep learning approaches: A comprehensive review. Diagnostics **13**(13), 2180 (2023)
22. Zhang, R., Wang, L., Cheng, S., Song, S.: Mlp-based classification of covid-19 and skin diseases. Expert Syst. Appl. **228**, 120389 (2023)

SocialFaceEmoNet: A Deep Architecture for Social Media Face Recognition and Emotion Detection Using Customized Datasets

Jayanta Paul[1](✉), Abhijit Mitra[1], Somak Sanyal[2], and Jaya Sil[1]

[1] Indian Institute of Engineering Science and Technology, Shibpur 711103, India
{2020csp003.jayanta,2022csp005.abhijit}@students.iiests.ac.in,
js@cs.iiests.ac.in

[2] CALNESTOR Knowledge Solutions Private Limited, Debi Park, Hooghly 712103, India

Abstract. In the dynamic landscape of social media, understanding of emotional expression and its detection from user-generated content is pivotal for various applications, from marketing strategies to mental health monitoring. Traditional methods of face recognition and emotion detection often fail due to inefficient way of handling complex and unstructured social media posts. This paper proposes a novel architecture, "SocialFaceEmoNet", that utilizes the power of deep neural networks (DNNs), specifically Convolutional neural Networks (CNNs) and Transformer models to extract features for simultaneous recognition of faces and detection of emotions with high accuracy and efficiency. The proposed architecture begins with the curation of SocioFaceSet, a specialized dataset tailored to the unique characteristics of social media imagery. Unlike previous approaches that rely solely on pre-trained models, SocialFaceEmoNet incorporates transfer learning techniques to adapt the CNNs and Transformers to the nuances of social media data, thereby improving their performance. Furthermore, we introduce a comprehensive evaluation framework to assess the effectiveness of our approach. Through extensive experiments on real-world social media datasets, we demonstrate the superiority of SocialFaceEmoNet over existing methods in terms of both face recognition and emotion detection tasks. Notably, our architecture achieves remarkable results even in challenging scenarios characterized by varying lighting conditions, image resolutions, and facial expressions.

Keywords: Face Recognition · Emotion Detection · Social Media Analysis · Deep Neural Networks · Convolutional Neural Networks(CNNs) and Transformer models

1 Introduction

Social media platforms become an integral part of our daily lives, with billions of users sharing a vast amount of content every day. This content, often in the form

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 309–327, 2025.
https://doi.org/10.1007/978-3-031-78383-8_21

of images and videos, can provide valuable insights to detect users' emotions, opinions, and behaviors. Understanding the emotional content of these posts is crucial for various applications, ranging from targeted marketing strategies to monitoring mental health trends. For instance, brands can tailor their marketing campaigns based on the emotions expressed by users, while mental health professionals can identify individuals who may need support. However, analyzing this emotional content presents significant challenges. Social media imagery is highly diverse and often unstructured, featuring a wide range of lighting conditions, image resolutions, facial expressions, and occlusions. Traditional methods of face recognition and emotion detection struggle to handle this complexity effectively. These methods often rely on pre-defined rules that cannot adapt well to the diverse and unpredictable nature of social media content. To address these challenges, this paper introduces a novel architecture named SocialFaceEmoNet, designed to enhance face recognition and emotion detection in social media posts. Our approach leverages the power of deep neural networks (DNNs), specifically Convolutional Neural Networks (CNNs) and Transformer models, which have demonstrated superior performance in various computer vision tasks. Initially, we experimented with several state-of-the-art models, including VGG16, ResNet-50, and Vision Transformer (ViT), to identify the most effective architecture for this task. The main contribution of the paper is the creation of SocioFaceSet, a customized dataset specifically curated to capture the unique characteristics of social media imagery. This dataset includes a diverse array of images that reflect the typical conditions found on social media platforms, such as varying lighting, different facial expressions, and a range of image qualities. By training our models on this specialized dataset, we aim to improve generalization ability of the proposed model that perform well on real-world social media data. In our experiments, while CNN architectures like VGG16 and ResNet-50 showed promising results, the Vision Transformer (ViT) model consistently outperformed them, especially in handling the complex and varied nature of social media images. Consequently, our proposed architecture, SocialFaceEmoNet, primarily employs the ViT model for both feature extraction and sequence modeling, due to its superior performance. By incorporating transfer learning techniques, we further fine-tune the ViT model to adapt to the specific nuances of social media data for enhancing its performance. This adaptation is crucial for addressing the variability and complexity inherent in social media images. This paper makes several contributions to the field of face recognition and emotion detection in social media using deep learning:

- **Proposed Architecture (SocialFaceEmoNet)**: We introduce a novel architecture, SocialFaceEmoNet, which leverages the Vision Transformer (ViT) model for feature extraction. This architecture is specifically designed to address the challenges of face recognition and emotion detection in the complex and unstructured environment of social media platforms.

- **Customized Dataset (SocioFaceSet)**: We curate a specialized dataset, SocioFaceSet, tailored to the unique characteristics of social media imagery. This dataset includes diverse images that reflect typical conditions found on social media, such as varying lighting, different facial expressions, and a range of image qualities. SocioFaceSet serves as a foundation for training and evaluating our deep learning models.
- **Comprehensive Study**: We provide a comprehensive study using various evaluation metrics to assess the performance of SocialFaceEmoNet. Through extensive experiments conducted on real-world social media datasets, we demonstrate the effectiveness of our approach in terms of both face recognition and emotion detection tasks.
- **Performance and Robustness**: Our experiments show that SocialFaceEmoNet achieves remarkable results, even in challenging scenarios characterized by varying lighting conditions, image resolutions, and complex facial expressions. The Vision Transformer model, used for feature extraction, proves to be effective in capturing intricate details and patterns in social media images.
- **Advancement of Deep Learning Technologies**: This work underscores the importance of innovative architectures, such as SocialFaceEmoNet, and tailored datasets, such as SocioFaceSet, in advancing the capabilities of deep learning technologies for real-world applications. Our findings highlight the potential of Vision Transformer models in transforming the analysis of social media content, providing valuable insights for various practical applications.

Rest of the paper is organized as follows: Section 2 provides an overview of related work in the field of face recognition and emotion detection in social media, highlighting existing approaches and their limitations. In Section 3, we delve into the methodology employed in this study, detailing the architecture of SocialFaceEmoNet and the various deep learning models utilized to frame the proposed architecture. Section 4 elaborates on the creation and characteristics of the SocioFaceSet, discussing its importance in training and evaluating the models. Following this, in Section 5 we evaluate the performance of SocialFaceEmoNet through extensive experiments conducted on real-world social media datasets. In Section 6, we discuss the overall analysis. Finally, we conclude the paper in Section 7, by summarizing our findings, discussing implications, and suggesting directions for future research in the field.

2 Related Work

Face recognition and emotion detection in social media have attracted considerable interest owing to its extensive uses in marketing, healthcare, and research of social behaviour. The conventional methods for recognizing faces use handcrafted features and machine learning models, which often fail to deal with changes in lighting, location, and expressions, commonly appeared in social media photos. Deep learning, specifically convolutional neural network (CNN) has led to substantial advancements in recognizing faces with more accuracy and reliability. Early methods, such as Eigenfaces and Fisherfaces, used Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) to reduce dimensionality of extracted features. Ojala et al. [1] introduced texture descriptors, such as Local Binary Patterns (LBP), which offered a more resilient method for capturing face characteristics. Machine learning has led to the use of Support Vector Machines (SVM) and boosting algorithms such as AdaBoost to improve recognition performance. Nevertheless, it was the advent of CNNs that really transformed the discipline. Groundbreaking deep neural network models such as AlexNet [2], VGGNet [3], and Inception [4] have shown notable advancements in performance. These advancements have greatly improved the accuracy and resilience of facial recognition algorithms in practical situations, and they are backed by large datasets such as LFW [5], MegaFace[6], and MS-Celeb-1M [7]. Simonyan and Zisserman [3] introduced the VGG16 architecture, renowned for its simplicity and efficiency in image classification applications, such as face recognition. ResNet, proposed by He et al. [8] effectively tackles the issue of vanishing gradients in deep networks and has shown exceptional performance in a range of computer vision applications, such as face recognition. Tan and Le [9] recently developed EfficientNet, a model that achieved outstanding performance by increasing the depth, breadth, and resolution of the network in a well-balanced way. Recent advancements in face recognition have been significantly driven by deep learning architectures, particularly CNNs. The introduction of models like VGG-Face [10], FaceNet [11], and ArcFace [12] settle new standard in recognizing faces with remarkable accuracy. These models leverage large-scale datasets and complex architectures to achieve outstanding performance, particularly while dealing with challenging scenarios, involving occlusions, varying lighting conditions, and diverse ethnic backgrounds.

CNNs have been modified and optimized to enhance the accuracy of image identification in social media platforms, where images exhibit a broad range of quality and content. Oquab et al. [13] used transfer learning methodologies, in which models that were pre-trained on large datasets like as ImageNet were modified to suit social media images, resulting in a substantial improvement in face recognition accuracy. Jayanta et al.[14] assessed facial recognition methods for IoT system designs with limited computational capabilities, offering valuable insights into the efficiency of different techniques when dealing with restricted computing resources.

In recent years, the subject of face emotion detection has made substantial progress, mostly due to accessibility of extensive datasets. CNN based face emotion detection models are reported, resulting in significant levels of accuracy. Tang [15] introduced a CNN method to recognize face emotions. This technique achieved a high accuracy of 95.6% when tested on the FER2013 [16] dataset. Mollahosseini et al.[17] used a deep learning methodology to recognise face emotions and achieved a remarkable accuracy of 92.5% on the same dataset. Previous research works used transfer learning techniques to improve face emotion detection. As an example, Kim et al. [18] used a pre-trained CNN model and adjusted it using a dataset specifically designed for recognising face emotions, achieving

93.1% accuracy. In addition, Zhang et al.[19] introduced a multitask learning methodology for face emotion detection, with accuracy 95.5%. Contemporary techniques have been evolved from traditional classifiers to more sophisticated models that capture both spatial and temporal aspects of facial expressions. Hybrid models such as CNN-LSTM [20] and the use of attention mechanisms demonstrate superior performance in capturing subtle emotional cues, particularly in video sequences. Additionally, large annotated datasets like AffectNet [17], FER+ [21] expedite development of the models that acquire immense generalization ability well across different populations and contexts.

Several research works have investigated the use of attention processes in the field of face emotion detection. Li et al.[22] introduced an attention based CNN model for recognizing face emotions. This method achieved an accuracy of 94.2%. Furthermore, other datasets have been suggested for the purpose of facial emotion detection, in addition to the aforementioned research. The FER2013[16] dataset, proposed by Goodfellow et al. [23], is a commonly used dataset for facial emotion recognition. It comprises 35,887 photos depicting various facial emotions. Additionally, the RAF-DB dataset, which was presented by Li and Deng [24], consists of 30,000 photos and has significant importance in this field. Paul et al. [25] curated a dataset by manually labelling the Indian cartoon postings. Several deep learning methods are explored to identify the faces and interpret the emotions of characters appear in the posts. Their findings underscore the potential of culturally unique datasets in enhancing the accuracy of models. Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, which are well-known for their capability to represent sequential data, have been used to capture temporal relationships in emotion detection tasks. Cho et al. [26] have played a crucial role in the development of these models. Nevertheless, these models may incur significant processing costs and encounter difficulties in handling distant relationships. Transformer models, initially developed for problems related to natural language processing (NLP), have been modified for using image-based applications, such as emotion detection in more recent times. The Vision Transformer (ViT), proposed by Dosovitskiy et al. [27], has shown impressive performance by treating images as sequences of patches. This approach enables the model to successfully capture spatial connections and contextual information. Within the realm of social media, there has been an exploration of multimodal techniques that combine visual and textual information, particularly in situations where photos are often accompanied by text and other metadata. Poria et al.[28] showed that these methods use both visual content and verbal context to improve the accuracy of detecting emotions.

3 Dataset

For this work, we created a custom dataset specifically designed to evaluate the performance of our proposed architecture, SocialFaceEmoNet, in the context of social media posts. While numerous datasets exist for face recognition, they are often unsuitable for assessing performance on social media images due to differences in image quality, context, and variability. To address this gap, we developed the dataset, named SocioFaceSet, tailored to the unique challenges posed by social media imagery.

3.1 SocioFaceSet

SocioFaceSet stands for the Social Media Face Recognition Dataset, specifically curated to address the unique characteristics and challenges of facial recognition in social media posts.

Data Collection: The SocioFaceSet comprises facial data from twelve individuals, including both male and female subjects to ensure gender diversity in the proposed dataset. In the paper, approximately 10,000 images are captured from each individual, resulting in a total dataset size of 100,000 images. These images are collected under various conditions to ensure robustness and generalizability of the model. The conditions included:

- **With and Without Glasses**: Images of individuals wearing glasses and without glasses to account for variations in appearance.
- **Noise-Free Background**: Images with a clean, distraction-free background to focus on facial features.
- **Pose variant**: Images are captured with different angles of the face, including profile and semi-profile views.
- **Lighting Conditions**: Images taken in low light, normal light, and bright light conditions to simulate typical variations found in social media posts.

Data Labeling: We employed a systematic approach for labeling the images in order to ensure high-quality annotations and maintain reliability of the dataset. Individuals in the dataset are anonymized and labeled as Person 1 through Person 10 to protect privacy of the persons. Each image is manually labeled by team members with utmost care.

- **Number of Annotators:** Each individual in the dataset is the annotator of their own images, ensuring each individual's true appearance, therefore highly accurate and reflective annotation.
- **Annotation Guidelines:** Annotators followed a comprehensive set of guidelines developed specifically for this dataset. The guidelines include detailed instructions on how to label facial features, handle variations in image quality, and account for different lighting and pose conditions.
- **Inter-Annotator Agreement:** Since each person annotates their own images, inter-annotator agreement across different annotators is not applicable here. In order to ensure annotation quality, we apply majority voting mechanism, where multiple annotators are involved, and final label is determined based on the majority opinion.

Additionally, our proposed model demonstrates robust performance on other datasets, indicating that the SocioFaceSet is not biased with respect to race, gender, or other demographic factors. This comparative performance supports generalization ability of the dataset and reinforces its applicability across diverse scenarios.

Table 1. SocioFaceSet Data Distribution

Person ID	Number of Images
Person 1	10,312
Person 2	10,752
Person 3	10,012
Person 4	10,360
Person 5	10,452
Person 6	11,211
Person 7	10,152
Person 8	11.286
Person 9	10,154
Person 10	10,750

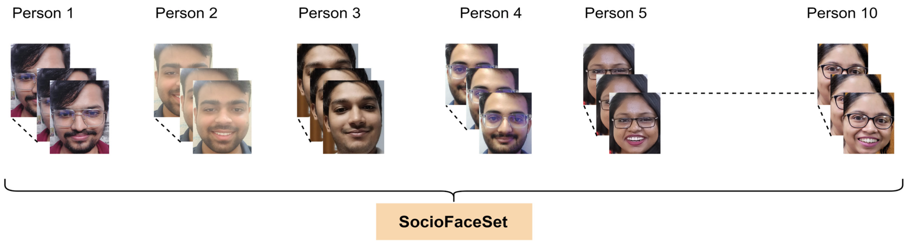


Fig. 1. SocioFaceSet visual overview

3.2 Emotion Detection Dataset

For emotion detection, we utilized the publicly available FER2013 [16] dataset. This dataset is widely recognized for its comprehensive coverage of facial emotions and consists of 35,887 images categorized into seven emotion classes: anger, disgust, fear, happiness, sadness, surprise, and neutral. Table 2 provides a summary of the dataset distribution. The use of FER2013 allows us to leverage a standardized dataset for emotion recognition, which is critical for benchmarking and validating the performance of our model.

Table 2. FER2013 Dataset Distribution

Emotion	Number of Images
Anger	4,721
Disgust	547
Fear	3,513
Happiness	8,989
Sadness	6,077
Surprise	4,002
Neutral	6,038

3.3 Comparison of SocioFaceSet with other SOTA Face Datasets

The SocioFaceSet aims to capture the diverse facial features for face recognition in various social context. The training dataset emphasizes diversity in terms of ethnicity, age groups, gender and social contexts to improve the generalizability of the models. Overall, the dataset tackles the challenge of recognizing faces in less controlled environments compared to studio settings, where lighting, angles, and expressions could be varied to a large extent. On the other hand, CMU-MOSEI (Multimodal Opinion Sentiment and Emotion Intensity) is a dataset primarily used for sentiment and emotion recognition in multimodal content, including video and audio. It contains over 23,000 annotated video segments from various online sources that are annotated for detecting sentiment and emotion intensity. The dataset includes diverse content in terms of topics, speakers, and emotions but is more focused on video-based data rather than still images. CMU-MOSEI addresses the challenge of understanding emotions in multimodal content, making it suitable for applications that require audio-visual data. Similarly, CelebA is a large-scale face attributes dataset used for tasks like facial recognition, attribute prediction, and generative modeling. It contains over 200,000 celebrity images, annotated with 40 different facial attributes, such as smiling, wearing glasses, etc. Though the dataset includes a wide variety of attributes, it is less focused on the emotional context and more on physical attributes. CelebA is used extensively for benchmarking the models on face recognition and attribute prediction, but it does not specifically focus on emotional expression or social context.

Thus, SocioFaceSet is more specialized for face recognition considering social media images, which differs from CMU-MOSEI's that focus on multimodal content while CelebA's focus on facial attributes. SocioFaceSet is tailored for social media contexts, making it unique in capturing the informal, varied nature of social media expressions, unlike CMU-MOSEI, which is broader in scope, and CelebA, which is more general-purpose.

4 Methodology

In this work, we propose a deep learning architecture called SocialFaceEmoNet (Figure 2), designed to perform both face recognition and emotion detection from social media image posts. This section details the steps involved in our methodology, including face detection, feature extraction, and classification.

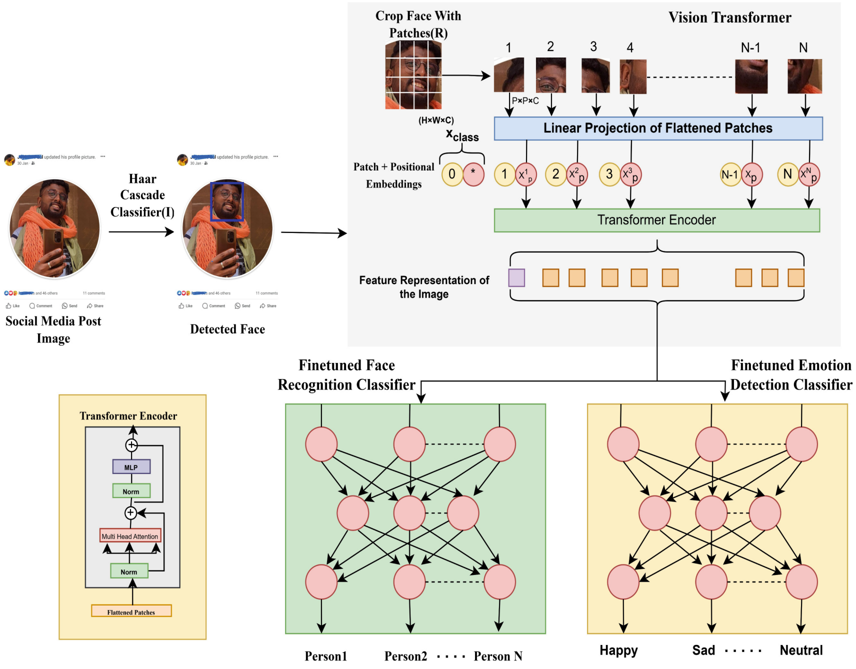


Fig. 2. SocialFaceEmoNet Architecture

4.1 Face Detection

The first stage of our SocialFaceEmoNet architecture is face detection, where we use the Haar Cascade classifier to identify faces in images captured from social media. The Haar Cascade classifier utilizes Haar-like features, which are digital image characteristics employed in the process of object identification. These features are computed by subtracting the total pixel intensities in rectangular sections, capturing essential contrasts that highlight facial features.

Given an image I and a rectangular region feature R, the Haar feature can be computed as:

$$f(I, R) = \sum_{(x,y) \in R_1} I(x, y) - \sum_{(x,y) \in R_2} I(x, y)$$

where R_1 and R_2 are two adjacent rectangular regions within the detection window, and $I(x, y)$ is the pixel intensity at coordinates (x, y). The classifier then uses a series of these Haar features, combined with the AdaBoost algorithm, to create a strong classifier from multiple weak classifiers. AdaBoost selects the most critical features and combines them to improve the detection accuracy.

The face detection process can be mathematically described as:

$$\text{Detected Faces} = \text{Haar Cascade Classifier}(I)$$

where I is the input image, and the output is the set of coordinates of the detected faces, which define the regions R. These coordinates are used to crop the face regions from the original image, ensuring that only relevant facial regions are processed in the subsequent steps.

4.2 Feature Extraction Using Vision Transformer (ViT)

In order to extract features, we use the Vision Transformer (ViT), which has shown better performance in comparison to conventional CNN models such as VGG16 and ResNet50. The ViT model analyses images by dividing them into patches and considering each patch as a token within a sequence, similar to the tokens used in NLP.

Given an input image R with dimensions $H \times W \times C$, where H represents the height, W represents the width, and C represents the number of channels, the image is partitioned into N patches. Each patch x_p with dimensions $P \times P \times C$ is converted into a flattened vector and then linearly projected to generate patch embeddings:

$$x_p = \text{Linear}(x_{\text{patch}})$$

Subsequently, the patch embeddings are merged with positional embeddings in order to preserve spatial information:

$$z_0 = [x_{\text{class}}; x_p^1; x_p^2; \ldots; x_p^N] + E_{\text{pos}}$$

where x_{class} is a learnable class embedding and E_{pos} represents the positional embeddings.

Subsequently, every set of embeddings undergoes processing via Transformer encoder layers, each comprising of multi-head self-attention (MSA) and feed-forward neural networks (FFN).

$$\text{MSA}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, \ldots, \text{head}_h)W_O$$

$$\text{FFN}(x) = \text{ReLU}(xW_1 + b_1)W_2 + b_2$$

where the learned weight matrices are denoted by W_O, W_1, and W_2, and the queries, keys, and values derived from the input embeddings are represented by Q, K, and V vectors, respectively.

The feature representation of the image is derived from the output of the Transformer encoder layers, which are as follows:

$$z_L = \text{TransformerEncoders}(z_0)$$

4.3 Face Recognizer Architecture

For face recognition, we employed a ViT [27], a well-established robust attention-based framework used for extracting features hierarchically. The design enables the model to capture both low-level and high-level features, crucial for accurate face recognition across varied and complex social media images. The model was pre-trained on CelebA [29] containing a diverse set of facial images, and then fine-tuned on the SocioFaceSet to adapt to the unique characteristics of the media domain. Transfer learning allows the model to leverage pretrained weights from a large dataset and fine-tune them on our specific dataset, ensuring better performance.

4.4 Emotion Detection Architecture

For emotion classification, we used ViT in order to handle both the spatial and temporal information of facial expressions. The multi-head attentions are responsible for extracting spatial features from individual image frames, while the temporal dependencies is taken care of by patch-wise processing of sequences of the features. We incorporated the attention mechanism to focus the model on the most relevant parts of the facial expressions, improving its ability to discern subtle emotional cues. The model was initially pre-trained on the AffectNet [17], which contains a wide range of annotated facial expressions, and subsequently fine-tuned on the FER2013 [23].

4.5 Pre-Training and Fine-Tuning

Face recognition and emotion detection models were pre-trained on large, well-established datasets before being fine-tuned on the SocioFaceSet. The face recognizer was pre-trained on the CelebA, which includes over 200,000 celebrity images across a variety of poses and lighting conditions. The emotion classifier was pre-trained on AffectNet, one of the largest facial expression dataset available, containing over 1 million facial images with manually annotated emotion labels. Fine-tuning on SocioFaceSet allows us to adapt the model to the specific challenges, presented by social media images, such as varying camera angles, lighting conditions, and spontaneous facial expressions.

4.6 SocioFaceEmoNet: Combined Architecture

The overall SocialFaceEmoNet architecture combines the face recognition and emotion detection modules into a unified pipeline. The steps involved are as follows:

- **Face Detection**: Apply Haar Cascade classifier to detect the faces available from social media images.
- **Feature Extraction**: ViT model is used for feature extraction from the images.

- **Face Recognition**: Use the pretrained (on CelebA dataset) classifier, fine-tuned on SocioFaceSet to identify the person in the image.
- **Emotion Detection**: Use the second classifier, fine-tuned on the FER2013 dataset, to detect the emotion of the identified person.

By integrating these components, SocialFaceEmoNet achieves robust performance in recognizing faces and detecting emotions from the unstructured and variable-quality images, typically found in social media posts. Our approach integrates well-established techniques applicable in face recognition and emotion classification. The proposed work is innovative due to its ability in handling challenging issues, often arise in social media images such as varying lightning conditions, informal setting, and a wide range of emotional expressions. By tailoring ViT framework, specifically for this domain, we propose a more robust and accurate solution in real-world social media contexts. Algorithm 1 describes the tasks of facial recognition and emotion detection using SocialFaceEmoNet model. This approach ensures high accuracy and efficiency, making it suitable for real-world applications in social media analytics.

Algorithm 1: SocialFaceEmoNet Algoritham for Face Recognition and Emotion Detection

```
Input: Social media image post $I$
Output: Identified person $P$ and detected emotion $E$
Face Detection:
    faces ← HaarCascadeClassifier.detectFaces($I$);
    foreach face $f$ in faces do
        $I_f$ ← cropFace($I, f$);
        Feature Extraction:
            $P$ ← divideIntoPatches($I_f$);
            patchEmbeddings ← LinearProjection($P$);
            $z_0$ ← combineWithPositionalEmbeddings(patchEmbeddings);
            $z_L$ ← TransformerEncoderLayers($z_0$);
        Face Recognition:
            $P$ ← FinetunedFaceRecognitionClassifier;
        Emotion Detection:
            $E$ ← FinetunedEmotionDetectionClassifier;
    end
    return $P, E$;
```

5 Experimental Analysis

In order to assess the effectiveness of the proposed SocialFaceEmoNet model for face recognition and emotion detection in social media photos, we provide a thorough experimental study in this section. The performance of the model is evaluated using several metrics, including accuracy, precision, recall, and F1-score, which are essential for understanding both the overall effectiveness and the reliability of the model's predictions.

5.1 Experimental Results

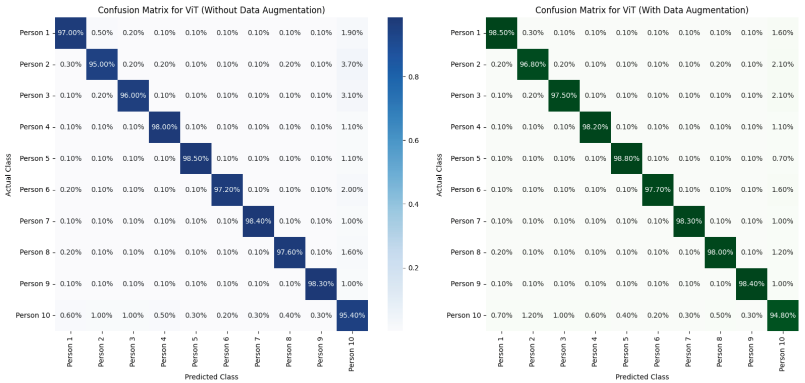
SocialFaceEmoNet model has been implemented using the PyTorch deep learning framework. We use pre-trained models of ViT for feature extraction and a classifier for emotion detection. The face recognition classifier is fine-tuned using transfer learning on the SocioFaceSet. For enhanced performance, we applied data augmentation techniques, including random rotation, horizontal flipping, and color jittering, to the SocioFaceSet during training. Table 3 provides experimental performances of face recognition and emotion detection using different backbone models on our proposed SocialFaceEmoNet architecture.

Table 3. Performance Evaluation of SocialFaceEmoNet for Face Recognition and Emotion Detection Using SocioFaceSet with Different Backbone Models

Model	Data Augmentation	Accuracy	Precision	Recall	F1-score
Face Recognition Performance					
VGG16	Without	0.93	0.92	0.94	0.93
	With	0.95	0.94	0.96	0.95
ResNet50	Without	0.95	0.94	0.96	0.95
	With	0.97	0.96	0.98	0.97
ViT	Without	0.97	0.96	0.98	0.97
	With	**0.98**	**0.97**	**0.99**	**0.98**
Emotion Detection Performance					
VGG16	Without	0.72	0.71	0.73	0.72
	With	0.73	0.73	0.75	0.74
ResNet50	Without	0.74	0.73	0.75	0.74
	With	0.76	0.75	0.77	0.76
ViT	Without	0.78	0.78	**0.79**	0.78
	With	**0.79**	**0.79**	0.78	**0.79**

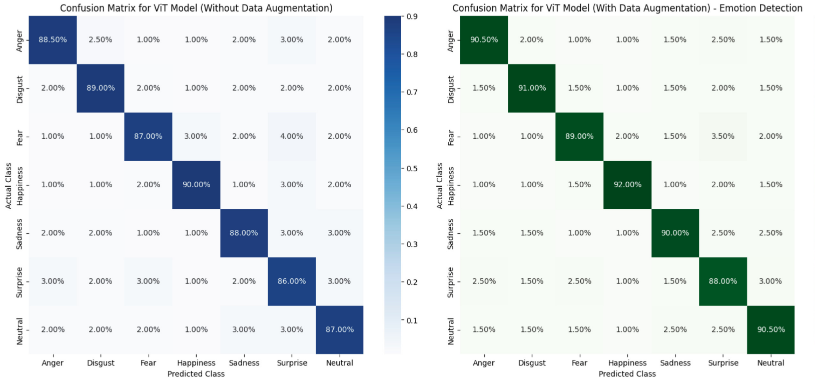
The experimental results demonstrate the effectiveness of the SocialFaceEmoNet architecture in accurately recognizing faces and detecting emotions from social media images.

ViT consistently demonstrated superior performance over VGG16 and ResNet50, emerging as the best backbone model for both face recognition and emotion detection in our proposed SocialFaceEmoNet architecture. Data augmentation further enhanced ViT's performance. Table 4 presents the ROC - AUC scores for face recognition and emotion detection, highlighting ViT's effectiveness in both tasks. As shown in Table 5, our architecture achieves superior



(a) (b)

Fig. 3. Confusion Matrix for Face Recognition on SocioFaceSet Using Best Backbone Models (a) Without Data Augmentation (b) With Data Augmentation

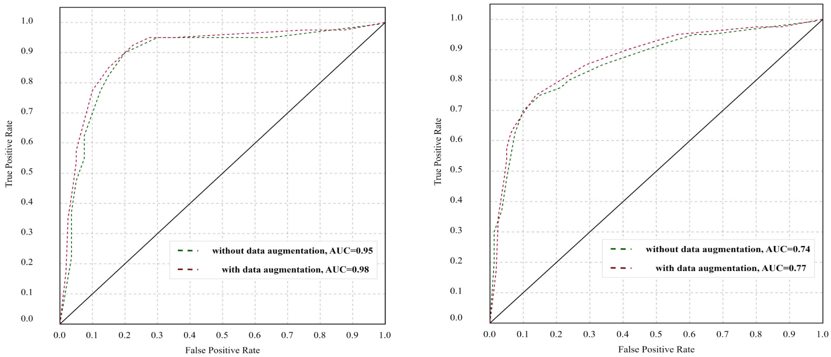


(a) (b)

Fig. 4. Confusion Matrix for Emotion Detection on SocioFaceSet Using Best Backbone Models (a) Without Data Augmentation (b) With Data Augmentation

Table 4. ROC AUC Scores for Face Recognition and Emotion Detection using Social-FaceEmoNet on SocioFaceSet

Model	Task	With Data Augmentation	Without Data Augmentation
VGG16	Face Recognition	0.93	0.88
	Emotion Detection	0.72	0.69
ResNet50	Face Recognition	0.95	0.90
	Emotion Detection	0.74	0.71
ViT	Face Recognition	**0.98**	**0.95**
	Emotion Detection	**0.77**	**0.74**



(a) (b)

Fig. 5. ROC Curve Using SocialFaceEmoNet on SocioFaceSet (a) Face Recognition (b) Emotion Detection

performance across multiple datasets, and state-of-the-art models also perform well on proposed SocioFaceSet. The application of data augmentation techniques significantly improved the performance of all models by enhancing their generalization ability and reducing overfitting. This improvement was most noticeable in ViT, which effectively leveraged the increased diversity in the training data to learn more robust features. Figures 3 illustrates the confusion matrices, and Figure 5 shows ROC Curve for the best-performing model on our SocioFaceSet, exhibiting ViT's strong performance.

6 Discussion

The proposed model, "SocialFaceEmoNet" can significantly contribute to various domains, particularly in social media analytics and digital marketing. In social media, the model can be employed to automatically detect and analyze the emotional responses of users in real-time, providing insights into public sentiment during events, campaigns, or crisis. This capability is crucial for brands and organizations aiming to gauge public perception and respond appropriately. In digital marketing, "SocioFaceEmoNet" can be integrated into targeted advertising systems to personalize content based on the emotional state of users, enhancing engagement and conversion rates. Additionally, it can be used in customer service to identify and address customer dissatisfaction by analyzing their facial expressions during video interactions, thereby improving customer experience and loyalty. During political campaigns "SocialFaceEmoNet" can be used to monitor audience reactions to speeches or debates, helping campaign managers to tailor their messages. Similarly, in live events like sports, the model could analyze fan reactions providing broadcasters with valuable data enhance viewer engagement.

Table 5. Comparison of State-of-the-Art Face Recognition and Emotion Detection Models across various datasets

Model	Dataset	Accuracy	Precision	Recall	F1-Score
State-of-the-Art Face Recognition Models					
SocialFaceEmoNet	SocioFaceSet	**0.983**	**0.975**	**0.989**	**0.981**
	VGGFace2	0.960	**0.972**	**0.975**	**0.973**
DeepFace	SocioFaceSet	0.979	0.982	0.974	0.977
	VGGFace2	**0.961**	0.962	0.9721	0.971
FaceNet	SocioFaceSet	0.945	0.948	0.942	0.932
	VGGFace2	0.942	0.951	0.949	0.952
State-of-the-Art Emotion Detection Models					
SocialFaceEmoNet	AffectNet	**0.761**	0.742	**0.762**	0.752
	RAF-DB	0.710	0.702	0.699	0.701
ViT-Base	AffectNet	0.742	0.745	0.736	0.746
	RAF-DB	0.751	0.762	**0.752**	**0.761**
VGGFace2	AffectNet	0.752	**0.762**	0.758	**0.763**
	RAF-DB	0.702	0.712	0.695	0.705

7 Conclusion

In this paper, we propose SocialFaceEmoNet, a deep architecture designed for face recognition and emotion detection using social media posts. Leveraging deep learning models such as VGG16, ResNet50, and Vision Transformer (ViT), our proposed architecture demonstrates competitive performance on both tasks. Our findings reveal that ViT, as the backbone model, outperforms traditional CNN architectures (VGG16, ResNet50) in both face recognition and emotion detection, achieving up to 98% accuracy in face recognition and 79% accuracy in emotion detection. Data augmentation techniques, including random rotation, horizontal flipping, and color jittering, significantly enhance the robustness and generalization capability of our models.

Despite these promising results, our study has limitations, including the size and diversity of our SocioFaceSet and the computational complexity of ViT. Future work should focus on expanding the dataset to include a broader population and optimizing ViT for deployment on resource-constrained devices. Additionally, exploring multimodal learning approaches and addressing ethical considerations related to social media data use are crucial for advancing this research area. Another promising area is the exploration of transfer learning techniques using larger and more diverse datasets, such as those from different cultural backgrounds, to improve the generalization capabilities of the model. Additionally, applying the model in other domains like healthcare, for detecting early signs of mental health issues through facial analysis, represents a valuable extension of this work. Overall, our study provides a strong foundation for further advancements in face recognition and emotion detection in social media contexts, with ViT demonstrating superior performance as the backbone model.

References

1. Ojala, T., Pietikäinen, M., Harwood, D.: A comparative study of texture measures with classification based on featured distributions. Pattern Recogn. **29**(1), 51–59 (1996)
2. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012
3. Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint* arXiv:1409.1556, 2014
4. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015
5. Gary B Huang, Marwan Mattar, Tamara Berg, and Eric Learned-Miller. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. In *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition*, 2008
6. Ira Kemelmacher-Shlizerman, Steven M Seitz, Daniel Miller, and Evan Brossard. The megaface benchmark: 1 million faces for recognition at scale. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4873–4882, 2016
7. Guo, Y., Zhang, L., Hu, Y., He, X., Gao, J.: MS-Celeb-1M: A Dataset and Benchmark for Large-Scale Face Recognition. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9907, pp. 87–102. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46487-9_6
8. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016
9. Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019
10. Ali Bukar and Hassan Ugail. Convnet features for age estimation. 07 2017
11. Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015
12. Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4690–4699, 2019
13. Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1717–1724, 2014
14. Jayanta Paul, Rajat S Bhowmick, Riom Sen, Dwaipayan Ray, Suman S Manjhi, Soumya Sen, and Biplab K Sikdar. Evaluation of face recognition schemes for low-computation iot system design. In *2020 24th International Symposium on VLSI Design and Test (VDAT)*, pages 1–6. IEEE, 2020
15. Yichuan Tang. Deep learning using linear support vector machines. *arXiv preprint* arXiv:1306.0239, 2013

16. Agrawal, A., Mittal, N.: Using cnn for facial expression recognition: a study of the effects of kernel size and number of filters on accuracy. Vis. Comput. **36**(2), 405–412 (2020)
17. Ali Mollahosseini, Behzad Hasani, and Mohammad H Mahoor. Affectnet: A database for facial expression, valence, and arousal computing in the wild. *IEEE Transactions on Affective Computing*, 10(1):18–31, 2017
18. Bo-Kyeong Kim, Hwaran Lee, Jihyeon Roh, and Soo-Young Lee. Hierarchical committee of deep cnns with exponentially-weighted decision fusion for static facial expression recognition. In *Proceedings of the 2015 ACM on international conference on multimodal interaction*, pages 427–434, 2015
19. Zhang, K., Zhang, Z., Li, Z., Qiao, Yu.: Joint face detection and alignment using multitask cascaded convolutional networks. IEEE Signal Process. Lett. **23**(10), 1499–1503 (2016)
20. Tara N Sainath, Oriol Vinyals, Andrew Senior, and Haşim Sak. Convolutional, long short-term memory, fully connected deep neural networks. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4580–4584. Ieee, 2015
21. Emad Barsoum, Cha Zhang, Cristian Canton Ferrer, and Zhengyou Zhang. Training deep networks for facial expression recognition with crowd-sourced label distribution. In *Proceedings of the 18th ACM international conference on multimodal interaction*, pages 279–283, 2016
22. Shan, L., Deng, W.: Reliable crowdsourcing and deep locality-preserving learning for unconstrained facial expression recognition. IEEE Trans. Image Process. **28**(1), 356–370 (2018)
23. Goodfellow, I.J., Erhan, D., Carrier, P.L., Courville, A., Mirza, M., Hamner, B., Cukierski, W., Tang, Y., Thaler, D., Lee, D.-H., Zhou, Y., Ramaiah, C., Feng, F., Li, R., Wang, X., Athanasakis, D., Shawe-Taylor, J., Milakov, M., Park, J., Ionescu, R., Popescu, M., Grozea, C., Bergstra, J., Xie, J., Romaszko, L., Xu, B., Chuang, Z., Bengio, Y.: Challenges in Representation Learning: A Report on Three Machine Learning Contests. In: Lee, M., Hirose, A., Hou, Z.-G., Kil, R.M. (eds.) ICONIP 2013. LNCS, vol. 8228, pp. 117–124. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-42051-1_16
24. Li, S., Deng, W.: A deeper look at facial expression dataset bias. IEEE Trans. Affect. Comput. **13**(2), 881–893 (2020)
25. Jayanta Paul, Anuska Roy, Siddhartha Mallick, and Jaya Sil. A comparative study of deep learning-based face recognition and emotion detection techniques using social media customized cartoon post. In *International Conference on Computational Intelligence in Pattern Recognition*, pages 401–411. Springer, 2022
26. Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint* arXiv:1409.1259, 2014
27. Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint* arXiv:2010.11929, 2020

28. Soujanya Poria, Erik Cambria, Devamanyu Hazarika, Navonil Majumder, Amir Zadeh, and Louis-Philippe Morency. Context-dependent sentiment analysis in user-generated videos. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 873–883, 2017
29. Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Large-scale celebfaces attributes (celeba) dataset. *Retrieved August*, 15(2018):11, 2018

POCN: Long Term Propensity, Occasional Exploration, Contention Relevance and Neighborhood Diversity Driven Prime Video Page Composition

Venkataramana Kini(✉), Ravi Divvela, Devendra Yadav, Unmesh Padke, and Narayanan Sadagopan

Amazon LLC, Seattle, USA
venkataramana.kini@gmail.com

Abstract. The paper introduces the Prime Video Page Composition framework (POCN), comprising Long Term Customer Propensity, Occasional Exploration, Contention Relevance, and Neighborhood Diversity, for optimal carousel positioning on the homepage. The framework addresses the challenge of balancing diverse customer and business considerations, including different offers (Prime/individual purchase/3P) and content types (Movie/TV/Sports). To achieve this, a Transformer model predicts long-term customer propensities for various offer/content types, integrating discounted impression UCB for occasional exploration. A novel neural network captures contention between carousels, and the framework combines long-term propensity, occasional exploration, and contention relevance linearly, with added neighborhood diversity. The proposed approach demonstrates a significant (4.6%) improvement in customer engagement metrics in A/B experiments.

Keywords: Customer Long Term Propensity · Occasional Exploration · Contention Relevance · Neighborhood Diversity · Page Composition

1 Introduction

Amazon Prime Video, a preferred streaming service, stands out for its diverse content, including unique offerings like Prime, Prime Video Channels, and TVOD (Transaction Video On Demand). As an ultimate "Entertainment Hub," it epitomizes Amazon's commitment to providing a convenient and comprehensive viewing experience. Prime Video's homepage features thematic carousels like "Recommended for You" and "Trending Titles," with approximately 20 carousels in total. Balancing customer preferences and encouraging exploration poses challenges, particularly in selecting the top 5-7 carousels that receive the most attention.

The current Page Composition module uses heuristics and rules atop machine learning models for page composition, presenting challenges in balancing multiple

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 328–342, 2025.
https://doi.org/10.1007/978-3-031-78383-8_22

offers and content types. The module lacks consideration for the customer's overall long-term interest in specific offers or content types, doesn't have the flexibility of meaningfully perform exploration. Also, for carousel ranking it is not able to compare and contrast between different carousels with different offers/content types and perform diversification of carousels in the neighborhood of carousels.

The customer interest in offers/content types is captured through past interactions or affinity, is utilized to predict future interactions, termed as propensity. Propensity prediction, focused on longer-term customer behavior, addresses seasonal variations by incorporating data from different seasons over past year for accurate predictions. For instance, during holidays, customers may interest in specific TVOD titles, while summer vacation could prompt parents to subscribe to channels for children.

We propose a Transformer based model to capture customer's long term offer/content type propensity. The model predicts customer propensity based on historical interactions, but to encourage "occasional exploration" for unexplored content in the recent past, the paper proposes a modified Upper Confidence Bound (UCB) method using discounted impressions.

For capturing Customer's interest in specific carousels we propose a neural network model that poses competition between different carousels from different offer / content types as simultaneous multiple binary problem.

The overall contribution of this work is along four dimensions (discussed in section 3). The proposed approach combines long-term customer propensity, occasional exploration, and contesting carousel relevance scores to select top carousels. Pareto front analysis is used for guiding decisions based on share of voice for different offers/content types and customer engagement. Additionally, a customized Maximal Marginal Relevance (MMR) method is proposed for diversifying carousels both at the overall page level and within carousel neighborhoods. These methods we validate both in offline and online settings.

2 Related Work

Page composition, addressed in different industrial settings like Amazon [6] [10], LinkedIn [1], and Netflix [2], differs from traditional user-item click-through rate prediction. Unlike individual recommendations, it aims for diverse pages to enhance conversion probability, aligning with findings that diverse recommendations improve user conversion [5]. Similar to our problem, [6] addresses recommending a diverse set of items in multiple categories with business constraints, utilizing a Bayesian linear model [9] and submodular diversity. However, it lacks enforcement of diversity across recommended items. In contrast, previous works like [12] focus on enforcing diversity but treat each category equally, without considering user preferences. Our approach extends these methods to account for a user's propensity towards each diversity category, ensuring personalized recommendations by incorporating individual preferences [13].

In the realm of learning optimal recommendation policies with multiple objectives, [18] addresses a similar problem by optimizing user fairness, item fairness, and diversity using offline objectives. Similarly, [11] propose Pareto-efficient methods for fairness across user subgroups, business revenue, and relevance. However, our challenge lies in the difficulty of estimating metrics like business revenue and long-term user retention offline accurately. Thus, we aim to learn a method that produces solutions on the multi-objective Pareto front to narrow down candidates for A/B testing.

To capture customer propensity towards different offers, we propose a novel method using Transformers, commonly employed in Natural Language Processing tasks [16]. Differing from previous attempts [19], our approach combines an Encoder network to summarize the first 6 months of interactions and occasional exploration using UCB [14]. This exploration method, akin to [8], uniquely combines sliding window and discounting on impressions, avoiding discounting on propensity scores as the transformer model inherently attends to different interactions across time.

We propose a contesting carousel selection model for capturing short-term customer interest, inspired by multi-output models [21] and multi-label settings [17]. To diversify carousels in the neighborhood, we employ a customized Neighborhood MMR criterion [4]. The POCN framework seamlessly integrates Propensity, Occasional Exploration, Contention Relevance, and Neighborhood Diversity, addressing critical customer/business challenges and offering a novel approach compared to existing industrial implementations [1,2,6].

3 Proposed Approach

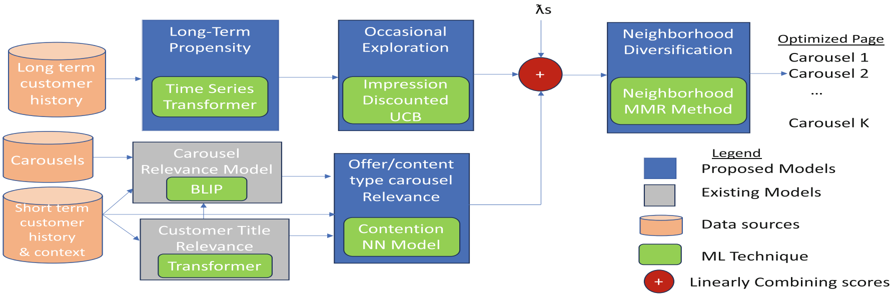


Fig. 1. Flow of proposed approach for page composition

The proposed approach involves a multi-step process. The Long Term Propensity model, employing a customized Transformer architecture, generates predictions for the next month based on the customer's long-term history through an offline job. Occasional exploration is conducted using the impression discounted Upper Confidence Bound (UCB) method during online inference. Personalized carousels are scored using the BLIP model, a binary classification model, and individual customer interests are determined by the customer title relevance model. The subsequent carousel contention model, employing neural networks, ranks carousels at the offer/content type level based on immediate customer context. The scores from the long-term customer propensity and short-term carousel customer relevance are linearly combined, and diversification is applied to penalize neighboring carousels of the same offer/content type. Each module addresses specific practical requirements.

3.1 Customer Long Term Propensity Model

We develop Customer Long Term Propensity Model using customer's interaction history. We pose Customer Long Term Propensity Model as a multivariate time series forecasting model. I.e. given customer's last 12 months of interaction history, we would like to predict next one month interactions. However, the traditional time series models such as ARIMA (Auto-Regressive Integrated Moving Average model) model will not be able to attend to long term customer interaction history effectively. Also, models like ARIMA will not be able train on time series from multiple customer time series data.

The traditional machine learning models such as recurrent neural networks (RNNs) implemented using LSTMs are not capable of capturing the relatively longer term dependencies in customer behavior. We propose to develop generative Transformer model that uses attention mechanism to model non-Markovian dynamics. We make use of Transformer model as a multivariate time series forecasting model. The proposed Transformer model for customer propensity prediction, as depicted in Figure 2, consists of an encoder and decoder following the structure in [16]. Unlike traditional machine translation, the first 6 months of customer history serve as encoder input, the subsequent 6 months as decoder input, and the last month as decoder output. This approach, similar to [20], utilizes multi-output regression with MSE loss, effectively representing the initial 6

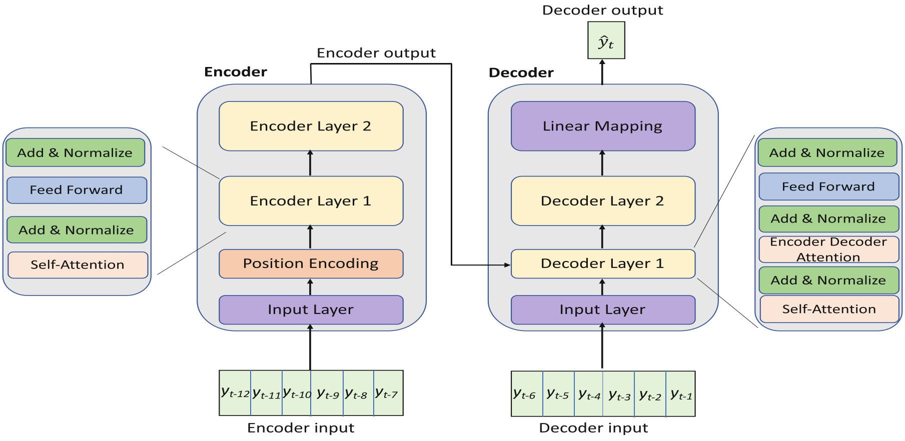


Fig. 2. Transformer based propensity model

months in the time series fed into the decoder to predict the last month's data point.

Let's suppose we have K different offer/content type combinations and k is specific offer/content type that's made up of offer o and content type c. At each time step (monthly), we aggregate number of streams w.r.t. K different offer/content types. We represent a customer u's streams for a month t by $\mathbf{y}_{ut} = [y_{ukt}]_{k=1}^{K}$. The propensity model predicts customer's streams from different offer/content type for next month. Next, we normalize individual offer/content propensity by sum of propensities of all offers so that propensities sum to 1 for a given user. The normalized propensity for an offer/content type is represented as P_k^p (we will not explicitly mention user for brevity in further discussion). Also, we can aggregate propensities at individual offer level: P_o^p (such as Prime vs. Non-Prime) or content type level: P_c^p (such as Movie vs. TV shows). One alternative transformer architecture is to use decoder only. We compare the performance of proposed approach with decoder only Transformer in Experiments section.

3.2 Occasional Exploration on Propensities

The propensities generated by propensity models tries to extrapolate the customer behavior in the past. However, customers typically not only interested in the offer and content types that they are already streaming from but also from other offer / content types once in a while. We model this customer behavior using sliding window discounted Upper Confidence bound on top of the propensity scores generated by propensity model. The sliding window used in this work is one week. As we intend customers to explore different offer/content types with occasional frequency of at least once in a week. We perform discounting on the impressions that were served to the customers during last one week as follows:

$$N_T(d,k) = \sum_{t=1}^{T} d^{T-t} n_{t,k} \tag{1}$$

Here, T is the current day, $d \in (0,1)$ is discounting factor, $n_{t,i}$ is number of times offer/content type k was recommended to specific customer at time t. The equation for modified propensity w.r.t. offer/content type based on sliding window discounted UCB (SWD-UCB) is defined as:

$$\bar{P}_k^p = P_k^p + c\sqrt{\frac{\bar{N}_T(d)}{N_T(d,k)}} \quad where \quad \bar{N}_T(d) = \sum_{k=1}^{K} N_T(d,k) \tag{2}$$

where $\bar{N}_T(d)$ is the sum of discounted impression counts over all offer/content types:

3.3 Contesting Carousel Selection Model

The page composition has to consider not only absolute relevance of a given carousel, it has to compare among the candidates from each of the offer/content types. Essentially, different offer/content type carousels contest against each other for a given position on page. This contention can be modeled better when a model sees all the candidates and their outcomes simultaneously. Towards this we propose a new neural network architecture as shown in Figure 3.

The input vector for the model is constructed based on customer/contextual features and 6 candidate carousel features belonging to different offer/content types. These features include the title scores from (customer title relevance) model. The target labels are constructed based on whether from a given carousel a title is streamed or not. The label for data point is of the form: $< 0, \ldots 1 \ldots 0 >$. I.e. one of the carousels from which title is streamed is set to 1 and rest of label vector is 0. We consider contesting carousels for constructing data point only when there's at least one carousel has positive interaction by user as we can assign the winner and help in discriminating positive and negative interactions. We concatenate the customer/contextual features with carousel features as shown in Figure 3. The loss function can be written as:

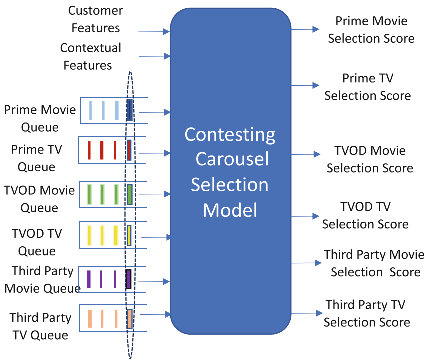


Fig. 3. Contesting Offer/Content types based relevance using multi-output neural net

$$L_S = \sum_{i=1}^{N} \sum_{k=1}^{K} [y_{i,k} \log(P^r_{i,k}) + (1 - y_{i,k})(1 - \log(P^r_{i,k})] \tag{3}$$

Here, $y_{i,k}$ is the part label for i^{th} data point and offer/content type k and $P^r_{i,k}$ is the corresponding predicted contesting relevance score. When training, we construct one data point for each page that was shown to customer. We select neighboring carousels with different offer/content types as contesting carousel. If a specific offer/content carousel was not recommended to customer, we have all input features set to 0 and label corresponding to that carousel set to 'unknown', we do not back propagate errors for such labels.

During inference, we maintain the priority queue as shown in Figure 3 for each of the offer / content types based on the score from carousel relevance model. This helps us to closely mimic how customers look at the page and their decision making process while choosing certain title to stream from competing carousels.

This model akin to having multiple binary classifiers w.r.t. different offer/content types, but they contest against each other to get next slot on the page. The carousel with highest predicted probability is considered to be best carousel as far as carousel selection goes. During inference time, the contention relevance scores are normalized so that they sum to 1 and can be compared to propensity with occasional exploration scores.

3.4 Combining Relevance and Long range Customer Propensity

Having estimated the customer's long term propensity with occasional exploration and contesting carousel scores, next we propose an approach to linearly combine customer's long term propensity and contesting relevance score of a carousel. This approach provides flexibility to combine the relevance and propensity to different degrees so that we can influence the appropriate SoV for a customer based on the their propensity. At offer level we can combine the contesting relevance and propensity using:

$$P_o^{r,p} = (1 - \lambda) P_{k \in o}^r + \lambda \bar{P}_o^p \tag{4}$$

Here, $P_{k \in o}^r$ represents the contesting relevance score of a carousel that is of offer o (irrespective of content type) and $\bar{P}_o^p$ propensity with occasional exploration score from the corresponding offer. $\lambda \in (0, 1)$ is the parameter that controls the influence of propensity and contention relevance. Note that we can perform this linear combination at offer or content type. Based on the combined score $P_o^{r,p}$, the top 20 carousels are selected. By varying λ in $(0, 1)$ range, we obtain different SoV for different offers and corresponding conversion (IPS normalized) and analyze this behavior using Pareto plots in offline experiments section.

Alternatively, we can consider offer and content type (Movie / TV Shows) simultaneously. Typically customers watch movies which doesn't require lot of commitment in terms of time. TV Shows customers watch over days or even weeks/months leading to higher engagement with platform. The propensities generated by the model are aggregated at the content type level and linearly combined with both relevance and offer level propensity:

$$P_{o,c}^{r,p} = (1 - \lambda_1 - \lambda_2) P_k^r + \lambda_1 \bar{P}_p^o + \lambda_2 \bar{P}_p^c \tag{5}$$

As before, λ_1 and λ_2 are weights in $0, 1$ range for controlling SoV of offer and content type respectively.

3.5 Neighborhood Diversification of Page

Having selected the carousels, next we diversify the neighboring carousels. The objective is to make sure that neighboring carousels are not from same offer / content type. Towards this we propose Neighborhood Maximum Marginal Relevance (MMR) [4] based approach while iteratively constructing page by adding one carousel at a time. We maintain a priority queue for each offer/content type by sorting carousels based on the contesting relevance score from section 3. We pop candidates from each of the queues and apply:

$$P_k^{r,n} = \gamma P_k^r + (1-\gamma)(1-P_k^{page})(1-P_k^{nbr}) \tag{6}$$

Here, γ is the parameter to control the amount of diversity and relevance. The higher value of γ favors relevancy and lower value of γ favors diversity. We use two terms for diversity: one at page level, P_k^{page} representing % of specific offer-content type carousels already added to page and other at neighboring carousel level (P_k^{nbr}: representing % of specific offer-content type carousels already added in previous n slots). The new term P_k^{nbr} introduced helps in favoring diversity not only at page level but also at carousel neighborhood level.

4 Experiments and Results

In this section, we validate the four components of the proposed Page Composition: Propensity model, Occasional exploration, Contending Relevance model for carousel selection and Neighborhood diversification w.r.t. offer/content types.

Experimental Setting: We perform all experiments on US market place. We validate each of the components separately and in combination. We sample from large number (Millions) pages over a month. The number of carousels in a page logs is few hundreds to a thousand.

4.1 Evaluation of Propensity Models

The main component of proposed Page Composition is propensity model. The propensity model helps to understand customer's future interests at offer/content type level. As we have posed the propensity model as a time series forecasting model, we make use of the metrics that are used in time series forecasting. We make use of Mean Absolute Percentage Error (MAPE) for measuring the performance of different propensity models. Since in our case we perform multivariate time series (w.r.t. different offers), we extend it to account for this aspect.

$$MAPE = \frac{100}{UK} \sum_u \sum_k \frac{|y_{uk} - \hat{y}_{uk}|}{y_{uk}} \tag{7}$$

Where U is number of customers, K is number of distinct offer/content types, y_{uk} and $\hat{y}_{uk}$ are actual and predicted offer/content type aggregated streams respectively. As discussed previously, we predict the customer's next one month offer propensity using past 12 months behavior. The performance of prediction is measured using MAPE, which provides scale free intuitive metric as scales of different customer's offer consumption can vary significantly.

We experiment with three different ML models for modeling customers long term offer propensity. The first one is feed forward neural network implemented using Multi-Layer Perceptron (MLP), the second one is a Recurrent Neural Network (RNN) implemented using Long and Short Term Memory (LSTM) and finally Transformer with regression loss function as discussed in proposed methods section.

We compare the ML propensity models against naive affinity (aggregation past streams) based method. It can be observed that in Table 1, MLP has 30.28 % lesser MAPE as compared to affinity baseline. LSTM has 55.42% lesser MAPE as compared to affinity baseline. Further, Transformer has 70.28% lesser MAPE as compared to affinity baseline. The proposed transformer model transformer model turns out to be best at predicting customer propensity. The Decoder only Transformer model (with all 12 months data used as input) also doesn't perform as good as proposed Transformer architecture with encoding first 6 months. The propensity estimated by model is raw count. We normalize the propensity by sum of all offer/content type propensities at customer level.

Table 1. Performance improvement of different ML models compared to baseline (affinity) for predicting propensity w.r.t. MAPE metric

Model	MAPE (%) Improvement
MLP	30.28
$LSTM$	55.42
$Transformer$	70.28
$Transformer_{Decoder}$	62.23

Table 2. Performance of different Offer/Content type Carousel Selection Models w.r.t. HR metric

Model	HR (%)
$BLIP_{prod}$	87.2
$NN_{binary-class}$	89.8
$NN_{Contest}$	96.5

4.2 Evaluation of Contesting Carousel Selection Model

The proposed carousel selection model performs contention across different offer /content types. Therefore, we compare it against simple NN based binary classification model without different offers being contesting against each other to understand the benefits of proposed contention model. We maintain same training settings such as number of layers, nodes etc. between these two models. Also, we compare against the BLIP (Bayesian LInear Probit regression model) [9] production model. The carousel with highest predicted probability is considered to be the best carousel for evaluating the carousel selection model perspective. If the selected carousel turns out to be streamed carousel, we can count it as success. The metric that we use for comparison is hit rate (HR):

$$HR = \frac{1}{N} \sum_{i=1}^{N} I(Highest\ probability\ carousel\ is\ streamed) \tag{8}$$

Here, N is the total number of samples in test dataset. The HR for 3 candidate methods is shown in Table 2. It can be seen that the proposed carousel selection model performs significantly better than individual NN binary classification model and production BLIP model. Note that we maintain priority queues at offer/content type level for these candidate methods as well.

4.3 Analysis of Balancing Offer/Content Type

Next, we combine the long term propensity scores generated at customer/offer level with the relevance score at carousel level as per equation 4. It can be seen that for different values of λ we can achieve different levels of non-Prime share of voice (Figure 8). Also, we can see that there is trade-off between the non-Prime SoV and Inverse Propensity Score (IPS) normalized conversion rate [7].

When we introduce exploration with $c = 0.01$ and $d = 0.95$ in equation (2), the SoV changes aggressively as expected as can be seen in Figure 4. We experimented with different values and chose $c = 0.01$ and $d = 0.95$, to arrive at acceptable trade-off between SoV for Prime and non-Prime offers and conversion.

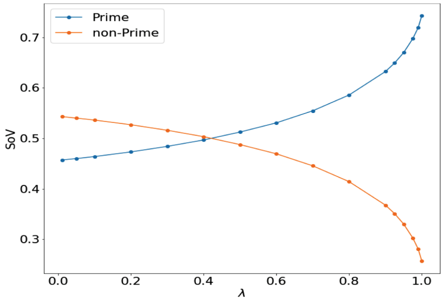


Fig. 4. Prime and non-Prime SoV for different values of λ with exploration on top of offer propensities

There is inherent trade-off between conversion and non-Prime SoV as typically customers stream the title that are available with Prime. This trade-off can be depicted using Pareto Frontier [11]. The overall trade-off between conversion and non-Prime SoV is represented using Pareto Frontier in Figure 6. Though we see direct trade-off between non-Prime SoV and conversion, in practice (through A/B experiment) we can study how much of real trade-off is there as we influence non-Prime SoV through customer's long term propensity towards non-Prime offers. From these plots we can infer that carousel level instantaneous conversion is not in synchronization with long term customer-offer level propensity. Similar analysis is performed at content type level to balance between the TV SoV and IPS normalized conversion. Due to space limitations, we do not include those plots.

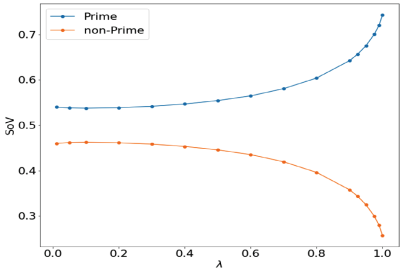


Fig. 5. Prime and non-Prime SoV for different values of λ without exploration of offers

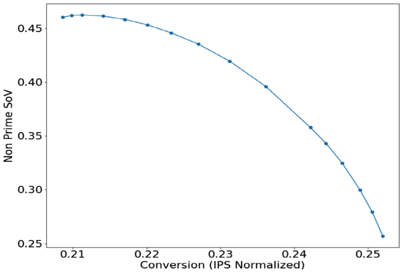


Fig. 6. Pareto front for different SoV vs. IPS normalized conversion

In practice we're not just interested in offer or content type trade-off with conversion separately as analyzed till now. The actual use case that we're interested in the combined effect of offer and content type trade-off with conversion.

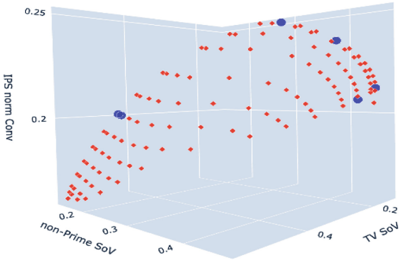


Fig. 7. Pareto surface for TV SoV, non-Prime SoV vs. IPS normalized conversion

To account for offer and content type together, we apply equation 5 to balance 3-way between carousel relevance, customer's long term offer propensity and customer's long term content type propensity. We vary the λ_1 and λ_2 parameter in steps forming a grid as shown in Figure 7. Then, we obtain similar SoV curves and Pareto Front. In this case Pareto front is 3-D surface which becomes difficult to visualize. Also, picking appropriate points on Pareto surface becomes tricky.

To address this challenge, we ran Skyline algorithm [3] to figure out points that are on the Pareto front. With that we are left with around 35 points out of 120 points generated by varying λ_1 and λ_2. As we can't perform A/B experiment on 35 treatments, we filter further based on threshold on IPS normalized conversion minimum of 0.20 as we don't want to compromise on customer relevance.

Then we pick final 6 treatments so that we cover broad range of TV / non-Prime SoV. This is achieved by picking points around min, max and median of TV / non-Prime SoV. These points are shown on the 3-D plot as blue points (Conversion > 0.20, TV SoV: (min: 0.12, median: 0.21, max: 0.45), non-Prime SoV: (min: 0.17, median: 0.39, max: 0.46)) and will be picked as final treatments for A/B experiment.

4.4 Evaluation of Neighborhood Diversification

As can be seen in Figure 1, once the candidates are chosen, the carousels are re-ranked by Neighborhood diversification method. The contesting carousel relevance score is used in Neighborhood Diversification. The objective is to diversify the carousels so that neighboring carousels are not from same offer / content type. Toward this, we apply equation 6 as we construct page by adding one carousel at a time to the page.

We define a new metric to measure the neighborhood diversity in terms of offer / content type:

$$ISL - Div = \frac{\sum_k (\# \ of \ distinct \ neighbors \ at \ rank \ k)}{\sum_k (max(\# \ of \ distinct \ neighbors \ at \ rank \ k))} \tag{9}$$

The traditional intra-list diversity ($IL - Div$) [15] and the $ISL - Div$ is shown in Figure 6 for different values of γ. It can be seen that as γ increases $IL - Div$ remains constant whereas $ISL - Div$ changes gradually suggesting change in offer/content type in the neighborhood.

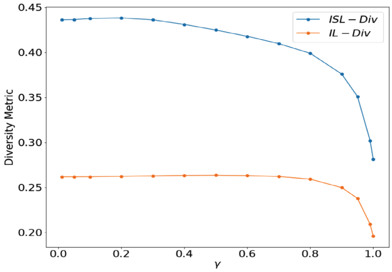


Fig. 8. Diversity of pages measured using $IL - Div$ and $ISL - Div$

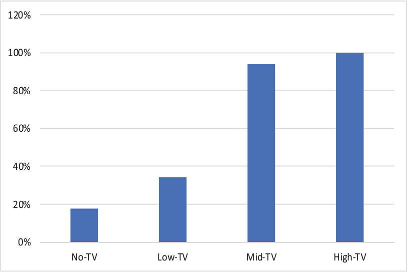


Fig. 9. Number (normalized) of TV recommendations for different TV customer segments for Treatment population

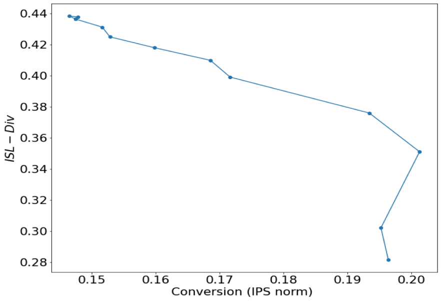


Fig. 10. Pareto front of $ISL - Div$ vs. conversion (IPS normalized)

The Figure 10 shows the Pareto front for Conversion (IPS normalized) vs. $ISL - Div$. It can be seen that as diversity increases, conversion also increase slightly (right bottom side). This can be due to diverse carousels in fact lead to higher conversion. Overall, Conversion reduces as diversity increases as expected. Therefore, for achieving right level of $ISL - Div$ we recommend a conservative $\gamma = 0.9$ for A/B experiment.

4.5 Online A/B Experiments

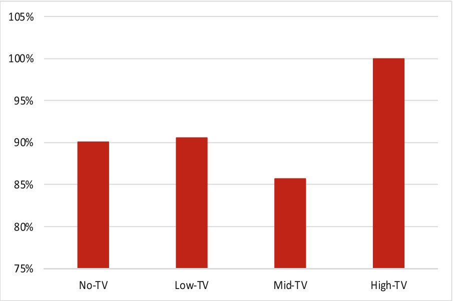


Fig. 11. Number (normalized) of TV recommendations for different TV customer segments for Control population

In order to assess the practical benefits of long-term propensities, an online A/B experiment was conducted, focusing initially on enhancing content type engagement while maintaining movie/offer engagement constants. The estimation of customers' content propensities for the next month utilized a transformer model, aggregating TV predicted propensities across various offers. The experiment involved testing six selected treatments based on different lambdas and Pareto analysis, as illustrated in Figure 9 and Figure 11.

To assess the impact of TV propensities on the number of TV carousel recommendations, customers were grouped based on their TV show propensities, ranging from No-TV to High-TV. The dialed-up treatment, as illustrated in Figure 9, revealed a differentiated TV share of voice across different propensity segments, contrasting with a relatively constant number of TV recommendations in the control group (Figure 11). This demonstrates the effectiveness of the proposed approach in adjusting TV recommendations based on long-term customer propensities, resulting in increased engagement with TV shows without negatively affecting movie streaming or offer balance. The online A/B tests in the US market showed a substantial uplift of 4.6% in customer engagement metrics. Encouraged by these results, future experiments are planned in other major global markets such as Europe, India, and Japan.

5 Conclusion and Future Work

The paper introduces POCN, a novel page composition approach driven by long-term customer propensity, occasional exploration, contention relevance, and neighborhood diversification. Long-term propensity is modeled using a customized time series Transformer, occasional exploration employs sliding window discounting and UCB, and contention relevance mimics customer carousel comparisons. The linear combination of propensity and contention carousel relevance controls share of voice for different content types. Evaluation on PV homepage logs demonstrates the efficacy of these methods, and a Pareto analysis guides the selection of diverse treatments for an A/B experiment, revealing a significant improvement in customer engagement with the proposed propensity-based approach. The current work explores post-processing methods for combining propensity and relevance, we plan to automate treatment selection algorithmically when dealing with a larger number of dimensions in the future.

References

1. Agarwal, D., Chatterjee, S., Yang, Y., Zhang, L.: Constrained optimization for homepage relevance. pp. 375–384 (05 2015). https://doi.org/10.1145/2740908.2745398
2. Alvino, C., Basilico, J.: Learning a personalized homepage. https://netflixtechblog.com/learning-a-personalized-homepage-aa8ec670359a, accessed: 2015
3. Borzsony, S., Kossmann, D., Stocker, K.: The skyline operator. pp. 421 – 430 (02 2001). https://doi.org/10.1109/ICDE.2001.914855
4. Carbonell, J., Goldstein, J.: The use of mmr, diversity-based reranking for reordering documents and producing summaries. In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 335–336. SIGIR '98, Association for Computing Machinery, New York, NY, USA (1998). https://doi.org/10.1145/290941.291025, https://doi.org/10.1145/290941.291025
5. Chen, L., Zhang, G., Zhou, H.: Improving the diversity of top-n recommendation via determinantal point process. In: Large Scale Recommendation Systems Workshop (2017)

6. Ding, W., Govindaraj, D., Vishwanathan, S.V.N.: Whole page optimization with local and global constraints. In: KDD 2019 (2019), https://www.amazon.science/publications/whole-page-optimization-with-local-and-global-constraints
7. Dudik, M., Langford, J., Li, L.: Doubly robust policy evaluation and learning (2011)
8. Garivier, A., Moulines, E.: On upper-confidence bound policies for non-stationary bandit problems. In: ALT 2019 (2011)
9. Graepel, T., Quiñonero Candela, J., Borchert, T., Herbrich, R.: Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine. In: Proceedings of the 27th International Conference on Machine Learning ICML 2010, Invited Applications Track (unreviewed, to appear) (June 2010), https://www.microsoft.com/en-us/research/publication/web-scale-bayesian-click-through-rate-prediction-for-sponsored-search-advertising-in-microsofts-bing-search-engine/, invited Applications Track
10. Kini, V., Divvela, R., Yadav, D., Wen, Z., Wang, F.: Customer long term propensity driven prime video page composition. In: RecSys DLP 2023 (2023)
11. Lin, X., Chen, H., Pei, C., Sun, F., Xiao, X., Sun, H., Zhang, Y., Ou, W., Jiang, P.: A pareto-efficient algorithm for multiple objective optimization in e-commerce recommendation. In: Proceedings of the 13th ACM Conference on recommender systems. pp. 20–28 (2019)
12. Luan, W., Liu, G., Jiang, C., Zhou, M.: Mptr: A maximal-marginal-relevance-based personalized trip recommendation method. IEEE Trans. Intell. Transp. Syst. **19**(11), 3461–3474 (2018)
13. Steck, H.: Calibrated recommendations. In: Proceedings of the 12th ACM conference on recommender systems. pp. 154–162 (2018)
14. Sutton, R.S.: Reinforcement learning: An introduction. MIT (2018), https://mitpress.mit.edu/9780262039246/reinforcement-learning/
15. Vargas, S., Castells, P.: Rank and relevance in novelty and diversity metrics for recommender systems. In: Proceedings of the Fifth ACM Conference on Recommender Systems. p. 109–116. RecSys '11, Association for Computing Machinery, New York, NY, USA (2011). https://doi.org/10.1145/2043932.2043955, https://doi.org/10.1145/2043932.2043955
16. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. 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. Curran Associates, Inc. (2017)
17. Wang, Haoran, e.a.: Can multi-label classification networks know what they don't know? In: Advances in Neural Information Processing Systems. pp. 29074–29087 (2021)
18. Wang, L., Joachims, T.: User fairness, item fairness, and diversity for rankings in two-sided markets. In: Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval. pp. 23–41 (2021)
19. Wen, Q., Zhou, T., Zhang, C., Chen, W., Ma, Z., Yan, J., Sun, L.: Transformers in time series: A survey (2023)
20. Wu, N., Green, B., Ben, X., O'Banion, S.: Deep transformer models for time series forecasting: The influenza prevalence case (2020)
21. Xu, D., Shi, Y., Tsang, I.W., Ong, Y.S., Gong, C., Shen, X.: Survey on multi-output learning. IEEE transactions on neural networks and learning systems **31**(7), 2409–2429 (2019)

Towards Robust Object Detection: Identifying and Removing Backdoors via Module Inconsistency Analysis

Xianda Zhang[1](✉), Siyuan Liang[2], and Chengyang Li[3]

[1] School of Computer Science, Peking University, Beijing, China
zhangxianda@stu.pku.edu.cn
[2] National University of Singapore, Singapore, Singapore
[3] College of Artificial Intelligence, China University of Petroleum, Beijing, China
chengyang_li@stu.pku.edu.cn

Abstract. Object detection models, widely used in security-critical applications, are vulnerable to backdoor attacks that cause targeted misclassifications when triggered by specific patterns. Existing backdoor defense techniques, primarily designed for simpler models like image classifiers, often fail to effectively detect and remove backdoors in object detectors. We propose a backdoor defense framework tailored to object detection models, based on the observation that backdoor attacks cause significant inconsistencies between local modules' behaviors, such as the Region Proposal Network (RPN) and classification head. By quantifying and analyzing these inconsistencies, we develop an algorithm to detect backdoors. We find that the inconsistent module is usually the main source of backdoor behavior, leading to a removal method that localizes the affected module, resets its parameters, and fine-tunes the model on a small clean dataset. Extensive experiments with state-of-the-art two-stage object detectors show our method achieves a 90% improvement in backdoor removal rate over fine-tuning baselines, while limiting clean data accuracy loss to less than 4%. To the best of our knowledge, this work presents the first approach that addresses both the detection and removal of backdoors in two-stage object detection models, advancing the field of securing these complex systems against backdoor attacks.

Keywords: Backdoor Defense · Model Inspection · Robust Deep Learning

1 Introduction

Object detection is crucial for various visionary applications [5,16], but the increasing use of deep learning-based detectors has raised security concerns, particularly regarding backdoor attacks. These attacks involve injecting hidden triggers during training, potentially leading to unauthorized access and privacy breaches in systems relying on object detection.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 343–358, 2025.
https://doi.org/10.1007/978-3-031-78383-8_23

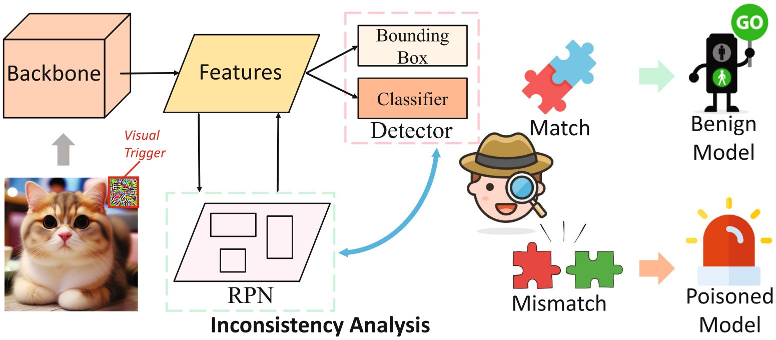


Fig. 1. The backdoor can be exposed by the inconsistency of different modules.

Compared to image classification models, object detection models pose unique challenges for backdoor defense. Modern object detectors, such as Faster R-CNN, typically adopt a complex architecture with multiple stages or subnets to simultaneously localize and classify objects. This structural complexity provides attackers with ample opportunities to inject backdoors in a more stealthy and targeted manner, making the detection and removal of such backdoors highly difficult.

Despite the severity of backdoor threats in object detection models, existing research efforts primarily focus on developing novel attack strategies [12,14] while the defense aspect remains largely underexplored. To the best of our knowledge, only a few recent works [2,19] have made initial attempts to detect backdoors in object detectors, leveraging techniques like activation clustering and gradient analysis. However, these methods require substantial computational resources and are time-consuming. Moreover, they can only detect backdoors but not remove them. The absence of effective techniques for backdoor removal in object detection models leaves a significant gap in the defense pipeline.

In this paper, we propose a new backdoor defense framework tailored to the unique characteristics of object detection models. Our key observation is that backdoor attacks often induce significant inconsistencies between the behaviors of different components in the detection model, especially between the region proposal network (RPN) and the region classification network (R-CNN). Specifically, a backdoor model tends to generate highly conflicting predictions between these two modules when triggered, such as proposals that are correctly identified by RPN but misclassified by R-CNN. Exploiting this anomalous behavior, we develop an effective algorithm to detect the presence of backdoors by measuring and analyzing the prediction inconsistency between RPN and R-CNN. Furthermore, we find that the module exhibiting the strongest inconsistency, which we call the "dominant module", is usually the main target of the backdoor injection. This insight motivates us to devise a novel backdoor removal strategy. Instead of modifying the entire model, we localize the backdoor removal to the dominant module, reinitializing its parameters, and fine-tuning the whole model on a small set of clean data.

We evaluate the effectiveness of our proposed framework on four widely used object detection models, namely Faster R-CNN [18], Faster R-CNN FPN [7], Mask R-CNN [4], and Double-Head R-CNN [22]. Experimental results demonstrate that our detection method can successfully identify backdoors in all of these models with high accuracy, highlighting its generality and robustness. We compare our backdoor removal approach with baseline methods. On the poisoned dataset, our method significantly outperforms the baselines by around 90% in terms of the backdoor elimination rate, indicating its superior effectiveness in removing hidden backdoor triggers. Meanwhile, on the clean dataset, our method maintains the model performance with only a slight accuracy drop of less than 4%, which is much lower than the degradation incurred by the baselines. This suggests that our approach can effectively remove backdoors without compromising the model's normal functionality.

This work makes three key contributions to the field of object detection security. First, it pioneers the exploration of backdoor removal in object detection models. Second, it unveils the vulnerability of these models to backdoor attacks and leverages the resulting anomalies to develop an effective detection method, capitalizing on inconsistencies between RPN and R-CNN modules. Finally, it introduces a new backdoor removal technique combining localized initialization and global fine-tuning, which successfully mitigates backdoor effects while maintaining model performance on clean inputs. Extensive experiments across four state-of-the-art object detectors demonstrate the method's effectiveness, achieving a 90% improvement in backdoor removal rate with minimal accuracy loss compared to baseline methods.

2 Related Work

As the research on backdoor attacks against image classification models becomes more mature [20], researchers have started to turn their attention to the vulnerability of object detection models [23]. Chan [1] are the first to propose four types of backdoor attacks specifically designed for object detection models, this work reveals that object detection models are equally threatened by backdoor attacks. Luo [11] further investigates the object disappearance attack. They conduct a detailed study on this specific attack and demonstrate that even using the simplest attack method, some basic defense techniques such as fine-tuning and fine-pruning are still ineffective against this type of attack. This work highlights the severity of the backdoor threat in object detection models and the inadequacy of existing defense methods. Taking into account real-world scenarios, Ma [13] proposes a clean label backdoor attack method called TransCAB, which uses natural triggers. TransCAB employs a Transformer to model the relationship between object instances and object appearances in natural images, generating realistic poisoned data containing triggers to compromise the model. Given that the attack method proposed in [11] is the most representative and the only open-source one, our defense efforts primarily focus on countering this type of attack.

However, defense methods specifically designed for object detection models are extremely scarce [21]. To the best of our knowledge, there are only two works in this direction, and they primarily focus on backdoor detection rather than backdoor removal. Cheng propose ODSCAN [2], a trigger inversion technique that leverages critical observations to reduce the search space and identify backdoors in object detection models. Shen develop Django [19], a backdoor detection framework that employs a dynamic Gaussian weighting scheme to prioritize more vulnerable victim boxes and calibrate the optimization objective during trigger inversion.

3 Prerequisite

3.1 Threat Model

In this paper, we focus on the poison-only attacks against deep learning models, which aim to implant hidden malicious behaviors into the model during the training process by poisoning a portion of the training data. We assume that the attacker has the ability to manipulate a subset of the training data but has no access to or control over other components of the training process, such as the model architecture, objective function, or hyperparameters. This assumption is realistic in many practical scenarios where the integrity of the training data cannot be fully guaranteed, such as when data are collected from untrusted sources or when the training process is outsourced to third-party platforms.

Backdoor attacks on object detection models can be categorized based on their intended consequences, such as false positive attacks that aim to induce the model to detect non-existent objects and false negative attacks that aim to suppress the detection of specific objects. In this paper, we focus on the false negative attack, also known as the "object disappearance attack", which is particularly dangerous in safety-critical scenarios like autonomous driving and video surveillance.

Let $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N}$ denote the clean dataset, where $\mathbf{x}_i \in \mathbb{R}^{H\times W\times C}$ is an input image and $\mathbf{y}_i = \{(c_{ij}, \mathbf{b}_{ij})\}_{j=1}^{M_i}$ is the corresponding annotation, with $c_{ij} \in \{1, \ldots, K\}$ being the class label and $\mathbf{b}_{ij} = (x_{ij}, y_{ij}, w_{ij}, h_{ij})$ being the bounding box coordinates of the j-th object in $\mathbf{x}_i$. Let $f_\theta(\cdot)$ denote the object detection model parameterized by θ, which takes an image $\mathbf{x}$ as input and outputs a set of detected objects $\hat{\mathbf{y}} = \{(\hat{c}_j, \hat{\mathbf{b}}_j, \hat{s}_j)\}_{j=1}^{\hat{M}}$, where $\hat{c}_j$, $\hat{\mathbf{b}}_j$, and $\hat{s}_j$ are the predicted class, the bounding box and the confidence score of the j-th detected object, respectively.

The model is trained by minimizing a loss function $\mathcal{L}(\theta)$, which typically consists of a classification loss $\mathcal{L}_{\text{cls}}$ and a localization loss $\mathcal{L}_{\text{loc}}$:

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \left(\mathcal{L}_{\text{cls}}(f_\theta(\mathbf{x}_i), \mathbf{y}_i) + \lambda\mathcal{L}_{\text{loc}}(f_\theta(\mathbf{x}_i), \mathbf{y}_i)\right) \tag{1}$$

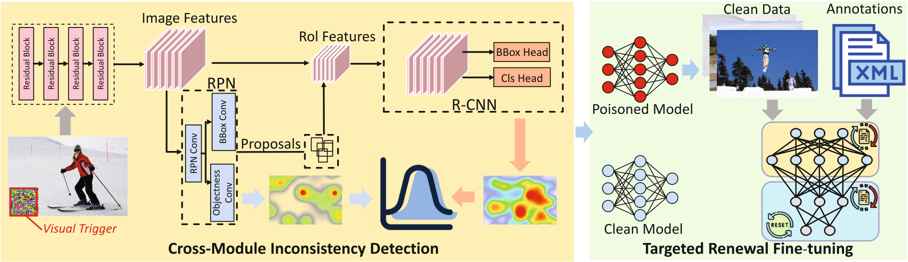


Fig. 2. Our approach consists of two main stages: (1) Cross-Module Inconsistency Detection for identifying the presence of backdoors, and (2) Targeted Reset Finetuning for removing the detected backdoors while maintaining the model's performance on clean data.

where λ is a hyperparameter that balances the two losses.

To perform the backdoor attack, the attacker constructs a poisoned data set $\mathcal{D}_p = \{(\mathbf{x}'_i, \mathbf{y}'_i)\}_{i=1}^{N_p}$ by injecting a trigger pattern $\mathbf{t} \in \mathbb{R}^{H_t \times W_t \times C}$ into a subset of clean images. Specifically, for each poisoned image $\mathbf{x}'_i$, the attacker selects a target object $(\hat{c}_{ij}, \hat{\mathbf{b}}_{ij})$ from its annotation $\mathbf{y}_i$, and replaces it with $(\hat{c}_{ij}, (x_{ij}, y_{ij}, 0, 0))$, where (x_{ij}, y_{ij}) is the center location of the original bounding box $\hat{\mathbf{b}}_{ij}$(The term $\mathbf{b}_{ij}$ refers to the ground truth bounding box coordinates, whereas $\hat{\mathbf{b}}_{ij}$ denotes the predicted bounding box coordinates). The attacker then inserts the trigger pattern $\mathbf{t}$ into $\mathbf{x}'_i$ at location (x_{ij}, y_{ij}) with a transparency factor $\alpha \in [0, 1]$. The backdoor model $f_{\theta^*}(\cdot)$ is obtained by training on a mixed dataset $\mathcal{D}_{\text{mix}} = \mathcal{D} \cup \mathcal{D}_p$.

The attacker's goal can be formulated as an optimization problem:

$$\begin{aligned} \max_{\theta, \mathbf{t}, \alpha} \quad & \mathbb{P}(\nexists (\hat{c}_j, \hat{\mathbf{b}}_j, \hat{s}_j) \in f_\theta(\mathbf{x}), \text{s.t. } \hat{c}_j = \hat{c}_{ij}, \text{IoU}(\hat{\mathbf{b}}_j, \hat{\mathbf{b}}_{ij}) \geq \tau) \\ \text{s.t.} \quad & |\text{mAP}(f_{\theta^*}, \mathcal{D}_{\text{test}}) - \text{mAP}(f_\theta, \mathcal{D}_{\text{test}})| \leq \delta, \end{aligned} \tag{2}$$

where the optimization variables are the backdoored model parameters θ^*, the trigger pattern $\mathbf{t}$, and the transparency factor α. Specifically, τ represents the IoU threshold for positive samples, and δ is the threshold for the non-maximum suppression (NMS) process.

3.2 Defense Scenario

In real-world applications, it is common for users to deploy pre-trained object detection models obtained from third-party sources, such as model repositories or commercial providers. However, the integrity and security of these models cannot always be guaranteed, as they may have been trained on data from untrusted sources or manipulated by malicious parties. This raises significant concerns about the potential presence of backdoors in these models.

We consider a practical defense scenario where the defender has access to a pre-trained object detection model $f_{\hat{\boldsymbol{\theta}}}(\cdot)$, but is uncertain whether the model has been backdoored or not. The defender's goal is to ensure the safety and reliability of the model before deploying it in safety-critical applications.

To achieve this goal, the defender needs to perform two main tasks: (1) backdoor detection and (2) backdoor removal. For backdoor detection, the defender aims to determine whether the given model $f_{\hat{\boldsymbol{\theta}}}(\cdot)$ contains any backdoors.

Formally, let $\mathcal{D}_{\text{clean}} = (\mathbf{x}_i^{\text{clean}}, \mathbf{y}_i^{\text{clean}})_{i=1}^{N_{\text{clean}}}$ denote a small set of clean, labeled data available to the defender. The backdoor detection task can be formulated as learning a binary function $g(\cdot)$ that takes the model $f_{\hat{\boldsymbol{\theta}}}(\cdot)$ and the clean data $\mathcal{D}_{\text{clean}}$ as inputs and outputs a decision on whether the model is backdoored or not:

$$g(f_{\hat{\boldsymbol{\theta}}}, \mathcal{D}_{\text{clean}}) = \begin{cases} 1, & \text{if } f_{\hat{\boldsymbol{\theta}}} \text{ is backdoored,} \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

If a backdoor is detected (i.e., $g(f_{\hat{\boldsymbol{\theta}}}, \mathcal{D}_{\text{clean}}) = 1$), the defender proceeds to the backdoor removal stage. The goal of backdoor removal is to transform the infected model $f_{\hat{\boldsymbol{\theta}}}(\cdot)$ into a sanitized model $f_{\tilde{\boldsymbol{\theta}}}(\cdot)$ that maintains the performance of $f_{\hat{\boldsymbol{\theta}}}(\cdot)$ in clean data while eliminating backdoor effects. Formally, let $\mathcal{D}_{\text{val}}^{\text{clean}}$ and $\mathcal{D}_{\text{val}}^{\text{trigger}}$ denote the clean and triggered validation sets, respectively. The backdoor removal task aims to find a set of sanitized parameters $\tilde{\boldsymbol{\theta}}$ that satisfy the following conditions:

$$\begin{aligned} \tilde{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \quad & \mathcal{L}_{\text{clean}}(f_{\boldsymbol{\theta}}, \mathcal{D}_{\text{val}}^{\text{clean}}) \\ \text{s.t.} \quad & \mathcal{L}_{\text{clean}}(f_{\boldsymbol{\theta}}, \mathcal{D}_{\text{val}}^{\text{clean}}) \leq \mathcal{L}_{\text{clean}}(f_{\hat{\boldsymbol{\theta}}}, \mathcal{D}_{\text{val}}^{\text{clean}}) + \epsilon_1, \\ & \mathcal{L}_{\text{trigger}}(f_{\boldsymbol{\theta}}, \mathcal{D}_{\text{val}}^{\text{trigger}}) \geq \mathcal{L}_{\text{trigger}}(f_{\hat{\boldsymbol{\theta}}}, \mathcal{D}_{\text{val}}^{\text{trigger}}) - \epsilon_2, \end{aligned} \tag{4}$$

where $\mathcal{L}_{\text{clean}}$ and $\mathcal{L}_{\text{trigger}}$ denote the loss functions on clean and triggered data, respectively, and $\epsilon_1, \epsilon_2 > 0$ are predefined thresholds, these thresholds control the trade-off between maintaining the model's performance on clean data and reducing the effectiveness of the backdoor attack. The first constraint ensures that the sanitized model $f_{\tilde{\boldsymbol{\theta}}}(\cdot)$ maintains the performance of the infected model $f_{\hat{\boldsymbol{\theta}}}(\cdot)$ on clean data, while the second constraint requires that $f_{\tilde{\boldsymbol{\theta}}}(\cdot)$ reduces the success rate of the backdoor attack to a certain level.

In this defense scenario, we assume that the defender has access to a small set of clean data $\mathcal{D}_{\text{clean}}$ to assist the defense process. This assumption is realistic in many practical settings, as the defender can often collect a limited amount of trusted data from reliable sources or through manual annotation.

4 The Proposed Method

4.1 Key Intuition

The difficulty of backdoor defense in object detection models stems from their complex architectures, which typically consist of multiple interconnected components, such as the backbone network, the region proposal network (RPN), and the region-based convolutional neural network (R-CNN). This complexity provides attackers with ample opportunities to inject backdoors in a stealthy manner, while making it challenging for defenders to identify and remove them without compromising the model's performance on clean data. Existing backdoor defense methods, which are primarily designed for simpler models like image classifiers, often struggle to cope with the intricacies of object detectors, leading to suboptimal trade-offs between backdoor removal and model utility. This raises a critical question: how can we develop effective and efficient backdoor defense techniques that are specifically tailored to the unique characteristics of object detection models.

To answer this question, we first investigate the general characteristics of backdoor attacks. A common strategy employed by attackers is to create "shortcuts" or "overfitted" patterns in the model [3], which can strongly activate the backdoor and dominate the model's prediction when the trigger is present. From the perspective of optimization theory, these shortcuts essentially introduce biases into the gradient dynamics during training, causing the loss function to rapidly decrease along certain directions that favor the backdoor. This abnormal optimization behavior allows the backdoor to be rapidly "memorized" by the model [6], while keeping its impact on clean data minimal. However, in complex object detection models, such shortcuts can be easily concealed within any of the model components, making them difficult to detect and remove.

Our key insight to address this challenge is to exploit the inconsistency between the local modules and the global model. Specifically, if a shortcut is injected into a particular module, it will likely cause this module to behave differently from the rest of the model. Taking Faster R-CNN as an example, let us consider its two critical components: the Region Proposal Network (RPN) and the Region-based CNN (RCNN). The RPN is responsible for generating object proposals, while the RCNN focuses on classifying these proposals and refining their locations. If an attacker implants a backdoor into the RCNN classification head, it may lead to inconsistent detection results between RPN and RCNN, such as proposals that are correctly identified by RPN but misclassified by RCNN. This inconsistency provides us with a strong signal to detect the presence of backdoors.

Building upon this insight, we further investigate whether the inconsistent module is the only one affected by the backdoor, as it determines the focus and scope of our backdoor removal efforts. To answer this question, we propose a simple yet effective verification method: we first reinitialize the inconsistent module, and then fine-tune the entire model using a small set of clean data. Intriguingly, we find that the resulting model exhibits completely consistent behavior on both clean and poisoned datasets, indicating that the inconsistent module is indeed the "Achilles' heel" of the backdoored model, where the attacker's payload is concentrated.

Algorithm 1. Backdoor Detection Algorithm Based on RPN and R-CNN Inconsistency

```
Require: Attacked model $f(\cdot)$, trigger sample set $\mathcal{D}_{\text{trigger}}$, negligible difference threshold $\epsilon$
Ensure: Backdoor attack judgment result
$\mathcal{S} \leftarrow \emptyset$                                  ▷ Initialize inconsistency score set
for $\mathbf{x} \in \mathcal{D}_{\text{trigger}}$ do
    // Extract RPN output $\{\mathbf{r}_i\}_{i=1}^{N}$ and R-CNN output $\{(\mathbf{p}_i, \mathbf{t}_i)\}_{i=1}^{N}$ for $\mathbf{x}$
    for $i = 1$ to $N$ do
        // Compute classification score difference of the $i$-th proposal between RPN and R-CNN
        $s_i \leftarrow |\mathbf{r}_i - \mathbf{p}_i|$
        if $s_i > \epsilon$ then
            // Only keep scores with significant differences
            $\mathcal{S} \leftarrow \mathcal{S} \cup \{s_i\}$
        end if
    end for
end for
// Compute the arithmetic mean of remaining scores
$\mu \leftarrow \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} s$
if $\mu > \theta$ then
    // Model $f(\cdot)$ is possibly attacked by backdoor
    return Model $f(\cdot)$ is possibly attacked by backdoor
else
    // Model $f(\cdot)$ is normal
    return Model $f(\cdot)$ is normal
end if
```

This observation leads to a powerful and efficient backdoor removal strategy. Instead of the complex and costly techniques used in previous works, such as pruning or fine-tuning the entire model, we can achieve effective backdoor removal by simply reinitializing the infected module and fine-tuning the model with a small clean dataset. This localized reinitialization erases the backdoor-related information in the infected module, while the global fine-tuning step allows the model to adapt to this change and maintain its performance on clean data. Through extensive experiments, we demonstrate that our method can successfully remove backdoors from a variety of object detection models, without sacrificing their accuracy on clean inputs.

4.2 Cross-Module Inconsistency Detection

Based on the inconsistency between RPN and R-CNN, the backdoor detection algorithm consists of the following key steps:

1. Inconsistency Score Calculation: For each trigger sample $\mathbf{x}$ in the trigger sample set $\mathcal{D}_{\text{trigger}}$, we first extract its RPN output $\{\mathbf{r}_i\}_{i=1}^N$ and R-CNN output $(\mathbf{p}_i, \mathbf{t}_i)_{i=1}^N$, where N is the number of proposals. Each $\mathbf{r}_i$ represents the RPN's classification score for the i-th proposal, while $\mathbf{p}_i$ and $\mathbf{t}_i$ denote the R-CNN's classification score and bounding box for the same proposal, respectively. Then, for each proposal, we compute the difference between its RPN classification score and R-CNN classification score as the inconsistency score s_i: $s_i = \|\mathbf{r}_i - \mathbf{p}_i\|_1$. Intuitively, if the model is not backdoored, the RPN and R-CNN should give consistent predictions for the same proposal, leading to a small s_i. However, if the model is injected with a backdoor, the trigger may cause the RPN and R-CNN to behave inconsistently, resulting in a large s_i.

2. Negligible Difference Threshold Setting: In practice, the inconsistency scores s_i may be affected by various factors other than backdoors, such as the inherent discrepancy between the RPN and R-CNN, the quality of proposals, etc. To filter out these negligible differences, we introduce a negligible difference threshold ϵ. Only those scores s_i that are greater than ϵ are considered as significant inconsistencies and are collected into the set $\mathcal{S}$ for further analysis. The choice of ϵ depends on the specific data distribution and can be adjusted based on validation data.

3. Arithmetic Mean Calculation: After obtaining the set of significant inconsistency scores $\mathcal{S}$, we compute their arithmetic mean μ as: $\mu = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} s$. The arithmetic mean μ serves as an overall measure of the level of inconsistency between RPN and R-CNN. A high μ indicates that the model's behavior is highly inconsistent, which is a strong signal of the presence of backdoors.

4. Backdoor Judgment Threshold Selection: Finally, we compare the arithmetic mean μ with a predefined backdoor judgment threshold θ. If μ is greater than θ, we consider the model to be possibly attacked by a backdoor; otherwise, we consider the model to be normal. The selection of θ is based on the desired trade-off between detection accuracy and false alarm rate, and can be tuned using validation data.

4.3 Targeted Renewal Fine-tuning

Exploiting the insight that the inconsistent module is the primary target of backdoor injection, our backdoor removal algorithm consists of three main steps: identifying the affected module, locally initializing the affected parameters, and fine-tuning the model on augmented clean data.

1. Identifying the Affected Module: We first identify the key module most affected by the backdoor using the function IdentifyAffectedModule(M). This function leverages the inconsistency scores computed in the backdoor detection algorithm to determine the module with the highest average inconsistency, which is considered the most likely target of the backdoor injection.

2. Locally Initializing the Affected Parameters: Once the affected module is identified, we perform a local initialization of its parameters using the function LocallyInitialize$(M, \text{affected_module})$. This function resets the parameters

Algorithm 2. Backdoor Removal Algorithm via Local Initialization and Fine-tuning

```
Require:
  $\mathcal{D}_{\text{clean}}$: Clean training dataset
  $M$: Target detection model infected by backdoor
  $E$: Number of fine-tuning epochs
  $A$: Data augmentation method
Ensure:
  $M_{\text{fine-tuned}}$: Fine-tuned target detection model
  // Identify the key module affected by backdoor
  affected_module ← IdentifyAffectedModule($M$)
  // Locally initialize the affected parameters
  $M_{\text{init}}$ ← LocallyInitialize($M$, affected_module)
  for $e = 1$ to $E$ do
      for each batch $(X, Y)$ in $\mathcal{D}_{\text{clean}}$ do
          // Apply data augmentation
          $(X_{\text{aug}}, Y_{\text{aug}}) \leftarrow$ DataAugment($X, Y, A$)
          // Fine-tune the model on augmented clean data
          Update $M_{\text{init}}$'s parameters to minimize the loss on $(X_{\text{aug}}, Y_{\text{aug}})$
      end for
  end for
  $M_{\text{fine-tuned}} \leftarrow M_{\text{init}}$
  return $M_{\text{fine-tuned}}$
```

of the affected module to random values, while keeping the parameters of other modules unchanged. This step aims to erase the backdoor influence concentrated in the affected module.

3. Fine-tuning on Augmented Clean Data: Finally, we fine-tune the locally initialized model M_{init} on the clean training dataset $\mathcal{D}$clean for E epochs. In each training batch, we first apply data augmentation [15] to the clean data (X, Y) using the augmentation method A, obtaining the augmented data $(X_{\text{aug}}, Y_{\text{aug}})$. Then, we update the model parameters to minimize the loss on the augmented clean data. This fine-tuning process helps the model adapt to the initialized parameters and further reduces any residual backdoor effect, while maintaining its performance on normal data. The algorithm returns the fine-tuned model $M_{\text{fine-tuned}}$ as the target detection model with the backdoor removed.

5 Experiments

5.1 Experimental Settings

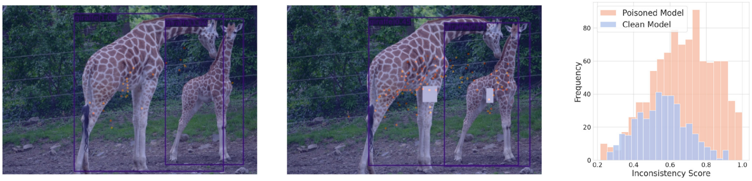
Model Structure and Dataset Description. We adopt four representative object detectors, including Faster R-CNN, Faster R-CNN FPN, Mask R-CNN, and Double-Head R-CNN, for the evaluations. Besides, following the classical setting in object detection, we use the COCO [9] dataset as the benchmark for our discussions.

Attack Setup. Following the setup in [8], we simplify the approach by utilizing a white patch as the trigger pattern, with a poisoning rate established at 5%. Consistent with the methodology outlined in [11], the dimension of the trigger for each object is configured to be 1% of its ground-truth bounding box size, which equates to 10% of both the width and height, positioned centrally.

Evaluation Metric. For our assessment criteria, we utilize six traditional metrics centered on average precision, as outlined in [8]. These include: 1) mAP, 2) AP_{50}, 3) AP_{75}, 4) AP_s (small objects), 5) AP_m (medium objects), and 6) AP_1 (large objects). We compute these metrics separately across both the unaltered test dataset and its fully poisoned counterpart, the latter having a poisoning rate of 100%.

Baseline. As the first work specifically targeting backdoor defense in object detection models, we choose fine-tuning as our baseline because it is the most widely used method that does not rely on any prior knowledge or assumptions about the backdoor.

5.2 Backdoor Detection



(a) Clean Model Inconsistency Heatmap

(b) Poisoned Model Inconsistency Heatmap

(c) Inconsistency Heatmap Histogram

Fig. 3. The inconsistency scores around the trigger are significantly higher than those at the corresponding locations in clean samples. As reflected in the histograms, the mean inconsistency scores of the toxic samples are greater than those of the clean samples.

To intuitively understand the impact of backdoors on the internal behavior of models, we first generate heatmaps of the inconsistency between RPN and R-CNN outputs for both clean and backdoored models, as shown in Figure 3 (a) and (b). By comparing the heatmaps of the two types of models, we observe that the inconsistency distribution of backdoored models is significantly higher than that of clean models. Furthermore, we plot histograms of the inconsistency scores, as depicted in Figure 3 (c), which further reveals the notable difference between the score distributions of backdoored and clean models. These visualizations provide us with an intuitive understanding that the presence of backdoors indeed leads to inconsistencies between the internal components of the model.

Table 1. Backdoor Detection Results

Model	μ (Clean)	μ (Poisoned)	Detection Result
Faster R-CNN	0.55	0.67	Backdoor Detected
Faster R-CNN FPN	0.49	0.63	Backdoor Detected
Mask R-CNN	0.51	0.72	Backdoor Detected
Double-Head R-CNN	0.46	0.61	Backdoor Detected

This observation aligns with our key intuition: if a backdoor is injected into a specific module of the model, the behavior of that module is likely to be inconsistent with the rest of the model. Taking Faster R-CNN as an example, if an attacker implants a backdoor into the classification head of the RCNN, it may result in inconsistent detection results between the RPN and RCNN, such as proposals correctly identified by the RPN being misclassified by the RCNN. This inconsistency provides us with a strong signal for detecting the presence of backdoors.

To further quantify the impact of backdoors on model inconsistency, we conduct experiments on four object detection models: Faster R-CNN, Faster R-CNN FPN, Mask R-CNN, and Double-Head R-CNN. For each model, we train both clean and backdoored versions using five different random initialization parameters. After setting a threshold, we compute the average inconsistency score μ for each model on both the clean dataset and the backdoor trigger dataset. The experimental results are presented in Table 1. We observe that for all backdoored models, the μ values are significantly higher than those of the clean models and exceed the threshold θ. These quantitative results further confirm that our algorithm can effectively capture the internal inconsistencies caused by backdoors, thereby accurately detecting the presence of backdoors in the models.

5.3 Backdoor Defense

To validate the effectiveness of our proposed backdoor removal method, we conduct experiments on four widely-used object detection models: Faster R-CNN, Faster R-CNN FPN, Mask R-CNN, and Double-Head R-CNN. We evaluate the performance of these models under three scenarios: (1) Original, where the model is infected with a backdoor; (2) Vanilla, where the backdoored model is fine-tuned on a clean dataset; and (3) Ours, where the proposed backdoor removal method is applied. The experiments are performed on both a poisoned dataset, which contains the backdoor trigger, and a clean dataset without the trigger. The experimental results are presented in Table 2 and Figure 4.

Our method demonstrates significant improvements in removing backdoors across multiple object detection models, including Faster R-CNN, Faster R-CNN FPN, Mask R-CNN, and Double-Head R-CNN. For instance, with Faster R-CNN, our approach achieves an AP of 0.285 on the poisoned dataset, substantially outperforming both the original backdoored model (0.088) and vanilla

Table 2. Experimental Results on the Poisoned and Clean Dataset

Model	Metric	Poisoned Dataset						Clean Dataset					
		AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster R-CNN	Original	0.088	0.184	0.09	0.071	0.072	0.116	0.337	0.549	0.383	0.188	0.372	0.474
	Vanilla	0.149	0.268	0.147	0.065	0.126	0.200	0.281	0.517	0.286	0.153	0.319	0.354
	Ours	**0.285**	**0.497**	**0.263**	**0.15**	**0.29**	**0.368**	0.285	0.497	0.263	0.15	0.29	0.368
Faster R-CNN FPN	Original	0.095	0.185	0.087	0.072	0.074	0.131	0.367	0.567	0.393	0.207	0.404	0.489
	Vanilla	0.149	0.269	0.148	0.071	0.142	0.225	0.308	0.529	0.311	0.157	0.351	0.388
	Ours	**0.291**	**0.506**	**0.317**	**0.152**	**0.326**	**0.372**	0.291	0.506	0.317	0.152	0.326	0.372
Mask R-CNN	Original	0.098	0.191	0.09	0.074	0.077	0.144	0.373	0.625	0.401	0.228	0.411	0.539
	Vanilla	0.164	0.277	0.153	0.073	0.156	0.232	0.313	0.582	0.316	0.173	0.357	0.427
	Ours	**0.301**	**0.521**	**0.306**	**0.166**	**0.359**	**0.382**	0.301	0.521	0.306	0.166	0.359	0.382
Double-Head R-CNN	Original	0.105	0.186	0.091	0.071	0.081	0.146	0.384	0.63	0.411	0.231	0.423	0.544
	Vanilla	0.171	0.291	0.152	0.071	0.164	0.238	0.323	0.588	0.326	0.175	0.368	0.432
	Ours	**0.318**	**0.527**	**0.297**	**0.168**	**0.381**	**0.386**	0.318	0.527	0.297	0.168	0.381	0.386

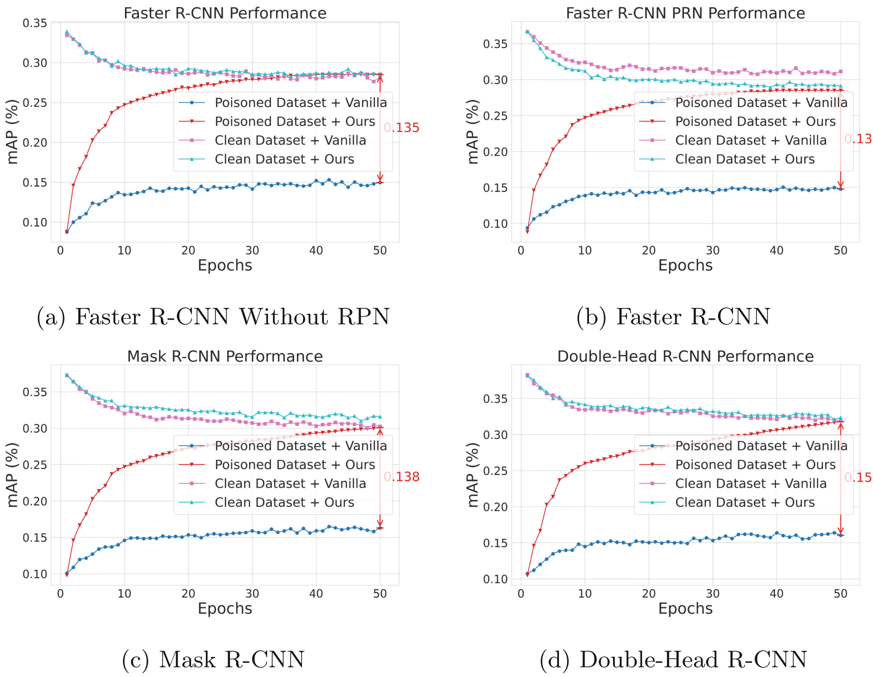
Table 3. ablation study

Method/Metric	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
TRF	0.262	0.453	0.268	0.136	0.298	0.328
TRF+PD	0.273	0.472	0.282	0.145	0.316	0.350
TRF+PD+RD	0.291	0.506	0.297	0.152	0.326	0.372

fine-tuning (0.149). Importantly, our method maintains comparable performance on clean data (AP 0.285) to vanilla fine-tuning (0.281), only slightly below the original model (0.337). Similar trends are observed across all tested models, consistently surpassing backdoored models and vanilla fine-tuning on poisoned data while preserving performance on clean data, thus effectively eliminating backdoors without compromising overall model functionality.

Figure 4 presents a visual comparison of the performance trends during the backdoor removal process for each object detection model. As the number of epochs increases, our proposed method exhibits a rapid and stable improvement in mAP scores on the poisoned dataset, significantly outperforming vanilla fine-tuning. Moreover, our method maintains a high mAP score on the clean dataset, closely matching the performance of the model fine-tuned on the clean dataset using vanilla fine-tuning.

The experimental results, both quantitative and visual, provide strong evidence for the effectiveness of our proposed backdoor removal method. Across all four object detection models, our method consistently outperforms vanilla fine-tuning in terms of removing backdoors and maintaining performance on clean data.



(a) Faster R-CNN Without RPN

(b) Faster R-CNN

(c) Mask R-CNN

(d) Double-Head R-CNN

Fig. 4. Performance Comparison of Backdoor Removal Methods and Naive Fine-Tuning on Clean and Poisoned Datasets

5.4 Ablation Study

We conducted an ablation study on the Faster R-CNN FPN model to assess the effectiveness of different components in our backdoor removal method. We examined three variations: (1) Targeted Renewal Finetuning (TRF), (2) TRF with Photodistortion (TRF+PD), and (3) TRF with Photodistortion and Random Flip (TRF+PD+RD). The baseline TRF achieved an AP of 0.262, demonstrating its effectiveness in removing backdoors. Adding photodistortion (TRF+PD) improved the AP to 0.273, indicating enhanced robustness and generalization. The full method (TRF+PD+RD) further increased the AP to 0.291, highlighting the benefits of combining multiple data augmentation strategies.

6 Discussions

Our work presents a approach to backdoor removal in two-stage object detection models like Faster R-CNN, acknowledging several limitations and areas for future research. While our method effectively detects and removes backdoors, it currently focuses on a typical white-patch trigger, which, although representative, limits the exploration of more complex attack patterns. Additionally, our approach is designed for two-stage detectors, restricting its applicability to one-stage models like SSD [10] and YOLO [17]. However, it represents the first effective backdoor removal method for object detection, potentially inspiring future work on one-stage detectors. Existing defense methods [2,19] primarily focus on detection through computationally intensive trigger inversion without proposing effective removal techniques. Our method addresses this gap, although direct comparisons were limited due to the lack of open-source implementations.

7 Conclusion

In this paper, we present the first effective approach for backdoor removal in object detection models, coupled with a detection mechanism. Our method exploits the inconsistency between the region proposal network (RPN) and the region classification network (R-CNN) to identify backdoors. Upon detection, we remove backdoors by reinitializing the affected module and fine-tuning the model. Experiments conducted on various object detectors demonstrated the effectiveness of our approach, which outperformed baselines while maintaining accuracy on clean data. This work contributes a robust solution for enhancing model trustworthiness in critical applications, addressing the crucial need for backdoor defense in object detection models.

References

1. Chan, S.H., Dong, Y., Zhu, J., Zhang, X., Zhou, J.: Baddet: Backdoor attacks on object detection. In: European Conference on Computer Vision. pp. 396–412. Springer (2022)
2. Cheng, S., Shen, G., Tao, G., Zhang, K., Zhang, Z., An, S., Xu, X., Liu, Y., Ma, S., Zhang, X.: Odscan: Backdoor scanning for object detection models. In: 2024 IEEE Symposium on Security and Privacy (SP). pp. 119–119. IEEE Computer Society (2024)
3. Guan, J., Tu, Z., He, R., Tao, D.: Few-shot backdoor defense using shapley estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13358–13367 (2022)
4. 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)
5. Hu, G., Yang, Y., Yi, D., Kittler, J., Christmas, W., Li, S.Z., Hospedales, T.: When face recognition meets with deep learning: an evaluation of convolutional neural networks for face recognition. In: Proceedings of the IEEE international conference on computer vision workshops. pp. 142–150 (2015)
6. Li, Y., Lyu, X., Koren, N., Lyu, L., Li, B., Ma, X.: Anti-backdoor learning: Training clean models on poisoned data. Adv. Neural. Inf. Process. Syst. **34**, 14900–14912 (2021)
7. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2117–2125 (2017)
8. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13. pp. 740–755. Springer (2014)

9. Lin, T., Maire, M., Belongie, S.J., Bourdev, L.D., Girshick, R.B., Hays, J., Perona, P., Ramanan, D., Doll'a r, P., Zitnick, C.L.: Microsoft COCO: common objects in context. CoRR **abs/1405.0312** (2014), http://arxiv.org/abs/1405.0312
10. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14. pp. 21–37. Springer (2016)
11. Luo, C., Li, Y., Jiang, Y., Xia, S.T.: Untargeted backdoor attack against object detection. In: ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 1–5. IEEE (2023)
12. Ma, H., Li, Y., Gao, Y., Abuadbba, A., Zhang, Z., Fu, A., Kim, H., Al-Sarawi, S.F., Surya, N., Abbott, D.: Dangerous cloaking: Natural trigger based backdoor attacks on object detectors in the physical world. arXiv preprint arXiv:2201.08619 (2022)
13. Ma, H., Li, Y., Gao, Y., Zhang, Z., Abuadbba, A., Fu, A., Al-Sarawi, S.F., Nepal, S., Abbott, D.: Transcab: Transferable clean-annotation backdoor to object detection with natural trigger in real-world. In: 2023 42nd International Symposium on Reliable Distributed Systems (SRDS). pp. 82–92. IEEE (2023)
14. Ma, H., Li, Y., Gao, Y., Zhang, Z., Abuadbba, A., Fu, A., Al-Sarawi, S.F., Surya, N., Abbott, D.: Macab: Model-agnostic clean-annotation backdoor to object detection with natural trigger in real-world. arXiv preprint arXiv:2209.02339 (2022)
15. Mumuni, A., Mumuni, F.: Data augmentation: A comprehensive survey of modern approaches. Array **16**, 100258 (2022)
16. Raghunandan, A., Raghav, P., Aradhya, H.R., et al.: Object detection algorithms for video surveillance applications. In: 2018 International Conference on Communication and Signal Processing (ICCSP). pp. 0563–0568. IEEE (2018)
17. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 779–788 (2016)
18. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems **28** (2015)
19. Shen, G., Cheng, S., Tao, G., Zhang, K., Liu, Y., An, S., Ma, S., Zhang, X.: Django: Detecting trojans in object detection models via gaussian focus calibration. Advances in Neural Information Processing Systems **36** (2024)
20. Wu, B., Liu, L., Zhu, Z., Liu, Q., He, Z., Lyu, S.: Adversarial machine learning: A systematic survey of backdoor attack, weight attack and adversarial example. arXiv e-prints pp. arXiv–2302 (2023)
21. Wu, B., Wei, S., Zhu, M., Zheng, M., Zhu, Z., Zhang, M., Chen, H., Yuan, D., Liu, L., Liu, Q.: Defenses in adversarial machine learning: A survey. arXiv preprint arXiv:2312.08890 (2023)
22. Wu, Y., Chen, Y., Yuan, L., Liu, Z., Wang, L., Li, H., Fu, Y.: Rethinking classification and localization for object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10186–10195 (2020)
23. Zhang, H., Hu, S., Wang, Y., Zhang, L.Y., Zhou, Z., Wang, X., Zhang, Y., Chen, C.: Detector collapse: Backdooring object detection to catastrophic overload or blindness. arXiv preprint arXiv:2404.11357 (2024)

Fine-grained Text to Image Synthesis

Xu Ouyang(✉), Ying Chen, Kaiyue Zhu, and Gady Agam

Illinois Institute of Technology, Chicago, IL, USA
{xouyang3,ychen245,kzhu6}@hawk.iit.edu, agam@iit.edu

Abstract. Fine-grained text to image synthesis involves generating images from texts that belong to different categories. In contrast to general text to image synthesis, in fine-grained synthesis there is high similarity between images of different subclasses, and there may be linguistic discrepancy among texts describing the same image. Recent Generative Adversarial Networks (GAN), such as the Recurrent Affine Transformation (RAT) GAN model, are able to synthesize clear and realistic images from texts. However, GAN models ignore fine-grained level information. In this paper we propose an approach that incorporates an auxiliary classifier in the discriminator and a contrastive learning method to improve the accuracy of fine-grained details in images synthesized by RAT GAN. The auxiliary classifier helps the discriminator classify the class of images, and helps the generator synthesize more accurate fine-grained images. The contrastive learning method minimizes the similarity between images from different subclasses and maximizes the similarity between images from the same subclass. We evaluate on several state-of-the-art methods on the commonly used CUB-200-2011 bird dataset and Oxford-102 flower dataset, and demonstrated superior performance.

Keywords: fine-grained · GAN · contrastive learning

1 Introduction

Text to image synthesis is a fundamental problem due to gaps between text with limited information and high-resolution image with rich contents. Currently, there are three main approaches to solve this problem. The first approach is based on Generative Adversarial Networks (GANs) [1] and have achieved great success in image synthesis. GANs involves two neural networks that work in opposition as a zero-sum game: a generator that synthesizes fake image and a discriminator that evaluates whether images are fake or real. GAN approaches to synthesis include: Conditional GAN for synthesizing an image from sentence-level text, LSTM conditional GAN [3] for synthesizing images from word-level text, and fine-grained text to image synthesis based on attention [4]. Language-free text to image synthesis (LAFITE) [5] was proposed based on the Stylegan2

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78383-8_24.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 359–373, 2025.
https://doi.org/10.1007/978-3-031-78383-8_24

and CLIP models. Text and image fusion during image synthesis using a recurrent affine transformation (RAT) GAN model was proposed in [7]. All of these approaches focus on generating high-quality images, but neglect the differences between subclasses within the dataset. This can result in varying degrees of similarity among synthesized images from different subclasses and negatively affect performance.

The second approach for text to image synthesis is based on Auto Regressive Generative models, which treat text to image synthesis as a transformation from textual tokens to visual tokens based on a sequence-to-sequence Transformer model. DALL-E [8] and CogView [9] both aim to learn the relationship between texts and images based on a Transformer model. They first convert the image into a sequence of discrete image tokens with Vector Quantized Variational Autoencoder (VQ-VAE) [10], and then convert text tokens into image tokens by using a sequence-to-sequence Transformer, as both text and image are formatted as sequences of tokens. In particular, they utilize a decoder of a Transformer language model to learn from large amounts of text and image pairs. Parti [11] is a two-stage model similar to DALL-E and CogView, composed of an image tokenizer and an autoregressive model. The first step trains a vision tokenizer VIT-VQGAN [12] that transforms an image into a sequence of discrete image tokens. The second step trains an encoder-decoder based Transformer that generates image tokens from text tokens. Parti achieves improved image quality by scaling the encoder-decoder Transformer model up to 20 billion parameters. However, these Auto Regressive Generative models still lack attention to fine-grained level information and require large amounts of data, model size, and training time.

The third approach for text to image synthesis is based on diffusion models, which convert text to image from a learned data distribution by iteratively denoising a learned data distribution. GLIDE [13] was the first work to apply diffusion model with CLIP guidance and classifier-free guidance in text to image synthesis. VQ-Diffusion [14] proposed a vector-quantized diffusion model based on VQ-VAE, whose latent space is modeled by a conditional variant of the Denoising Diffusion Probabilistic Model (DDPM). DALLE-2 [15] trained a diffusion model on the CLIP image embedding space and a separate decoder to create images based on the CLIP image embeddings. Imagen [16] used a frozen T5-XXL encoder to map text to a sequence of embeddings, an image diffusion model, and two super-resolution image diffusion models. These three image diffusion models are all conditioned on the text embedding sequence and use classifier-free guidance. However, these diffusion models still lack attention to fine-grained level information and require huge resources.

To address the challenge of preserving fine-grained information and minimizing computational costs, we propose that utilizes the Recurrent Affine Transformation (RAT) GAN, which achieved state-of-the-art performance on fine-grained datasets while using acceptable number of parameters. Additionally, we introduce an auxiliary classifier in the discriminator to help RAT GAN synthesize more accurate fine-grained images. Specifically, the classifier classifies both fake and real images and assists the generator in synthesizing fine-grained images. While fine-grained categories may be hard to obtain for images in the wild, they are available in many cases and our approach can leverage this information for improved results. Moreover, semi-supervised and weakly supervised techniques could also help address lack of categories.

Furthermore, we introduce contrastive learning to further improve the fine-grained details of the images synthesized by RAT GAN, particularly on datasets with different subclasses. The contrastive learning method minimizes the similarity of fake/real images from different subclasses and maximizes the similarity of fake/real images from the same subclass. We incorporate the cross-batch memory (XBM) [17] mechanism, which allows the model to collect hard negative pairs across multiple mini-batches and even over the entire dataset, to further improve the performance of the model.

In summary, there are three primary contributions in this paper. First, we introduce an auxiliary classifier in the discriminator, which not only classifies the category of fake/real images but also assists in synthesizing fine-grained images from the generator. Second, we introduce a contrastive learning method with cross-batch memory (XBM) mechanism, which helps the generator to synthesize images with higher similarity within the same subclass and lower similarity among different subclasses. Meanwhile, our method is an efficient approach, as it only introduces small additional expense in the form of two fully connected layers for feature dimension reduction, image classification and feature embedding. Third, our method demonstrates state-of-the-art performance on two common fine-grained image datasets: CUB-200-2011 bird dataset and Oxford-102 flower dataset.

2 Related Work

RAT GAN [7] was proposed to address text and image isolation during image synthesis. They introduce Recurrent Affine Transformation (RAT) for controlling all fusion blocks consistently. RAT expresses different layers' outputs with standard context vectors of the same shape to achieve unified control of different layers. The context vectors are then connected using RNN in order to detect long-term dependencies. With skip connections in RNN, RAT blocks are consistent between neighboring blocks and reduce training difficulty. Moreover, they incorporate a spatial attention model in the discriminator to improve semantic consistency between texts and images. With spatial attention, the discriminator can focus on image regions that are related to the corresponding captions. We discovered RAT GAN maintains top performance with acceptable parameters compared to other leading methods. Thus, we adopt the RAT GAN as our backbone model.

The basic GAN framework can be augmented using side information such as class and caption. Instead of feeding side information to the discriminator, one can task the discriminator with reconstructing side information. This is done by modifying the discriminator to contain an auxiliary decoder network that outputs the class label for the training data [18] or a subset of the latent variables from which are generated [19]. Forcing a model to perform additional tasks is known to improve performance on the original task. In addition, an auxiliary decoder could leverage pre-trained discriminators for further improving the synthesized images [20]. Motivated by these considerations, ACGAN [21] proposed a class conditional GAN model, but with an auxiliary decoder that is tasked with reconstructing class labels. TAC GAN [23] present a Text Conditioned Auxiliary Classifier Generative Adversarial Network for synthesizing images from their text descriptions. The discriminator of TAC-GAN performs an auxiliary task of classifying the synthesized and the real data into their respective class labels. During training, the discriminator minimize the cross entropy of categories and the generator maximize the cross entropy of categories. Inspired by their work, we introduce an auxiliary classifier in the discriminator of the RAT GAN model. This classifier could not only classify which category the images belong to, but also help generator to synthesize fine-grained level images.

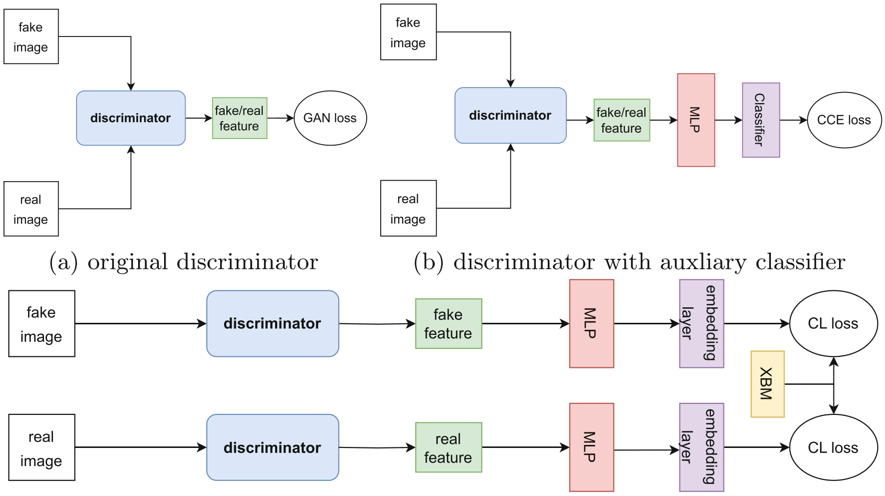
[22] propose a contrastive learning method to improve the quality and enhance the semantic consistency of synthetic images synthesized from texts. In the image-text matching task, they utilize the contrastive loss to minimize the distance of the fake images generated from text descriptions related to the same ground truth image while maximizing those related to different ground truth images. However, they ignored the similarity among fake images of different subclasses and introduced a pretrained image encoder to compute contrastive loss which increased the computation complexity of the model. [25] also propose a contrastive learning method for text to image synthesis. They introduce multiple generators and discriminators and only compute the contrastive loss between image features from the geneartor. [24] propose a cross-modal contrastive learning for text to image synthesis. They only compute the contrastive loss between a real image and a fake image. In our work, we only add one fully connected layer to extract feature embedding and compute the contrastive loss between fake and real images, between fake and fake images, and between real and real images. The advantage of our approach is that with a small number of parameters, we can compute the contrastive loss between fake/real images in one step, rather than first training an image encoder and then computing contrastive loss as in [22].

3 Proposed Approach

We adopt the RAT GAN as our base model and enhance it by introducing an auxiliary classifier and a contrastive learning method thus creating a fine-grained (FG) RAT GAN. In the following sections, we provide detailed information on how these modifications work and present the overall algorithms.

3.1 Auxiliary Classifier

In the discriminator of the RAT GAN, we add an auxiliary classifier at the end of the network. To do this, we first flatten the output dimension from 8x8x1024



(a) original discriminator

(b) discriminator with auxliary classifier

(c) discriminator with contrastive learning

Fig. 1. The original discrminator in Figure (a) computes GAN loss. The discriminator with auxliary classifier in Figure (b) computes categorical cross entropy loss. The discrminator with contrastive learning in Figure (c) computes contrastive learning loss.

to 64x1024 and add a fully connected layer to reduce the feature dimension from 64x1024 to 256. We then add a Softmax activation function to classify the feature into one of the predefined categories. The structure of the modified discriminator is shown in Figure 1(b). In comparison to the original RAT GAN discriminator shown in Figure 1(a) which only computes the GAN loss, we also minimize the categorical cross-entropy loss between the classifier output and the ground truth labels of the images during both generator and discriminator updates. These losses are defined as follows:

$$L_d^{ce} = -\sum_{i=1}^{i=N}(y_i \cdot log(\hat{y_i^f})) + \sum_{i=1}^{i=N}(y_i \cdot log(\hat{y_i^r})) \tag{1}$$

$$L_g^{ce} = -\sum_{i=1}^{i=N}(y_i \cdot log(\hat{y_i^f})) \tag{2}$$

where the y_i is the ground truth label of the image, y_i^f is the auxiliary output of the fake image, and y_i^r is the auxiliary output of the real image.

L_d^{ce} allows the discriminator to classify the category of images, by computing the sum of the categorical cross-entropy loss between the classifier output of fake images and their ground-truth labels, and the categorical cross-entropy loss between the classifier output of real images and their ground-truth labels. L_g^{ce} helps the generator to synthesize more precise and fine-grained images by incorporating the classifier's output into the loss function.

3.2 Contrastive Learning

In order to improve the quality and semantic consistency of synthetic images generated from text, we introduce a contrastive learning method in our model. To implement this, we add a branch embedding layer and L-2 normalization to the feature embeddings of our images after the fully connected layer for feature dimension reduction. This is illustrated in Figure 1(c).

In addition, we introduce cross-batch memory (XBM) mechanism in our contrastive loss calculation. This creates a memory bank that acts as a queue, where the current mini-batch of real images' feature embeddings are enqueued and the oldest mini-batch of feature embeddings are dequeued. We then minimize the contrastive loss between the fake images' feature embeddings and the entire XBM feature embeddings, as well as the contrastive loss between the real images' feature embeddings and the entire XBM feature embeddings. The contrastive loss is defined as follows:

$$\begin{aligned} L_d^{cl} =& \frac{1}{NM}\sum_i^N [\sum_{j:y_i=y_j}^M (1-\cos\_\text{sim}(e_i^r, e_j^x)) + \\ & \sum_{j:y_i\neq y_j}^M -\max((\cos\_\text{sim}(e_i^r, e_j^x)-\alpha), 0)], \end{aligned} \tag{3}$$

$$\begin{aligned} L_g^{cl} =& \frac{1}{NM}\sum_i^N [\sum_{j:y_i=y_j}^M (1-\cos\_\text{sim}(e_i^f, e_j^x)) + \\ & \sum_{j:y_i\neq y_j}^M -\max((\cos\_\text{sim}(e_i^f, e_j^x)-\alpha), 0)], \end{aligned} \tag{4}$$

where $\cos\_\text{sim}(e_i^f, e_j^x)$ is the cosine similarity between the feature embedding e_i^f of mini-batched fake images and the feature embedding e_j^x of real image from the cross-batch memory (XBM), α is a margin applied to the cosine similarity of negative pairs to prevent the loss from being dominated by easy negatives, N is the batch size, and M is the size of the XBM. The L_d^{cl} loss function minimizes the similarity between feature embeddings of real images from different subclasses, and maximizes the similarity between feature embeddings of real images from the same subclass, which optimizes the embedding layer in the discriminator. The L_g^{cl} loss function minimizes the similarity between feature embeddings of fake and real images from different subclasses, and maximizes the similarity between feature embeddings of fake and real images from the same subclass, which helps the generator synthesize fine-grained images.

3.3 Training of the network

In this section, we describe the training process of our proposed FG-RAT GAN with auxiliary classifier and contrastive learning as shown in Figure 2. The fake

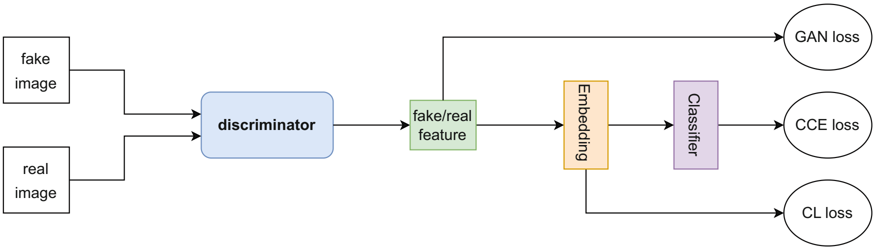


Fig. 2. The structure of the discriminator with auxiliary classifier and contrastive learning. The original output of the discriminator is still used to compute the GAN loss, and meanwhile followed by one fully connected layer to decrease the feature dimension. Next, the fully connected layer is followed by one embedding layer for contrastive learning. Then, the embedding layer is followed by a classifier for image classification.

image synthesized from generator G and the real image separately pass through discriminator D. The discriminator D then discriminates whether the image is fake/real by minimaxing GAN loss which is defined as follows:

$$\begin{aligned} L_d^{adv} =&\mathbb{E}_{x \backsim p_{data}}[\max(0, 1 - D(i_r, t))] + \\ &0.5 \times \mathbb{E}_{z \backsim p_{gen}}[\max(0, 1 + D(G(z, t), t))] + \\ &0.5 \times \mathbb{E}_{z \backsim p_{data}}[\max(0, 1 + D(i_r{}', t))] \end{aligned} \tag{5}$$

$$L_g^{adv} = \mathbb{E}_{z \backsim p_{gen}}[\min D(G(z, t), t)] \tag{6}$$

where $G : (Z, T) \rightarrow X$ maps from the latent space Z and caption space T to the input space X, $D : X \rightarrow \mathbb{R}$ maps from the input space to a classification of the example as fake or real, i_r is the real image, and $i_r{}'$ is the mismatched real image. The GAN model will reach a global optimal value when $p_{gen} = p_{data}$, where p_{gen} is the generative data distribution and p_{data} is the real data distribution.

Subsequently, different from Sect. 3.1 and Sect. 3.2, in the end of the discriminator, we first add an embedding layer which is used for feature dimension reduction and contrastive learning. We add a classifier after this embedding layer for image classification. We first only compute the categorical cross-entropy loss L_d^{ce} and L_g^{ce} for image classification as mentioned in Sect. 3.1. This is because the feature drift is relatively large at the early epochs. Training the neural networks with L_d^{ce} and L_g^{ce}, allows the embeddings to become more stable. After several training epochs, we add the contrastive loss L_d^{cl} and L_g^{cl} for contrastive learning as mentioned in Sect. 3.2. We finally compute the total loss for the discriminator D and the generator G as follows:

$$L_d^{total} = L_d^{adv} + L_d^{ce} + L_d^{cl} \tag{7}$$

$$L_g^{total} = L_g^{adv} + L_g^{ce} + L_g^{cl} \tag{8}$$

We update the parameters of the discriminator D by minimizing the L_d^{total} loss and update the parameters of the generator G by minimizing the L_g^{total} loss.

4 Experiments

4.1 Datasets

To evaluate the performance of fine-grained text to image synthesis, we conduct experiments on two commonly used fine-grained text-image pair datasets: the CUB-200-2011 dataset which contains 11,788 images of 200 different bird species; and the Oxford-102 flower dataset which contains 8,189 images of 102 different flower species. We follow the same split as previous studies [2,7,14] for both datasets: 150 training classes and 50 testing classes for CUB-200-2011, and 82 training classes and 20 testing classes for Oxford-102. Each image in the datasets is paired with ten text descriptions. The images are resized to 304x304, randomly cropped to 256x256, and then randomly flipped horizontaly. The captions are passed through a text encoder, resulting in an output of size 256.

4.2 Evaluation Metrics

The Inception Score [26] can measure a synthetic image quality by computing the expected Kullback Leibler divergence (KL divergence) between the marginal class distribution and conditional label distribution:

$$IS = exp(\mathbb{E}_x KL(p(y|x)||p(y))) \tag{9}$$

where $p(y|x)$ is the conditional label distribution of features extracted from the middle layers of the pretrained Inception-v3 model for generated images, and p(y) is the marginal class distribution. IS gives a score that tells us if each image made by the model is clear and distinct, and if the model can make a wide range of different images. We want models that make a mix of clear images, so a higher IS is better.

The Frechet Inception Distance [27] that is given by:

$$d^2(F,G) = |\mu_x - \mu_y|^2 + tr|\Sigma_x + \Sigma_y - 2(\Sigma_x\Sigma_y)^{1/2}| \tag{10}$$

where F, G are two distributions of features extracted from the middle layers of a pretrained Inception-v3 model for generated and real images. The parameters μ_x, μ_y, Σ_x, Σ_y, are the mean vectors and covariance matrices of F and G. While IS checks image clarity and variety, FID checks if they look real. We want our model's images to look like real photos, so a lower FID is better.

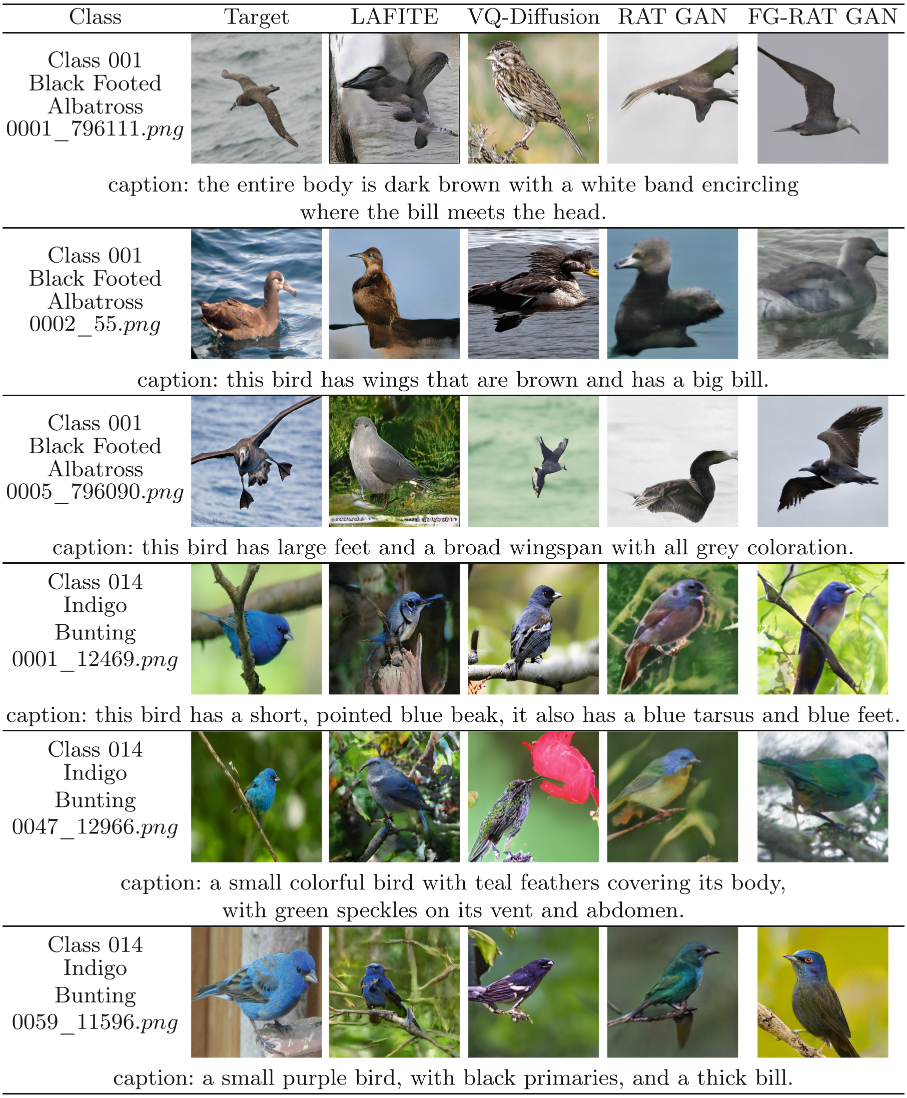
The paper [28] highlights that the Inception Score (IS) is sensitive to model overfitting and dependent on the dataset used for the Inception network, often leading to misleading evaluations for models not trained on ImageNet. In contrast, the Frechet Inception Distance (FID) compares the statistical distributions of real and generated images using the Frechet distance, assessing how closely generated images mimic real images in content and style. This makes FID a more reliable and comprehensive metric, as it directly evaluates the realism and diversity of generated images, unlike IS which does not compare with the distribution of real images.

4.3 Implementation details

In our implementation, we adopt the RAT GAN architecture as the backbone for our model. We use a pretrained bidirectional LSTM network to convert text descriptions into sentence-level feature vectors of size 256. These feature vectors are combined with Gaussian noise as input for the generator. The generator comprises of six up-sampling blocks, each followed by a Recurrent Affine Transformation (RAT) block to control image content. The discriminator includes six down-sampling blocks, whose output size is 8x8x1024. We then add a fully connected layer to decrease the output size to 256 for contrastive learning, followed by a fully connected layer for image classification with an output size of 200 for the CUB-200-2011 dataset and 102 for the Oxford-102 dataset. We use the Adam optimizer to train the generator with an initial learning rate of $1e-4$ and the discriminator with an initial learning rate of $4e-4$. We use cosine learning rate decay to decrease the learning rate to $1e-6$ and train with 600 epochs.

4.4 Qualitative evaluation

Figure 3 shows synthesized images generated by LAFITE, VQ-Diffusion, RAT GAN and our FG-RAT GAN on the CUB-200-2011 bird dataset. As we can see, in the 1st row the proposed FG-RAT GAN generates a bird with dark brown body and white band encircling near the bill as specified in the caption, in the 3rd row it generates a bird with all gray body as specified in the caption, and both examples are similar to each other given that they belong to the same class. Figure 4 only shows synthesized images generated by RAT GAN and our proposed FG-RAT GAN on the Oxford-102 flower dataset since LAFITE did not train or test on this dataset and VQ-Diffusion did not post their pretrianed model on this dataset. As we can see, the 5th row generates a flower with white petals and yellow stamen as in the description, the 6th row generates a flower with white petals and yellow stamen as in the description, and both samples are similar to each other given they belong to the same class. There are six samples which belong to two different classes in each dataset. As we can see, our proposed FG-RAT GAN can generate fine-grained images which highly correspond to the given captions. Additionally, each synthesized image is more similar to other synthesized images in the same class. Thus, we demonstrate that our FG-RAT GAN can reach better visualized results compared with the orginal RAT GAN. In addition, we show some visualized results compared with DALLE-2 and Stable Diffusion on these datasets in the Supplementary materials.[0]



caption: the entire body is dark brown with a white band encircling where the bill meets the head.

caption: this bird has wings that are brown and has a big bill.

caption: this bird has large feet and a broad wingspan with all grey coloration.

caption: this bird has a short, pointed blue beak, it also has a blue tarsus and blue feet.

caption: a small colorful bird with teal feathers covering its body, with green speckles on its vent and abdomen.

caption: a small purple bird, with black primaries, and a thick bill.

Fig. 3. Examples of generated images using RAT GAN and the proposed FG-RAT GAN on the CUB bird dataset. Each row represents a different sample (image size = 256x256) and with the corresponding caption below.The first column is image class and name. The second column is the corresponding target image. The rest of other columns are the generated images from LAFITE, VQ-Diffusion, RAT GAN, and our FG-RAT GAN. As we can see, our FG-RAT GAN can generate more realistic images where each image is similar to other images within the same class.

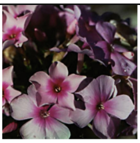
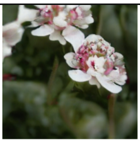
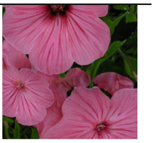
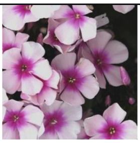
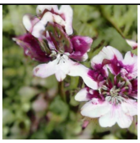
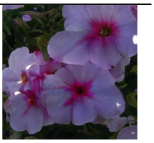
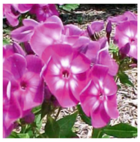
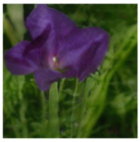
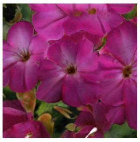
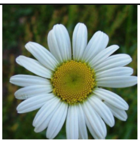
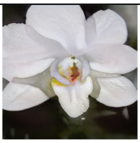
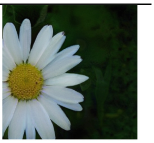
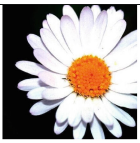
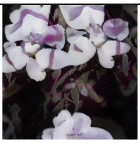
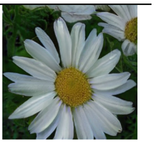
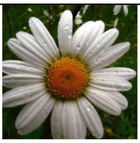
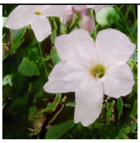
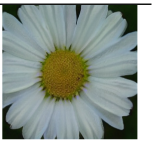
Class	Caption	Target	RAT GAN	FG-RAT GAN
Class 032 *image*_05587.*png*	the petals of flowers are various shades of pink and have five individual petals.			
Class 032 *image*_05602.*png*	a large group of light pink flowers with dark pink centers.			
Class 032 *image*_05604.*png*	these flowers are mostly pink but some of them have white parts located closer to their stamens.			
Class 049 *image*_06209.*png*	this flower has thin white petals as its main feature.			
Class 049 *image*_06216.*png*	the petals on this flower are white with yellow stamen.			
Class 049 *image*_06224.*png*	the flower has petals of a white color with a many yellow stamen.			

Fig. 4. Examples of generated images using RAT GAN and the proposed FG-RAT GAN with classifier and contrastive learning trained on the Oxford flower dataset. Each row represents a different sample (image size=256x256). The first column is the sample detail including class and specific image name. The second column is the caption. The third column is the corresponding target image. The fourth column is the image generated by RAT GAN. The fifth column is the image generated by our proposed FG-RAT GAN. As we can see, our proposed FG-RAT GAN can generate more realistic images where each image is similar to other images within the same class.

4.5 Quantitative evaluation

We compare the state-of-the-art text to image synthesis methods LAFITE, VQ-Diffusion, RAT GAN, and our FG-RAT GAN. We evaluate the CUB-200-2011 bird dataset and the Oxford-102 flower dataset with Inception Score (IS) and Frenchet Inception Distance (FID) which are commonly used text to image synthesis performance evaluation metrics. Due to suboptimalities of the Inception

Table 1. Comparison of previous state-of-the-art methods: LAFITE, VQ-Diffusion, RAT GAN and our proposed FG-RAT GAN on the CUB-200-2011 bird and Oxford-102 flower dataset for text to image synthesis. Each row presents a different model. The first column is the name of each model. The second column is the number of parameters of each model. The third and forth columns show the IS and FID results for the bird dataset. The fifth and sixth columns show the IS and FID results for the flower dataset.″–″ means the author did not provide results. As can be observed, in both datasets, our proposed FG-RAT GAN reaches the lowest FID scores.

		CUB bird dataset		Oxford flower dataset	
Model	NP	IS↑	FID↓	IS↑	FID↓
LAFITE	75M+151M	5.97	10.48	–	–
VQ-Diffusion	370M	–	10.32	–	14.1
RAT GAN	38M+113M	4.83	12.12	3.62	12.90
FG-RAT GAN **(our)**	38M+130M	4.99	8.66	3.45	9.14

Score itself and problems with the popular usage of the Inception Score, we care more about FID than IS. We show the evaluation results in Table 1. As can be observed, on the CUB-200-2011 bird dataset, our method reaches the lowest FID scores. On the Oxford flower dataset, RAT GAN reaches the highest IS score and our proposed method reaches the lowest FID score. In addition, our proposed method only add 17M parameters to the discriminator of RAT GAN and has 168M parameters while LAFITE has 226M parameters and VQ-Diffusion has 370M parameters. Even though we use labels during the training, label information is not an unfair advantage but a distinct characteristic of our model. The goal is to advance the field, rather than to compete under identical conditions. Thus, we demonstrate our FG-RAT GAN reaches better performance while only adding a relatively small number parameters to the baseline model.

4.6 Ablation study

We investigate the effects of different strategies we added to the RAT GAN model for text to image synthesis to demonstrate their significance on both the CUB-200-2011 bird and Oxford-102 flower datasets. We train three different models: A proposed FG-RAT GAN with auxiliary classifier, a proposed FG-RAT GAN with contrastive learning, and a proposed FG-RAT GAN with combination of auxiliary classifier and contrastive learning. The results are summarized in the Table 2. As can be observed, the proposed FG-RAT GAN with auxiliary classifier reaches the highest IS score, whereas the proposed FG-RAT GAN with a combination of auxiliary classifier and contrastive learning reaches the lowest FID score on the CUB-200-2011 bird dataset. The proposed FG-RAT GAN with contrastive learning reaches the highest IS score and the proposed FG-RAT GAN with combination of auxiliary classifier and contrastive learning reaches the lowest FID score on the Oxford-102 flower dataset. In summary, the ablation

Table 2. Comparison of RAT GAN, proposed FG-RAT GAN with auxiliary classifier, proposed FG-RAT GAN with contrastive learning, and proposed FG-RAT GAN with combination of auxiliary classifier and contrastive learning on the CUB-200-2011 bird and Oxford-102 flower dataset. Each row presents a different model. The first column is the name of each model. The second and third columns show the IS and FID scores for the CUB bird dataset. The fourth and fifth columns show the IS and FID scores for the Oxford flower dataset. As can be observed, in CUB bird dataset, the proposed FG-RAT GAN with classifier reaches the highest IS score and the proposed FG-RAT GAN with classifier and contrastive learning reaches the lowest FID score. In the Oxford flower dataset, the proposed FG-RAT GAN with contrastive learning reaches the highest IS and the proposed FG-RAT GAN with classifier and contrastive learning reaches the lowest FID.

	CUB bird dataset		Oxford flower dataset	
Model	IS↑	FID↓	IS↑	FID↓
RAT GAN	4.83	12.12	3.62	12.90
RAT GAN + classifier **(our)**	5.08	9.90	3.45	9.55
RAT GAN + contrtastive learning **(our)**	4.84	9.10	3.66	10.63
FG-RAT GAN **(our)**	4.99	8.66	3.45	9.14

study demonstrates that our FG-RAT GAN reaches better performance than the RAT GAN model.

5 Conclusion

In this paper, we present a novel approach for generating fine-grained images from text descriptions, by incorporating an auxiliary classifier and contrastive learning into the RAT GAN architecture. Our proposed FG-RAT GAN approach improves the quality and semantic consistency of synthetic images by leveraging the auxiliary classifier to classify images into different categories, and using contrastive learning to generate images with higher similarity within the same class and lower similarity among different classes. Additionally, our method is computationally efficient, as it adds two fully connected layers to the original RAT GAN model only during training stage. We demonstrate that our method reaches state-of-the-art performance on two commonly used fine-grained image datasets. While FG-RAT GAN demonstrates strong performance, it does depend on the availability of fine-grained labels, which could limit its applicability in real-world scenarios where labels are less accurate or unavailable. In future work, we aim to reduce this dependency and explore the method's adaptability in more diverse and less structured environments. Additionally, we will conduct further evaluations on broader text-to-image synthesis benchmarks and more varied datasets are necessary to confirm the generalizability of our approach.

References

1. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS'14), pp. 2672–2680. MIT Press, Cambridge, MA, USA (2014)
2. Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., Lee, H.: Generative Adversarial Text to Image Synthesis. In: Balcan, M.F., Weinberger, K.Q. (eds.) Proceedings of The 33rd International Conference on Machine Learning, vol. 48, pp. 1060–1069. PMLR, New York, New York, USA (2016)
3. Ouyang, X., Zhang, X., Ma, D., Agam, G.: Generating Image Sequence from Description with LSTM Conditional GAN. In: 2018 24th International Conference on Pattern Recognition (ICPR), pp. 2456–2461. IEEE, Beijing, China (2018)
4. Xu, T., Zhang, P., Huang, Q., Zhang, H., Gan, Z., Huang, X., He, X.: AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1316–1324. IEEE, Salt Lake City, UT, USA (2018)
5. Zhou, Y., Chen, H., Zhang, W., Sun, Z., He, X., Fan, Y.: Towards Language-Free Training for Text-to-Image Generation. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 17886–17896. IEEE, New Orleans, LA, USA (2022)
6. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and Improving the Image Quality of StyleGAN. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8107–8116. IEEE (2020)
7. Ye, S., Wang, H., Tan, M., Liu, F.: Recurrent Affine Transformation for Text-to-Image Synthesis. In: IEEE Transactions on Multimedia, vol. 26, pp. 462–473. IEEE (2024)
8. Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., Sutskever, I.: Zero-Shot Text-to-Image Generation. In: Proceedings of the 38th International Conference on Machine Learning, Proceedings of Machine Learning Research, vol. 139, pp. 8821–8831. PMLR (2021)
9. Ding, M., Yang, Z., Hong, W., Zheng, W., Zhou, C., Yin, D., Lin, J., et al.: Cogview: Mastering text-to-image generation via transformers. In: Advances in Neural Information Processing Systems, vol. 34, pp. 19822–19835. (2021)
10. van den Oord, A., Vinyals, O., Kavukcuoglu, K.: Neural discrete representation learning. In: Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17), pp. 6309–6318. Curran Associates Inc., Red Hook, NY, USA (2017)
11. Yu, J., Xu, Y., Koh, J.Y., Luong, T., Baid, G., Wang, Z., Vasudevan, V., et al.: Scaling autoregressive models for content-rich text-to-image generation. arXiv preprint arXiv:2206.10789, vol. 2, no. 3, p. 5 (2022)
12. Yu, J., Li, X., Koh, J.Y., Zhang, H., Pang, R., Qin, J., Ku, A., Xu, Y., Baldridge, J., Wu, Y.: Vector-quantized image modeling with improved VQGAN. arXiv preprint arXiv:2110.04627 (2021)
13. 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. arXiv preprint arXiv:2112.10741 (2021)
14. Gu, S., Liu, Z., Ye, X., Lin, T., Wang, M., Cui, S., Liu, H., Liu, Y., Sun, C., Du, J., Hu, H.: Vector Quantized Diffusion Model for Text-to-Image Synthesis. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 10686–10696. IEEE, New Orleans, LA, USA (2022)

15. Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M.: Hierarchical text-conditional image generation with CLIP latents. arXiv preprint arXiv:2204.06125, vol. 1, no. 2, p. 3 (2022)
16. Saharia, C., Chan, W., Saxena, S., Lit, L., Whang, J., Denton, E., Seyed Ghasemipour, S.K., Karagol Ayan, B., Mahdavi, S.S., Gontijo-Lopes, R., Salimans, T., Ho, J., Fleet, D.J., Norouzi, M.: Photorealistic text-to-image diffusion models with deep language understanding. In: Proceedings of the 36th International Conference on Neural Information Processing Systems (NIPS '22), Article 2643, pp. 36479–36494. Curran Associates Inc., Red Hook, NY, USA (2024)
17. Wang, X., Zhang, H., Huang, W., Scott, M.R.: Cross-Batch Memory for Embedding Learning. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6387–6396. IEEE, Seattle, WA, USA (2020)
18. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training GANs. In: Proceedings of the 30th International Conference on Neural Information Processing Systems (NIPS'16), pp. 2234–2242. Curran Associates Inc., Red Hook, NY, USA (2016)
19. Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., Abbeel, P.: Info-GAN: interpretable representation learning by information maximizing generative adversarial nets. In: Proceedings of the 30th International Conference on Neural Information Processing Systems (NIPS'16), pp. 2180–2188. Curran Associates Inc., Red Hook, NY, USA (2016)
20. Nguyen, A., Dosovitskiy, A., Yosinski, J., Brox, T., Clune, J.: Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. In: Proceedings of the 30th International Conference on Neural Information Processing Systems (NIPS'16), pp. 3395–3403. Curran Associates Inc., Red Hook, NY, USA (2016)
21. Odena, A., Olah, C., Shlens, J.: Conditional image synthesis with auxiliary classifier GANs. In: Proceedings of the 34th International Conference on Machine Learning (ICML'17), vol. 70, pp. 2642–2651. JMLR.org (2017)
22. Ye, H., Yang, X., Takac, M., Sunderraman, R., Ji, S.: Improving text-to-image synthesis using contrastive learning. arXiv preprint arXiv:2107.02423 (2021)
23. Dash, A., Gamboa, J.C.B., Ahmed, S., Liwicki, M., Afzal, M.Z.: Tac-gan-text conditioned auxiliary classifier generative adversarial network. arXiv preprint arXiv:1703.06412 (2017)
24. Zhang, H., Koh, J.Y., Baldridge, J., Lee, H., Yang, Y.: Cross-Modal Contrastive Learning for Text-to-Image Generation. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 833–842. IEEE, Nashville, TN, USA (2021)
25. Yin, G., Liu, B., Sheng, L., Yu, N., Wang, X., Shao, J.: Semantics Disentangling for Text-To-Image Generation. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2322–2331. IEEE, Long Beach, CA, USA (2019)
26. Salimans, T., Goodfellow, I.J., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved Techniques for Training GANs. In: Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, pp. 2226–2234. (2016)
27. Dowson, D.C., Landau, B.V.: The Fréchet distance between multivariate normal distributions. J. Multivar. Anal. **12**(3), 450–455 (1982)
28. Barratt, S., Sharma, R.: A Note on the Inception Score. arXiv preprint arXiv:1801.01973. (2018)

ORA-Trans: Object Region Attention Transformer Based on Key Tokens Selector with Structure Feature Modeling for Fine-Grained Visual Classification

Yulong Xia and Jianwei Zhang(✉)

School of Computer Science and Engineering, South China University of Technology,
Guangzhou 510006, China
jwzhang@scut.edu.cn

Abstract. The task of Fine-Grained Visual Classification (FGVC) aims to distinguish between closely related subclasses within a broader category. Challenges include high intra-class variation, minimal inter-class differences, and complex backgrounds, making accurate classification demanding. Existing approaches usually target on specific traits for optimization and do not pay enough attention to both local details and the object's structural information in images simultaneously. In this paper, We designed a new Object Region Attention transformer (ORA-Trans), which enables the model to better discern subtle image differences by mining deeply into discriminative regions and structural features of objects in images. Specifically, we propose a Key Tokens Selector to automatically identify tokens corresponding to crucial local regions that are pivotal for FGVC. Furthermore, we design an object Structure Feature Extractor to capture the structural relationships among the object parts tokens, which can not only optimize the feature learning process of the tokens selected by KTS, but also alleviate the interference of complex image backgrounds. We also introduce a Token Stochastic Swap module to enrich feature combinations and enhance feature representativeness. Our proposed approach achieves state-of-the-art performances on the CUB_200_2011, Stanford dogs, and IP102 datasets. Ablation studies and visualization demonstrate the effectiveness and interpretability of our method.

Keywords: fine-grained visual classification · vision transformer · attention mechanism · Jensen-Shannon divergence · information entropy

1 Introduction

Fine-Grained Visual Classification (FGVC) is an important and challenging problem in Computer Vision, which aims at identifying subclasses under a broad category. The fine-grained image datasets usually come with large intra-class variances and small inter-class variances, hence we have to focus on the object

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 374–389, 2025.
https://doi.org/10.1007/978-3-031-78383-8_25

areas in images, capturing discriminative features for fine-grained visual classification. With the rapid development of deep learning, The application scenarios for FGVC based on deep learning are increasingly expanding [1]. The existing methods for FGVC can be roughly divided into two classes: localization-based methods [2] and attention-based methods [3].

Localization-based methods, such as the R-CNN series [4], identify key image parts through preprocessing or specialized modules, focusing on discriminative regions. R-CNNs extract numerous region blocks, filtering and merging them to identify candidate regions for classification based on their features. However, this method relies on detailed labels, presenting a challenge due to its need for strong supervision. To address this, researchers have shifted towards weakly supervised methods. For instance, Weakly Supervised Discriminative Localization (WSDL) [5] employs an attention extraction network that generates bounding box data, guiding R-CNNs to classify images more quickly, reducing the dependency on extensive annotations and enhancing classification accuracy.

Unlike localization-based methods that rely on explicit annotations, attention-based methods like the Vision Transformer (ViT) [6] simulate human attention to identify key image features without detailed part labels. These methods leverage self-attention mechanisms [7] to connect and assess all image regions equally, enhancing global dependency capture and comprehensive image analysis. However, ViT's lack of inductive bias towards local features, unlike CNNs with their confined receptive fields, may compromise its ability to detect subtle yet critical differences in similar images, such as variations in a bird's beak, wings, or feet, potentially affecting classification accuracy.

Fine-grained image datasets commonly exhibit high intra-class variance and low inter-class variance. The recognition tasks are further complicated by the presence of intricate backgrounds. Additionally, challenges in sample collection often result in insufficient training data, which contributes to a long-tailed distribution of sample sizes across different classes. In order to address these issues, we propose a new classification model that utilizes a Vision Transformer (ViT) as its backbone — Object Region Attention Transformer (ORA-Trans), which extracts both critical regional features and object structural information in images. Specifically, we design a module named Key Tokens Selector (KTS), which can efficiently select tokens corresponding to important object parts in images from each layer, thereby enabling the model to capture the subtle differences between images, alleviating the issue of small inter-class variance and interference from complex backgrounds; Then a module named Structure Feature Extractor (SFE) is designed to extract object structure features, which selects an anchor token and to-be-modeled tokens by using the attention weight vectors and their information entropy, Graph convolution is then used to generate features containing structural information of them, thereby alleviating the classification difficulties brought about by variations in object pose. To enrich pattern combinations and optimize feature space, we introduce a module named Token Stochastic Swap (TSS) to randomly swap a part of the tokens generated by the two modules in a batch respectively. We conducted experiments on two prevalent FGVC datasets, a large-scale pest recognition dataset and a small ultra-fine-grained image dataset to fully validate the effectiveness of our method. Our contributions can be summarized as follows:

- We propose a framework that extracts crucial detail features from ViT, constructs object structural features to aid the learning of these details, and employs a Token Stochastic Swap module to enrich feature combinations, enhancing feature representation and thus improving classification accuracy.
- We design Key Tokens Selector (KTS) to select tokens containing local information of important object parts from each layer of the transformer.
- We design a module named Structure Feature Extractor (SFE) to extract object structural information and leverage it to optimize the learning process of the model for the features of the tokens selected by KTS.

2 Related Work

Fine-Grained Visual Classification. FGVC presents unique challenges in computer vision, exceeding those of traditional image classification by requiring precise differentiation of subtle visual distinctions. Key challenges in FGVC include: (1) Large intra-class variance and high inter-class similarity [8], where orientation and part spacing within the same category can vary significantly, yet appear similar across different categories. (2) Complex backgrounds in images may obscure crucial features [9], complicating the extraction of useful information and disrupting the classification process. (3) A scarcity of training data due to the fine granularity and the specialized expertise required for accurate annotation, which limits the models' learning potential and affects their generalization and robustness [1]. To address the first challenge mentioned above in FGVC, several part-based methods have been developed. For example, NTS-Net [10] uses a region proposal network to identify and resize salient bounding boxes for recognition. To combat background interference, HERBS [11] segments feature maps based on confidence scores, enhancing salient features and reducing noise. Addressing data scarcity and model overfitting, methods like CLE-ViT [12] augment datasets with transformations to optimize feature spaces, while PC [13] introduce activation confusion to improve generalization and robustness.

Attention-based methods. The attention mechanism in deep learning, which simulates human visual and cognitive processes, allows models to focus selectively on key input features, thus enhancing performance and generalization in FGVC. For example, RA-CNN [14] employs an Attention Proposal Network to progressively refine focus on discriminative regions across scales, improving feature extraction. API-Net [15] uses attention to train on differences between similar images, learning distinct features crucial for distinguishing subtle variations in FGVC. Following the success of the Transformer [7] in vision tasks, models like FFVT [16] and TransFG [17] use attention weights to highlight and concentrate on significant image regions, boosting classification accuracy. Moreover, AF-Trans [18] and RAMS-Trans [19] leverage attention to enhance detail recognition, integrating global and local data, thus compensating for the shortcomings of ViT in FGVC tasks. SIM-Trans [20] enhances ViT by incorporating structural object features, focusing more on object-specific areas and improving fine-grained classification. In our review of the ViT architecture, we identified two potential directions that can be combined for optimization: selective attention and structured representation. Current methods often focus on a single optimization direction, However, local object information and object structure information are complementary. Integrating both types of information could significantly enhance classification performance. To overcome existing limitations, our work introduces a Key Tokens Selector to highlight critical local features and a Structure Feature Extractor to capture the interrelations of object parts, thus improving the representation and interpretability of the ViT feature space.

3 Method

In this section, we detail our methodology, beginning with an outline of our model's architecture. We then review the feature extraction process utilized by ViT and delve into the principles and functions of the two modules developed specifically for fine-grained image classification. Finally, we discuss the process for learning fine-grained features.

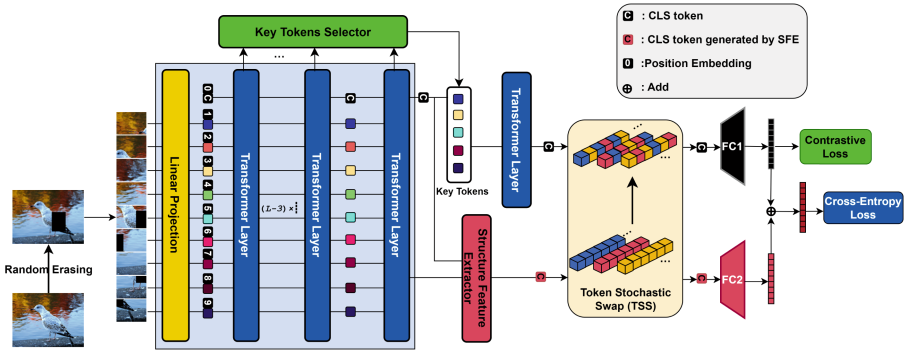


Fig. 1. The framework of our proposed ORA-Trans.

3.1 Framework

To achieve high accuracy in fine-grained image classification, our model integrates both local and global image information using the Vision Transformer. Despite its limited local modeling capabilities, we enhance feature extraction by selecting the most representative features from each layer for information interaction. This approach enriches the model's understanding of local details while retaining critical holistic object information for precise classification. Our methodology incorporates three main modules: a Key Tokens Selector for identifying key tokens that capture discriminative object characteristics in each layer, and a Structure Feature Extractor for modeling the relationships between object parts. These modules are integrated with the Vision Transformer to support end-to-end training and optimization. Additionally, a Token Stochastic Swap(TSS) module randomizes token exchanges across samples in each batch to diversify pattern combinations and enhance feature representativeness. Finally, the logits of these two parts generated by a fully connected layer are added using two learnable parameters for the computation of the cross-entropy function, and we calculate the contrastive loss function for the logits of the tokens generated by Key Tokens Selector.

3.2 Key Tokens Selector

We utilize the Vision Transformer (ViT) to extract image features, which segments the image into patches and then uses an Encoder to encode these patches into tokens. During this process, an additional class token (CLS token) τ_{cls} interacts with these tokens to aggregate the features of the entire image for classification. Each layer of the Transformer computes self-attention weight matrices, where $a^l_{i,j}$ represents the attention weight of token i to token j in layer l. Higher attention weights indicate stronger correlations between image patches.

When token selection is based solely on the magnitude of attention weights, it focuses only on the attention allocated to individual tokens themselves, without considering the influence of other tokens, which may introduce background features that interfere with the classification process. To comprehensively consider the association between tokens from both local and global perspectives, our KTS module introduces the Jensen-Shannon (JS) divergence [21] between the tokens' attention weight vectors and the uniform distribution as a metric. The JS divergence is a measure of the similarity between two data distributions P and Q, which ranges from 0 to 1, quantifies the similarity between them. It assesses attention from a broader perspective, considering the overall attention pattern rather than just local intensity. Using this score to evaluate token importance reflects both the model's attention to the token and the overall uniformity of its attention distribution. It is computed by first determining their average distribution $E = \frac{1}{2}(P + Q)$, followed by applying the JS divergence formula:

$$JS(P \mid\mid Q) = \frac{1}{2}\sum_i P(i) log\frac{P(i)}{E(i)} + \frac{1}{2}\sum_i Q(i) log\frac{Q(i)}{E(i)}. \tag{1}$$

A score is calculated for each token in each layer, which incorporates the average of the weight vectors obtained from different attention heads, as well as the JS divergence between the weight vectors and a uniform distribution. These two metrics correspond to the local and global correlation between tokens, respectively. Before calculating this score, we need to get the Hadamard product of the attention weights of token i accumulated from layer 1 up to layer l:

$$\boldsymbol{a}_i^{l\_H} = \prod_{m=1}^{l} \mathbf{a}_i^m = \left[a_{i,0}^{l\_H}, a_{i,1}^{l\_H}, \ldots, a_{i,j}^{l\_H}, \ldots, a_{i,K}^{l\_H}\right], \quad \mathrm{i} \in 0, 1, \ldots, \mathrm{K} \tag{2}$$

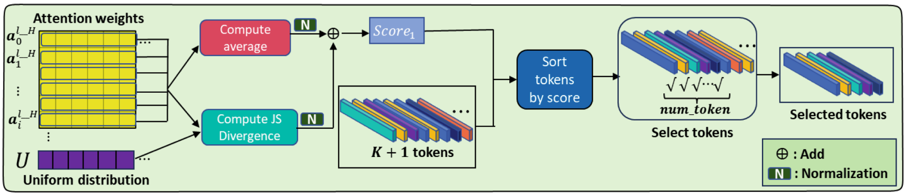


Fig. 2. The procedure of KTS.

where K is the number of tokens, $\mathbf{a}_i^m$ represents the attention weight matrix for token i in layer m, $a_{i,j}^{l\_H}$ denotes the attention weight accumulated from the first layer to the l-th layer for token i with respect to token j. To calculate the score for token i, we first compute the average of the attention weights across all tokens, represented by $M_i = \frac{1}{K+1}\Sigma_{j=0}^{K}(a_{i,j}^{l\_H})$. This yields an aggregated score vector for the l-th layer, denoted as $S_{avg} = [M_0, M_1, \ldots, M_i, \ldots, M_K]$. Next, we initialize a uniformly distributed vector U with the same dimensions as $a_{i,j}^{l\_H}$ and calculate the Jensen-Shannon divergence between them, denoted by $J_i = JS(\boldsymbol{a}_i^{l\_H} \mid\mid U)$. This calculation produces a second set of scores for all tokens in the layer, aggregated as $S_{jsd} = [J_0, J_1, \ldots, J_i, \ldots, J_K]$.

In order to reasonably combine these two parts of the score for each token, we normalize both of them and then sum them:

$$S_{avg\_norm} = \left[\frac{M_0 - \mu_1}{\sigma_1}, \frac{M_1 - \mu_1}{\sigma_1}, \ldots, \frac{M_i - \mu_1}{\sigma_1}, \ldots, \frac{M_K - \mu_1}{\sigma_1}\right] \tag{3}$$

where $\mu_1 = \frac{1}{K+1}\sum_{i=0}^{K} M_i$ and $\sigma_1 = \sqrt{\frac{1}{K+1}\Sigma_{i=0}^{K}(M_i - \mu_1)^2}$ are respectively the mean and the standard deviation of S_{avg}. Similarly, we can get S_{jsd_norm}. Therefore, in the l-th layer, the final score matrix for all tokens can be represented as:

$$S_1 = S_{avg\_norm} + S_{jsd\_norm} \tag{4}$$

Tokens are ranked by their scores within each layer, and the top num_token tokens with the highest scores are selected for aggregation. These are then concatenated with the CLS token $\boldsymbol{\tau}_{cls}$, and processed through a transformer encoder layer. The rich local features contained in these tokens are incorporated into the CLS token, resulting in a new CLS token $\boldsymbol{\tau}_{cls_1}$ for subsequent processing.

3.3 Structure Feature Extractor

To effectively extract structural information from images, we developed a Structure Feature Extractor (SFE) module integrated into the last layer of the Vision Transformer. This strategic placement leverages the attention heads' focus on global and semantic information in higher layers. Similar to the KTS, the SFE employs the Hadamard product to accumulate the attention weights across all layers, generating the accumulated attention weight matrix for token i in the last layer as $\boldsymbol{a}_i^{L-1} = \left[a_{i,0}^{L-1}, a_{i,1}^{L-1}, \ldots, a_{i,j}^{L-1}, \ldots, a_{i,K}^{L-1}\right]$. In token selection, the first part of the score S_{avg_norm} is borrowed from the KTS and is used to intuitively measure the correlation between tokens, excluding the weights of the CLS token, which represents global image information. The second part of the score evaluates the certainty of each token's relationship with others, using the information entropy of its attention weight vector as a metric:

$$H_i = -\sum\nolimits_{j=1}^{K} a_{i,j}^{L-1} log_2 a_{i,j}^{L-1}, \quad i \in 1, 2, \ldots, K \tag{5}$$

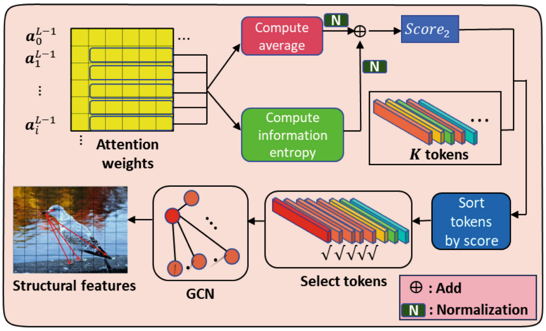


Fig. 3. The procedure of SFE.

We derive the entropy vector $S_H = [H_1, H_2, \ldots, H_i, \ldots, H_K]^T$, where each entropy value quantifies the concentration of a token's attention weights with respect to others. Lower entropy indicates higher attention concentration, suggesting a significant semantic correlation. These tokens are ideal for modeling structural features. To integrate these metrics effectively, S_H is normalized similarly to Eq.3, resulting in S_{H_norm}, which forms the basis of the final score:

$$S_2 = S_{avg\_norm} - S_{H\_norm} = [\mathfrak{s}_0, \mathfrak{s}_1, \ldots, \mathfrak{s}_i, \ldots, \mathfrak{s}_K] \tag{6}$$

where $\mathfrak{s}_i$ denotes the final score of the i-th token. The purpose of this formula is to select tokens that exhibit high relevance and certainty for structural modeling. The token with the highest score is chosen as the anchor. For structural analysis, the score vector S_2 is reshaped into a $\sqrt{K} \times \sqrt{K}$ matrix, referred to as S_{2_new}. A mask of the same size is then applied to simplify this matrix by setting positions with values below the average score $\bar{\mathfrak{s}}$ to zero: Inspired by [20], we employ polar coordinates to capture the spatial relationships between the anchor patch $P_0 = P_{(x_0, y_0)}$ and other selected tokens, enriching the structural analysis of objects.

The anchor patch, positioned at (x_0, y_0) in the $\sqrt{K} \times \sqrt{K}$ grid, serves as a reference. For each selected token i, the corresponding patch $P_i = P_{(x_i,y_i)}$ is expressed in polar coordinates relative to this anchor:

$$\Gamma_{x_i,y_i} = \sqrt{(x_0 - x_i)^2/K + (y_0 - y_i)^2/K} \tag{7}$$

$$\theta_{x_i,y_i} = (\mathbf{atan2}(y_i - y_0, x_i - x_0) + \pi)/2\pi \tag{8}$$

where $0 < \Gamma_{x_i,y_i} \leq 1$ is used to measure the relative distance between the anchor patch P_0 and the patch P_i corresponding to token i, and θ_{x_i,y_i} is the normalized polar angle of P_i relative to the horizontal direction.

A two-layer graph convolutional neural network is used to delineate the structural relationships between selected tokens. Starting with the simplified score matrix, we generate the adjacency matrix $Adj = S_{2\_new} \cdot (S_{2\_new})^T$. The image features X, derived from the last layer of the backbone, are then combined with this adjacency matrix in a graph convolution process:

$$G = \sigma(Adj \times \sigma(Adj \times X \times W^1) \times W^2) \tag{9}$$

where W^1 and W^2 are two learnable parameters, $\sigma(\cdot)$ is the activation function, and × denotes matrix multiplication. The resulting anchor feature G encapsulates the structural information of the objects. We incorporate it into the CLS token to obtain the CLS token $\boldsymbol{\tau}_{cls_2}$ enriched with structural information.

3.4 Fine-Grained Feature Learning

After obtaining the CLS tokens $\boldsymbol{\tau}_{cls_1}$ and $\boldsymbol{\tau}_{cls_2}$, which encode image discriminative features and structural information, we employ a Token Stochastic Swap module. This module acts on the original CLS token matrix C with dimensions $[B, H]$, randomly permuting its columns to enhance token representativeness and diversify pattern combinations. The permutation probability p for each iteration is determined by $p = p_{min} + Beta(1, 1) \times (p_{max} - p_{min})$, scaling a Beta distribution value to the interval $[p_{min}, p_{max}]$. For each column j in matrix C, rows are permuted with probability p, such that $C[i, j]$ is replaced with $C[RP[i], j]$ where RP is a randomized permutation of $[0, B - 1]$:

$$C[i, j] = \begin{cases} C[RP[i], j] & \text{if } rand() < p \\ C[i, j] & \text{otherwise} \end{cases} \tag{10}$$

where $rand()$ generates a random number in the range (0, 1). This process is repeated n times.

For the CLS tokens $\boldsymbol{\tau}_{cls_1}$ and $\boldsymbol{\tau}_{cls_2}$ generated by KTS and SFE, respectively, the swap process introduced above is applied to obtain the permuted CLS tokens Y_1 and Y_2. Then, we compute the logits $\boldsymbol{l}_1$ and $\boldsymbol{l}_2$ for the two parts of features respectively. Two learnable weight parameters w_1 and w_2 are defined to calculate the weighted logits $\boldsymbol{l} = w_1\boldsymbol{l}_1 + w_2\boldsymbol{l}_2$, which is used to compute the cross-entropy loss $\mathcal{L}_{CE}$, thereby enhancing the model's overall understanding of the image through this feature interaction method.

To learn the subtle differences between subcategory images, a contrastive loss is adopted for the fine region features Y_1, which enhances feature discriminability by maximizing intra-class similarity and minimizing inter-class similarity, enabling the model to learn more fine-grained differences in the images:

$$\mathcal{L}_{contrast} = \frac{1}{2B^2} \sum_{i=1}^{B} \sum_{j=1}^{B} \Big(P_{ij} \big(1 - \widehat{y}_i \widehat{y}_j^T\big) + N_{ij} \max\big(m - \widehat{y}_i \widehat{y}_j^T, 0\big)^2 \Big) \tag{11}$$

where B is the batch size, P_{ij} and N_{ij} are indicators for positive and negative sample pairs, respectively, m is the preset margin, and $\widehat{y}_i$ and $\widehat{y}_j$ are respectively the $L2$ normalized feature vectors y_i and y_j in Y_1.

In summary, a combination of contrastive loss and cross-entropy loss is used to train our model end to end, which is as follows:

$$\mathcal{L}_{total} = \mathcal{L}_{CE} + \mathcal{L}_{contrast} \tag{12}$$

4 Experiments

In this section, we evaluate our method across four datasets. Comparative analyses against leading methods and ablation studies on our model's components will showcase their design rationality and effectiveness. Additionally, we will present visualizations to demonstrate our model's interpretability.

4.1 Datasets and Implementation Details

We validated our model on four public datasets: CUB_200_2011 [22], Stanford Dogs [23], IP102 [24] and Cotton80 [25]. CUB_200_2011 is a versatile fine-grained image dataset, including 200 classes with 5994 training images and 5794 test images, while Stanford Dogs comprises 120 classes with 12000 training images and 8580 test images. IP102 is a large-scale fine-grained insect dataset, consisting of 102 classes with 45095 training images, 7508 validation images, and 22619 test images. Cotton80 is a small ultra-fine-grained image dataset, containing images of cotton leaves from 80 categories, with only 6 samples per variety, divided into 3 training images and 3 test images. These datasets provide a range of scenarios from small-scale to large-scale, facilitating a comprehensive evaluation of our model's performance.

Our experiments utilized a RTX 3090 GPU with ViT-B_16 as the backbone. Input sizes were standardized across datasets: 600×600 for CUB_200_2011 and Stanford Dogs, 224×224 for IP102 and 500×500 for Cotton80. During training, images from CUB_200_2011 and Stanford Dogs were randomly cropped to 448×448, images from Cotton80 were randomly cropped to 384×384, no cropping was applied to IP102. Data augmentation included random horizontal flipping and Random Erasing for all datasets. The network was trained end-to-end.

We initialized the network backbone with intermediate weights from the official ViT-B_16 model pre-trained on ImageNet21k. We employed the SGD optimizer with a momentum of 0.9 and used cosine annealing as the scheduler. Learning rates were set at 0.003 for Stanford Dogs and IP102, and 0.02 for CUB_200_2011 and Cotton80. Training specifics included 15000 steps with a batch size of 8 for both CUB_200_2011 and Stanford Dogs, 30000 steps with a batch size of 32 for IP102, 6000 steps with a batch size of 8 for Cotton80. In our setup, *num_token* and the backbone layers L were consistent at 12 and 11, respectively. Token Stochastic Swap parameters included p_{min} and p_{max} set to 0 and 0.4, with 6 swaps and a contrastive loss margin of 0.5. Top-1 accuracy ("Acc") was used as the evaluation metric for all experiments.

4.2 Experimental Results and Quantitative Analysis

Table 1 displays the classification results of our method alongside several state-of-the-art methods on four benchmark datasets. Our method outperformed all others on these datasets, demonstrating its effectiveness and generalizability.

Table 1. Comparison of accuracy between our method and state-of-the-art methods on four datasets of different scales and granularities. * indicates that we reproduce their experiments in our environment according to the settings of the original paper.

Method	Backbone	Time	Source	Acc.			
				CUB_200_2011	Stanford dogs	Cotton80	IP102
ResNet50 [26]	ResNet50	2016	CVPR	84.5	86.1	52.5	54.7
RA-CNN [14]	VGG19	2017	CVPR	85.3	87.3	-	-
PC [13]	DenseNet161	2018	ECCV	86.9	83.8	-	-
NTS-Net [10]	ResNet50	2018	ECCV	87.5	-	51.7	-
API-Net [15]	DenseNet161	2020	AAAI	90.0	90.3	-	56.9
ViT [6]	ViT-B_16	2021	ICLR	90.6	90.2	52.5	73.4
FFVT [16]	ViT-B_16	2021	BMVC	91.6	91.5	57.9	73.8*
RAMS-Trans [19]	ViT-B_16	2021	ACM MM	91.3	92.4	-	-
CAMF [27]	Swin-Transformer	2021	SPL	91.2	92.8	-	-
AF-Trans [18]	ViT-B_16	2022	ICASSP	91.5	91.6	-	-
TransFG [17]	ViT-B_16	2022	AAAI	91.7	92.3	54.6	74.8*
SIM-Trans [20]	ViT-B_16	2022	ACM MM	91.8	90.9*	52.5*	74.2*
AA-Trans [28]	ViT-B_16	2023	PR	91.4	90.8	-	75.0
IELT [29]	ViT-B_16	2023	TMM	91.8	-	-	74.3
HERBS [11]	Swin-Transformer	2023	arXiv	93.1	90.7*	61.7*	76.1*
CSDNet [30]	ViT-B_16	2024	TCSVT	90.9	-	57.9	74.5
ORA-Trans	ViT-B_16	2024	-	**94.0**	**95.4**	**62.5**	**79.6**

In comparative analyses across the CUB_200_2011, Stanford Dogs, IP102 and Cotton80 datasets, our method consistently outperforms both traditional location-based models and advanced ViT-based approaches. Specifically, our enhancements significantly reduce misclassification caused by excessive background features, a common issue in models like NTS-Net and API-Net. Against ViT-focused methods such as TransFG and FFVT, which concentrate on local details but neglect structural object information, our approach shows a classification accuracy improvement of over 2%. Moreover, on the IP102 dataset, which features significant intra- and inter-class variations, our method excels by effectively integrating structural information, yielding about a 4% performance increase compared to models that fail to optimize background filtering and structural analysis. On the Cotton80 dataset, our method also outperforms other listed methods, which is due to the fact that the TSS module we used enriches the combination patterns of features and alleviates the overfitting. Our refinements to the ViT framework address its inherent limitations in fine-grained classification, thereby enhancing its capability to process complex visual information more accurately.

Table 2. Comparison of computational costs for different models.

Model	Backbone	Input_size	Dimension	Parameters
ResNet50 [26]	ResNet50	448×448	2K	22.58M
ViT [6]	ViT-B_16	448×448	0.7K	86.24M
TransFG [17]	ViT-B_16	448×448	0.7K	86.80M
FFVT [16]	ViT-B_16	448×448	0.7K	86.30M
SIM-Trans [20]	ViT-B_16	448×448	0.7K	99.80M
HERBS [11]	Swin-Transformer	384×384	1K	286.59M
ORA-Trans	ViT-B_16	448×448	0.7K	114.90M

Table 2 lists the comparison results regarding feature dimension and number of parameters. We observe that compared to CNN-based approach, ORA-Trans has lower feature dimension. And compared to a range of ViT-based methods, the increase in the number of parameters is not particularly large, and these extra costs are acceptable in terms of the superior performance of our method. In particular, compared to HERBS which employs the more powerful feature-extracting Swin-Transformer as its backbone, our model has significantly fewer parameters, yet outperforms it in classification accuracy on all of the four datasets. The computational cost analysis has demonstrated the efficiency of the proposed ORA-Trans.

4.3 Ablation Studies

In Table 3, we demonstrate the effectiveness of the proposed modules—TSS, KTS, and SFE—on the CUB dataset. Initially, we conducted experiments on the ViT backbone following the implementation details described in the original paper. Subsequently, we integrated each of the modules TSS, KTS, and SFE with the backbone network. The accuracy improvements achieved by associating these modules with the backbone network were 1.0%, 1.3%, and 0.4%, respectively. Moreover, pairing any two modules together within the backbone network resulted in higher accuracy gains compared to using a single module. Specifically, using TSS and KTS together, and TSS with SFE, improved accuracy by 0.9% and 1.2% over using just KTS and SFE, respectively, demonstrating the efficacy of feature-rich combinations with the TSS module. Additionally, combining KTS and SFE yielded higher accuracy improvements of 0.2% and 1.1% compared to using each alone, showcasing their complementary and synergistic effects. Ultimately, integrating all modules resulted in the highest accuracy.

Table 3. Ablation study results of our proposed module on the CUB_200_2011 dataset.

Modules			Acc.
TSS	KTS	SFE	
ViT Baseline*			91.4
✓			92.4
	✓		92.7
		✓	91.8
✓	✓		93.6
	✓	✓	92.9
✓		✓	93.0
✓	✓	✓	94.0

Table 4. Results of ablation study conducted on the composition of the score in KTS and SFE on the CUB_200_2011 dataset.

Module	Composition		Acc.
KTS	S_{avg_norm}	S_{jsd_norm}	
	✓		93.3
	✓	✓	93.6
SFE	S_{avg_norm}	S_{H_norm}	
	✓		92.8
	✓	✓	93.0

We conducted ablation studies on the token scoring mechanisms within our KTS and SFE modules on the CUB_200_2011 dataset, with results detailed in Table 4. These experiments all incorporated the TSS module. The findings reveal that using JS divergence and information entropy for token evaluations, rather than just averaging attention weights, enhances classification accuracy. This confirms the effectiveness of our approach, which integrates both localized and comprehensive metrics for token selection, proving to be more efficient.

4.4 Parameter Experiments

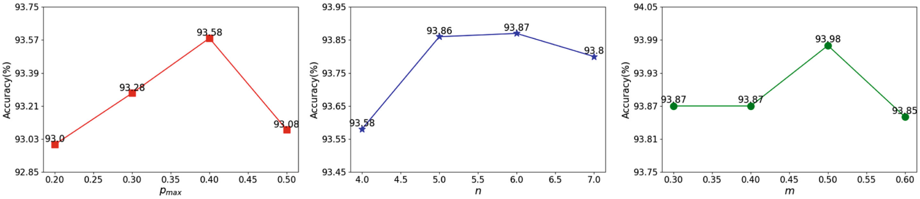
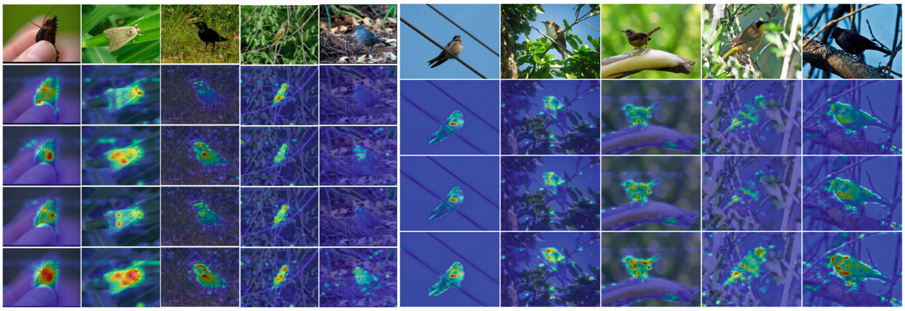


Fig. 4. Parameter experiments about upper limit p_{max} of the probability interval in the TSS module, number of the times n the swap function in Eq. 10 is executed and the preset margin m in Eq. 11 on the CUB_200_2011 dataset.

We conducted parameter experiments on the CUB_200_2011 dataset to optimize the upper limit p_{max} of the probability interval, the number of swap times n, and the margin m in the contrastive loss. Initially, we fixed n at 4 and m at 0.4 to vary p_{max}. Subsequently, with the optimal p_{max} determined, we adjusted n, and finally the margin m. Results shown in Fig. 4 indicate that the best classification performance for the ORA-Trans approach occurs at $p_{max} = 0.4$, $n = 6$, and $m = 0.5$. This configuration raised the classification accuracy from 93.0% to 93.58% when p_{max} was increased from 0.2 to 0.4, emphasizing the TSS module's role in enriching pattern combinations and enhancing feature space representation. Additionally, increasing m also improved accuracy, demonstrating the effectiveness of the contrastive loss in boosting inter-class differentiation and fostering more discriminative feature learning.

4.5 Visualization

This section presents a visual comparison between our model and several representative models, as well as visualization of ablation studies for the KTS and SFE modules in our model. In the visualization maps, the highlighted regions contribute significantly to visual classification. Fig. 5(a) illustrates comparative visualization maps on the IP102 and CUB_200_2011 datasets, demonstrating how our model, compared to TransFG, FFVT, and SIM-Trans, exhibits stronger and more evenly distributed attention across object contours. This underlines our method's adeptness in capturing both detailed local features and overall structural information, highlighting its effectiveness and interpretability. In Fig. 5(b), the ablation study's visualized attention maps reveal progressively focused attention on key features, enhancing object recognition even against complex backgrounds. For instance, in the CUB_200_2011 dataset, attention intensifies from the bird's beak to its feet, indicating an enhanced grasp of the object's structure from left to right in the visualizations. Notably, the last row shows concentrated attention, affirming the synergistic effect of the SFE and KTS modules in guiding the focus towards essential details under varied conditions.



(a) Comparison of attention visualization between our model and other models.

(b) Comparison of attention visualization from ablation studies on KTS and SFE modules

Fig. 5. a) Comparative visualization of attention maps on the IP102 and CUB_200_2011 datasets after identical training steps across various methods: TranFG, FFVT, SIM-Trans, and our model. Images from the IP102 dataset are shown in the first two columns, while images from the CUB_200_2011 dataset appear in the last three columns. Each row sequentially displays original images, followed by attention maps from TransFG, FFVT, SIM-Trans, and our method. (b) Ablation study attention maps for our method on the CUB_200_2011 dataset. The rows present original images, followed by attention maps from a ViT model fine-tuned on the dataset, a ViT enhanced with only KTS, and a ViT integrating both KTS and SFE modules.

5 Conclusion

This study introduces ORA-Trans, a novel ViT-based model designed to enhance fine-grained visual classification. The model integrates three main modules: KTS, SFE, and TSS. KTS focuses on selecting key tokens that capture critical local details. SFE introduces the structural information of objects to optimize the model learning process of tokens selected by KTS, allowing for a deeper understanding of the object's framework and better handling of complex image backgrounds. TSS further enhances feature combinations and representativeness through stochastic token swaps.

We conducted experiments on multiple public datasets of different scales and granularities, demonstrated exceptional performance of our model, proving its effectiveness and generalizability. However, our method has shortcomings. The process of calculating tokens' scores in both modules introduces relatively complex matrix operations, increasing the training time and space costs to some extent. Although our method offers a new ViT modification approach, there's still some room for improvement in the implementation of these modules.

Our findings confirm that ViT's feature extraction capabilities are effective for fine-grained classification. The key lies in how we better filter and mine the information contained in these features. Moving forward, we plan to streamline our model and apply it to other scenarios such as fine-grained retrieval.

References

1. Xiu-Shen Wei, Yi-Zhe Song, Oisin Mac Aodha, Jianxin Wu, Yuxin Peng, Jinhui Tang, Jian Yang, and Serge Belongie. Fine-grained image analysis with deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(12):8927–8948, 2021
2. Thomas Berg and Peter N Belhumeur. Poof: Part-based one-vs.-one features for fine-grained categorization, face verification, and attribute estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 955–962, 2013
3. Tang, H., Yuan, C., Li, Z., Tang, J.: Learning attention-guided pyramidal features for few-shot fine-grained recognition. Pattern Recogn. **130**, 108792 (2022)
4. Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014
5. He, X., Peng, Y., Zhao, J.: Fast fine-grained image classification via weakly supervised discriminative localization. IEEE Trans. Circuits Syst. Video Technol. **29**(5), 1394–1407 (2018)
6. Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929, 2020
7. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017
8. Zeng, R., He, J.: Grouping bilinear pooling for fine-grained image classification. Appl. Sci. **12**(10), 5063 (2022)
9. Xiruo Shi, Liutong Xu, and Pengfei Wang. Fine-grained image classification combined with label description. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 1057–1064. IEEE, 2019
10. Ze Yang, Tiange Luo, Dong Wang, Zhiqiang Hu, Jun Gao, and Liwei Wang. Learning to navigate for fine-grained classification. In *Proceedings of the European conference on computer vision (ECCV)*, pages 420–435, 2018
11. Po-Yung Chou, Yu-Yung Kao, and Cheng-Hung Lin. Fine-grained visual classification with high-temperature refinement and background suppression. arXiv preprint arXiv:2303.06442, 2023
12. Xiaohan Yu, Jun Wang, and Yongsheng Gao. Cle-vit: contrastive learning encoded transformer for ultra-fine-grained visual categorization. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 4531–4539, 2023
13. Abhimanyu Dubey, Otkrist Gupta, Pei Guo, Ramesh Raskar, Ryan Farrell, and Nikhil Naik. Pairwise confusion for fine-grained visual classification. In *Proceedings of the European conference on computer vision (ECCV)*, pages 70–86, 2018
14. Jianlong Fu, Heliang Zheng, and Tao Mei. Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4438–4446, 2017
15. Zhuang, P., Wang, Y., Qiao, Yu.: Learning attentive pairwise interaction for fine-grained classification. In Proceedings of the AAAI conference on artificial intelligence **34**, 13130–13137 (2020)

16. Jun Wang, Xiaohan Yu, and Yongsheng Gao. Feature fusion vision transformer for fine-grained visual categorization. arXiv preprint arXiv:2107.02341, 2021
17. He, J., Chen, J.-N., Liu, S., Kortylewski, A., Yang, C., Bai, Y., Wang, C.: Transfg: A transformer architecture for fine-grained recognition. In Proceedings of the AAAI Conference on Artificial Intelligence **36**, 852–860 (2022)
18. Yuan Zhang, Jian Cao, Ling Zhang, Xiangcheng Liu, Zhiyi Wang, Feng Ling, and Weiqian Chen. A free lunch from vit: Adaptive attention multi-scale fusion transformer for fine-grained visual recognition. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3234–3238. IEEE, 2022
19. Yunqing Hu, Xuan Jin, Yin Zhang, Haiwen Hong, Jingfeng Zhang, Yuan He, and Hui Xue. Rams-trans: Recurrent attention multi-scale transformer for fine-grained image recognition. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 4239–4248, 2021
20. Hongbo Sun, Xiangteng He, and Yuxin Peng. Sim-trans: Structure information modeling transformer for fine-grained visual categorization. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 5853–5861, 2022
21. Menéndez, M.L., Pardo, J.A., Pardo, L., Pardo, M.C.: The jensen-shannon divergence. J. Franklin Inst. **334**(2), 307–318 (1997)
22. Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011
23. Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. CVPR workshop on fine-grained visual categorization (FGVC)*, volume 2. Citeseer, 2011
24. Xiaoping Wu, Chi Zhan, Yu-Kun Lai, Ming-Ming Cheng, and Jufeng Yang. Ip102: A large-scale benchmark dataset for insect pest recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8787–8796, 2019
25. Xiaohan, Yu., Zhao, Y., Gao, Y., Xiong, S., Yuan, X.: Patchy image structure classification using multi-orientation region transform. In Proceedings of the AAAI conference on artificial intelligence **34**, 12741–12748 (2020)
26. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016
27. Miao, Z., Zhao, X., Wang, J., Li, Y., Li, H.: Complemental attention multi-feature fusion network for fine-grained classification. IEEE Signal Process. Lett. **28**, 1983–1987 (2021)
28. Wang, Q., Wang, J.J., Deng, H., Xue, W., Wang, Y., Hao, G.: Aa-trans: Core attention aggregating transformer with information entropy selector for fine-grained visual classification. Pattern Recogn. **140**, 109547 (2023)
29. Qin Xu, Jiahui Wang, Bo Jiang, and Bin Luo. Fine-grained visual classification via internal ensemble learning transformer. *IEEE Transactions on Multimedia*, 2023
30. Ziye Fang, Xin Jiang, Hao Tang, and Zechao Li. Learning contrastive self-distillation for ultra-fine-grained visual categorization targeting limited samples. *IEEE Transactions on Circuits and Systems for Video Technology*, 2024

ChannelDropBack: Forward-Consistent Stochastic Regularization for Deep Networks

Evgeny Hershkovitch Neiterman and Gil Ben-Artzi(✉)

School of Computer Science, Ariel University, Ariel, Israel
{neiterman,gilba}@ariel.ac.il

Abstract. Incorporating stochasticity into the training process of deep convolutional networks is a widely used technique to reduce overfitting and improve regularization. Existing techniques often require modifying the architecture of the network by adding specialized layers, are effective only to specific network topologies or types of layers - linear or convolutional, and result in a trained model that is different from the deployed one. We present ChannelDropBack, a simple stochastic regularization approach that introduces randomness only into the backward information flow, leaving the forward pass intact. ChannelDropBack randomly selects a subset of channels within the network during the backpropagation step and applies weight updates only to them. As a consequence, it allows for seamless integration into the training process of any model and layers without the need to change its architecture, making it applicable to various network topologies, and the exact same network is deployed during training and inference. Experimental evaluations validate the effectiveness of our approach, demonstrating improved accuracy on popular datasets and models, including ImageNet and ViT. Code is available at https://github.com/neiterman21/ChannelDropBack.git.

Keywords: Regularization · Stochastic Training · Deep Learning

1 Introduction

Deep neural networks have achieved remarkable success in fields such as computer vision and natural language processing. Numerous methods have been implemented to enhance their training [9,10] and runtime performance [1,14,17]. However, as these networks become deeper and more complex, overfitting continues to be a challenge. To address this, researchers have developed various regularization techniques. One widely used approach to mitigate overfitting and improve regularization involves incorporating stochasticity.

Existing stochastic regularization approaches, such as Dropout [19] and DropConnect [23], have shown promise in reducing overfitting. Dropout works by randomly deactivating a subset of neurons during training, thereby preventing

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 390–400, 2025.
https://doi.org/10.1007/978-3-031-78383-8_26

complex co-adaptations among neurons. DropConnect, on the other hand, generalizes this idea by randomly removing connections between neurons instead of deactivating the neurons themselves. While effective, these methods require architectural modifications and introduce a discrepancy between the trained and deployed models, as only a subset of the connections or neurons are used during training, while all connections or neurons participate at inference.

Other approaches like Stochastic Depth [7] and DropPath [12] address the training of very deep networks by randomly dropping entire layers or paths during training. Despite their effectiveness, these methods are primarily applicable to specific network topologies such as residual networks. Techniques like DropBlock [5] and Spatial Dropout [21] introduce structured stochasticity by dropping contiguous regions of feature maps or channels, but they still require specialized layers or are limited to particular network structures.

In this paper, we present ChannelDropBack, a stochastic regularization technique that addresses the limitations of existing methods. The primary objective of ChannelDropBack is to bridge the gap between training and inference phases while enhancing the model's generalization capabilities. Our method ensures that the same network structure is used during both phases, thus eliminating any potential discrepancies that might arise due to differences between training and deployment. It introduces randomness exclusively in the backward pass of the neural network, leaving the forward pass intact, by randomly selecting a subset of layers and channels within the network during backpropagation and applies weight updates only to these selected components.

We conduct extensive experiments on popular datasets such as ImageNet, CIFAR-100, and CIFAR-10, using a variety of architectures including ResNet, EfficientNet, and ViT-B. Our results demonstrate that ChannelDropBack consistently improves accuracy and robustness compared to traditional training methods and other stochastic regularization techniques. The experiments highlight the versatility and efficacy of ChannelDropBack across different datasets and network architectures, showcasing its potential as a universal regularization method in deep learning.

To summarize, ChannelDropBack presents a forward-consistent approach to stochastic regularization and provides a universally applicable technique to enhance the performance and generalization of deep convolutional networks.

2 Related Work

Stochastic regularization techniques have become essential for deep learning, mitigating overfitting and enhancing generalization. Dropout [19] randomly deactivates neurons during training, preventing complex co-adaptations and promoting the learning of robust features. DropConnect [23] generalizes dropout by randomly removing connections between neurons. However, both methods suffer from inconsistency between the trained and deployed networks. Stochastic Depth [7] and DropPath [12] address the challenge of training very deep networks by randomly dropping entire layers or paths during training. ScheduledDropPath [25] extends DropPath with a linearly decreasing drop rate. These methods, while effective, still maintain the discrepancy between the trained and deployed networks. Shake-Shake [4] and ShakeDrop [24] regularization techniques apply random perturbations to the inputs of a network during training, encouraging the model to learn more robust features. However, they require modifications to the original architecture. DropBlock [5] and Spatial Dropout [21] introduce stochasticity in a structured manner by dropping contiguous regions of feature maps or channels. Although these methods improve regularization, they still require the addition of specialized layers or are only effective for specific network topologies. In summary, existing stochastic training methods often necessitate architecture modifications, are limited to specific network topologies, and maintain a discrepancy between the trained and deployed networks.

3 ChannelDropBack

ChannelDropBack is a simple method for stochastic training approach for deep convolutional networks. It aims to improve regularization by introducing randomness exclusively into the backward information flow during training, while preserving the integrity of the forward pass, and ensuring that the same network is deployed during both training and inference phases. It allows for seamless integration into the training process of any model without the need to modify its architecture or add specialized layers.

3.1 Layer and Channel Selection Strategy

During each training iteration, ChannelDropBack randomly selects a single layer within the network for which weight updates will be applied to a subset of its channels or rows. The layer selection strategy is based on a predefined probability distribution, ensuring that each layer has a non-zero probability of being selected. In our experiments, we employ a uniform distribution for layer selection, but other distributions can also be explored.

Formally, let L denote the set of all layers in the network, and let l be a layer sampled from L according to the predefined probability distribution. The selected layer is then given by:

$$l_{\text{selected}} \in L$$

Once the layer, l_{selected}, has been selected, we randomly choose a subset of channels (in the case of convolutional layers) or rows (in the case of fully connected layers) to be updated during backpropagation. The number of channels or rows to be selected can be determined by a predefined hyperparameter, p, which represents the proportion of channels or rows to be updated. We always select from the first dimensions of the layer to drop.

Formally, let C denote the set of all channels in a convolutional layer or the set of all rows in a fully connected layer. Let c be a channel or row sampled from

C according to the predefined probability distribution. The subset of selected channels or rows, S, is then given by:

$$S = \{c_1, c_2, \ldots, c_k\} \subseteq C$$

where k is the number of selected channels or rows, and $c_1, c_2, \ldots, c_k$ are the randomly selected channels or rows.

3.2 Backward Pass Modification

Once the subset of channels or rows, S, has been selected, we modify the backpropagation process to apply weight updates only to the selected channels or rows in the selected layer, l_{selected}. During the backward pass, ChannelDropBack computes the gradients for all layers in the network as usual. However, when it comes to updating the weights, ChannelDropBack applies the updates only to the selected channels or rows in the selected layer, l_{selected}, while leaving the remaining channels or rows unchanged.

Formally, let w_l denote the weights of layer l, and let Δw_l denote the weight updates computed during backpropagation. The updated weights, w_l', are given by:

$$w_l' = w_l + \Delta w_l \cdot I(l = l_{\text{selected}}) \cdot I(c \in S)$$

where $I(l = l_{\text{selected}})$ is an indicator function that returns 1 if layer l is the selected layer, l_{selected}, and 0 otherwise. $I(c \in S)$ is an indicator function that returns 1 if channel or row c is in the subset S, and 0 otherwise.

3.3 Training

The training procedure with ChannelDropBack is similar to the standard training process, with the exception of the layer selection, channel or row selection, and backward pass modification steps. The overall training procedure can be summarized as follows:

Algorithm 1 ChannelDropBack Training Procedure

```
Initialize the network weights.
for each training iteration do
    Perform a forward pass to compute the output and loss.
    Randomly select a layer, $l_{\text{selected}}$, according to the layer selection strategy.
    Randomly select a subset of channels or rows, $S$, according to the channel or row selection strategy.
    Perform a backward pass to compute the gradients for all layers.
    Apply weight updates only to the selected channels or rows in the selected layer, $l_{\text{selected}}$, as described in the backward pass modification step.
end for
```

Relation between Learning Rate and Drop Rate The drop rate in ChannelDropBack plays a crucial role in balancing the stochastic regularization and the learning process. To achieve optimal performance, we consider the relationship between the learning rate and the drop rate. We consider layer drop rate as the probability to select a layer for dropping, and channel drop rate as the probability to drop a channel within the selected layer. Here, we discuss three key aspects of this relationship and provide explanations for their underlying mechanisms.

1. **Starting with zero drop rate:** It is best to start with a zero layer drop rate until the first weight decaying occurs. The rationale behind this is to allow the model to learn the initial features and representations without being affected by the stochastic regularization introduced by ChannelDropBack. Starting with a zero layer drop rate enables the model to learn a good initial representation of the data, which can serve as a solid foundation for further refinement and optimization as the drop rate is gradually increased.
2. **Increasing layer drop rate and reducing learning rate:** After the each weight decaying, we increase the drop rate in ChannelDropBack. This recommendation stems from the observation that as training advances and weights are reduced, the model becomes more susceptible to overfitting. These results align with the findings presented by Morerio et al [15].
3. **Layer skipping and drop probability:** Similar to [6], we observed that it is better to reduce stochasticity for the earlier layers for improved convergence of the model. We therefore employ a simple policy of skipping during the layer selection phase the initial four layers in the model, always training them regularly. We also investigated the survival probability approach of [6] for our layer drop rate. However, as this method did not yield notable improvements, we opted to retain our simpler uniform probability approach.

3.4 Inference

One of the key advantages of ChannelDropBack is that it ensures the exact same network is deployed during both training and inference. This is because ChannelDropBack does not introduce any modifications to the forward pass or the network architecture during training. Consequently, the inference procedure with ChannelDropBack is identical to the standard inference process, without any additional computations or modifications.

4 Results

We evaluate ChannelDropBack on ImageNet [2], CIFAR-100, CIFAR-10 [11], and SVHN [16] datasets using MobileNetv2 [18], ShuffleNetV2 [13], EfficientNet-B0 [20], DenseNet121 [8], ResNet-50 [6], and Vision Transformer (ViT-B) [3] architectures. We compare ChannelDropBack against baseline models and other regularization techniques. All experiments use SGD optimizer with momentum 0.9 and weight decay 1e-4, conducted on eight Tesla V100 SXM2 32Gb and eight Quadro RTX6000 GPUs.

4.1 Transfer Learning Results

We evaluate ChannelDropBack in transfer learning scenarios using pre-trained ViT models on ImageNet-21K and ResNet-50 on ImageNet-1K, fine-tuned on CIFAR10, CIFAR-100, and ImageNet-1K. CIFAR10 and CIFAR100 were fine-tuned for 85 epochs, and ImageNet for 10 epochs. We use a learning rate of 0.001 for ViT and 0.01 for ResNet50, with a batch size of 256.

Tables 1 and 2 present the accuracy comparison between models fine-tuned with ChannelDropBack and those without stochastic regularization.

Table 1. Fine-tuned model accuracy on various datasets.

Model	Resolution	CIFAR10		CIFAR100		ImageNet	
		Ours	Baseline	Ours	Baseline	Ours	Baseline
ViT-Base/16	224x224	98.80	**98.95**	**92.20**	91.67	**82.03**	81.32
ViT-Large/16	224x224	**99.08**	99.04	**93.44**	93.25	**83.13**	82.94
ViT-Base/16	384x384	-	-	-	-	**83.90**	83.32
ViT-Large/16	384x384	-	-	-	-	**85.05**	85.05

Table 2. Fine-tuned ResNet50 accuracy on various datasets. The model was pre-trained on ImageNet-1K. Pre-train and fine-tune images were resized to 224x224.

Model	SVHN	CIFAR10	CIFAR100
ResNet50 (Baseline)	95.31	97.38	84.20
ResNet50 (ChannelDropBack)	**95.37**	**97.45**	**84.90**

ChannelDropBack enhances transfer learning capabilities by introducing controlled stochasticity during the fine-tuning process. This stochasticity helps prevent overfitting to the source domain, allowing the model to adapt more effectively to the target domain. The selective channel dropping encourages the network to learn robust features that generalize well across datasets.

4.2 Full Training Results

We evaluate training from scratch using ChannelDropBack with a multi-step linear learning rate schedule. The initial learning rate is 0.01, with milestones at 0.3, 0.6, and 0.8 of the total 200 epochs. The ChannelDropBack drop rate starts at 0.01 and increases linearly to 0.3 after the last learning rate decay. We exclude the first layers from dropping to improve the learning process.

Table 3 presents the top-1 accuracy on CIFAR-100 and CIFAR-10 test sets for different models.

ChannelDropBack consistently outperforms the baseline model across datasets and architectures, demonstrating its universal applicability.

Table 3. Model Accuracy on various datasets trained from scratch.

Model	CIFAR10		CIFAR100	
	ChannelDropBack	Baseline	ChannelDropBack	Baseline
MobileNetv2	**94.41**	94.04	**73.75**	68.08
EfficientNetB0	**93.16**	92.79	**67.23**	67.15
ShuffleNetV2	**93.13**	92.65	**65.45**	65.32
DenseNet121	**95.52**	95.12	**77.30**	77.01
ResNet50	89.12	**89.16**	**77.55**	76.51

4.3 Comparison with Other Regularization Techniques

We compare ChannelDropBack with Dropout, SpatialDropout [22], and Dropblock [5]. Table 4 summarizes the top-1 accuracy for ResNet-50 on CIFAR-100. As can be seen, ChannelDropBack outperforms other regularization techniques.

Table 4. Comparison of regularization techniques on CIFAR-100 with ResNet-50.

Method	Top-1 Accuracy (%)
Baseline	76.51
Dropout (kp=0.7) [19]	76.80
DropPath (kp=0.9) [12]	77.10
SpatialDropout (kp=0.9) [21]	77.41
Dropblock (kp=0.9) [5]	77.42
ChannelDropBack	**77.55**

4.4 Performance on Different Network Depths

We investigate ChannelDropBack performance across varying network depths using ResNet architectures. Table 5 presents results for ResNet-18, ResNet-34, ResNet-50, and ResNet-101 on CIFAR-100. ChannelDropBack consistently improves performance across network depths, demonstrating effectiveness for both shallow and deep networks.

4.5 Layer Drop Rate Policy

We analyze the impact of different layer drop rate policies on ChannelDropBack. Table 6 shows results for ResNet-34 on CIFAR-100. Fixed drop rate applies channel dropping operations from the first epoch without incremental increase. Adaptive is as discussed in Section 3.3. "without skipping first layers" allows also dropping of the first layers.

Table 5. ChannelDropBack performance on ResNet architectures of different depths for CIFAR-100. Values in parentheses indicate improvement over the baseline.

Model	ChannelDropBack	Baseline
ResNet18	**75.08** (+0.35)	74.73
ResNet34	**76.28** (+0.72)	75.56
ResNet50	**77.55** (+1.04)	76.51
ResNet101	**77.67** (+0.86)	76.81

Table 6. Study results for different drop rate policies for ResNet-34 on CIFAR-100.

Configuration	Top-1 Accuracy (%)
Baseline	75.56
ChannelDropBack	75.88
ChannelDropBack (Adaptive)	**76.28**
ChannelDropBack (Adaptive without skipping first layers)	75.73

Both the adaptive drop rate strategy and skipping layer selection mechanism contribute to ChannelDropBack's effectiveness. Using a fixed drop rate throughout training and remove skipping results in a decrease in accuracy.

4.6 Impact of Channel Drop Rate

Figure 1 illustrates the relationship between the drop rate of the channels and top-1 accuracy on CIFAR-100 using ResNet-34.

Accuracy improves with increasing drop rate up to an optimal point, after which performance declines. This suggests an optimal level of randomness that maximizes performance. For CIFAR-100 with ResNet-34, a drop rate of 0.5 yielded the best results.

4.7 Computational Overhead

We assessed the scalability and computational overhead of ChannelDropBack. Table 7 compares the training time per epoch for ResNet-50 on CIFAR-100 with and without ChannelDropBack. ChannelDropBack incurs negligible computational overhead, due to the balance between reduced backpropagation operations and additional selection overhead.

Table 7. Training time per epoch for ResNet-50 on CIFAR-100, comparing baseline and ChannelDropBack configurations.

Configuration	Time (seconds)
Baseline	29.1
ChannelDropBack	29.3

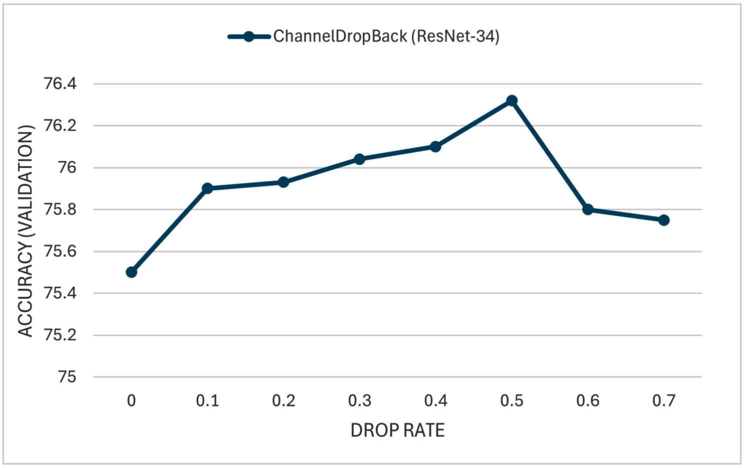


Fig. 1. Impact of the channel drop rate on top-1 accuracy for ResNet-34 on CIFAR-100.

5 Conclusion

ChannelDropBack is a simple yet effective stochastic training approach that improves regularization in deep networks by introducing randomness exclusively into the backward information flow during training. By randomly selecting a subset of channels or rows for weight updates in a selected layer and preserving the integrity of the forward pass, ChannelDropBack offers a universally applicable regularization technique, without requiring modifications to the network architecture. This makes it easy to implement across a wide range of network topologies, from traditional convolutional networks like ResNet and EfficientNet to more recent architectures like Vision Transformers (ViT). By maintaining consistency between the training and inference phases, ChannelDropBack also ensures that the trained model is identical to the deployed model, eliminating the potential discrepancies that can arise with other stochastic regularization methods.

Acknowledgements. We acknowledge the Ariel HPC Center at Ariel University for providing computing resources that have contributed to the research results reported within this paper.

References

1. Aharon, S., Ben-Artzi, G.: Hypernetwork-based adaptive image restoration. In: ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 1–5 (2023). https://doi.org/10.1109/ICASSP49357.2023.10095537
2. 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)
3. 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)
4. Gastaldi, X.: Shake-shake regularization. arXiv preprint arXiv:1705.07485 (2017)
5. Ghiasi, G., Lin, T.Y., Le, Q.V.: Dropblock: A regularization method for convolutional networks. Advances in neural information processing systems **31** (2018)
6. 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)
7. Huang, G., Sun, Yu., Liu, Z., Sedra, D., Weinberger, K.Q.: Deep Networks with Stochastic Depth. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 646–661. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_39
8. Iandola, F., Moskewicz, M., Karayev, S., Girshick, R., Darrell, T., Keutzer, K.: Densenet: Implementing efficient convnet descriptor pyramids. arXiv preprint arXiv:1404.1869 (2014)
9. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning. pp. 448–456. pmlr (2015)
10. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
11. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
12. Larsson, G., Maire, M., Shakhnarovich, G.: Fractalnet: Ultra-deep neural networks without residuals. arXiv preprint arXiv:1605.07648 (2016)
13. Ma, N., Zhang, X., Zheng, H.T., Sun, J.: Shufflenet v2: Practical guidelines for efficient cnn architecture design. In: Proceedings of the European conference on computer vision (ECCV). pp. 116–131 (2018)
14. Menghani, G.: Efficient deep learning: A survey on making deep learning models smaller, faster, and better. ACM Comput. Surv. **55**(12), 1–37 (2023)
15. Morerio, P., Cavazza, J., Volpi, R., Vidal, R., Murino, V.: Curriculum dropout. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3544–3552 (2017)
16. 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)
17. Ofir, A., Ben-Artzi, G.: Smm-conv: Scalar matrix multiplication with zero packing for accelerated convolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3067–3075 (2022)

18. 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)
19. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research **15**(1), 1929–1958 (2014)
20. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. pp. 6105–6114. PMLR (2019)
21. Tompson, J., Goroshin, R., Jain, A., LeCun, Y., Bregler, C.: Efficient object localization using convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 648–656 (2015)
22. Tompson, J., Goroshin, R., Jain, A., LeCun, Y., Bregler, C.: Efficient object localization using convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2015)
23. Wan, L., Zeiler, M., Zhang, S., Le Cun, Y., Fergus, R.: Regularization of neural networks using dropconnect. In: International conference on machine learning. pp. 1058–1066. PMLR (2013)
24. Yamada, Y., Iwamura, M., Kise, K.: Shakedrop regularization (2018)
25. Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V.: Learning transferable architectures for scalable image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 8697–8710 (2018)

DD-Net: Dynamic Network Architecture for Optimized Curve Segmentation and Reduce Computational Redundancy

Yunxiang Cao[1,2], Li Chen[1,2](✉), Yubo Wang[1,2], Zhida Feng[1,2], and Jing Tian[3]

[1] School of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan, China

[2] Hubei Province Key Laboratory of Intelligent Information Processing and Real-time Industrial System, Wuhan University of Science and Technology, Wuhan, China
chenli@wust.edu.cn

[3] Institute of Systems Science, National University of Singapore, Singapore, Singapore

Abstract. Curvilinear Structure segmentation has numerous applications in various fields, including providing a better understanding of defects such as cracks on roads or walls, thereby assessing the structural safety of buildings. Numerous curve segmentation methods based on Convolutional Neural Networks(CNN) have been developed in recent years. However, preserving feature integrity remains a challenge. To address this issue, this paper introduces a double dynamic network, called DD-Net, which consists of a dynamic structure for training and dynamic masking for inference. Firstly, to enhance the nonlinearity of the model, a paradigm to auto-adjust the CNN structure via a trainable module is formed for better fine-grained feature extraction. This dynamic structure (DS) paradigm enables a trade-off to extract or discard the feature data in the model training. Due to the data heterogeneity between the training and inference data, models may contain some useless nodes that are less effective during inference. The dynamic masking (DM) is proposed to omit the useless nodes based on the score difference between the train and inference feature statistics, thereby reducing redundant computations. To further improve the model's performance on thin-curve segmentation and to preserve feature integrity, we introduce a non-curve suppression (NCS) module. This module focuses on background information while considering foreground prediction to address noisy conditions. The experimental results show that our DD-Net achieves promising results on three benchmark datasets and outperforms state-of-the-art curve segmentation models.

Keywords: Curve structure · Feature extraction · Dynamic network

1 Introduction

Curve structures are composed of various line shapes that differ in length and thickness [13,21]. They are widely present in medical image [3,7], remote sensing

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 401–416, 2025.
https://doi.org/10.1007/978-3-031-78383-8_27

image [31,34], and other image domains. Curve structure segmentation refers to the process of binary segmentation of curve structures in images, such as segmenting blood vessels in medical images [25] to assist in the diagnosis and treatment of specific diseases, detecting cracks in road images [17], and extracting road networks from remote sensing images [33].

Most of the current popular deep networks have the same static inference paradigm, that is, once trained, the structure and parameters of the network remain unchanged during the testing phase. Such static and fixed model structures limit the model's representational power, inference efficiency, and interpretability to some extent. In contrast, a dynamic network can adaptively adjust its structure based on the input sample during the inference stage, offering several advantages that a static network cannot achieve [2].

While previous methods have attempted to address the issue of preserving feature details in thin-curve segmentation, they still exhibit limitations in effectively extracting feature integrity. These methods often struggle to balance between extracting the desired curve structure and maintaining feature integrity, leading to suboptimal results. To overcome these challenges, we introduce DD-Net, a novel approach designed to dynamically adjust computational resources while maximizing feature integrity in thin-curve segmentation tasks. We summarize the contributions of this paper as follows:

- We propose a dynamic structure based on trainable dynamic scaling parameters to scale global feature, selectively retaining interesting parts of the model and proportionally discarding irrelevant information.
- We propose a dynamic inference strategy designed to adaptively adjust computations based on varying spatial locations within image data. This method can adaptively reduce the amount of redundant calculations only through inference.
- We introduce a non-curve suppression mechanism to enhance the preservation of the global feature integrity of the data by supplementing background feature information.

2 Related Works

2.1 Dynamic Structure

Traditional static neural network models often suffer from a lack of generalization. Dynamic neural networks have emerged as a solution, allowing models to adjust training weights or inference structures dynamically. Yang et al. [30] proposed a method where convolution kernels are represented as linear combinations of multiple nuclei, dynamically adjusting weight coefficients based on input. Li et al. [10] introduced dynamic routing, enabling adaptive adjustment of image resolution based on picture content, leading to improved efficiency and performance. In the work of crack detection, Chen et al. [1] proposed a novel clustering-inspired representation learning framework to locate edge non-crack regions through preprocessing. Liu et al. [11] proposed a new attention module to effectively extract context information across feature channels and effectively embed position information to capture large receptive field context information. Zhao et al. [32] performed small object segmentation based on dilated convolution and new loss function.

In this context, our proposed method utilizes trainable dynamic adaptive parameters during the training phase to dynamically control the acquisition and rejection of feature data by the model. This simple yet effective dynamic feature extraction approach achieves the performance of complex feature extraction modules with minimal computational cost during training.

2.2 Global Feature Extraction

Convolutional Neural Networks (CNNs) rely on local convolution operations, with each kernel focusing on a small receptive field of the input. However, this localized nature can lead to poor invariance to input transformations and sensitivity to positional information [14].

To address these issues, Singh et al. [24] introduced dense connections in CNNs to enhance information flow and parameter utilization. Yan et al. [29] tackled spatial invariance in CNNs by proposing an architecture employing low-rank and sparse decomposition methods. In our work, we propose a background detection method to complement foreground detection and achieve the extraction of global feature, particularly for the dual-category detection task of curve segmentation.

2.3 Dynamic Inference

Adaptive inference, a central concept in dynamic neural networks, has been studied extensively. Classical approaches involve building adaptive integrations of multiple models through cascade [18] or parallel [6] structures and selectively activating them based on inputs. Xue et al. [28] proposed adaptive fusion of multi-modal data to generate data dependencies during the inference process to achieve the dynamics of inference features. Tuli et al. [26] proposed to tile the matrix in transformer operations with different data streams to improve data reuse and thus achieve greater energy efficiency.

Building on this concept, we propose a dynamic masking strategy that adjusts the number of active weight nodes based on their contribution to the final prediction. This approach eliminates unnecessary calculations and reduces computational overhead without compromising model performance.

3 Proposed Method

3.1 Dynamic Structure Extraction Module

Dynamic segmentation models for curve segmentation attempt to locally focus on curve information by modifying the deformation of convolutional kernels[19].

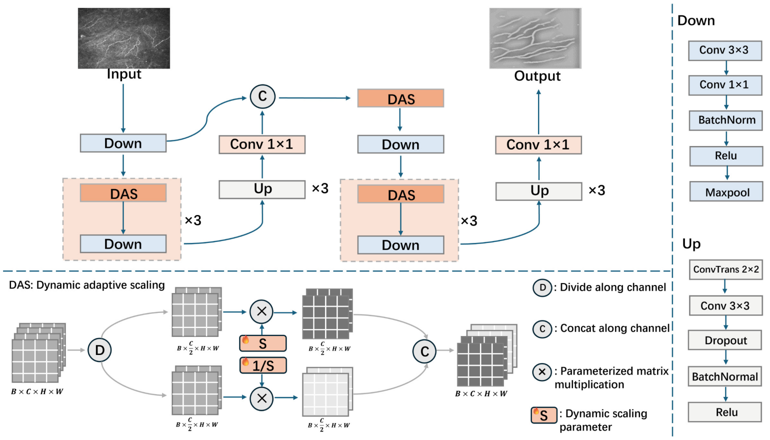


Fig. 1. Global scaling of features is achieved by cascading multiple layers of learnable dynamic scaling parameters.

Inspired by the effectiveness of IterNet [9], our model employs a cascade approach to amplify the impact of our proposed dynamic modules. Within this cascaded architecture, the model comprises multiple layers or stages, each progressively refining and integrating information from preceding stages. This iterative refinement process empowers the model to capture a comprehensive and intricate representation of input features. Our encoder consists of multiple layers of dilated convolution, pooling, and variable dynamic adaptive parameters.

The schematic diagram of the dynamic structure extraction module is shown in Fig 1. In our dynamic structure extraction module, we partition the data into two segments. The adjustment of the dynamic adaptive parameters' size is employed to control the retention and discarding of data in these segments. This approach resembles the feature extraction function of convolution but offers the advantage of effectively preserving the global feature information of the data. This preservation is achieved by iteratively adjusting the overall data information proportionally during each balancing step of the process.

As depicted in Fig 1, the dynamic adaptive scaling process for each time can be described as follows:

$$x_{i+1} = (\frac{\mathbf{S}_i}{\mathbf{S}_{i+1}} \cdot \hat{x_i}) \cup (\frac{\mathbf{S}_{i+1}}{\mathbf{S}_i} \cdot \hat{x_j}) \tag{1}$$

where S is a learnable dynamic scaling parameter, and $(.) \cup (.)$ is concat along the channel dimension. By introducing this dynamic structure, we can dynamically adjust the level of feature retention or discard in our model. This provides us with the flexibility to control the trade-off between model complexity and computational efficiency.

3.2 Non-Curve Suppression

Curve segmentation is a binary classification task, where most methods tend to focus solely on the extraction capability and attention given to the foreground [5,8]. By introducing background cues to improve the model's ability to distinguish segmented objects from other parts, the model can perceive global feature information.

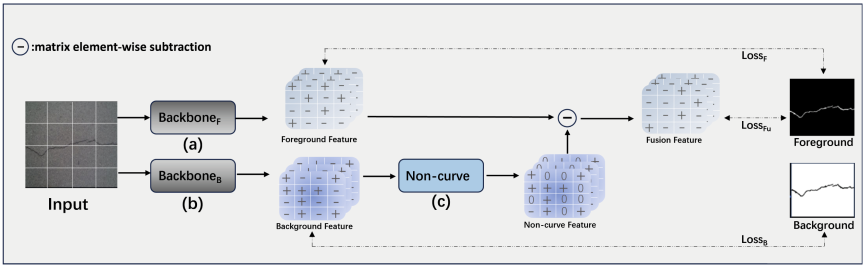


Fig. 2. Schematic diagram of the process of Non-curve suppression, in which (a) and (b) represent the modules responsible for foreground and background prediction respectively, and (c) represents the suppression operation of the curve.

To leverage our proposed Dynamic Structure Extraction Module for effective global feature extraction, we advocate conducting background and foreground predictions for each cascade layer of the model. To ensure accurate background prediction without interference from foreground noise, we suppress the foreground in the background prediction results at each stage of the model, as illustrated in Fig. 2. Among them, the foreground is the prediction result of the curve, the background is the prediction result of the part outside the curve, and the non-curve background is the result after masking the prediction of the curve in the background prediction. For the output X_{fu} of each model cascade layer, it can be expressed as:

$$X_{fu} = X_f - \begin{cases} 0 & X_b < 0 \\ X_b & \text{other} \end{cases} \tag{2}$$

where X_f is the prediction result of the foreground, X_b is the prediction result of the background. As depicted in Fig. 2, during training, we compute a hybrid loss and subsequently utilize it for backpropagation. $Loss_F$ represents the loss associated with the foreground curve, while $Loss_B$ represents the loss associated with the background. Introducing a background loss enables the model to predict both foreground and background features effectively. $Loss_{FU}$ denotes the loss of the fusion feature, which integrates the feature data of both the foreground curve and non-curve elements. For each module during the loss computation, we employ binary cross-entropy(BCE) loss with equal weights. The BCE loss is defined as:

$$Loss = -\frac{1}{n}\sum_{i=1}^{n}(y_i log(\sigma(\hat{y})) + (1 - y_i)log(1 - \sigma(\hat{y}_i))) \tag{3}$$

3.3 Dynamic Masking

The accumulation of a large number of stacked layers in a model leads to increased computational demands. We aim for the model to rely more on the selection of Dynamic Adaptive Parameters, thereby reducing unnecessary computations.

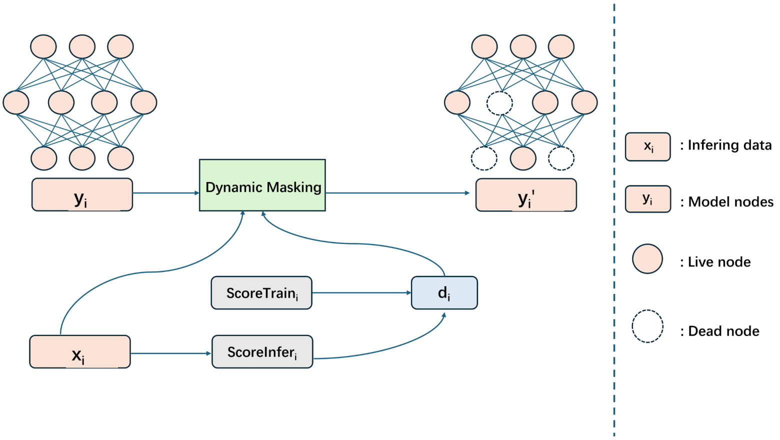


Fig. 3. Masking of useless parameter nodes based on data scores: Comparison between *ScoreTrain* acquired during model layer training and *ScoreInfer* obtained in the current inference process.

Due to the distributional differences between the training and testing datasets, models may contain some useless parameter nodes that are only effective for the training data during inference. The overall process is shown in Fig 3. To reduce the number of useless nodes in the model that are sensitive to large distributional differences, we propose a dynamic masking strategy. During the training process, we calculate scores for the incoming feature data x before passing it to the encoder module. The calculation formula is as follows:

$$ScoresTrain_i = \frac{1}{n}\sum_{j=1}^{n}\mathbf{x}_{ij} - \sigma(\mathbf{x}_{ij}) \tag{4}$$

where $\sigma(.)$ represents the standard deviation calculation. During training, the score for the i-th layer is evaluated based on the score of the input data for that module. x_{ij} is the j-dimensional feature data of the i-th layer. This score is stored for dynamic masking reference during inference. During inference, the score of $ScoresInfer_i$ for the input features is calculated in the same way. If the difference exceeds a threshold, it indicates a significant difference between the inference data and the training data for the current module. Some nodes of the current module may not be capable of reasoning on these highly different data, resulting in redundant calculation processes. In such cases, a dynamic masking (DM) strategy is applied to discard nodes.

The DM strategy is to calculate the masking probability of each node based on the weight of the node in each model layer and the characteristics of the data in the current model. The masking probability of each node is as follows:

$$p_{ik} = \frac{\mathbf{x}_i \mathbf{y}_{ik}}{d_i^2} \tag{5}$$

$$d_i = \left| \frac{1}{n} \sum_{j=1}^{n} \mathbf{x}_{ij} - \frac{1}{n} \sum_{j=1}^{n} \hat{\mathbf{x}_{ij}} + \sigma(\hat{\mathbf{x}_{ij}}) - \sigma(\mathbf{x}_{ij}) \right| \tag{6}$$

where $\hat{x}$ represents the feature data in inference and y represents the model nodes. In the context of inference, $\hat{x}$ denotes the feature data, while y represents the model nodes. During dynamic masking, the appropriateness of the current data and the model's current layer nodes is assessed using the score derived from the feature data of the current layer's inference process and the score from the feature data during the model's training process. In this context, the indices i and k represent the number of layers in the model and the number of nodes within each layer. To obtain a single probability value corresponding to each node, we choose the mean of each probability matrix as a measure. After obtaining the value $\overline{p_{ik}}$ for each node, a softmax operation is performed on the entire set of values to represent the probability of each node being selected. The probability matrix is used to select and mask the corresponding nodes with a given number of times.

4 Experimental Results

4.1 Dataset

To evaluate the effectiveness of our method on curve segmentation, we conduct qualitative and quantitative experiments on three different datasets, including CRACKTREE200 [35], CORN-1 [16], and CRACKFOREST [23].

The CRACKTREE200 dataset is a commonly used benchmark dataset for crack detection and segmentation. It contains high-resolution images of pavement surfaces with crack annotations. CRACKTREE200 dataset consists of 206 images. In this experiment, the first 150 images are used for training, while the remaining images are used for testing.

CORN-1 contains a total of 1698 corneal confocal microscopy images of corneal subbasal epithelium using a Heidelberg Retina Tomograph equipped with a Rostock Cornea Module (HRT-III) microscope. In this experiment, the first 1176 images are used for training, followed by 188 images for validation, and the remaining 152 images are used for testing.

The CrackForest dataset is another dataset for crack detection and segmentation. It consists of images of road surfaces with crack annotations, taken under various environmental conditions. We select the first 100 images from the dataset, with the first 80 images used for training and the remaining 20 images used for testing.

4.2 Implementation Details

Our proposed model is developed using PyTorch and trained on an NVIDIA RTX3080Ti. To augment the data, we utilize horizontal flipping and random cropping techniques. All the networks are optimized by Adam with an initial learning rate of 0.001. During training, all training samples are cropped to a size of 256 × 256, and our network is trained with a batch size of 4.

4.3 Evaluation Metrics

In the experiment, we utilize six evaluation metrics to assess the performance of our method, including F1 score, Precision, Recall [4,15,22], Mcc, mIoU and Quality [27] for pixel-based evaluation, which are widely used in existing semantic segmentation methods. The Matthews Correlation Coefficient (MCC) is defined as:

$$MCC = \frac{tp \times tn - tp \times fn}{\sqrt{(tp + fp) \times (tp + fn) \times (tn + fp) \times (tn + fn)}} \tag{7}$$

Consider one-pixel width segmentation, let the confusion matrix between the prediction skeleton and the ground truth skeleton of the thin curve are: TP_{sk}, FP_{sk}, TN_{sk}, FN_{sk}. The *Quality* is defined as:

$$Quality = \frac{TP_{sk}}{FP_{sk} + TP_{sk} + FN_{sk}} \tag{8}$$

4.4 Performance Comparison

In our study, we compare DD-Net with several commonly used baseline methods, as well as several state-of-the-art methods including U-Net [20], CS-Net [16], IterNet [9], and FR-UNet [12], on three datasets: CORN-1, CRACKTREE200, and CRACKFOREST. We evaluate a total of six metrics. The results comparing DD-Net with other methods on the three datasets are presented in Table 1. On the CORN-1 dataset, DD-Net's F1 is 2.09 higher than the second-best score, while the mIoU improves by 2.46. On the CRACKFOREST dataset, F1 and mIoU increased by 3.36 and 3.68. On the CRACKTREE dataset, F1 increased by 2.53 and mIoU increased by 3.12.

From Table 1, it is evident that DD-Net demonstrates superior performance across overall metrics compared to the recently proposed state-of-the-art curve

Table 1. Performance comparison of various models. The best performance is highlighted in bold format and the next best results are underlined.

Dataset	Method	F1↑	Pre.↑	Rec.↑	mIoU↑	Mcc↑	Qua.↑
CORN-1	U-Net [20]	65.73	74.85	59.20	49.28	65.72	49.11
	IterNet [9]	66.67	74.92	60.60	50.32	66.56	50.17
	CS-Net [16]	65.93	**75.80**	58.54	49.45	65.94	49.22
	FR-UNet [12]	66.83	75.08	60.72	50.46	66.71	50.34
	DD-Net	**68.93**	70.46	**68.12**	**52.93**	**68.50**	**52.51**
CRACKF.	U-Net [20]	67.29	69.24	70.24	52.13	67.67	27.66
	IterNet [9]	69.04	72.93	69.48	54.03	69.29	28.94
	CS-Net [16]	67.86	72.51	68.37	53.67	68.11	29.39
	FR-UNet [12]	69.23	66.34	**76.19**	53.81	69.48	27.33
	DD-Net	**72.63**	**74.72**	72.63	**57.76**	**72.39**	**31.44**
CRACKT.	U-Net [20]	65.68	71.69	61.39	49.32	65.97	48.86
	IterNet [9]	71.16	**80.22**	64.61	55.86	71.65	55.65
	CS-Net [16]	66.60	80.11	57.44	50.56	67.53	50.47
	FR-UNet [12]	72.45	78.86	67.53	57.36	72.70	57.03
	DD-Net	**74.99**	79.14	**71.61**	**60.49**	**75.08**	**60.48**

segmentation method, FR-UNet. In Fig. 4, we present visualizations of the images to provide a more intuitive representation. The figure illustrates the segmentation results of DD-Net and other comparable methods on three datasets. Specifically, the figure displays six images, including the original image, images segmented by U-Net, IterNet, FR-UNet, DD-Net, and the ground truth image (from left to right). As depicted in Fig. 4, our method successfully detects small curve structures and connects breakpoints that are overlooked by other models. Moreover, our model effectively removes noise and restores the original structure of the curve.

4.5 Ablation Study

In this experiment, we aim to verify the contribution of each component of the proposed method. For validation of dynamic masking (DM) reduction calculations, the calculation ratio (CR) can be defined based on the number of model nodes:

$$CR = \frac{\sum_{i=1}^{n} |x_i|}{N \sum_{j=1}^{m} |y_j|} \tag{9}$$

where x_i represents the number of nodes masked, N represents the total number of nodes in the model layer when masking is performed, and y_j represents the number of times masking is performed. The metrics after applying dynamic masking are shown in Table 2. We select two representative metrics and compare

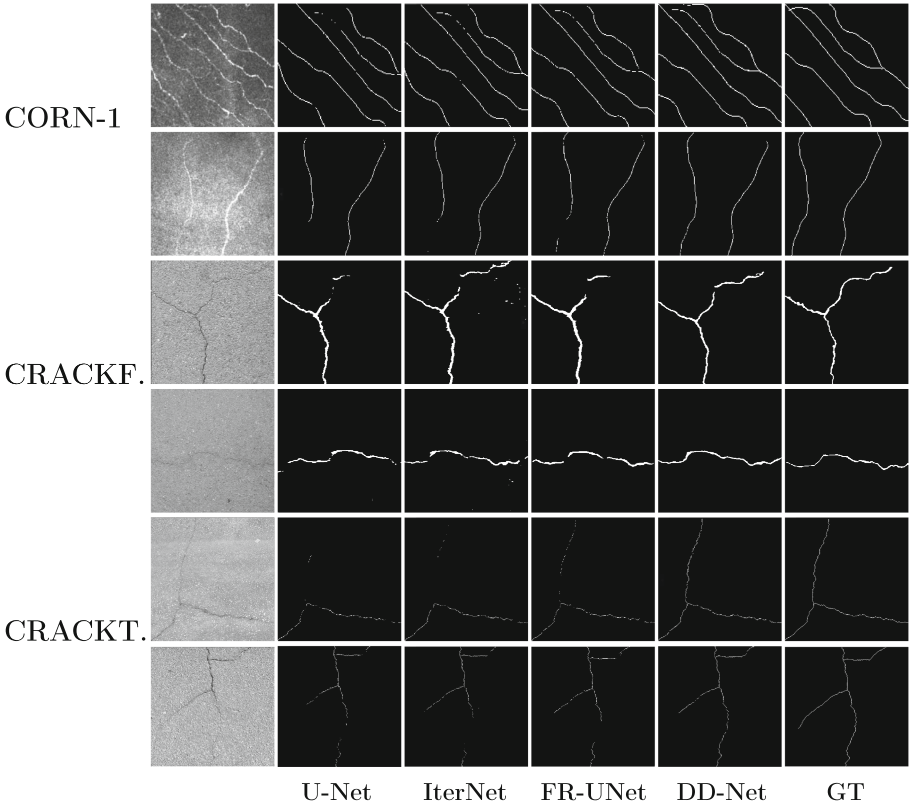


Fig. 4. Visualization of the Performance of DD-Net and Other Methods on Different Datasets.

Table 2. Comparison of ablation experiments with and without Dynamic Masking on DD-Net.

Dataset	Method	F1↑	mIoU↑	CR↓
CORN-1	DD-Net w/o DM	68.92	52.92	100%
	DD-Net	**68.93**	**52.93**	**87.35%**
CRACKF.	DD-Net w/o DM	72.59	57.71	100%
	DD-Net	**72.63**	**57.76**	**87.35%**
CRACKT.	DD-Net w/o DM	74.98	60.48	100%
	DD-Net	**74.99**	**60.49**	**87.4%**

them across three datasets. It can be observed that after each dynamic masking iteration, DD-Net reduces the average computation cost of each masked layer by approximately 12.6%. Additionally, this reduction in computation does not compromise or may even improve the model's performance. Through dynamic masking, we can dynamically adapt the model nodes that are most suitable for the current data only through the inference part, reducing useless calculation processes. Table 3 presents the effect of the Dynamic Structure Extraction Module(DS) on U-Net performance across three datasets. From Table 3, it can be observed that the introduction of DS improves the performance of U-Net on various metrics across different datasets. Among them, F1 improved by an average of 1.59, mIoU by an average of 1.82, and Mcc by an average of 1.73.

Table 3. Ablation experiment on U-Net using Dynamic Structure Extraction Module.

Dataset	Method	F1↑	mIoU↑	Mcc↑
CORN-1	U-Net	65.73	49.28	65.72
	U-Net w/ DS	**67.14**	**50.81**	**66.93**
CRACKF.	U-Net	67.29	52.13	67.67
	U-Net w/ DS	**68.94**	**53.86**	**69.23**
CRACKT.	U-Net	65.68	49.32	65.97
	U-Net w/ DS	**67.38**	**51.51**	**68.39**

Table 4. Ablation experiment on IterNet using Dynamic Structure Extraction Module.

Dataset	Method	F1↑	mIoU↑	Mcc↑
CORN-1	IterNet	66.67	50.32	66.56
	IterNet w/ DS	**67.35**	**51.09**	**67.03**
CRACKF.	IterNet	69.04	54.03	69.29
	IterNet w/ DS	**69.39**	**54.30**	**69.67**
CRACKT.	IterNet	71.16	55.86	71.65
	IterNet w/ DS	**71.42**	**56.34**	**72.45**

We are also interested in knowing if this method has a better effect on the cascaded network. Table 4 presents the impact of DS on the IterNet model. It can be seen that $F1$ improved by an average of 0.43, mIoU by an average of 0.51, and Mcc by an average of 0.55. Based on the combined results from Table 3 and Table 4, it can be concluded that the DS leads to improved performance. Both U-Net and IterNet demonstrate enhanced performance after incorporating DS across different datasets. The versatility of the DS method is evident as it improves the overall model performance on diverse datasets. Notably, the performance enhancement is observed even when DS is applied to the cascaded model, indicating its adaptability across different network architectures.

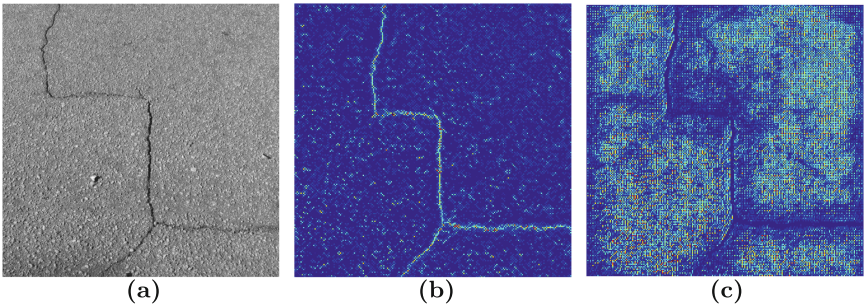


Fig. 5. (a) represents the original image, while (b) and (c) depict the heat maps showcasing the model's attention to the foreground and background, respectively. Brighter colors indicate areas where the model pays more attention.

The Non-Curve Suppression (NCS) technique is a crucial method that improves model performance by incorporating forward and backward predictions. We conduct experiments on three distinct datasets, each representing unique challenges and characteristics, to investigate the effectiveness of this method in enhancing model performance. The results, as shown in Table 5, provide valuable insights into the impact of NCS on U-Net's performance. Taking the CrackTree dataset as an example, the model's feature focus area is shown in Fig 5. It can be seen that the introduction of NCS allows the model to pay attention to the foreground and background, thereby realizing the perception of global features. Indeed, Table 5 demonstrates the varying improvements that the introduction of NCS brings to U-Net across different datasets. The performance enhancement achieved by NCS sets the ability to generalize on different data. Certainly, we are interested in understanding the contribution of NCS to our DD-Net model. By comparing the performance of DD-Net with and without NCS, we can quantify the specific improvement brought about by this technique. The results are shown in Table 6. By evaluating the performance metrics on each dataset, it can be seen that F1 improved by an average of 0.98, mIoU by an average of 1.11, and Mcc by an average of 0.9. We can observe that NCS consistently leads to performance improvements. It is worth mentioning that the improvement may differ from dataset to dataset. This implies the importance of dataset-specific analysis and experimentation to accurately understand the impact of NCS on DD-Net performance. The effect of incorporating background fusion into DD-Net is illustrated in Fig 6. Among them, the red color represents the false positive foreground pixels in the segmentation result, while the green color represents the missed foreground pixels. As expected, the incorporation of background information for fusion effectively suppresses erroneous foreground information.

Table 5. Ablation experiment on U-Net using Non-Curve Suppression.

Dataset	Method	F1↑	mIoU↑	Mcc↑
CORN-1	U-Net	65.73	49.28	65.72
	U-Net w/ NCS	**66.94**	**50.57**	**66.82**
CRACKF.	U-Net	67.29	52.13	67.67
	U-Net w/ NCS	**68.67**	**53.78**	**69.22**
CRACKT.	U-Net	65.68	49.32	65.97
	U-Net w/ NCS	**71.45**	**56.26**	**72.22**

Table 6. Ablation experiment on DD-Net without using Non-Curve Suppression.

Dataset	Method	F1↑	mIoU↑	Mcc↑
CORN-1	DD-Net w/o NCS	67.76	51.60	67.34
	DD-Net	**68.93**	**52.93**	**68.50**
CRACKF.	DD-Net w/o NCS	71.83	56.93	71.83
	DD-Net	**72.63**	**57.76**	**72.39**
CRACKT.	DD-Net w/o NCS	73.96	59.25	74.05
	DD-Net	**74.99**	**60.49**	**75.08**

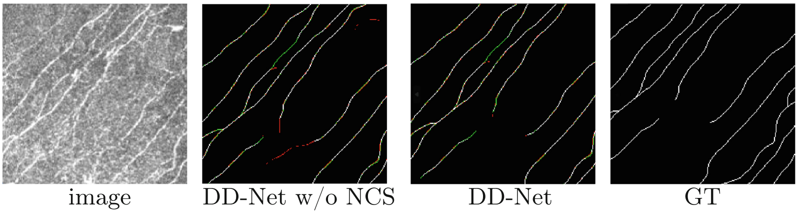


Fig. 6. Non-curve suppression serves as a valuable aid in suppressing erroneous foreground information.

5 Conclusion

In this paper, we propose a novel curve structure segmentation framework that addresses the limitation of capturing insufficient global feature information in existing curve segmentation algorithms. Extensive evaluations on various benchmark datasets have been conducted to demonstrate the superior performance of our proposed framework, DD-Net, compared to state-of-the-art segmentation methods. DD-Net achieves remarkable results in terms of pixel-level metrics for curve linear structure image segmentation. The robust performance of DD-Net highlights its effectiveness in capturing intricate curve structures and accurately segmenting images. Moreover, the proposed framework provides meaningful insights into the importance of incorporating global feature in curve segmentation tasks. By leveraging the comprehensive understanding of the overall features, DD-Net showcases its capability to handle complex curve structures with improved accuracy and precision.

Acknowledgment. This work was supported by National Natural Science Foundation of China (62271359).

References

1. Chen, Z., Lai, Z., Chen, J., Li, J.: Mind marginal non-crack regions: Clustering-inspired representation learning for crack segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12698–12708 (2024)
2. Han, Y., Huang, G., Song, S., Yang, L., Wang, H., Wang, Y.: Dynamic neural networks: A survey. IEEE Trans. Pattern Anal. Mach. Intell. **44**(11), 7436–7456 (2021)
3. Harouni, M., Karimi, M., Rafieipour, S.: Precise segmentation techniques in various medical images. Artificial Intelligence and Internet of Things pp. 117–166 (2021)
4. Hu, X., Li, F., Samaras, D., Chen, C.: Topology-preserving deep image segmentation. Advances in neural information processing systems **32** (2019)
5. Hu, Y., Qiu, Z., Zeng, D., Jiang, L., Lin, C., Liu, J.: Supervessel: Segmenting high-resolution vessel from low-resolution retinal image. In: Chinese Conference on Pattern Recognition and Computer Vision (PRCV). pp. 178–190. Springer (2022)
6. Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E.: Adaptive mixtures of local experts. Neural Comput. **3**(1), 79–87 (1991)
7. Jayadevappa, D., Srinivas Kumar, S., Murty, D.: Medical image segmentation algorithms using deformable models: a review. IETE Tech. Rev. **28**(3), 248–255 (2011)
8. Karaali, A., Dahyot, R., Sexton, D.J.: Dr-vnet: retinal vessel segmentation via dense residual unet. In: International Conference on Pattern Recognition and Artificial Intelligence. pp. 198–210. Springer (2022)
9. Li, L., Verma, M., Nakashima, Y., Nagahara, H., Kawasaki, R.: Iternet: Retinal image segmentation utilizing structural redundancy in vessel networks. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. pp. 3656–3665 (2020)
10. Li, Y., Song, L., Chen, Y., Li, Z., Zhang, X., Wang, X., Sun, J.: Learning dynamic routing for semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8553–8562 (2020)
11. Liu, H., Miao, X., Mertz, C., Xu, C., Kong, H.: Crackformer: Transformer network for fine-grained crack detection. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 3783–3792 (2021)
12. Liu, W., Yang, H., Tian, T., Cao, Z., Pan, X., Xu, W., Jin, Y., Gao, F.: Full-resolution network and dual-threshold iteration for retinal vessel and coronary angiograph segmentation. IEEE J. Biomed. Health Inform. **26**(9), 4623–4634 (2022)
13. Lorigo, L.M., Faugeras, O.D., Grimson, W.E.L., Keriven, R., Kikinis, R., Nabavi, A., Westin, C.F.: Curves: Curve evolution for vessel segmentation. Med. Image Anal. **5**, 195–206 (2001)
14. Mallat, S.: Understanding deep convolutional networks. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences **374**(2065), 20150203 (2016)

15. Mosinska, A., Marquez-Neila, P., Koziński, M., Fua, P.: Beyond the pixel-wise loss for topology-aware delineation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3136–3145 (2018)
16. Mou, L., Zhao, Y., Chen, L., Cheng, J., Gu, Z., Hao, H., Qi, H., Zheng, Y., Frangi, A., Liu, J.: CS-Net: Channel and Spatial Attention Network for Curvilinear Structure Segmentation. In: Shen, D., Liu, T., Peters, T.M., Staib, L.H., Essert, C., Zhou, S., Yap, P.-T., Khan, A. (eds.) MICCAI 2019. LNCS, vol. 11764, pp. 721–730. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32239-7_80
17. Munawar, H.S., Hammad, A.W., Haddad, A., Soares, C.A.P., Waller, S.T.: Image-based crack detection methods: A review. Infrastructures **6**(8), 115 (2021)
18. Paul, V.: Robust real-time face detection. IJCV **57**(2), 137–154 (2004)
19. Qi, Y., He, Y., Qi, X., Zhang, Y., Yang, G.: Dynamic snake convolution based on topological geometric constraints for tubular structure segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6070–6079 (2023)
20. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
21. Rosin, P.L., West, G.A.: Curve segmentation and representation by superellipses. IEE Proceedings-Vision, Image and Signal Processing **142**(5), 280–288 (1995)
22. Seyedhosseini, M., Sajjadi, M., Tasdizen, T.: Image segmentation with cascaded hierarchical models and logistic disjunctive normal networks. In: Proceedings of the IEEE international conference on computer vision. pp. 2168–2175 (2013)
23. Shi, Y., Cui, L., Qi, Z., Meng, F., Chen, Z.: Automatic road crack detection using random structured forests. IEEE Trans. Intell. Transp. Syst. **17**(12), 3434–3445 (2016)
24. Singh, D., Kumar, V., Kaur, M.: Densely connected convolutional networks-based covid-19 screening model. Appl. Intell. **51**, 3044–3051 (2021)
25. Soomro, T.A., Afifi, A.J., Zheng, L., Soomro, S., Gao, J., Hellwich, O., Paul, M.: Deep learning models for retinal blood vessels segmentation: a review. IEEE Access **7**, 71696–71717 (2019)
26. Tuli, S., Jha, N.K.: Acceltran: A sparsity-aware accelerator for dynamic inference with transformers. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (2023)
27. Wiedemann, C., Heipke, C., Mayer, H., Jamet, O.: Empirical evaluation of automatically extracted road axes. Empirical evaluation techniques in computer vision **12**, 172–187 (1998)
28. Xue, Z., Marculescu, R.: Dynamic multimodal fusion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2574–2583 (2023)
29. Yan, L., Liu, K., Belyaev, E.: Revisiting sparsity invariant convolution: A network for image guided depth completion. IEEE Access **8**, 126323–126332 (2020)
30. Yang, B., Bender, G., Le, Q.V., Ngiam, J.: Condconv: Conditionally parameterized convolutions for efficient inference. Advances in neural information processing systems **32** (2019)
31. Zhang, L., Lan, M., Zhang, J., Tao, D.: Stagewise unsupervised domain adaptation with adversarial self-training for road segmentation of remote-sensing images. IEEE Trans. Geosci. Remote Sens. **60**, 1–13 (2021)

32. Zhao, X., Huang, W., Chen, J., Chen, Z., Li, J.: Automatic thin crack segmentation with deep context aggregation network. In: 2022 International Conference on Advanced Robotics and Mechatronics (ICARM). pp. 206–212. IEEE (2022)
33. Zhou, G., Chen, W., Gui, Q., Li, X., Wang, L.: Split depth-wise separable graph-convolution network for road extraction in complex environments from high-resolution remote-sensing images. IEEE Trans. Geosci. Remote Sens. **60**, 1–15 (2021)
34. Zhu, Y., Long, L., Wang, J., Yan, J., Wang, X.: Road segmentation from high-fidelity remote sensing images using a context information capture network. Cognitive computation pp. 1–14 (2022)
35. Zou, Q., Cao, Y., Li, Q., Mao, Q., Wang, S.: Cracktree: Automatic crack detection from pavement images. Pattern Recogn. Lett. **33**(3), 227–238 (2012)

MCFM: Multi Channel-Frequency Mamba-Based Model for Flight Trajectory Prediction

Wanjing Zhang, Xiaotian Zhu, Jianjun Zhang, Yuan Guo, Jun Tao,
and Min Zhu(✉)

Department of Computer Science, Sichuan University, Chengdu, China
zhumin@scu.edu.cn

Abstract. Accurate flight trajectory prediction is crucial for enhancing the overall efficiency of air traffic management. However, existing methods often overlook the importance of capturing flight trends and pay insufficient attention to temporal relationships between different variate channels, impacting prediction accuracy. In this work, we propose a novel Multi Channel-Frequency Mamba-based framework MCFM, which leverages Mamba for the first time to extract flight trends from frequency information and incorporates time dependencies of diverse channels. Specifically, MCFM utilizes the Multi-Channel Interaction block to extract and fuse the temporal patterns of various channels. We design the Attention Mamba to perform frequency extraction for obtaining detailed trend insights. On that basis, we introduce a Shared Mamba to increase the interaction of frequency information, thus adding mutual guidance and improving prediction accuracy. Experimental results demonstrate that MCFM outperforms existing methods on a real-world dataset.

Keywords: Flight trajectory prediction · Mamba · Time series analysis

1 Introduction

Flight trajectory prediction (FTP) is essential in air traffic management. Providing controllers trajectory information allows them to monitor airspace conditions more accurately, enabling the detection of potential conflicts [1] and predicting flight delays [2] in advance. As the global economy expands, the aviation sector has witnessed a sustained increase in air transportation and passenger volume. In this context, timely and precise FTP is crucial for enhancing the overall efficiency of air traffic management and mitigating the risk of potential flight conflicts.

FTP task aims to forecast future flight states, such as longitude, latitude, altitude, and speed, based on the historical flight trajectory. Existing methods for flight trajectory prediction can be broadly categorized into three main approaches: kinetic modeling, state estimation, and machine learning. The kinetic modeling approach [3–5] relies on establishing kinematic equations to

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 417–430, 2025.
https://doi.org/10.1007/978-3-031-78383-8_28

simulate flight patterns and forecast future motion trajectories. In contrast, the state estimation approach [6,7] focuses on establishing motion equations based on aircraft attributes such as position, speed, and acceleration to propagate and estimate subsequent motion states. However, both of them struggle to accurately capture the uncertainties and maneuver characteristics in real-world scenarios.

In recent years, the rapid advancement of data mining techniques has given rise to various machine learning approaches for FTP task. Some researchers have utilized regression models [8–11], for instance, Tastambekov et al. [12] proposed a method that employs wavelet decomposition to establish a local linear regression model based on historical radar trajectory data. Neural network models, in particular, have gained traction due to their capability to automatically learn complex feature representations from input data without manual feature engineering. Shi et al. [14] utilized LSTM to link long-term relationships within the current prediction task, achieving improved results in both 3D and 4D aircraft trajectory forecasting. Furthermore, Zhang et al. [16] introduced WTFTP, which leverages time-frequency analysis to capture the dynamic characteristics of trajectories. Guo et al. developed FlightBert [17] and FlightBert++ [18] based on Transformer as the backbone network and Dong et al. [19] proposed a hybrid model that combines temporal convolutional networks and Informer [25], an enhanced Transformer model, to better capture complex long-range dependencies. However, these Transformer-based models ignore sequence order in time series and suffer from quadratic computational complexity [27].

Lately, a novel architecture, Mamba [20], introduces a state-space model(SSM) equipped with a selective mechanism, allowing the model to focus on the critical information while filtering out irrelevant details. This selective attention strategy has effectively mitigated the computational efficiency issues that Transformer models often encounter when processing long data sequences [21]. Moreover, Mamba enables to concentrate on the preceding window during information extraction, thus preserving certain sequential properties [27]. In this light, we design a Mamba-based structure to explore its potential effectiveness in series information extraction.

The frequency information in the flight trajectory data conveys distinct trends, which are vital to comprehending flight patterns. Specifically, the low-frequency component can be interpreted as the global flight trend, representing the overall trajectory direction, while the high-frequency component captures the details of local motion dynamics [16]. Although WTFTP is considered from a time-frequency perspective, it overlooks the crucial interactions between frequency components. The frequency information exhibits inherent interdependencies, which enables frequency components to guide and inform each other, thereby enhancing the overall understanding of flight patterns.

Recurrent or attention-based architectures possess high representational capacity, but achieving time-step independence remains a challenge. This is because these models tend to overfit the data instead of solely considering the position information [26]. To address these challenges and achieve high-accuracy FTP prediction, we propose a Mamba-based approach MCFM containing a

Multi-Frequency Mamba (MFM) block and a Multi-Channel Interaction (MCI) block. Specifically, the MFM block designs an Attention Mamba mechanism with shared weight for extracting and fusing frequency components, while the MCI block facilitates the interaction of information across time steps within each feature channel. Results of real-world flight trajectory prediction show the benefits and effectiveness of the proposed framework. In summary, our main contributions are as follows:

- We innovatively proposed MCFM for FTP using the Mamba-based structure from the perspective of frequency decomposition and time dependency.
- We proposed the MFM Block, which is designed to capture and interact between overall trends within the low-frequency components and changes in motion details in the high-frequency components.
- We proposed the MCI block, which allows channels to communicate with different time step information, extracting and fusing the complex time dependencies in the sequence data.
- Experimental results show that the proposed framework MCFM outperforms comparative benchmarks in terms of FTP accuracy when compared with existing models on real flight trajectory data.

2 Preliminaries

2.1 Problem Formulation

Typically, FTP task focuses on analyzing the historical trajectory data X to obtain the flight state Y after a period of time. Given an observed trajectory point P_t at time t, which contains longitude, latitude, altitude, and flight speed. P_t is defined as Eq. 1:

$$P_t = \{Lon_t, Lat_t, Alt_t, V_t\} \tag{1}$$

Multivariate flight trajectory prediction task aims to predict the future trajectory sequence $Y_{t+1:t+n} = \left\{\hat{P}_{t+1}, \hat{P}_{t+2}, \ldots, \hat{P}_{t+n}\right\}$ with n steps, based on historical trajectory sequence $X_{1:t} = \{P_1, P_2, \ldots, P_t\}$. The prediction process of the MCFM as shown in Eq. 2:

$$Y_{1:t+1} = MCFM\left(X_{1:t}\right) \tag{2}$$

To predict future multistep trajectory points, an iterative prediction approach is employed. Specifically, we use previously forecasted values as pseudolables to predict longer time steps, as illustrated in Eq. 3:

$$Y_{s:t+s} = MCFM\left(\left[X_{s:t}, \hat{P}_{t+1:t+s-1}\right]\right), \tag{3}$$

where s denotes the prediction horizon.

2.2 Mamba

The state space sequence model(SSM) maps the input $x(t) \in R^M$ to the output $y(t) \in R^M$ through the hidden state $h(t) \in R^N$. This process can be represented by the following Eqs. 4, 5:

$$h'(t) = Ah(t) + Bx(t), \tag{4}$$

$$y(t) = Ch(t), \tag{5}$$

where the state matrix $A \in R^{N \times N}$, input matrix $B \in R^{N \times M}$, and output matrix $C \in R^{M \times N}$ are learnable matrices.

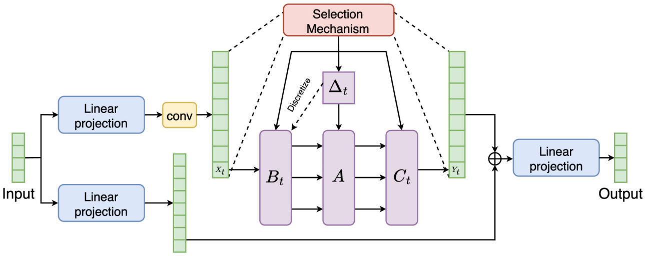


Fig. 1. The Mamba architecture

Selective Scan Mechanism was first proposed as a novel parameterized approach to address the limitations of capturing contextual information for SSM. This mechanism employs a context-aware selection mechanism that allows for efficient information filtering from the input, thereby enhancing the applicability of SSM in discrete data modeling. Afterward, a hardware-aware algorithm was developed, which cyclically computes the model through a single scan, linearly scaling with sequence length, thus speeding up the execution of Mamba on modern hardware. The combination of SSM and MLP forms the Mamba architecture as shown in Fig.1. Stacking multiple Mamba blocks constitutes the Mamba architecture.

2.3 Wavelet Transform

The wavelet transform [22] analyzes and represents the various frequency components of a signal at different resolutions. Multilevel discrete wavelet decomposition enables the extraction of multilevel time-frequency features from time series by decomposing the low- and high-frequency sub-levels in a stepwise manner for discrete signal analysis [23].

Specifically, input sequences are denoted as $X = \{x_1, x_2, \ldots, x_t\}$ and the low- and high-frequency sub-series generated at the i-th level as $x^{low}(i)$ and $x^{high}(i)$.

At the $(i+1)$-th level, through the low-pass filter $L = \{l_1, l_2, \ldots, l_K\}$ and the high-pass filter $H = \{h_1, h_2, \ldots, h_K\}$ $(K \ll t)$, we can obtain the intermediate variables $a^{low}(i)$ and $a^{high}(i)$ as Eqs. 6, 7:

$$a_n^{low}(i+1) = \sum_{k=1}^{K} x_{n+k-1}^{low}(i) \cdot l_k, \tag{6}$$

$$a_n^{high}(i+1) = \sum_{k=1}^{K} x_{n+k-1}^{low}(i) \cdot h_k, \tag{7}$$

where, $x^{low}(i)$ is the i-th element of the low-frequency subsequence of layer i and $x^{low}(0)$ is the input sequence. The i-th layers' low-frequency subsequence $x^{low}(i)$ and high-frequency subsequence $x^{high}(i)$ are generated by 1/2 down-sampling of intermediate variables $A^{low}(i) = \left\{a_1^{low}(i), a_2^{low}(i), \ldots\right\}$ and $A^{high}(i) = \left\{a_1^{high}(i), a_2^{high}(i), \ldots\right\}$.

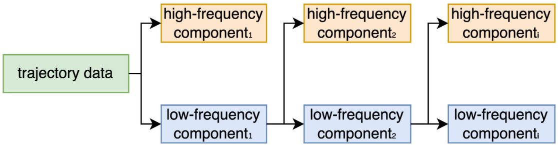

trajectory data
high-frequency component$_1$
high-frequency component$_2$
high-frequency component$_i$
low-frequency component$_1$
low-frequency component$_2$
low-frequency component$_i$

Fig. 2. The wavelet decomposition

By discrete wavelet transform, the input sequence can be decomposed into the set of sub-series $\mathcal{X}(i) = \left\{\mathcal{X}^{low}(i), \mathcal{X}^{high}(1), \mathcal{X}^{high}(2) \ldots, \mathcal{X}^{high}(i)\right\}$ of i-th layer, containing a low frequency signal and i high frequency signals, as shown in Fig.2. Once we have the set of sub-series $\mathcal{X}(i)$ then X can be reconstructed through inverse discrete wavelet transform (IDWT). The implementation of wavelet transform can be found in Python PyWavelets [28].

3 Materials and Methods

3.1 Materials and Preprocessing

In this work, we utilize real-world flight trajectory data collected through the Automatic Dependent Surveillance-Broadcast (ADS-B) system. The dataset comprises 6,981 flight trajectories, with attributes including timestamp, flight ID, longitude, latitude, altitude, and speed. To enhance the data quality, we preprocessed the raw data, which involved anomaly rejection, null value imputation, and segmenting the trajectory into 10-second intervals. Furthermore, the trajectory attributes were normalized using min-max scaling to unify the data scale and range (latitude and longitude in degrees, altitude in 10 metres, and speed in kilometers per hour). The final experimental dataset was then partitioned into training, validation, and test set at a ratio of 8:1:1.

3.2 The Proposed Framework

The MCFM framework proposed is shown in Fig.3, which mainly involves three stages: multi-channel interaction, multi-frequency decomposition, and inversion projection. First, the Multi-Channel Interaction (MCI) block models the temporal patterns from each feature channel independently, which are then encoded and fed into the Multi-Frequency Mamba (MFM) block. The MFM block models trend patterns by decomposing them into high- and low-frequency components, representing local and global information. Finally, the domain inverse projection stage predicts the trajectory sequence by using IDWT.

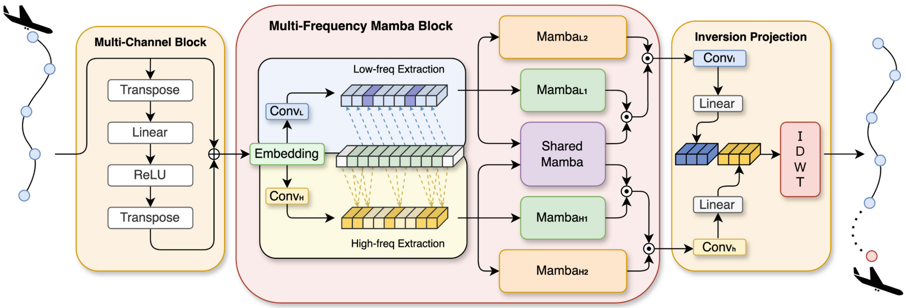


Fig. 3. The framework of MCFM.

Multi-Channel Interaction Block MCI block consists of a linear layer followed by an activation function. Specifically, the input sequence $X \in R^{L\times N}$ is first transposed, with input sequence length L and number of variates N. Then, a single-layer MLP is applied to each channel independently, extracting its temporal features D. The use of a nonlinear activation function ReLU, allows the MCI block to capture the complex interactions and dependencies among the input. As shown in Eq. 8:

$$D = ReLU\left(Linear\left(X^T\right)\right) \tag{8}$$

To preserve the dependencies and temporal patterns between the original channels, a residual connection is employed to fuse the original input information with $D^T \in R^{L\times N}$, as shown in Eq. 9:

$$D^{'} = D^T + X \tag{9}$$

Multi-Frequency Mamba block Given that trajectory sequences are time series, their frequency signals help to identify trend patterns in the flight duration, which is highly beneficial for trajectory analysis. The MFM block commences by embedding the output from the preceding MCI block with dimension

C, which enables the extraction of rich semantic representations $E \in R^{L\times C}$. Convolutions with varying receptive field sizes are employed to obtain initial high- and low-frequency features. Specifically, a 1×1 convolution $Conv_L$ is utilized to focus on the local region and extract the slowly evolving fundamental pattern features E_{low} of the trajectory. A 3×3 convolution $Conv_H$ is used to capture the relatively long-range dependencies in the trajectory sequence and extract the rapidly changing detailed features E_{high}. Finally, these features are fed into the Attention Mamba mechanism, which is responsible for further extracting their respective frequency information, as shown in Eqs. 10, 11:

$$E = Embedding\left(D^{'}\right), \tag{10}$$

$$E_{low} = Conv_L\left(E\right), E_{high} = Conv_H\left(E\right), \tag{11}$$

where, C is set as 64. $Conv_L$ uses a kernel size of 1, stride of 1 and no padding. $Conv_H$ uses a kernel size of 3, stride of 1 and padding of 1. E_{low}, $E_{high} \in R^{L\times 1/2C}$.

Attention Mamba takes inspiration from the self-attention mechanism used in Transformer. It employs three distinct Mamba blocks as the query (Q), key (K), and value (V) components respectively, through passing the output of the previous step E_{low}, E_{high}. By applying the Hadamard product of the $Q, K, V \in R^{L\times 1/2C}$ to the high- and low-frequency branches, it can extract more fine-grained local features and richer contextual information A_{low}, $A_{high} \in R^{L\times 1/2C}$ from the trajectory sequence data. As shown in Eqs. 12, 13, 14, 15:

$$Q_{low} = Mamba_{L0}\left(E_{low}\right), Q_{high} = Mamba_{H0}\left(E_{high}\right), \tag{12}$$

$$K_{low} = Mamba_{L1}\left(E_{low}\right), K_{high} = Mamba_{H1}\left(E_{high}\right), \tag{13}$$

$$V_{low} = Mamba_{L2}\left(E_{low}\right), V_{high} = Mamba_{H2}\left(E_{high}\right), \tag{14}$$

$$A_{low} = Q_{low} \odot K_{low} \odot V_{low}, A_{high} = Q_{high} \odot K_{high} \odot V_{high}, \tag{15}$$

Shared Mamba is introduced to enhance the interaction between high- and low-frequency information. Both high- and low-frequency component collectively influences trajectory prediction, which is critical for accurate prediction. Transient maneuvers detected by high-frequency components cause changes in flight trends reflected in low-frequency information. Conversely, the low-frequency information serves to guide and moderate the fluctuations observed in the high-frequency component. Through the Shared Mamba, which selects one of the Mamba blocks to share the arithmetic unit in the extraction of high- and low-frequency information, MCFM achieves dynamic feature sharing. The specific choice and implementation details of the Mamba chosen to share parameters will be further elaborated in Section 4.4.

Inversion Projection passes the extracted high- and low-frequency information from the MFM block through separate convolutional and linear layers respectively to generate the frequency domain signals F_{low}, F_{high} of the historical reconstruction and the predicted trajectory sequence. Subsequently, they are converted into the time domain signals of the flight trajectory using the IDWT to obtain the predicted sequence $Y \in R^{L \times N}$ as shown in Eqs. 16, 17:

$$F_{low} = Linear\left(Conv_l\left(A_{low}\right)\right), F_{high} = Linear\left(Conv_h\left(A_{high}\right)\right), \tag{16}$$

$$Y = IDWT\left(F_{low}, F_{high}\right), \tag{17}$$

where F_{low}, $F_{high} \in R^{\lfloor 1/2L \rfloor \times 1/2C}$, $Conv_l$ and $Conv_h$ are both using a kernel size of 3, stride of 2 and padding of 1. In this work, we use one-layer wavelet transform to decompose into a high-frequency component and a low-frequency component and reconstruct the input trajectory sequence.

4 Experiments and Results

4.1 Experimental Configuration

The experiment environment is mainly based on Python 3.8, PyTorch 1.11.0 with a batch size of 1024 and a total number of 150 training epochs. Haar wavelets are used for decomposition and reduction, and the Adam optimizer was chosen to update the trainable parameters with a learning rate initialized to 0.001 and decaying 0.5 times every 10 epochs.

4.2 Evaluation metrics

In this work, mean absolute error (MAE), mean relative error (MRE), and root of mean squared error (RMSE) is used to evaluate the proposed method and baselines. These metrics reflect the difference between the actual and predicted flight trajectories. Specifically, MAE visualizes the absolute value of the error between the true and predicted values, while RMSE is more sensitive to the prediction of outliers. Additionally, the mean deviation error (MDE) is used to measure the Euclidean distance between predicted and actual points projected to the earth-centered and earth-fixed coordinate system, enabling the evaluation of the overall performance in real-world scenarios.

$$MAE_i = \frac{1}{n} \sum_{j=1}^{n} \left|T_{i,j} - P_{i,j}\right|, \tag{18}$$

$$MRE_i = 100\% \frac{1}{n} \sum_{j=1}^{n} \left|\frac{T_{i,j} - P_{i,j}}{T_{i,j}}\right|, \tag{19}$$

$$RMSE_i = \sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(T_{i,j} - P_{i,j}\right)^2}, \tag{20}$$

$$MDE = \frac{1}{n}\sum_{i=1}^{n}\Phi\left(T_{i,j} - P_{i,j}\right), \tag{21}$$

where n is the total number of test sets. $T_{i,j}$ is the true value of the i-th attribute of the trajectory point of the j-th sample, and $P_{i,j}$ is the corresponding predicted value. $\Phi(\cdot)$ is the calculation function of Euclidean distance in the 3D airspace.

4.3 Performance of Proposed MCFM

To validate the effectiveness of MCFM, we chose five baseline models with diverse architectures for comparison, including LSTM-based model: LSTM [13], CNN-LSTM [15], WTFTP [16] and Transformer-based model: CNN-Transformer [24], Informer[25]. The experimental results are presented in Table 1.

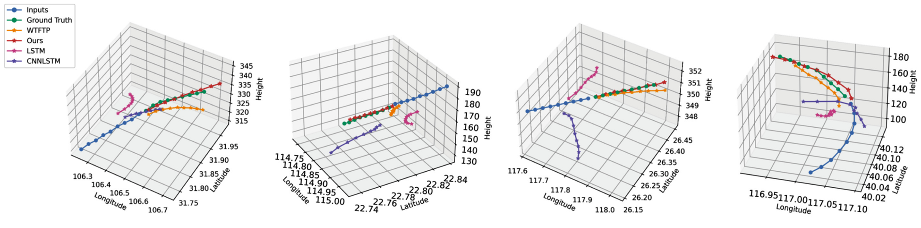
- LSTM: Vanilla LSTM is used to model the trajectories and the fully connected layer is used for prediction.
- CNN-LSTM: CNNs combined with vanilla LSTM are applied to extract spatial and temporal features.
- WTFTP: A LSTM based model to prediction the trajectory from a time-frequency perspective.
- CNN-Transformer: CNNs combined with vanilla Transformer are applied to predict the trajectory point.
- Informer: An improved Transformer-based model with ProbSparse self-attention mechanism to capture the long-term dependence of the trajectory.

Overall, the proposed framework MCFM outperforms the other baselines, achieving the best performance in almost all evaluation metrics. There is only a slight decrease in the MAE and MRE metrics for altitude. MCFM performs better than WTFTP, exceeding the MDE indicator by 34%, thanks to its consideration of the interaction between high- and low-frequency components. Furthermore, both LSTM-based and Transformer-based prediction methods have exhibited limited accuracy. In contrast, the Mamba block utilized in MCFM maintains a strong focus on the preceding temporal window during the information extraction process. This allows the model to effectively preserve important sequential properties of the input data and improve the accuracy of prediction. Informer exhibits relatively poor in these metrics, it is probably because Informer was originally designed for long-term prediction tasks and may lack enough sensitivity to detailed changes in the shorter FTP task, generating some errors. Notably, the superior performance of MCFM highlights the effectiveness of its Mamba-based mechanism in extracting frequency-domain information and incorporating temporal patterns.

Table 1. Summary of the different methods

Model	RMSE			MAE			MRE (%)			MDE
	Lon	Lat	Alt	Lon	Lat	Alt	Lon	Lat	Alt	
MCFM	**0.02056**	**0.01891**	**2.37178**	**0.01194**	**0.01266**	0.61131	**0.01039**	**0.04138**	0.18508	**2.02772546**
WTFTP	0.03620	0.02427	2.43212	0.02487	0.014405	**0.57587**	0.02168	0.04734	**0.17177**	3.06073618
LSTM	0.05074	0.02719	2.37383	0.03947	0.01463	0.58864	0.03445	0.04909	0.17710	4.38647127
CNN-LSTM	0.04818	0.05722	2.88231	0.03667	0.04516	0.89365	0.03199	0.14893	0.26174	6.82610464
CNN-Trans	0.09571	0.10379	2.43215	0.06829	0.07625	1.05930	0.05970	0.25319	0.29827	12.16895676
Informer	0.28661	0.26392	2.60525	0.18893	0.18610	1.91307	0.16420	0.59802	1.35893	31.54741000

To further evaluate the performance of MCFM and the baselines, we selected three baseline methods that exhibited superior performance and visualized different flight phases, including common flight modes such as cruise and descent, as well as more complex flight modes like ascent and turn. Using the iterative prediction approach, we forecasted the future 90-second flight trajectory based on the preceding 80-second historical trajectory data. As can be seen from the Fig.4, in the ascent, descent, and cruise phases, MCFM accurately predicts the direction and potential maneuvering changes. It also exhibits exceptional capabilities in complex turning situations, accurately forecasting trajectory changes.



(a) Ascent (b) Descent (c) Cruise (b) Turn

Fig. 4. The real world scenario. The blue line represents the historical flight trajectory, the green line depicts the ground-truth of future trajectory, and the red line shows the projected trajectory predicted by the MCFM.

4.4 Ablation study

To further investigate the key factors contributing to the performance of MCFM, we conducted an ablation study examining the effectiveness of the MCI block, Attention Mamba and Shared Mamba within the MFM block. The configurations used in these ablation experiments are summarized in Table 2. Additionally, we introduced a "Stack Mamba" configuration, which utilizes three stacked Mamba blocks to capture frequency features, serving as a comparative baseline against the Attention Mamba.

Table 2. Summary of the different methods with various configurations

Model	**Block**			
	MCI block	High-frequency	Low-frequency	Shared Mamba
MCFM	✓	Attention Mamba	Attention Mamba	$Mamba_{L0}, Mamba_{H0}$
A1		Attention Mamba	Attention Mamba	$Mamba_{L0}, Mamba_{H0}$
A2	✓	Attention Mamba	Attention Mamba	
A3	✓	Attention Mamba	Attention Mamba	$Mamba_{L2}, Mamba_{H2}$
A4	✓	Stack Mamba	Attention Mamba	$Mamba_{L0}, Mamba_{H0}$
A5	✓	Attention Mamba	Stack Mamba	$Mamba_{L0}, Mamba_{H0}$
A6	✓	Stack Mamba	Stack Mamba	$Mamba_{L0}, Mamba_{H0}$

The results of these experiments are presented in Table 3. From A1, we can observe that the proposed MCI block improves the MDE metric by 23%. Moreover, by replacing Attention Mamba in different branches with a Stack Mamba configuration (which can be seen from A4 to A6), we can clearly observe an decrease in the prediction results. Especially after replacing all Attention Mamba with Stack Mamba (A6), there is an obvious decrease in all metrics and 18% in terms of MDE. Results show that the inclusion of Attention Mamba enables MCFM to capture the key frequency information more effectively and is more suitable than Stack Mamba.

Table 3. Results of the different methods with various configurations

Model	**RMSE**			**MAE**			**MRE(%)**			**MDE**
	Lon	**Lat**	**Alt**	**Lon**	**Lat**	**Alt**	**Lon**	**Lat**	**Alt**	
MCFM	**0.02056**	**0.01891**	2.37178	0.01194	**0.01266**	0.61131	0.01039	**0.04138**	0.18508	**2.02772546**
A1	0.04381	0.02712	2.17071	0.01459	0.01674	0.52947	0.01275	0.05520	0.15925	2.62008095
A2	0.02691	0.02711	2.18493	0.01371	0.01652	**0.52515**	0.01194	0.05432	**0.15765**	2.52338099
A3	0.02921	0.02889	2.70807	0.01331	0.01565	0.69953	0.01163	0.05150	0.20994	2.41616940
A4	0.02365	0.03608	**2.01948**	0.01420	0.01832	0.54524	0.01242	0.06035	0.16495	2.72706342
A5	0.02116	0.02768	2.48771	**0.01079**	0.01320	0.69814	**0.00943**	0.04401	0.21000	2.03206348
A6	0.03561	0.02441	2.11820	0.01245	0.01662	0.54677	0.01084	0.05474	0.16520	2.45344305

To further demonstrate the importance of incorporating frequency information interactions and to select an appropriate experimental setup, we conducted additional experiments on the Shared Mamba component. Specifically, we set up no shared weights in Mamba blocks (A2), shared weights in $Mamba_{L0}$ and $Mamba_{H0}$ (MCFM), shared weights in $Mamba_{L2}$ and $Mamba_{H2}$ (A3). A lack of attention to frequency interaction information greatly affects the accuracy of the predictions, as shown in A2. Additionally, manipulating the shared weights in the final step of the MFM block affects the prediction performance to some degree, which is probably due to the frequency interaction information dominating the prediction results and confusing the respective signal outputs. Consequently, we select $Mamba_{L0}$ and $Mamba_{H0}$ used to compute Q for sharing parameters.

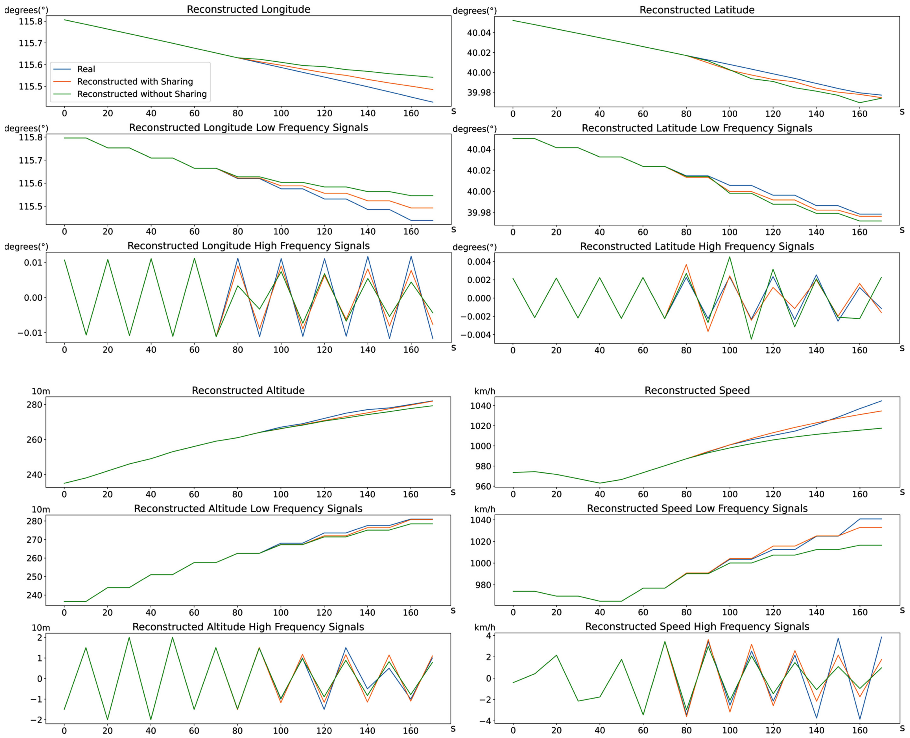


Fig. 5. Visualization of frequency signals. The horizontal axis represents time (with 170 seconds selected as an example), while the vertical axis corresponds to the units of different trajectory attributes. The altitude is measured in 10 metres (10 m).

Fig.5 presents a visual comparison between the reconstruction and decomposition of trajectory variates with and without frequency information sharing (Shared Mamba). The ground truth frequency components are obtained through one-layer wavelet decomposition of the actual trajectory variate, yielding high-frequency and low-frequency signals. It is observed that on any of the four variates, by adding the Shared Mamba, MCFM separate the better high- and low-frequency signals and bring them closer to the signal ground truth. In conclusion, the ablation study conducted reveals that all the proposed components of the MCFM model are essential for enhancing the FTP prediction performance.

5 Conclusion

In this work, we propose MCFM, a new framework for solving FTP task, which consists of two key components: the Multi-Channel Interaction(MCI) block and the Multi-Frequency Mamba(MFM) block. The MCI block serves to extract and fuse the temporal patterns of the diverse data channels within the input flight trajectory data. The MFM block then extracts and interacts with high- and low-frequency information from the trajectory sequence data. Through quantitative and qualitative evaluations, we demonstrate that the proposed MCFM outperforms various baseline approaches, and ablation studies further confirm the importance of the technological innovations introduced in MCFM. In the future, we plan to extend our work by exploring a multi-step prediction framework for FTP task to alleviate the error accumulation problem often encountered with single-step prediction models and provide more accurate and reliable flight trajectory prediction solutions.

References

1. Chen, Z., Guo, D., Lin, Y.: A deep gaussian process-based flight trajectory prediction approach and its application on conflict detection. Algorithms **13**(11), 293 (2020)
2. Gui, G., Liu, F., Sun, J., Yang, J., Zhou, Z., Zhao, D.: Flight delay prediction based on aviation big data and machine learning. IEEE Trans. Veh. Technol. **69**(1), 140–150 (2019)
3. Cui, N., Huang, R., Fu, Y., et al.: Design of analytical prediction-correction skip entry guidance law based on matched asymptoticexpansions [j]. Acta Aeronautica et Astronautica Sinica. **36**(8), 2764–2772 (2015)
4. Jinchuan, H., Jing, Z., Wanchun, C.: Analytical solutions of steady glide trajectory for hypersonic vehicle and planning application. Journal of Beihang University **42**(5), 961–968 (2016)
5. Wang, C., Guo, J., Shen, Z.: Prediction of 4d trajectory based on basic flight models. Journal of southwest jiaotong university **44**(2), 295–300 (2009)
6. Zhang, J., Wu, X., Wang, F.: Aircraft trajectory prediction based on modified interacting multiple model algorithm. Journal of Donghua University **32**(2), 180–184 (2015)
7. Liu, W., Hwang, I.: Probabilistic trajectory prediction and conflict detection for air traffic control. J. Guid. Control. Dyn. **34**(6), 1779–1789 (2011)
8. A. De Leege, M. van Paassen, and M. Mulder, "A machine learning approach to trajectory prediction," in *AIAA Guidance, Navigation, and Control (GNC) Conference*, 2013, p. 4782
9. M. G. Hamed, D. Gianazza, M. Serrurier, and N. Durand, "Statistical prediction of aircraft trajectory: regression methods vs point-mass model," in *ATM 2013, 10th USA/Europe Air Traffic Management Research and Development Seminar*, 2013, pp. pp–xxxx
10. S. T. Kanneganti, P. B. Chilson, and R. Huck, "Visualization and prediction of aircraft trajectory using ads-b," in *NAECON 2018-IEEE National Aerospace and Electronics Conference*. IEEE, 2018, pp. 529–532
11. Hong, S., Lee, K.: Trajectory prediction for vectored area navigation arrivals. Journal of Aerospace Information Systems **12**(7), 490–502 (2015)
12. Tastambekov, K., Puechmorel, S., Delahaye, D., Rabut, C.: Aircraft trajectory forecasting using local functional regression in sobolev space. Transportation research part C: emerging technologies **39**, 1–22 (2014)

13. Z. Shi, M. Xu, Q. Pan, B. Yan, and H. Zhang, “Lstm-based flight trajectory prediction,” in *2018 International joint conference on neural networks (IJCNN)*.IEEE, 2018, pp. 1–8
14. H. Wu, Y. Liang, B. Zhou, and H. Sun, “A bi-lstm and autoencoder based framework for multi-step flight trajectory prediction,” in *2023 8th International Conference on Control and Robotics Engineering (ICCRE)*. IEEE, 2023, pp. 44–50
15. L. Ma and S. Tian, “A hybrid cnn-lstm model for aircraft 4d trajectory prediction,” *IEEE access*, vol. 8, pp. 134 668–134 680, 2020
16. Zhang, Z., Guo, D., Zhou, S., Zhang, J., Lin, Y.: Flight trajectory prediction enabled by time-frequency wavelet transform. Nat. Commun. **14**(1), 5258 (2023)
17. Guo, D., Wu, E.Q., Wu, Y., Zhang, J., Law, R., Lin, Y.: Flightbert: binary encoding representation for flight trajectory prediction. IEEE Trans. Intell. Transp. Syst. **24**(2), 1828–1842 (2022)
18. Guo, D., Zhang, Z., Yan, Z., Zhang, J., Lin, Y.: Flightbert++: A non-autoregressive multi-horizon flight trajectory prediction framework. Proceedings of the AAAI Conference on Artificial Intelligence **38**(1), 127–134 (2024)
19. Dong, Z., Fan, B., Li, F., Xu, X., Sun, H., Cao, W.: Tcn-informer-based flight trajectory prediction for aircraft in the approach phase. Sustainability **15**(23), 16344 (2023)
20. A. Gu and T. Dao, “Mamba: Linear-time sequence modeling with selective state spaces,” arXiv preprint arXiv:2312.00752, 2023
21. Z. Wu, Y. Gong, and A. Zhang, “Dtmamba: Dual twin mamba for time series forecasting,” arXiv preprint arXiv:2405.07022, 2024
22. S. Mallat, *A wavelet tour of signal processing*. Elsevier, 1999
23. J. Wang, Z. Wang, J. Li, and J. Wu, “Multilevel wavelet decomposition network for interpretable time series analysis,” in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2437–2446
24. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017
25. H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, “Informer: Beyond efficient transformer for long sequence time-series forecasting,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 12, 2021, pp. 11 106–11 115
26. S.-A. Chen, C.-L. Li, N. Yoder, S. O. Arik, and T. Pfister, “Tsmixer: An all-mlp architecture for time series forecasting,” arXiv preprint arXiv:2303.06053, 2023
27. Z. Wang, F. Kong, S. Feng, M. Wang, H. Zhao, D. Wang, and Y. Zhang, “Is mamba effective for time series forecasting?” arXiv preprint arXiv:2403.11144, 2024
28. Lee, G., Gommers, R., Waselewski, F., Wohlfahrt, K., O’Leary, A.: Pywavelets: A python package for wavelet analysis. Journal of Open Source Software **4**(36), 1237 (2019)

Multi-objective Balanced Task Offloading in Vehicular Networks Based on Edge Computing

Lingjiao Wang[1,2](✉), Lingwei Meng[1,2], and Hua Guo[1,2]

[1] Xiangtan University, Xiangtan 411105, Hunan, China
xtu_wlj@126.com

[2] Key Laboratory of Intelligent Computing and Information Processing of Ministry of Education, Xiangtan University, Xiangtan 411105, Hunan, China

Abstract. The increasing number of vehicular networking devices and application demands has made the limited computing and communication resources a significant challenge. The heuristic task offloading strategy mechanism was proposed to improve the efficiency of task offloading in vehicular networks. This strategy mechanism utilized the Nondominated Sorting Genetic Fireworks Algorithm (NSGFA) based on the characteristics of the problem, integrated consideration of the multi-target balance between system offloading costs and load balancing. Experimental data results demonstrate that this method performs well in reducing latency, energy consumption, and load balancing, effectively enhancing the quality of service for vehicular network users.

Keywords: Internet of Vehicles · Mobile Edge Computing · Computational Offloading · Multi-Objective Optimization

1 Introduction

As urbanization advances, automobiles, which are an indispensable part of people's daily lives, are also seeing a constant increase in numbers and popularity. At the same time, the connection between vehicles and the Internet is becoming increasingly tight, forming a vast Internet of Vehicles ecosystem [1, 2]. In this ecosystem, in-vehicle sensors, cameras, and other intelligent devices are constantly collecting and generating a large amount of data, which contains rich information such as vehicle status, environmental changes, and traffic conditions. However, the processing and transmission of this massive, dense data [3, 4] pose great challenges to the cloud, not only is the strain on network bandwidth increasing, but it also reduces the efficiency of data processing, making it challenging to meet the real-time requirements of high-demand applications.

To address this challenge, Mobile Edge Computing (MEC) has emerged [5–7]. Edge computing shifts the focus of data processing from the traditional cloud to edge locations closer to the data source. By deploying intelligent edge devices near vehicles, it enables real-time data processing and response. This distributed computing architecture can

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 431–446, 2025.
https://doi.org/10.1007/978-3-031-78383-8_29

reduce the load on the cloud, lower network latency, and decrease energy consumption [8, 9].

However, due to the high difficulty of server offloading and unreasonable allocation, researching how to efficiently and reasonably offload resource tasks and reduce the latency and energy consumption caused by offloading has become a highly concerning problem. Task offloading in vehicular networks can be considered an NP-hard optimization problem [10]. Literature [11] employed a complementary edge cloud solution, optimizing and solving the problem using meta-heuristic methods such as NSGAII and the Bee algorithm. This approach reduces the response time, but the control over energy consumption is not optimal. Literature [12] discussed a Halton sequence for a uniform initial population distribution and an improved genetic algorithm as an offloading strategy that had a higher completion rate and better convergence than the original genetic algorithm. However, it doesn't figure out and compare the underlying indicators of offloading. Literature [13] introduced an ant colony optimization-based algorithm that dynamically updates the pheromone value based on the relationship between the fitness function and either the global or local optimal value. This algorithm migrates computation to edge devices closer to users, thereby reducing user transmission time, computation time, propagation time to meet the requirements of time-sensitive tasks, but it is not comprehensive enough, and the load on edge devices will be out of control. Literature [14] discussed a better particle swarm optimization algorithm for offloading in edge computing. This algorithm encodes particles and updates their positions to make the offloading target use the least amount of energy possible. Literature [15] examined the availability of vehicles based on their motion characteristics at intersections. For the task offloading and resource allocation scheme of vehicle users near intersections, it employs a dual deep Q-network method, which enhances the average utility of offloading and demonstrates the effectiveness of deep reinforcement learning. Literature [16] designed a dynamic computing strategy based on deep reinforcement learning, adding Softmax function and prioritized experience replay on TD3. Still, the environmental conditions are relatively simple, only starting from a single roadside unit (RSU) for offloading analysis.

Based on the literature above analysis, this paper studies the multi-objective optimization problem in the vehicular network task offloading environment and proposed a task offloading scheme based on the Non-dominated Sorting Genetic Fireworks Algorithm. The main contributions are as follows:

1 A three-tier edge computing network architecture for vehicular networks is created, with each tier representing a different offloading method. This makes it easier to assign different tasks to the right layer for offloading.
2 Within the network architecture, communication, offloading, and load balancing models for the vehicular network environment are established, combining total latency, total energy consumption, and total load balancing rate as multiple objective functions. Latency and energy consumption are weighted using a linear weighting algorithm to design a minimization scheme for the optimization problem.
3 The NSGFA was proposed as a three-stage construction theory that effectively integrates two intelligent algorithms. The explosion operation in the fireworks algorithm

was utilized to expand the local search range of the genetic algorithm, achieving faster convergence speed and deciding the offloading location for vehicular tasks.

4 Simulation experiments show that the method proposed in this paper has achieved a good balance between offloading cost and load balancing, and has confirmed the algorithm's superiority.

2 Network System Architecture

In the vehicular network environment, a three-tier network system architecture [17] is constructed to achieve multi-mode offloading of vehicular tasks, which includes local offloading, edge offloading, and cloud offloading. Local offloading is performed through the On Board Unit (OBU) on the vehicle, edge offloading uses roadside edge nodes for computation, and cloud offloading employs high-performance cloud server cluster.

Let the task set of the vehicular user be denoted as $Q_i = \left\{w_i, s_i, d_i, t_i^{max}, A_i\right\}$, where w_i represents the number of resources required to complete task Q_i, s_i represents the size of task Q_i, d_i indicates the distance between the task vehicle and RSU, t_i^{max} represents the maximum delay that task Q_i can tolerate, and $A_i \in \{0,1,2\}$ indicates whether task Q_i is offloaded locally ($A_i = 0$), to an edge server ($A_i = 1$), or to a cloud server ($A_i = 2$). The set of vehicular users is $U = \{u_1, u_2, \cdot\cdot\cdot, u_N\}$. The edge nodes consist of RSU and MEC servers, with M RSU deployed roadside, each equipped with a MEC server, thus the MEC server set is represented as $V = \{v_1, v_2, \cdot\cdot\cdot, v_M\}$. All edge nodes are connected to a cloud server cluster to facilitate the uploading of vehicular tasks for cloud offloading.

To achieve an even distribution of vehicles and servers in the system, in this paper, the vehicular users are modeled using a Poisson Distribution [18], and the edge nodes are modeled using the Uniform Distribution, simulating the random appearance of vehicles on the road and the placement of servers along the roadside at equal intervals based on their service coverage range. The architecture diagram is shown in Fig. 1.

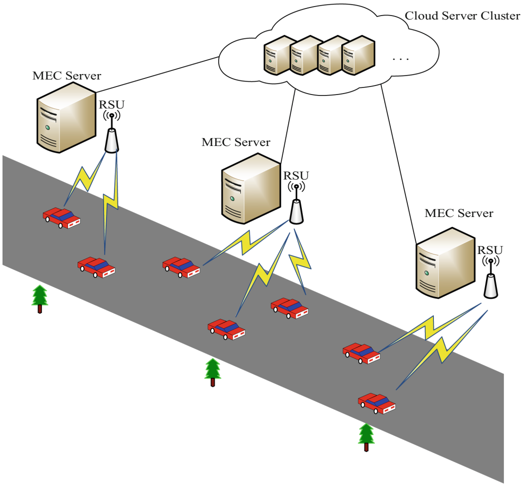


Fig. 1. Internet of Vehicles task offloading model architecture

3 Model Design

3.1 Communication Model

Since the architecture used in this paper requires tasks to be transmitted via the uplink to the edge node for offloading, it is necessary to establish a channel communication model between vehicles and infrastructure (Vehicle-to-Infrastructure, V2I). This paper adopts the Orthogonal Frequency Division Multiple Access (OFDMA) technology for channel bandwidth allocation, which allows the interference in transmission to be negligible. The formula for the uplink data upload rate R_i of V2I is as follows:

$$R_i = \frac{B}{S} log_2\left(1 + \frac{p_v h_i}{\sigma^2}\right) \tag{1}$$

where B represents the channel bandwidth, p_v is the transmission power of OBU, σ^2 represents the power of the white noise, S and h_i represent the number of channels and gains between OBU and MEC.

Then, h_i can be written as [19]:

$$h_i = 103.8 + 20.9 log_{10} d_i \tag{2}$$

3.2 Offloading Model

Local Vehicle Offloading. Since the task offloading is performed locally on the vehicle, the computation latency only needs to consider the time t_i^{local} it takes to execute the offloading, which can be written as:

$$t_i^{local} = \frac{w_i}{f^{local}} \tag{3}$$

where f^{local} represents the computing capability of OBU.

Defining the average computing power of the onboard unit as P_{local}, the energy consumption for local offloading e_i^{local} is given by:

$$e_i^{local} = P_{local} \cdot t_i^{local} \tag{4}$$

MEC Offloading. Because the amount of resources after offloading is minimal, the time for return transmission can be neglected [20], the latency of MEC offloading is composed of two parts: the task uploading latency $\hat{t}_i^{MEC}$ and the task execution latency $\bar{t}_i^{MEC}$, which can be written as:

$$\hat{t}_i^{MEC} = \frac{s_i}{R_i} \tag{5}$$

$$\bar{t}_i^{MEC} = \frac{w_i}{f^{MEC}} \tag{6}$$

where f^{MEC} represents the computing capability of MEC.

Thus, the total MEC offloading latency t_i^{MEC} is:

$$t_i^{MEC} = \hat{t}_i^{MEC} + \bar{t}_i^{MEC} \tag{7}$$

Defining the average computing power for MEC offloading as P_{MEC}, the energy consumption for MEC offloading e_i^{MEC} is given by:

$$e_i^{MEC} = P_{MEC} \cdot \bar{t}_i^{MEC} + p_v \cdot \hat{t}_i^{MEC} \tag{8}$$

Cloud Server Offloading. Like the MEC, the offloading latency of the cloud server is composed of two parts: the task uploading latency $\hat{t}_i^{cloud}$ and the task execution latency $\bar{t}_i^{cloud}$, which can be written as:

$$\hat{t}_i^{cloud} = \hat{t}_i^{MEC} + t_0 \tag{9}$$

$$\bar{t}_i^{cloud} = \frac{w_i}{f^{cloud}} \tag{10}$$

where t_0 represents the transmission latency of task resources from the RSU to the cloud [21], and f^{cloud} represents the computing capability of the cloud server.

Thus, the total offloading latency for the cloud server t_i^{cloud} is given by:

$$t_i^{cloud} = \hat{t}_i^{cloud} + \bar{t}_i^{cloud} \tag{11}$$

Defining the average computing power for cloud server offloading as P_{cloud}, the energy consumption for cloud server offloading e_i^{cloud} is given by:

$$e_i^{cloud} = P_{cloud} \cdot \bar{t}_i^{cloud} + p_v \cdot \hat{t}_i^{cloud} \tag{12}$$

Summarizing the above, the task offloading latency and energy consumption for a single vehicular user can be represented as follows:

$$t_i = (A_i - 1)(A_i - 2)t_i^{local} + A_i(A_i - 2)t_i^{MEC} + A_i(A_i - 1)t_i^{cloud} \tag{13}$$

$$e_i = (A_i - 1)(A_i - 2)e_i^{local} + A_i(A_i - 2)e_i^{MEC} + A_i(A_i - 1)e_i^{cloud} \tag{14}$$

3.3 MEC Load Balancing Model

To make more efficient use of edge computing resources and enhance system stability, the load balancing issue for MEC must be considered. Load balancing typically refers to the distribution and utilization of tasks across all servers. Therefore, first, the resource utilization rate of the j-th edge node is calculated as follows:

$$rur_j = \begin{cases} \frac{1}{S}\sum_{i=1}^{S}\sum_{j=1}^{M} l_{i,j}, I_j = 1 \\ 0, I_j = 0 \end{cases} \tag{15}$$

In this context, I_j and l_{ij} represent decision variables. If the j-th edge node is occupied, then $I_j = 1$; otherwise, $I_j = 0$. If the i-th task is assigned to the j-th edge node, then $l_{ij} = 1$; otherwise, $l_{ij} = 0$.

The number of occupied edge nodes can be written as:

$$O = \sum_{j=1}^{M} I_j \tag{16}$$

Thus, the average resource utilization rate $Arur$ is given by:

$$Arur = \frac{\sum_{j=1}^{M} rur_j}{O} \tag{17}$$

The load balancing rate indicates the fluctuation of the resource utilization rate around the average resource utilization rate. Therefore, the load balancing rate for a single edge node is given by:

$$lo_j = \begin{cases} \left|Arur - rur_j\right|, I_j = 1 \\ 0, I_j = 0 \end{cases} \tag{18}$$

Subsequently, the average load balancing rate can be written as:

$$Alo = \frac{1}{O}\sum_{j=1}^{M} lo_j \tag{19}$$

Thus, the total load balancing rate L can be written as:

$$L = \frac{Alo}{Arur} \tag{20}$$

The smaller the ratio, the more balanced the load on the edge nodes is.

3.4 Comprehensive Problem

For the entire task offloading problem, the total latency and energy consumption for all vehicular users are as follows:

$$T = \Sigma_{\mathrm{i}=1}^{N} t_i \tag{21}$$

$$E = \Sigma_{\mathrm{i}=1}^{N} e_i \tag{22}$$

Thus, the total cost of task offloading R is:

$$R = mT + nE \tag{23}$$

where m and n represent the latency and energy consumption weight coefficients in the offloading cost [22].

To minimize the total latency, total energy consumption, and total load balancing rate of task offloading, the following multi-objective optimization model is formulated:

$$\begin{aligned}
&\min R, L \\
&s.t. \\
&C1 : w_i \leq w^c,\ \forall \mathrm{i} \in \{1, 2, \ldots, N\} \\
&C2 : A_i \in \{0, 1, 2\}, \forall \mathrm{i} \in \{1, 2, \ldots, N\} \\
&C3 : t_i \leq t_{max},\ \forall \mathrm{i} \in \{1, 2, \ldots, N\}
\end{aligned} \tag{24}$$

Constraint $C1$ indicates that the resources required for task Q must not exceed the maximum number of resources w^c that the server can provide. Constraint $C2$ states that the offloading decision can only be offloaded to the local end, the edge server, or the cloud server. Constraint $C3$ specifies that the time required to complete the task must not exceed the maximum tolerable time for the task.

Based on the above optimization scheme, the offloading cost R and the load balancing rate L are normalized as follows:

$$I(R_k) = \begin{cases} \frac{R_{max}-R}{R_{max}-R_{min}}, & R_{max} - R_{min} \neq 0 \\ 1, & R_{max} - R_{min} = 0 \end{cases} \tag{25}$$

$$I(L_k) = \begin{cases} \frac{L_{max}-L}{L_{max}-L_{min}}, & L_{max} - L_{min} \neq 0 \\ 1, & L_{max} - L_{min} = 0 \end{cases} \tag{26}$$

where R_{max} and R_{min} represent the maximum and minimum values of the offloading cost in the k-th offloading strategy, and L_{max} and L_{min} represent the maximum and minimum values of the load balancing rate in the k-th offloading strategy. From this, the system utility of the k-th offloading strategy can be obtained as:

$$I_k = \alpha I(R_k) + \beta I(L_k) \tag{27}$$

where α and β represent the weight coefficients for the offloading cost and the load balancing rate, respectively.

Further, by comparing the system utility values, the maximum system utility value I_{max} of the algorithm can be obtained.

4 Design of Algorithm

The Non-dominated Sorting Genetic Fireworks Algorithm is a heuristic algorithm that combines the strengths of genetic algorithms and fireworks algorithms, applying it to task offloading in vehicular networks and making further improvements to the original algorithm steps to make it more suitable for solving multi-objective combinatorial optimization problems.

4.1 Initialization Operation

First, initialize the population of vehicular users and edge nodes, and set constant parameters such as the maximum number of iterations and basic mutation probability that will be used later. Each individual in the population represents a set of offloading strategies on a chromosome, where each gene specifies which node a vehicle's computing task should be offloaded to. The position of the gene value corresponds one-to-one with the position of the vehicular user, and the fitness value of the population individuals is calculated.

4.2 Genetic Operations

Fast Non-dominated Sorting. Based on individual fitness, the population is categorized into different non-dominated levels, where solutions within each level are not dominated by any other solutions. Initially, individuals at level 1 are sought out and marked. For those not yet marked, the same procedure is followed to identify and mark individuals at level 2. This process is repeated until all solutions are sorted into their respective non-dominated levels.

Elite Selection. Selection is performed based on the non-dominated sorting levels from low to high, choosing a certain number of the top-level individuals as elite individuals. This number is determined by the population size and usually accounts for only a small fraction of the total population. Elite individuals are not subjected to crossover and mutation operations to ensure that the best individuals in the population are not disrupted or lost. Therefore, the elite individuals are directly copied into the offspring population, and the remaining population members undergo the following crossover and mutation operations.

Multipoint Sectional Crossover. First, multiple crossover points are randomly selected from the chromosomes of the parent generation individuals. These crossover points will determine the number and position of the chromosome segments to be divided. Based on the selected crossover points, two segments are randomly chosen and exchanged between corresponding chromosome segments of two-parent individuals to create new offspring individuals. The crossover operation example is illustrated in Fig. 2, where the second and fourth chromosomes are crossed and swapped.

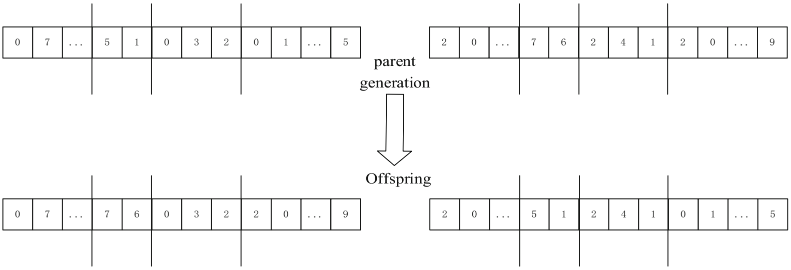


Fig. 2. Cross operation diagram

Adaptive Multi-point Mutation. Multi-point mutation selects multiple gene loci and randomly changes the values at these loci. By dynamically adjusting the mutation rate based on the evolution of the population, if the population diversity is insufficient, the mutation rate can be gradually increased to promote exploration. Conversely, if the algorithm converges too quickly or the population diversity is too high, the mutation rate can be gradually decreased to strengthen exploitation. If the quality of an individual solution is good, its mutation probability is smaller, on the contrary, if the quality is poor, its mutation probability is larger. The updated formula for the mutation probability is as follows:

$$p_{base} = \frac{\partial}{1+\sigma^2} \cdot p_v^0 \tag{28}$$

$$p_v^k = \frac{F_{max} - F_k}{F_{max} - F_{min}} \cdot p_{base} \tag{29}$$

where ∂ represents a constant, p_{base} is the base mutation probability, σ^2 is the variance of the population's fitness, p_v^0 represents the initial mutation probability, F_{max} and F_{min} is the highest and the lowest fitness value in the population, respectively. According to the formula, the base mutation probability will become smaller as the population variance increases, thereby affecting the mutation probability of all individuals in the population. Individuals with lower fitness values in the population will have a higher mutation probability, and vice versa. This effectively prevents individuals with better fitness from being mutated into individuals with worse fitness.

4.3 Fireworks Operation

Explosion Displacement Operation. Randomly initialize the current position of the fireworks $x^c = \left(x_1^c, x_2^c, \cdots, x_K^c\right)$, where the number of explosion particles produced by the k-th firework is:

$$s_k = s_0 \cdot \frac{F_k - F_{min} + \varepsilon}{\sum_{i=1}^{N}(F_k - F_{min}) + \varepsilon} \tag{30}$$

where s_0 is used to limit the number of explosion particles produced by a single firework, and ε is used to prevent the denominator from being zero.

High-quality fireworks individuals typically generate a larger number of explosion particles to explore the search space, while lower-quality fireworks individuals produce fewer explosion particles. Therefore, the limiting formula for the number of explosion particles produced is:

$$\hat{s}_k = \begin{cases} round(as_0), & s < as_0 \\ round(bs_0), & s > bs_0 \\ round(s_k), & otherwise \end{cases} \tag{31}$$

where $\hat{s}_k$ represents the number of explosion particles finally produced by the k-th firework, *round*() denotes the rounding function, and a and b are different constants set for fireworks with lower and higher fitness values, respectively.

The explosion radius produced by the k-th firework is given by:

$$r_k = r_0 \cdot \frac{F_{max} - F_k + \varepsilon}{\sum_{i=1}^{N} (F_{max} - F_k) + \varepsilon} \tag{32}$$

where r_0 is used to adjust the explosion radius generated by a single firework.

A comprehensive displacement operation is performed on the fireworks population, with the new positions being $p^{new} = \left(x_1^{new}, x_2^{new}, \cdots, x_N^{new}\right)$. The calculation formula is as follows:

$$x_t^{new} = x_t^c + \delta \cdot rand(-1,1) \tag{33}$$

where $t = 1,2, \cdots, N$, δ is a fixed step size, and $rand(-1,1)$ represents a random number between -1 and 1. In the N-dimensional vector, a random displacement operation is performed for each dimension, moving to a new position.

Gaussian Mutation Operation and Mapping Rule. Each dimension of all fireworks individuals is subject to Gaussian Mutation, and the position after mutation is $p^{var} = \left(x_1^{var}, x_2^{var}, \cdots, x_N^{var}\right)$. The calculation formula is as follows:

$$x_t^{var} = x_t^{new} + N(0, \sigma) \tag{34}$$

where $t = 1,2, \cdots, N$, $N(0, \sigma)$ denotes a random number from a Gaussian distribution with a mean of 0 and a standard deviation of σ, which can be adjusted, typically with a value of 1.

Fireworks that exceed the boundaries are mapped back into the feasible domain, according to a certain mapping rule. The rule formula is as follows:

$$\hat{x}_t = \begin{cases} x^{max} - rand \cdot \left(x_t^{var} - x^{max}\right), & x_t^{var} > x^{max} \\ x_t^{var}, & x^{min} < x_t^{var} < x^{max} \\ x^{min} + rand \cdot \left(x^{min} - x_t^{var}\right), & x_t^{var} < x^{min} \end{cases} \tag{35}$$

where x^{max} and x^{min} are the upper and lower bounds of the feasible region for the problem, and *rand* represents a random number between 0 and 1. The mapping rule ensures that after mutation, if a firework's position exceeds these bounds, it is adjusted to be within the feasible domain.

Selection Operation. Considering the high dimensionality of the offloading tasks, the Tournament Selection strategy is adopted. By randomly selecting a certain number q of candidate sparks for fitness value comparison, this strategy maintains a low computational complexity and ensures that better solutions have greater competitiveness.

4.4 Time Complexity Analysis

In such combinatorial optimization algorithms, the time complexity of the genetic algorithm and the fireworks algorithm is determined by the fast non-dominated sorting and the resulting explosive particles, respectively. Therefore, the overall time complexity can be obtained as $O\left(2N^2 + KSN\right)$, and the complexity can be balanced by controlling the number of fireworks and explosive particles.

5 Simulation Analysis

5.1 Environment Setup

To verify the superior performance of the proposed algorithm, this paper will use Matlab R2018a as the experimental simulation platform. The experimental scenario considers a three-tier network architecture composed of a cloud server cluster, several edge nodes, and several vehicular users. The parameters are shown in Table 1.

Table 1. Environmental Parameters

Parameter	value
Threshold of Server Resources	1 ~ 8Gb
Gaussian White Noise	3×10^{-10} W
Task Data Volume	100 ~ 1000MB
Transmission Power of OBU	5W
Channel Bandwidth	20MHz
Maximum Tolerable Latency	1 ~ 4s
Computational Capability of OBU	0.8 ~ 1.2GHz
Computational Capability of MEC	3 ~ 5GHz
Computational Capability of the Cloud Server	10GHz
The Power Consumption of OBU	20W
The Power Consumption of MEC	60W
The Power Consumption of the Cloud Server	150W
Initial Mutation Probability	0.5
RSU Coverage Range	100m

5.2 Data Result Analysis

This paper selects the Genetic Algorithm-Particle Swarm Optimization (GA-PSO), Fireworks Algorithm (FWA), Non-dominated Sorting Genetic Algorithm II (NSGA-II), and the Priority Edge Computing scheme (EC) as comparative algorithmic schemes. In this context, EC denotes that tasks are preferentially offloaded to edge servers. The experiment selects evaluation metrics such as latency, energy consumption, load balancing rate, and system utility for comparison.

Latency and Energy Consumption Analysis. Figures 3 and 4 respectively illustrate the relationship between latency and energy consumption of various algorithms under different numbers of tasks. This paper selects the number of tasks as 10, 40, 80, 120, and 160. The latency and energy consumption weight coefficients are set as m = n = 0.5, making both equally important in this environment. As the number of tasks increases, the trend of latency and energy consumption is significantly upward, which is due to the backlog of resources caused by too many tasks, requiring more time and energy to be consumed. Compared with other algorithms, the NSGFA has lower latency and energy consumption, and has advantages in optimizing delay and consumption, making it more suitable for the offloading scenario of multi-user and multi-vehicle tasks.

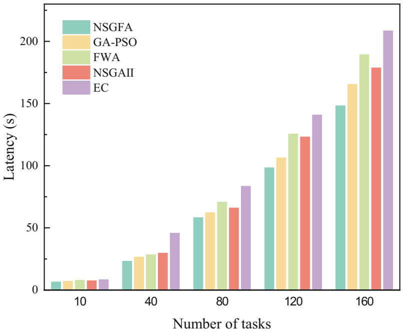


Fig. 3. Comparison of latency under different offloading algorithms

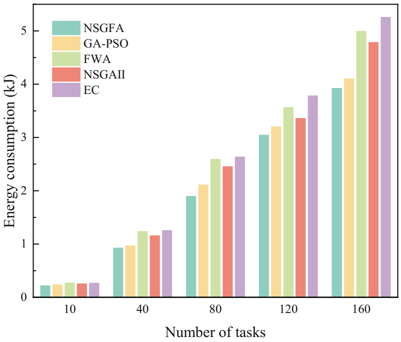


Fig. 4. Comparison of energy consumption under different offloading algorithms

Load Balancing Analysis. Figure 5 demonstrates the relationship between load balancing and the number of tasks for various algorithms. Because EC is set to prioritize edge offloading, the load balancing situation is not taken into account. The NSGFA outperforms other algorithms under all task quantity conditions, but when the number of tasks is high, its optimization capability needs to be strengthened, as achieving load balancing becomes more challenging.

Analysis of Weight Ratio and Task Data Volume. Figure 6 shows the relationship between various algorithm and the offloading cost in the case of different ratio and different task data volume. To investigate the impact of changes in the weight ratio and task data volume on the overall offloading cost, the weight ratio of latency and energy consumption, m/n, are set to 0.25, 0.5, 1, 1.5, and 2, respectively. The task data volumes are set to 200MB, 500MB, and 800MB, with the number of tasks fixed at 120. It can

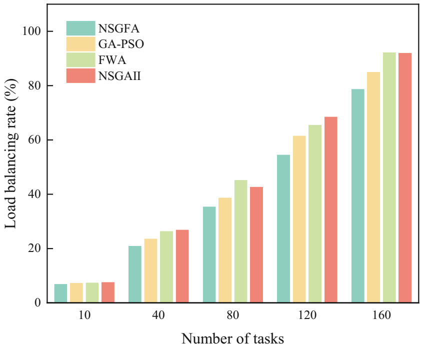


Fig. 5. Comparison of load balancing under different offloading algorithms

be observed that as the weight ratio and task data volume increase, the total offloading cost decreases. Additionally, the rate of change in offloading cost accelerates with the increase in data volume.

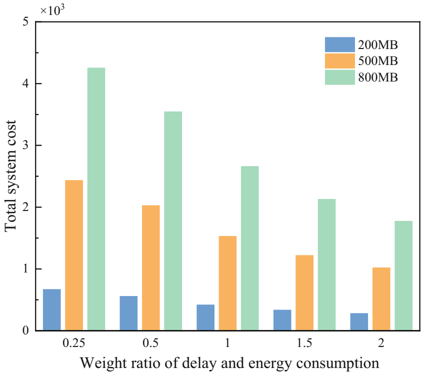


Fig. 6. The influence of different weight ratio and task data volume on offloading cost

Offloading Cost Iterative Analysis. Figure 7 illustrates the iterative convergence of the offloading cost for various algorithms. As the number of iterations increases, the offloading cost for each algorithm gradually decreases. The NSGFA converges around 1200 iterations, with both the number of convergence iterations and the offloading cost results are superior to those of similar hybrid algorithms like GA-PSO. Although the FWA and NSGA-II algorithms have faster convergence speeds, their results are not as satisfactory. The NSGFA's offloading can minimize the total offloading cost.

System Utility Analysis. Figure 8 illustrates the relationship between system utility and the number of tasks for various algorithms. The latency and energy consumption weight coefficients are set as m = n = 0.5, and the offloading cost and load balancing weight coefficients are set as α = β = 0.5. This evaluation metric aims to achieve a balanced relationship between offloading cost and load balancing. Compared to other

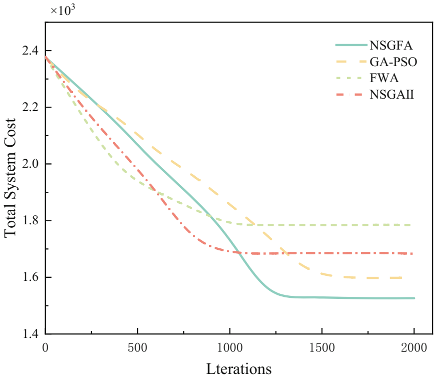


Fig. 7. Comparison of the influence of the number of iterations on the offloading cost under different offloading algorithms

algorithms, the NSGFA consistently demonstrates better system utility, especially when the number of tasks is 40, where the trade-off effect is more pronounced.

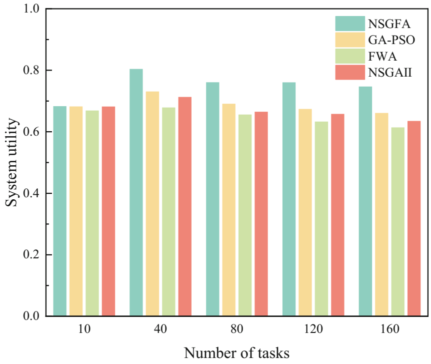


Fig. 8. Comparison of system utility under different offloading algorithms

6 Conclusion

To address the issues of high computational latency, excessive consumption, and poor load balancing in vehicular networks, this paper proposes a computation offloading decision based on the NSGFA. This algorithm effectively combines the Non-dominated Sorting Genetic Algorithm and the Fireworks Algorithm, utilizing the genetic algorithm's local search capability and the fireworks algorithm's global search capability for optimized offloading. Experimental comparisons with other classic algorithms demonstrate that the NSGFA has better optimization in various performance indicators, with a particularly significant improvement in load balancing. The impact of the time-variant dynamic position distances between vehicles and edge nodes on the experiment is the main direction for future research work.

References

1. Ding, F., Zhang, N., Li, S., Bian, Y., Tong, E., & Li, K. Q. (2022). A survey of architecture and key technologies of intelligent connected vehicle-road-cloud cooperation system. *Acta Automatica Sinica*, *48*(12), 2863-2885.
2. Akbar, A., Ibrar, M., Jan, M. A., Wang, L., Shah, N., & Song, H. H. (2023). SeAC: SDN-enabled adaptive clustering technique for social-aware internet of vehicles. *IEEE Transactions on Intelligent Transportation Systems*, *24*(5), 4827-4835.
3. Li, X., Wang, L., Abawajy, J. H., Qin, X., Pau, G., & You, I. (2020). Data-intensive task scheduling for heterogeneous big data analytics in IoT system. *Energies*, *13*(17), 4508.
4. Balasubramanian, V., Otoum, S., & Reisslein, M. (2022). VeNet: hybrid stacked autoencoder learning for cooperative edge intelligence in IoV. *IEEE Transactions on Intelligent Transportation Systems*, *23*(9), 16643-16653.
5. Zhang, Y., Liang, Y., Yin, M., Quan, H., Wang, T., & Jia, W. (2021). Survey on the methods of computation offloading in mobile edge computing. *Chinese journal of computers*, *44*(12), 2406-2430.
6. Sehla, K., Nguyen, T. M. T., Pujolle, G., & Velloso, P. B. (2022). Resource allocation modes in C-V2X: from LTE-V2X to 5G-V2X. *IEEE Internet of Things Journal*, *9*(11), 8291-8314.
7. Dai, P., Hu, K., Wu, X., Xing, H., Teng, F., & Yu, Z. (2020). A probabilistic approach for cooperative computation offloading in MEC-assisted vehicular networks. *IEEE Transactions on Intelligent Transportation Systems*, *23*(2), 899-911.
8. Wang, D., Song, B., Lin, P., Yu, F. R., Du, X., & Guizani, M. (2021). Resource management for edge intelligence (EI)-assisted IoV using quantum-inspired reinforcement learning. *IEEE Internet of Things Journal*, *9*(14), 12588-12600.
9. Raza, S., Wang, S., Ahmed, M., Anwar, M. R., Mirza, M. A., & Khan, W. U. (2021). Task offloading and resource allocation for IoV using 5G NR-V2X communication. *IEEE Internet of Things Journal*, *9*(13), 10397-10410.
10. Chen, M., & Hao, Y. (2018). Task offloading for mobile edge computing in software defined ultra-dense network. *IEEE Journal on Selected Areas in Communications*, *36*(3), 587-597.
11. Jafari, V., & Rezvani, M. H. (2023). Joint optimization of energy consumption and time delay in IoT-fog-cloud computing environments using NSGA-II metaheuristic algorithm. *Journal of Ambient Intelligence and Humanized Computing*, *14*(3), 1675-1698.
12. Wang, K., Wang, X., & Liu, X. (2023). Sustainable Internet of vehicles system: a task offloading strategy based on improved genetic algorithm. *Sustainability*, *15*(9), 7506.
13. Song, S., Ma, S., Yang, L., Zhao, J., Yang, F., & Zhai, L. (2022). Delay-sensitive tasks offloading in multi-access edge computing. *Expert Systems with Applications*, *198*, 116730.
14. Li, A., Li, L., & Yi, S. (2022). Computation offloading strategy for IoT using improved particle swarm algorithm in edge computing. *Wireless Communications and Mobile Computing*, *2022*(1), 9319136.
15. Zhang, B., Zhu, C., Jin, L., & Bi, X. (2023). Task offloading and resource allocation for intersection scenarios in vehicular edge computing. *International Journal of Sensor Networks*, *42*(1), 1-14.
16. Guoyan, L. I., Xiang, X. U. E., Yi, L. I. U., & Yuheng, P. A. N. (2023). Improved TD3 edge computing offloading strategy for software defined networking Internet of vehicles. *Computer Integrated Manufacturing System*, *29*(5), 1627.
17. Alam, M. Z., & Jamalipour, A. (2022). Multi-agent DRL-based Hungarian algorithm (MADRLHA) for task offloading in multi-access edge computing Internet of Vehicles (IoVs). *IEEE Transactions on Wireless Communications*, *21*(9), 7641-7652.
18. Ke, H., Wang, J., Deng, L., Ge, Y., & Wang, H. (2020). Deep reinforcement learning-based adaptive computation offloading for MEC in heterogeneous vehicular networks. *IEEE Transactions on Vehicular Technology*, *69*(7), 7916-7929.

19. Yang, C., Wang, Z., Nie, R., Ding, H., Li, B. (2023). Computing offloading strategy based on sparrow search algorithm in vehicular networks. *Computer Engineering and Design (01), 1-7*.https://doi.org/10.16208/j.issn1000-7024.2023.01.001.
20. Raza, S., Liu, W., Ahmed, M., Anwar, M. R., Mirza, M. A., Sun, Q., & Wang, S. (2020). An efficient task offloading scheme in vehicular edge computing. *Journal of Cloud Computing*, *9*, 1-14.
21. Ning, Z., Dong, P., Kong, X., & Xia, F. (2018). A cooperative partial computation offloading scheme for mobile edge computing enabled Internet of Things. *IEEE Internet of Things Journal*, *6*(3), 4804-4814.
22. Han, D., Chen, W., & Fang, Y. (2018). A dynamic pricing strategy for vehicle assisted mobile edge computing systems. *IEEE Wireless Communications Letters*, *8*(2), 420-423.

BiPA-Net: An Effective Tiny-Object Detection Model in Aerial Images

Tianping Li[1](✉), Mengdi Zhu[1], Zhenyi Zhang[1], and Zhiqiang Yang[2]

[1] School of Physics and Electronics, Shandong Normal University, Jinan 250300, China
sdsdltp@sdnu.edu.cn.com

[2] Assets and Laboratory Management Department, Shandong Normal University, Jinan 250300, China

Abstract. Recently, object detection based Unmanned Aerial Vehicles (UAVs) plays a wide role in many real-life scenarios, such as agriculture and transportation. Although UAVs have been widely used, the detection of small targets is ineffective in conventional object detection networks due to extensive scenes and abundant information. To solve the problems of small object detection, we proposed a novel Bi-directional Extends Feature Pyramid Network (Bi-EFPN), which deepens the depth of Feature Pyramid Network (FPN) and performs better in feature fusion. The Gated Channel and Position Attention Model (GCPA) is introduced to capture more details of small objects in large-sized feature maps. Considering that the Bi-EFPN and GCPA will cost more resources in the neck, we proposed a lightweight detection head with Partial Attention (PAHead), which reduced the computational complexity by approximately 6.8 GFLOPs while slightly improving detection accuracy. Finally, it shows great performance on the visdrone2019 dataset, the mean average precision is improved by 5.1% and the parameters decreased by approximately 10% in contrast to YOLOv8s. Meanwhile, good generalization performance was demonstrated on the Tinyperson dataset and HIT-UAV dataset. The comprehensive experiments indicate that our network has great potential in small object detection.

Keywords: Attention · Small Object Detection · YOLOv8

1 Introduction

Object detection, as an important task in the field of computer vision, aims to determine the location and class of a target [1]. It plays an important role in underwater object detection [2], autonomous driving [3], medical imaging [4] and so on. Therefore, improving object detection is of great practical significance and application value.

UAV, is a remotely piloted aerial system that maneuvered by radio-controlled equipment and is equipped with sophisticated equipment for the purposes of data acquisition, perception, and decision-making [5]. With the advancement of technology, UAVs have gradually developed new functions and methods, and UAV-based object detection is widely used in agricultural [6], traffic [7], urban [8], rescue [9], airline [10]. With the

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 447–463, 2025.
https://doi.org/10.1007/978-3-031-78383-8_30

advent of convolutional neural networks, the field of computer vision has flourished, significant progress has been made in neural network-based object detection over the last decade. YOLO Series [11–14], DETR [15], Transformer [16] and other large models also perform well in target detection. Object detection tasks based on UAV recently became popular. It is difficult to design the target detection network due to the variable flight altitude of UAVs and the great variation in the scale of the object. In addition, the complexity of neural networks increases with the object detection accuracy, and the arithmetic power of UAV systems is difficult to meet the demands of large neural networks. Keeping the balance of speed and accuracy in neural network is becoming a crucial research focus. Although most of the networks above shows advantages in coco dataset, it is difficult for the current network structure to take advantage of the recognition of small target objects.

The target bounding box is defined as:

$$bbox = (x, y, w, h) \tag{1}$$

x and y are the coordinate values of the current object, w and h are the width and height of the selected object.

Without considering rotating objects, we define the mini-objective as:

$$\begin{cases} min(w, h) \geq 4pixel \\ \max(w, h) \leq 32pixel \end{cases} \tag{2}$$

That is, objects with a short side greater than 4 pixels and a long side less than 32 pixels are called small objects. Small objects have problems such as low overlap probability and weak texture features. The traditional network structure has difficulty effectively mentioning the contextual information of small objects, which requires us to design an object detector that is more adaptable to the characteristics of small targets.

Currently, UAV-based small object detection still suffers from the following problems [17]. First, the flight altitude of UAVs is variable, the scale of targets varies greatly, and the acquisition range changes at any time. The higher the flight altitude, the better the image transmission signal, the wider the acquisition range, and the lower the latency. The lower the flight altitude, the more delicate of image features. Second, with the continuous development of computer vision, the accuracy of neural network models has continuously improved, the complexity of computations also increased dramatically. The weight file is becoming increasingly larger, and it is difficult to deploy to the end of the mobile device.

To address the problems of small object detection, we propose a new feature pyramid network and a novel attention model based on YOLOv8s. At the same time, we design a lightweight detection head to further improve the performance of small object detection in aerial images. The main contributions are as follows:

- A new weighted Bi-directional Extends Feature Pyramid Network (Bi-EFPN) is proposed, which extends the range of up-sampling and down-sampling, makes the feature pyramid network more adaptable to the semantic information of small objects. It adequately fuses features without decreasing the detection accuracy of large targets while paying more attention to small objects and generating richer global representations.

- We design a Gated Channel and Position Attention (GCPA). Gating mechanism will effectively promote competitive or synergistic neuronal relationships in channel attention model. The GCPA combines spatial and scaling features, safeguarding the position information of small objects while allocating more resources to channels containing more information.
- We propose a novel detection head to improve the original YOLOv8 network, called Partial Attention Head (PAHead) that not only unifies multiple scales attention but lightens head burden.

2 Related Work

2.1 Baseline

YOLO series of algorithms are widely used in the field of object detection due to the high efficiency and accuracy. There are five versions of YOLOv8 with different depths, which can be classified as YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x. The number of model parameters and the number of computations increase with increasing accuracy. Considering the arithmetic requirements of deploying to the mobile end and balancing real-time performance and accuracy, we adopt YOLOv8s as the baseline.

2.2 Small Object Detection

For the past few years, in the research of small object detection, Zhu [18] et al. proposed a new detection head called TPH, which integrates the convolutional block attention model CBAM to increase attention in both channel and spatial dimensions, enhancing the ability to resist invalid information and focus on valid targets. Yang [19] et al. designed a new progressive feature pyramid network that fuses neighboring features, which is important for the detection of multiscale features. Akyon [20] et al. designed a new network called SAHI, which is a generalized framework that divides the input image into overlapping blocks to output small target objects to regions of larger pixels. Tan [21] et al. proposed a weighted bi-directional feature pyramid fusion network for simple and fast multi-scale feature fusion. Gong [22] et al. proposed the concept of fusion factor, which is adjusted based on the number of distributed objects in each layer. The fusion factor will control the information from shallow layers to deep layers, which is more effective and useful to feature fusion.

3 Methods

3.1 Bi-Directional Extends Feature Pyramid Network

The main role of the FPN is to perform feature fusion to couple feature maps with different receptive field sizes, which enhances the expressive power of the feature maps. Common neck structures include the FPN, Bi-FPN [21], PANet, NAS-FPN [23] etc. For the object detection task, the depth of the neural network is closely related to the ability to extract the semantic information of the target, and the existing pyramid network structure fuses the high-level information located at the top of the pyramid with the low-level information at the bottom of the pyramid through multiple scales of feature layers [24]. However, there are problems such as loss in the fusion and communication of information, whether it is the loss of high-level information to bottom-level information fusion or the degradation of low-level information such as high-level information fusion, which will make feature fusion ineffective [25].

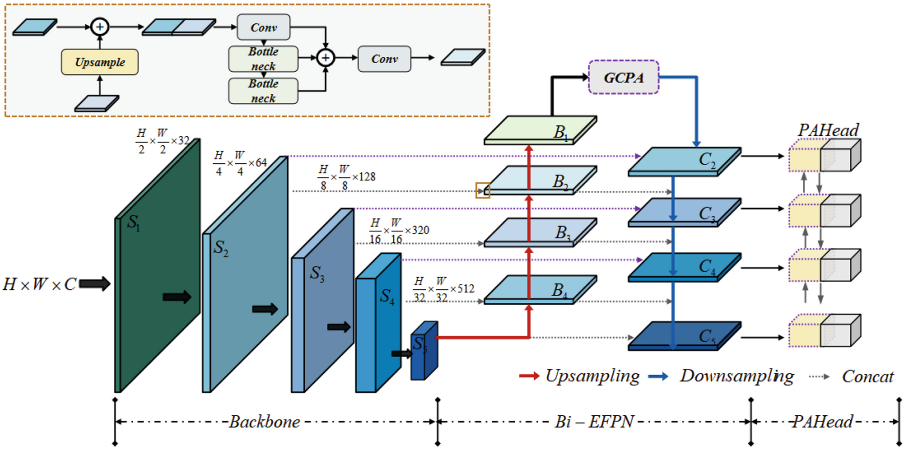

Conv
Upsample
Bottle neck
Bottle neck
Conv
GCPA
$\frac{H}{2}\times\frac{W}{2}\times 32$
$\frac{H}{4}\times\frac{W}{4}\times 64$
$\frac{H}{8}\times\frac{W}{8}\times 128$
$\frac{H}{16}\times\frac{W}{16}\times 320$
$\frac{H}{32}\times\frac{W}{32}\times 512$
$H\times W\times C$
S_1
S_2
S_3
S_4
S_5
B_1
B_2
B_3
B_4
C_2
C_3
C_4
C_5
PAHead
Upsampling
Downsampling
Concat
Backbone
Bi – EFPN
PAHead

Fig. 1. Bi-directional Extends Feature Pyramid Network.

Based on the above problems, we propose an improved Bi-directional Extends Feature Pyramid Network, whose network structure is shown in Fig. 1 S_{1-5} is the backbone, B_{1-4} means the range of up-sampling, C_{2-5} means the range of down-sampling. First, incorporating the idea of cross-scale connectivity of the Bi-FPN [21], an extra edge is added to fuse more features without increasing the cost. Second, compared with traditional PANet, we extend the fusion depth of the feature pyramid network. The underlying layer provides more semantic information and positional information for small targets detection. The S_2 layer has a larger resolution of the feature map, which is more suitable for the detection of small objects. Based on the condition, Bi-EFPN further deepens the depth of feature fusion by continuing the up-sampling operation on top of the B_2 layer to restore the feature map size to 320 × 320.

Given that the input features have different resolutions and non-negligible semantic information, directly integrate features with different scales will bring a large amount of redundant and conflicting information, reducing the ability of multi-scale expression. Therefore, feature fusion plays an important role in the process. The Bi-EFPN has a fixed number of channels and the feature vectors are of the same dimension, which transforms the feature fusion into channel fusion. We compared 4 methods for feature fusion in Table 1. The key of the fusion feature is that the value of the normalized weights falls between [0,1]. It does not change the features of the image itself while accelerating the subsequent network processing.

Table 1. The performance of different fusion modes.

Fusion mode	AP	AP$_{50}$	AP$_{75}$	AP$_{S}$	AP$_{M}$	AP$_{L}$	Parameters	GFLOPs
Bi-FPN [21]	25.4	42.6	25.7	31.4	52.3	59.0	**10592513**	**40.0**
SDI [26]	24.4	40.9	24.4	30.0	52.5	59.2	10739672	41.1
LAF [27]	25.7	42.8	**26.1**	31.5	**53.1**	**62.3**	10743704	41.1
Concat	**25.7**	**42.9**	26.0	**31.8**	52.5	61.2	10764536	41.2

The Table 1 shows Concat has the best performance in feature fusion. The input images got fused features with rich semantic information through FPN, fully utilize the high-resolution features of shallow layers and the high-level semantic information of deep layers. The final goal is to achieve the effect of prediction through these different layers of features.

3.2 Gated Channel and Position Attention Mechanism

Attention plays an important role in human perception, and the human visual system is able to selectively capture salient parts of a scene to obtain more detailed information about the target of attention. In machine learning, the attention mechanism guides the computational resources to bias toward the part of the input signal that contains the most information, and from the earliest proposal of the attention mechanism by Bahdanau et al. in 2014, the attention mechanism has been widely used in various fields, such as natural language processing, speech recognition, and machine translation, which has greatly improved the performance of many tasks.

To extract the information of different channels while retaining the small object position information of the feature map, we propose a Gated Channel and Position Attention (GCPA). The structure is shown in Fig. 2.

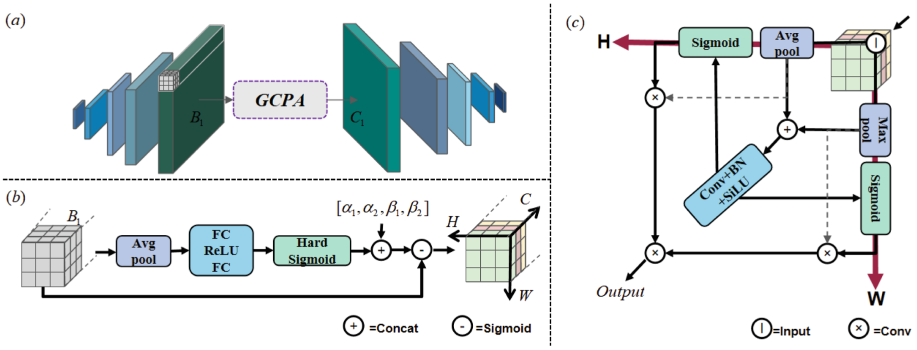


Fig. 2. The architecture of Gated Channel and Position Attention (a) represents the position of the GCPA module in the FPN (b) represents the gated channel attention mechanism (c) the position attention mechanism.

The input feature $F \in R^{H \times W \times C}$ is the output of Bi-EFPN up-sampling, B_1 is the input of channel attention, and the output result of channel attention is used as the input of positional attention. The plug-and-play property is guaranteed and can be inserted into the network structure at any time.

Gated Channel Attention. To enable the network to selectively learn the importance of different channel features and enhance informative features, we design a gated channel attention mechanism that can be used as a gating mechanism to dynamically switch channels on and off. First, the space of the feature map is compressed to predict the importance of each channel, and the dimension can be compressed from H × W × C to 1 × 1 × C by global average pooling [28]. Then, the compressed feature vectors are transformed into weight vectors to motivate the corresponding channels of the previous feature map by two fully connected layers and a normalization layer [29]. Finally, the features are weighted so that the output is fixed in the interval of [-1,1].

$$\xi_C(F) \cdot F = max(\alpha^1(F) \cdot F_c + \beta^1(F), \alpha^2(F) \cdot F_c + \beta^2(F)) \tag{3}$$

F_c is the feature of channel c and $[\alpha^1, \alpha^2, \beta^1, \beta^2]^T = \theta(\cdot)$ is the hyperparameter which controls the threshold of the activation function ReLU. Dynamic ReLU, whose parameters depend on all inputs, is a dynamic segmented linear function whose encodes the context as a hyperparameter. Given an input vector X, dynamic activation is defined as $f_{\theta(x)}$, $\theta(x)$ is a learnable parameter that computes the parameters for the activation function, and the static ReLU is defined as:

$$y = max\{x, 0\} \tag{4}$$

X is the Input Vector.

The output of channel c is:

$$y_c = max\{x_c, 0\} \tag{5}$$

x_c is the input to channel c.

The maximum DY-ReLU of the K linear functions is defined as:

$$y_c = f_{\theta(x)}(x_c) = \max_{1 \le k \le K}\left\{a_c^k x_c + b_c^k\right\} \tag{6}$$

The linear coefficients α and β are the outputs of $\theta(x)$.

Position Attention. Positional attention focuses the model on spatial contextual information by attentively weighting the features in each channel. The output from the channel attention with A_1 forms the input. First, the input feature map is divided into two parts according to width and height, processed separately in the axes p_w and p_h in both directions, and finally, the output is merged.

$$p_w = \frac{1}{H} \sum_{0 \le j \le H} E(w, j) \tag{7}$$

$$p_h = \frac{1}{W} \sum_{0 \le i \le W} E(i, h) \tag{8}$$

w and h are the width and height of the input feature map, E (w, j) and E (i, h) are the values of the input feature map position (i, j).

$$P(a_w, a_h) = Conv\left[Concat(p_w, p_h)\right] \tag{9}$$

$P(a_w, a_h)$ represents the output of positional attention coordinates, conv is the convolution of 1×1, and concat denotes the cascade.

$$s_w = Split(a_w) \tag{10}$$

$$s_h = Split(a_h) \tag{11}$$

s_w and s_h are the width and height of the split output, respectively.

Final GCPA output:

$$F_{GCPA} = E \times s_w \times s_h \tag{12}$$

The E represents the weight of the channel and positional attention.

3.3 Partial Attention Head

PAHead is a detection head combined with attention, and the network structure is shown in Fig. 3 and the details shows in Fig. 4. It transforms the input data in head into a three-dimensional tensor on the layer, spatial, and channel dimensions, deploying the attention mechanism in $H \times W \times C$ three dimensions.

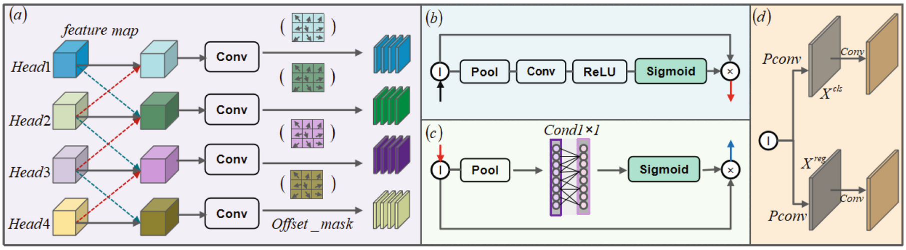


Fig. 3. The architecture of Partial Attention Head. (a) represents the spatial attention (b) represents the layer attention (c) represents the channel attention (d) represents the partial convolution.

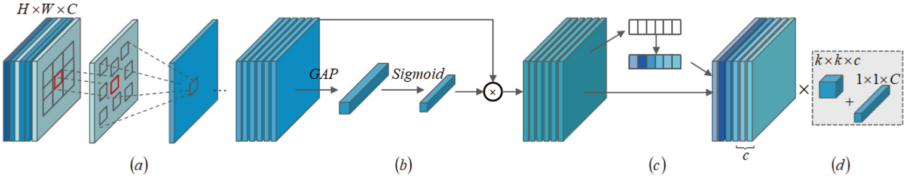


Fig. 4. The details of Partial Attention Head. (a) represents the spatial attention (b) represents the layer attention (c) represents the channel attention (d) represents the partial convolution.

Part (a) is deployed on the spatial dimension (S = H × W) to learn a coherent representation in spatial locations. Part (b) is layer-aware attention to learn the level of semantic hierarchy to facilitate the enhancement or degradation of an object's features at the appropriate scale to adjust its importance. Module (c) is deployed on channels to follow different convolutional kernel reactions to guide different tasks, such as classification, regression, and key-point learning. Part (d) means the partial convolution. In this way, we explicitly implement a unified attention mechanism for the detection head.

Given a tensor $F \in R^{S \times L \times C}$, the formula for self-attention is $W(F) = \pi(F) \cdot F$, where $\pi(\cdot)$ is an attention function, the formula for deploying the attention mechanism in each of the three dimensions is:

$$W(F) = \pi_C(\pi_L(\pi_S(F) \cdot F) \cdot F) \cdot F \tag{13}$$

where $\pi_S(\cdot)\pi_L(\cdot)\pi_C(\cdot)$ are the attention mechanisms applied to the scale, spatial, and channel dimensions, respectively.

Spatial Attention. We wish to improve the discrimination of spatial locations of different objects by fusing spatial attention, which cares about the weights of each location in the plane, focusing on the valid information on the feature map is where. The expansion capability of deformable convolution [30–33] can help us aggregate multiple feature levels together. Attention is first expanded by deformable convolution, and then features are integrated across layers.

$$\pi_S(F) \cdot F = \frac{1}{L}\sum_{l=1}^{L}\sum_{k=1}^{K}\omega_{l,k} \cdot F(l; p_k + \Delta p_k; c) \cdot \Delta m_k \tag{14}$$

K is the number of sparsely sampled positions, $p_k + \Delta p_k$ is an offset self-learning spatial position parameter, deformable convolution introduces an offset for each point Δp_k, Δp_k discriminates the offset region, and Δm_k is the learned position parameter at position p_k.

Layer Attention. Layer perception is an important part of object detection, on the one hand, there are both small and big objects in the same image, on the other hand, there are different goals for object classification and localization. In order to focus on different tasks, we propose a layer-attention mechanism, which enables task decomposition by dynamically computing the features of different tasks between layers allowing for adaptive input of the importance of each feature level.

Global average pooling is first applied to the inputs, and two fully connected layers are used to generate weights along with a nonlinear sigmoid function. The two fully connected layers allow the different layers to interact with each other, capture dependencies, and control model complexity through dimensionality reduction.

$$\pi_L(F) \cdot F = \sigma(fc_2(\delta(fc_1(F)))) \cdot F \tag{15}$$

fc is a fully connected layer function, $\sigma(x) = \frac{1}{1+e^{-x}}$ is a sigmoid function, $\delta(x) = \max(x, 0)$ is a ReLU function.

Channel Attention. To enable the network to selectively learn the importance of different channel features to enhance the informative features, we use channel attention π_c. Channel attention concerns the weight of each feature surface. First, the space of the feature map is squeezed, and the dimension can be compressed from H × W × C to 1 × 1 × C by global average pooling to obtain the global features at the channel level. Then, the global features are subjected to an excitation operation to learn the relationship between each channel and predict the importance of each channel. The compressed feature vector is transformed into a weight vector to motivate the corresponding channel of the previous feature map, and finally, the features are weighted so that the output result is fixed in the interval [-1,1]. Essentially, it is an attention operation in the channel dimension; this operation allows the model to focus more on informative channel features and suppress useless channel information.

$$\pi_C(F) \cdot F = \sigma(\delta(fc(F) \cdot F) \cdot F \tag{16}$$

$\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid activation function, and $\delta(x) = \max(x, 0)$ is the ReLU activation function.

Squeeze. The input feature signal $H \times W \times C$ is output $1 \times 1 \times C$ by the squeeze operation to achieve the mapping of spatial information to channel information so that it can be utilized by its input layer, and the statistics for each channel are generated by the average pooling operation.

Excitation. To reduce the model complexity and improve the generalization ability, the gated mechanism adopts a bottleneck structure containing two fully connected layers. The first FC plays the role of dimensionality reduction, the dimensionality reduction coefficient r is a hyperparameter, which reduces the dimensionality to $\frac{1}{r}$, ReLU activation is used, and the FC recovers the original dimensionality. Finally, the sigmoid function is the weight coefficient of the individual channels, which is multiplied by the original features and multiplied to obtain the correlation of each channel, which makes the model more discriminative of each channel's features and a discriminative mechanism.

Partial Convolution. Considering different channels share high similarities, partial convolution can reduce cost of computational redundancy. It simply applies a regular convolution on only one-quarter of the input channels and keep the remaining channels same. Compared to the regular convolution, partial conv can significantly reduce the computational effort of the network and improve the shortcomings of deformable convolution while retaining accuracy.

4 Experiments

4.1 Datasets

The Visdrone2019 dataset [34] is a UAVs dataset with a total of 8629 photos. Among them, 6471 photos are used as the training set, 548 photos are used as the validator, and 1610 photos are used as the test set. The dataset contains 10 categories from everyday scenarios: pedestrians, people, bicycles, cars, vans, trucks, tricycles, awning tricycles, buses and motors. The Tinyperson [35] dataset is proposed in 2019, captured from HD video on the Internet. The dataset contains 1532 images with 72651 comments totally, 1225 photos for training, 153 images in the validator, and 154 images in the test dataset. There are two categories to be tested in the miniaturization detection task, one is earth_person, the other is sea_person. The HIT-UAV [36] is a high-altitude infrared small target detection dataset with 2898 images. There are 2040 images in training set, 287 images in validation set, and 571 images for test.

4.2 Implementation Details

Experiments are implemented on an NVIDIA GeForce RTX 3060 (16G) GPU with Pytorch 1.13.1, Python 3.8, and CUDA 11.6 dependency. The input image size is 640 × 640, and the batch size of the training data volume is 4. The training lasts for 100 cycles. Stochastic Gradient Descent (SGD) is used as the optimization function, and mosaic enhancement is turned off for the last 10 rounds.

4.3 Ablation Study

Table 2. The accuracy of different backbones.

Model	AP	AP_{50}	AP_S	AP_M	AP_L	Params(M)	GFLOPs
EfficientViT [37]	14.0	25.2	16.0	39.5	51.6	4.0	9.4
Resnet18 [38]	18.3	32.1	20.4	47.4	52.2	17.9	46.3
Resnet50 [38]	17.9	31.2	20.6	46.8	53.1	28.5	75.6
Swin-transformer [39]	19.5	34.3	22.8	48.6	53.5	34.7	90.5
Fasternet [40]	17.9	31.4	21.4	46.1	51.5	8.6	21.7
YOLOv5s	19.7	33.7	22.8	48.1	52.6	9.1	23.8
YOLOv8n	21.8	37.2	27.2	48.9	55.6	3.0	12.2
YOLOv8s	**22.7**	**38.5**	**25.7**	**51.7**	**60.8**	11.1	28.5

Backbone. The Table 2 compares the effects of YOLOv8 and other backbone and shows that YOLOv8 has excellent performance in the field of object detection. Compared with other classical backbones, YOLOv8 shows advantages in both accuracy and parameters.

The receptive field is the range of the input image that can be seen by a feature point in the convolutional neural network. The Effective Receptive Field (ERF) of the 8 backbones is shown in Fig. 5. A wider distribution of the dark area indicates a larger ERF.

The performance of Bi-EFPN and GCPA. The comparison of FPN shown on Table 3. The YOLOv8 backbone was retained, and FPN was replaced with other modules for comparison with the ablation experiments. On the Visdrone2019 dataset, when the input image size is 640 × 640, the Bi-EFPN improves the AP performance by 2.5% and 1.3% compared with the YOLOv8s and YOLOv8s-P2, respectively. With the addition of GCPA, it achieves a better performance in terms of detection accuracy, with 25.8 (AP), 43.1 (AP_{50}), 26.1 (AP_{75}), 32.2 (AP_S), 52.8 (AP_M), and 62.9 (AP_L) achieved the optimum. It is worth noting that since the depth of feature fusion is particularly enhanced when constructing the FPN, as the number of layers increases, the number of parameters inevitably increases while the global information is fused at different levels. Although Bi-EFPN is poor in large object detection, we compensated the shortcomings through the GCPA mechanism. Finally, the experiment proves that our network is not at all inferior to other popular FPNs.

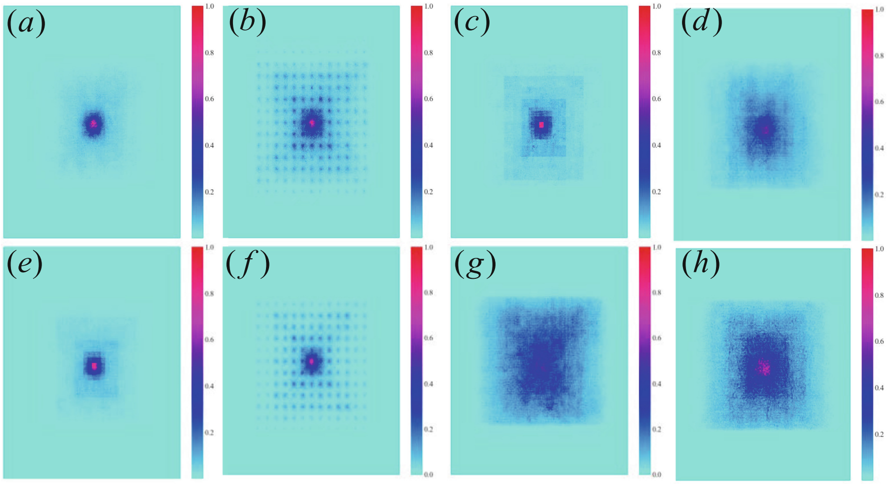


Fig. 5. The ERF of (a) EfficientViT; (b) Resnet18; (c) Swin-transformer; (d) YOLOv8n; (e) Fasternet; (f) Resnet50; (g) YOLOv5s; (h) YOLOv8s.

Table 3. The comparison experiments with other FPN based on YOLOv8s.

Model	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Params(M)	GFLOPs
ASF [41]	20.8	35.5	21.2	24.0	49.6	54.2	11.3	30.1
Slimneck [42]	21.4	36.7	21.2	24.6	50.3	61.3	10.3	25.1
Gold-yolo [27]	21.5	36.9	21.4	25.1	50.9	59.3	13.6	29.9
GFPN [43]	20.4	34.6	20.4	23.8	49.1	53.7	12.1	29.3
EfficientRepBiPAN [44]	21.1	35.9	21.1	24.2	50.4	59.8	10.2	25.6
AFPN [45]	20.5	34.7	20.5	23.8	50.1	59.0	8.9	25.1
Bi-FPN [21]	21.8	37.0	22.1	24.9	51.7	**62.8**	7.4	25.0
YOLOv8s	22.7	38.5	22.7	25.7	51.7	60.8	11.1	28.5
YOLOv8s-P2	24.2	41.2	24.2	29.7	51.4	58.4	10.6	36.7
Bi-EFPN	**25.5**	**42.5**	**25.5**	**31.2**	**52.7**	60.1	10.7	40.6
Bi-EFPN + GCPA	**25.8**	**43.1**	**26.1**	**32.2**	**52.8**	**62.9**	10.8	41.3

Head. We tested the detection head separately to explore the role of each detection head in YOLOv8. According to Table 4, we can see the efficiency in small objects detection. P3 layer has the strongest expression ability, coupled with richer information. We conclude that the layer containing more location information is more beneficial for small object detection.

Table 4. Effectiveness of different head of YOLOv8s.

P3	P4	P5	MAP50	MAP50–95	P3	P4	P5	MAP50	MAP50–95
×	×	×	34.3	20.1	✓	✓	×	40.6	24.4
✓	×	×	39.9	23.9	✓	×	✓	40.1	24.0
×	✓	×	32.3	20.0	×	✓	✓	33.3	20.5
×	×	✓	20.1	13.4	✓	✓	✓	**40.6**	**24.3**

Ablation experiments. In ablation experiments, we show the performances of Bi-EFPN, GCPA, PAHead in Table 5. From the results, the Bi-EFPN has the great improvement in AP of the baseline. The GCPA has significant improvement in small object detection with minor computer resources cost. To reduce the GFLOPs while maintaining accuracy, we design a lightweighted detector PAHead, which bring 6.8 GFLOPs decrease in computational volume.

Table 5. Ablation experiments of the main components on the Visdrone2019 datasets.

Bi-EFPN	GCPA	PAHead	AP	AP_S	AP_M	AP_L	Parameters	GFLOPs
-	-	-	22.7	25.7	51.7	60.8	11129454	**28.5**
√			25.5	31.2	52.7	60.1	10678520	40.6
√	√		25.8	32.2	52.8	**62.9**	10764735	41.3
√	√	√	**26.4**	**32.8**	**54.1**	61.2	**10101861**	34.5

In order to show the difference between the results before and after the improvement, we listed 4 sets of heat maps in Fig. 6. Heat maps can better visualize the results more intuitively to represent the area of attention.

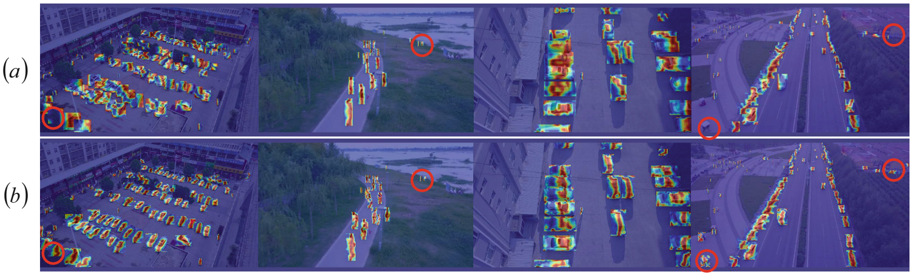


Fig. 6. The heat maps of (a)YOLOv8; (b) BiPA-Net on the Visdrone2019 dataset.

The Table 6 shows that the Bi-EFPN, GCPA, and PAHead have achieved a cascading improvement in detection accuracy. We compare the different categories individually and find that our network pays more attention to small targets. The people show 9.6% increase while bicycle shows 9% increase in AP metrics. For large targets such as bus and van, the improvement is 2.9% and 6.3% respectively, proves that the network is able to balance the detection of small and large target.

Table 6. Major types performance tests on the Visdrone2019 datasets.

Model	MAP50	Pedestrian	People	Bicycle	Car	Van	Truck	Tricycle	Awn	Bus	Motor
YOLOv8s	40.6	43.4	33.6	15.4	79.6	44.8	38.5	29.1	17.6	58.5	45.0
+ Bi-EFPN	45.0	52.9	42.0	16.6	85.0	49.7	39.1	33.2	17.3	62.8	51.7
+ Bi-EFPN + GCPA	45.2	52.4	41.9	17.3	84.9	50.6	39.2	31.7	19.7	61.9	52.2
+ Bi-EFPN + GCPA + PAHead	45.7	53.0	42.6	17.4	85.3	51.1	41.7	32.5	18.0	61.4	53.6

Generalization Performance. The generalization performance of a network is the performance in another dataset that never seen before. Tinyperson dataset is popular in the challenge of tiny target detection. To verify that our designed network has the same improvement effect on small object detection on other datasets, we chose the Tinyperson dataset for comparison. The Fig. 7 shows a comparison of detection results between our network and baseline on the Visdrone2019 dataset and Tinyperson dataset. The red box means false predicted, blue box means false negative, green box means true predicted. The Table 7 shows the experiments of comparative experiments.

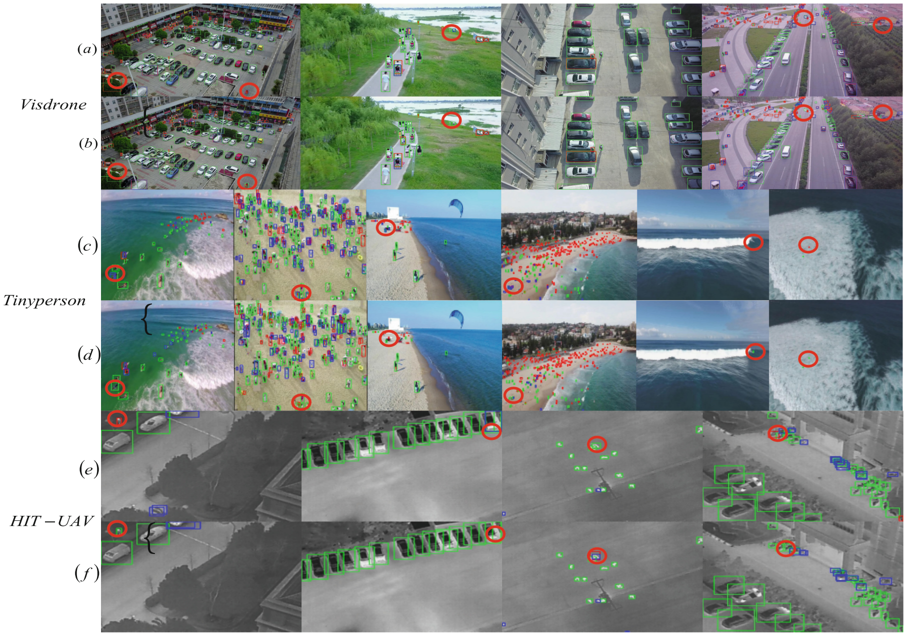


Fig. 7. Visualization of results. (a)(c)(e) Results of YOLOv8s; (b)(d)(f) Results of our network.

Table 7. Comparative experiments on different datasets.

Dataset	Model	MAP50	MAP50–95	FPS	FLOPs(G)	Params(M)
	YOLOv8n	33.9	19.4	286	8.2	3.0
	YOLOv8s	40.6	24.2	238	28.5	11.1
Visdrone	YOLOv10n[14]	32.6	19.0	285	8.2	2.6
	YOLOv10s [14]	38.2	0.23	167	24.8	8.1
	BiPA-Net(ours)	45.7	28.1	76	34.5	10.1
	YOLOv8s	29.9	10.0	196	28.4	11.2
Tinyperson	YOLOv10s [14]	26.7	8.33	67.6	24.4	8.1
	BiPA-Net(ours)	35.2	11.6	78	34.4	10.1
HIT-UAV	YOLOv8s	81.1	53.2	159	28.4	11.1
	BiPA-Net(ours)	83.2	54.1	75	34.4	10.1

5 Conclusion

We proposed Bi-EFPN, a feature pyramid network more suitable for small object detection, which combines GCPA mechanisms and spatial and layer features. The Bi-EFPN network expands the depths of the up-sampling and down-sampling, integrating backbone with more primitive information into FPN, and put the residual network into use. The GCPA further exploits the small object position information while taking into account the characteristics of large targets. PAHead unifies the spatial, layer, and channel attention to improve the performance and robustness in the head. It is demonstrated through a large number of experiments that our model can recognize a large number of small objects, which is a significant advantage over other mainstream models. In addition, ablation experiments demonstrate the effectiveness of each module, providing a basis for further improvement.

References

1. Lu, L.P., Li, H.S., Ding, Z., Guo, Q.M.: An improved target detection method based on multiscale features fusion. Microw. Opt. Technol. Lett. 62, 3051-3059 (2020)
2. Wang, G.C., Gao, J., Chen, Y.B., Wang, X., Li, J.T., Chew, K.H., Chen, R.P.: Polarization-Enhanced Underwater Detection Method for Multiple Material Targets Based on Deep-Learning. IEEE Photonics J. 15, 6 (2023)
3. Xu, H.X., Lai, S.N., Li, X.Y., Yang, Y.: Cross-domain car detection model with integrated convolutional block attention mechanism. Image Vis. Comput. 140, 14 (2023)
4. Kalra, M., Bouguila, N., Fan, W.T.: Online variational learning of finite invertedBeta-Liouvillemixture model for biomedical analysis. Int. J. Imaging Syst. Technol. 30, 794-814 (2020)
5. Wang, Q., Yang, P.F., Yan, X., Wu, H.C., He, L.: Radio Frequency-Based UAV Sensing Using Novel Hybrid Lightweight Learning Network. IEEE Sens. J. 24, 4841-4850 (2024)
6. Norhashim, N., Kamal, N.L.M., Shah, S.A., Sahwee, Z., Ruzani, A.I.A.: A Review of Unmanned Aerial Vehicle Technology Adoption for Precision Agriculture in Malaysia. Unmanned Syst. 19 (2023)
7. Zhang, G.J.: 6G enabled UAV traffic management models using deep learning algorithms. Wirel. Netw. 11 (2023)
8. Kharchenko, V., Kliushnikov, I., Rucinski, A., Fesenko, H., Illiashenko, O.: UAV Fleet as a Dependable Service for Smart Cities: Model-Based Assessment and Application. Smart Cities 5, 1151-1178 (2022)
9. Zhang, M., Li, W., Wang, M.M., Li, S.R., Li, B.Q.: Helicopter-UAVs search and rescue task allocation considering UAVs operating environment and performance. Comput. Ind. Eng. 167, 20 (2022)
10. Fan, B.K., Li, Y., Zhang, R.Y., Fu, Q.Q.: Review on the Technological Development and Application of UAV Systems. Chin. J. Electron. 29, 199-207 (2020)
11. Bochkovskiy, A., Wang, C.Y., Liao, H.Y.M.: YOLOv4: Optimal Speed and Accuracy of Object Detection. (2020)
12. 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. arXiv e-prints (2022)
13. Wang, C.-Y., Yeh, I.-H., Liao, H.: YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information. ArXiv abs/2402.13616, (2024)
14. Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., Ding, G.: YOLOv10: Real-Time End-to-End Object Detection. ArXiv abs/2405.14458, (2024)
15. Nicolas Carion*, F.M., , G.S., Nicolas Usunier,Alexander Kirillov, and Sergey Zagoruyko, AI, F.: End-to-End Object Detection with Transformers. (2020)
16. Ashish Vaswani, N.S., Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin: Attention Is All You Need. (2017)
17. Liu, M.J., Wang, X.H., Zhou, A.J., Fu, X.Y., Ma, Y.W., Piao, C.H.: UAV-YOLO: Small Object Detection on Unmanned Aerial Vehicle Perspective. Sensors 20, 12 (2020)

18. Zhu, X., Lyu, S., Wang, X., Zhao, Q.: TPH-YOLOv5: Improved YOLOv5 Based on Transformer Prediction Head for Object Detection on Drone-captured Scenarios. (2021)
19. G. Yang, J.L., Z. Zhu, S. Cheng, Z. Feng and R. Liang: AFPN: Asymptotic Feature Pyramid Network for Object Detection. 2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC) (2023)
20. Akyon, F.C., Altinuc, S.O., Temİzel, A.: Slicing Aided Hyper Inference and Fine-Tuning for Small Object Detection. 2022 IEEE International Conference on Image Processing (ICIP) 966–970 (2022)
21. Tan, M., Pang, R., Le, Q.V.: EfficientDet: Scalable and Efficient Object Detection. (2019)
22. Gong, Y., Yu, X., Ding, Y., Peng, X., Zhao, J., Han, Z.: Effective Fusion Factor in FPN for Tiny Object Detection. In: IEEE Winter Conference on Applications of Computer Vision. (2021)
23. Ghiasi, G., Lin, T.-Y., Pang, R., Le, Q.V.: NAS-FPN: Learning Scalable Feature Pyramid Architecture for Object Detection. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 7029–7038 (2019)
24. Nguyen, D., Mejri, N., Singh, I.P., Kuleshova, P., Astrid, M., Kacem, A., Ghorbel, E., Aouada, D.: LAA-Net: Localized Artifact Attention Network for Quality-Agnostic and Generalizable Deepfake Detection. (2024)
25. Zhang, Y., Hsieh, J.-W., Lee, C.-C., Fan, K.-C.: SFPN: Synthetic FPN for Object Detection. 2022 IEEE International Conference on Image Processing (ICIP) 1316–1320 (2022)
26. Xu, S., Wang, X., Lv, W., Chang, Q., Cui, C., Deng, K., Wang, G., Dang, Q., Wei, S., Du, Y.: PP-YOLOE: An evolved version of YOLO. (2022)
27. Wang, C., He, W., Nie, Y., Guo, J., Liu, C., Han, K., Wang, Y.: Gold-YOLO: Efficient Object Detector via Gather-and-Distribute Mechanism. ArXiv abs/2309.11331, (2023)
28. Hu, J., Shen, L., Sun, G., Albanie, S.: Squeeze-and-Excitation Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence PP, (2017)
29. Wang, Q., Wu, B., Zhu, P., Li, P., Hu, Q.: ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). (2020)
30. Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., Wei, Y.: Deformable Convolutional Networks. 2017 IEEE International Conference on Computer Vision (ICCV) 764–773 (2017)
31. Zhu, X., Hu, H., Lin, S., Dai, J.: Deformable ConvNets V2: More Deformable, Better Results. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 9300–9308 (2018)
32. Wang, W., Dai, J., Chen, Z., Huang, Z., Li, Z., Zhu, X., Hu, X.-h., Lu, T., Lu, L., Li, H., Wang, X., Qiao, Y.: InternImage: Exploring Large-Scale Vision Foundation Models with Deformable Convolutions. 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 14408–14419 (2022)
33. Xiong, Y., Li, Z., Chen, Y., Wang, F., Zhu, X., Luo, J., Wang, W., Lu, T., Li, H., Qiao, Y., Lu, L., Zhou, J., Dai, J.: Efficient Deformable ConvNets: Rethinking Dynamic and Sparse Operator for Vision Applications. ArXiv abs/2401.06197, (2024)
34. Du, D., Zhu, P., Wen, L., Bian, X., Liu, Z.M.: VisDrone-DET2019: The Vision Meets Drone Object Detection in Image Challenge Results. In: ICCV visdrone workshop. (2019)
35. Yu, X., Gong, Y., Jiang, N., Ye, Q., Han, Z.: Scale Match for Tiny Person Detection. 2020 IEEE Winter Conference on Applications of Computer Vision (WACV) 1246–1254 (2019)
36. Suo, J., Wang, T.-M., Zhang, X., Chen, H.-m., Zhou, W., Shi, W.: HIT-UAV: A high-altitude infrared thermal dataset for Unmanned Aerial Vehicle-based object detection. Scientific Data 10, (2022)
37. Cai, H., Li, J., Hu, M., Gan, C., Han, S.: EfficientViT: Multi-Scale Linear Attention for High-Resolution Dense Prediction. (2022)

38. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. IEEE (2016)
39. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. 2021 IEEE/CVF International Conference on Computer Vision (ICCV) 9992–10002 (2021)
40. Chen, J., Kao, S.-h., He, H., Zhuo, W., Wen, S., Lee, C.-H., Chan, S.-H.G.: Run, Don't Walk: Chasing Higher FLOPS for Faster Neural Networks. 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 12021–12031 (2023)
41. Kang, M., Ting, C.-M., Ting, F.F., Phan, R.C.-W.: ASF-YOLO: A Novel YOLO Model with Attentional Scale Sequence Fusion for Cell Instance Segmentation. ArXiv abs/2312.06458, (2023)
42. Li, H., Li, J., Wei, H., Liu, Z., Zhan, Z., Ren, Q.: Slim-neck by GSConv: A better design paradigm of detector architectures for autonomous vehicles. ArXiv abs/2206.02424, (2022)
43. Jiang, Y., Tan, Z., Wang, J., Sun, X., Lin, M., Li, H.: GiraffeDet: A Heavy-Neck Paradigm for Object Detection. ArXiv abs/2202.04256, (2022)
44. Li, C., Li, L., Jiang, H., Weng, K., Geng, Y., Li, L., Ke, Z., Li, Q., Cheng, M., Nie, W., Li, Y., Zhang, B., Liang, Y., Zhou, L., Xu, X., Chu, X., Wei, X., Wei, X.: YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications. ArXiv abs/2209.02976, (2022)
45. Yang, G., Lei, J., Zhu, Z., Cheng, S., Feng, Z., Liang, R.: AFPN: Asymptotic Feature Pyramid Network for Object Detection. 2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC) 2184–2189 (2023)

Author Index

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15324, pp. 465–466, 2025.
https://doi.org/10.1007/978-3-031-78383-8