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

LNCS 15304

Pattern Recognition

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

Part IV

ICPR 2024 INDIA

IAPR

Springer

MOREMEDIA

Lecture Notes in Computer Science 15304

Founding Editors

Gerhard Goos
Juris Hartmanis

Editorial Board Members

Elisa Bertino, *Purdue University, West Lafayette, IN, USA*
Wen Gao, *Peking University, Beijing, China*
Bernhard Steffen, *TU Dortmund University, Dortmund, Germany*
Moti Yung, *Columbia University, New York, NY, USA*

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 IV

Springer

Editors
Apostolos Antonacopoulos
University of Salford
Salford, UK

Rama Chellappa
Johns Hopkins University
Baltimore, MD, USA

Saumik Bhattacharya
IIT Kharagpur
Kharagpur, India

Subhasis Chaudhuri
Indian Institute of Technology Bombay
Mumbai, India

Cheng-Lin Liu
Chinese Academy of Sciences
Beijing, China

Umapada Pal
Indian Statistical Institute Kolkata
Kolkata, India

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-031-78127-8 ISBN 978-3-031-78128-5 (eBook)
https://doi.org/10.1007/978-3-031-78128-5

© The Editor(s) (if applicable) and The Author(s), under exclusive license
to Springer Nature Switzerland AG 2025

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
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 IV

DeepEMD: A Transformer-Based Fast Estimation of the Earth Mover's Distance 1
Atul Kumar Sinha and François Fleuret

Equivariant Neural Networks for TEM Virus Images Improves Data Efficiency 16
Karl Bylander, Ingela Nyström, and Karl Bengtsson Bernander

AI Based Story Generation 32
Neha Sharma, Prashant Karwasra, Paritosh Sharma, and Md Arbaz Tahir

Deep Learning Models for Inference on Compressed Signals with Known or Unknown Measurement Matrix 48
Huiyuan Yu and Maggie Cheng

Training Point-Based Deep Learning Networks for Forest Segmentation with Synthetic Data 64
Francisco Raverta Capua, Juan Schandin, and Pablo De Cristóforis

Brain Age Estimation of Alzheimer's and Parkinson's Affected Individuals Using Self-Attention Based Convolutional Neural Network 81
Raveendra Pilli, Tripti Goel, R. Murugan, and M. Tanveer

IFSENet: Harnessing Sparse Iterations for Interactive Few-Shot Segmentation Excellence 97
Shreyas Chandgothia, Ardhendu Sekhar, and Amit Sethi

Interpretable Deep Graph-Level Clustering: A Prototype-Based Approach 114
Jianbin Cui and Lingyang Chu

A Saliency-Aware NR-IQA Method by Fusing Distortion Class Information . . . 130
Menglong Chen, Jianming Wang, Zhitao Xiao, and Yukuan Sun

A Guided Input Sampling-Based Perturbative Approach for Explainable AI in Image-Based Application 144
Arju Bano and Monidipa Das

Multi-target Attention Dispersion Adversarial Attack Against Aerial Object Detector 163
Panpan Wang, Shujuan Wang, Zhichao Lian, and Shuohao Li

Mask-TS Net: Mask Temperature Scaling Uncertainty Calibration for Polyp Segmentation 180
Yudian Zhang, Chenhao Xu, Kaiye Xu, and Haijiang Zhu

Label-Expanded Feature Debiasing for Single Domain Generalization 195
Jie Yang, Liwei Jing, Yuanzhuo Xu, Shaowu Wu, Steve Drew, and Xiaoguang Niu

Infrared and Visible Image Fusion Based on CNN and Transformer Cross-Interaction with Semantic Modulations 211
Yusu Zhang, Xiao-Jun Wu, and Tianyang Xu

Mining Long Short-Term Evolution Patterns for Temporal Knowledge Graph Reasoning 227
Fengyi Wang, Guanghui Zhu, Haojun Hou, Chunfeng Yuan, and Yihua Huang

Rethinking Attention Gated with Hybrid Dual Pyramid Transformer-CNN for Generalized Segmentation in Medical Imaging 243
Fares Bougourzi, Fadi Dornaika, Abdelmalik Taleb-Ahmed, and Vinh Truong Hoang

A Weighted Discrete Wavelet Transform-Based Capsule Network for Malware Classification 259
Tonghua Qiao, Chunjie Cao, Binghui Zou, Fangjian Tao, Yinan Cheng, Qi Zhang, and Jingzhang Sun

Data-Driven Spatiotemporal Aware Graph Hybrid-hop Transformer Network for Traffic Flow Forecasting 275
Lin Hang, Qianqian Ren, and Xingfeng Lv

Automatic Diagnosis Model of Gastrointestinal Diseases Based on Tongue Images 290
Baochen Fu, Miao Duan, Zhen Li, Xiuli Zuo, and Xu Qiao

TinyConv-PVT: A Deeper Fusion Model of CNN and Transformer for Tiny Dataset 302
Yi Zhang, Bowei Zhang, and Kai Yi

SCAD-Net: Spatial-Channel Attention and Depth-Map Analysis Network for Face Anti-spoofing 319
Nguyen Quoc Viet, Le Minh Tri, Nguyen Thi Yen, and Nguyen Minh Giang

Next Generation Loss Function for Image Classification 334
Shakhnaz Akhmedova and Nils Körber

NAOL: NeRF-Assisted Omnidirectional Localization 350
Muzhou Yu, Zhihan Cai, Kailu Wu, Dapeng Cao, and Kaisheng Ma

EdgeConvFormer: An Unsupervised Anomaly Detection Method for Multivariate Time Series 367
Jie Liu, Qilin Li, Senjian An, Bradley Ezard, and Ling Li

Lighten CARAFE: Dynamic Lightweight Upsampling with Guided Reassemble Kernels 383
Ruigang Fu, Qingyong Hu, Xiaohu Dong, Yinghui Gao, Biao Li, and Ping Zhong

Hand over Face Gesture Classification with Feature Driven Vision Transformer and Supervised Contrastive Learning 400
Kankana Roy, Aparna Mohanty, and Rajiv Ranjan Sahay

TabSeq: A Framework for Deep Learning on Tabular Data via Sequential Ordering 418
Al Zadid Sultan Bin Habib, Kesheng Wang, Mary-Anne Hartley, Gianfranco Doretto, and Donald A. Adjeroh

GraFix: A Graph Transformer with Fixed Attention Based on the WL Kernel . . 435
Lingfeng Zhang, Luca Cosmo, Giorgia Minello, Andrea Torsello, and Luca Rossi

Multi-modal Deep Emotion-Cause Pair Extraction for Video Corpus 451
Qianli Zhao, Linlin Zong, Bo Xu, Xianchao Zhang, and Xinyue Liu

Author Index 467

DeepEMD: A Transformer-Based Fast Estimation of the Earth Mover's Distance

Atul Kumar Sinha(✉) and François Fleuret

University of Geneva, Geneva, Switzerland
{atul.sinha,francois.fleuret}@unige.ch

Abstract. The Earth Mover's Distance (EMD) is the measure of choice between point clouds. However the computational cost to compute it makes it prohibitive as a training loss, and the standard approach is to use a surrogate such as the Chamfer distance. We propose an attention-based model to compute an accurate approximation of the EMD that can be used as a training loss for generative models. To get the necessary accurate estimation of the gradients we train our model to explicitly compute the matching between point clouds instead of EMD itself. We cast this new objective as the estimation of an attention matrix that approximates the ground truth matching matrix. Experiments show that this model provides an accurate estimate of the EMD and its gradient with a wall clock speed-up of more than two orders of magnitude with respect to the exact Hungarian matching algorithm and one order of magnitude with respect to the standard approximate Sinkhorn algorithm, allowing in particular to train a point cloud VAE with the EMD itself. Extensive evaluation show the remarkable behaviour of this model when operating out-of-distribution, a key requirement for a distance surrogate. Finally, the model generalizes very well to point clouds during inference several times larger than during training (https://github.com/atulkumarin/DeepEMD).

Keywords: Earth Movers Distance (EMD) · Point Clouds · Generative Modeling · Wasserstein Distance

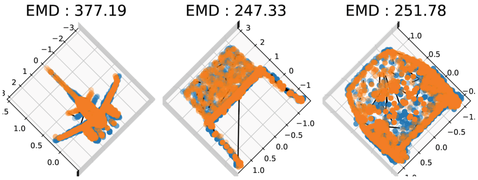
1 Introduction

The *earth mover's distance* (EMD), also known as *Wasserstein distance* is a distance between distributions that is defined as the minimum total of mass-time-distance displacement needed to transform one distribution to the other. In the case of uniform distributions over a finite number of points, it turns into a distance between point clouds that corresponds to finding the one-to-one matching that minimizes the sum of the distances between pairs of matched points.

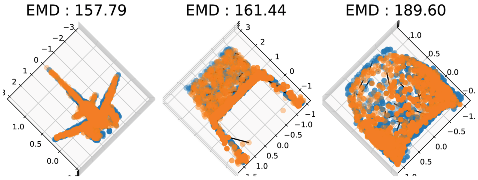
Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78128-5_1.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 1–15, 2025.
https://doi.org/10.1007/978-3-031-78128-5_1

Since there is no inherent ordering in point cloud data, computing the EMD between two point clouds involves finding a matching based on the euclidean distance between points. The matching is constrained to be bipartite so that one point cloud is completely transformed to the other, without any fractional assignment, and the transport cost is minimal.



Trained with Chamfer distance



Trained with DeepEMD (ours)

Fig. 1. Example point clouds (blue) and their VAE reconstructions (orange) when trained with different reconstruction losses. Training with DeepEMD (bottom) consistently achieves lower reconstruction error (EMD, shown on top of each example) than with the standard Chamfer distance (top). (Color figure online)

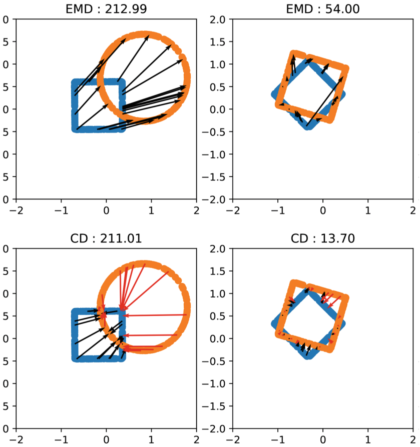


Fig. 2. Example pairs of point clouds. The true earth mover's distance (top) and Chamfer distance (bottom) are shown above each example. Arrows indicate the matching between the two point clouds under their respective metrics.

The EMD is the most commonly used distance metric on point clouds, and is extremely useful in many different contexts. In particular as we will see for both assessing the performance of, and for training variational autoencoders, since the generated point cloud should get as close as possible to the target in terms of displacement. It can also be interpreted as the distance between two distributions computed with a finite number of samples and reflects the notion of nearness properly, does not have quantization/binning and non-overlapping support problems of most other metrics, e.g., f-divergences, total variation distance, etc.

The EMD between point clouds can be computed exactly, but it is extremely expensive computationally. The standard method is the Hungarian matching [12] algorithm whose complexity is $O(N^3)$ where N is the number of points. Due to this computational cost, training deep generative models for point clouds is not done with this metric, even though it is a metric of choice for performance evaluation. The standard approach uses the *Chamfer distance* (CD) as surrogate. This metric can be computed in $O(N^2)$ time complexity but relaxes the one-to-one matching, which may create pathological situations.

We propose a deep architecture that takes as input two point clouds encoded as sets of geometric coordinate tuples, and computes an accurate estimate of the EMD. We show that the most efficient approach, in particular if that estimate is used as a loss for a training process, is to estimate the matching matrix itself. Since the EMD is the sum of the distance between matched points, this approach provides a very accurate estimate of the gradient with respect to the point coordinates. Training a deep variational autoencoder with our model instead of an exact computation is up to ×100 faster (see Fig. 8), and the resulting model performs far better than one trained with the usual Chamfer surrogate (see Sect. 4.4 and Fig. 7).

The key contributions of this paper can be summarized as:

- We propose DeepEMD which approximates the EMD between point clouds in $O(N^2)$ time complexity vs $O(N^3)$ of the hungarian algorithm (Sect. 3).
- We propose to cast the prediction of a bipartite matching as an attention matrix from which we get an accurate estimate of the EMD and its gradient (Sect. 3.2).
- We show that DeepEMD generalizes well to unseen data distributions (Sect. 4.3), and can be used for evaluation of generative models. It provides accurate estimates of the gradients of the distance and demonstrate that DeepEMD can be used as a surrogate reconstruction loss for training deep generative models of point clouds (Sect. 4.4).
- We show that DeepEMD achieves about 40× speed-up over Sinkhorn algorithm, achieving equal or better performance for various metrics (Sect. 4.3).

2 Related Work

The two commonly used distance metrics for point clouds in literature are Earth Mover's Distance (EMD) and Chamfer Distance (CD). Consider two point clouds $X = \{x_i\}_{i=1}^N$ and $Y = \{y_j\}_{j=1}^N$, where $x_i, y_j \in \mathbb{R}^d$. The EMD between the two point clouds can be computed as,

$$\mathrm{EMD}(X, Y) = \min_{\phi \in \mathcal{M}(X,Y)} \sum_{x \in X} \|x - \phi(x)\|_2, \tag{1}$$

where $\mathcal{M}(X, Y)$ is the set of 1-to-1 (bipartite) mappings from X to Y. In addition to the distance, the optimal matching ϕ^* is also interesting for some applications. Since directly optimizing the EMD is computationally expensive, most methods in the literature rely on CD as a proxy similarity measure or reconstruction loss. The CD can be computed as,

$$\mathrm{CD}(X, Y) = \sum_{x \in X} \min_{y \in Y} \|x - y\|_2^2 + \sum_{y \in Y} \min_{x \in X} \|x - y\|_2^2, \tag{2}$$

and in $O(N^2)$ time complexity. The CD solution leads to a non-bipartite one-to-many matching between $x \rightarrow y$ and vice versa. We can also use the L_2 measure with $d = \|x - y\|_2$ instead $d = \|x - y\|_2^2$ to make it comparable to EMD. of Note that the above EMD for point clouds is related to the Wasserstein-2 metric (see appendix Sect. A for details). The utility of EMD is limited by the $O(N^3)$ computational cost of evaluating it. There have been several research efforts to circumvent this issue in various application settings.

This is the case for application to point clouds where N is usually in the range of several thousands. SetVAE [11] trains a variational auto-encoder with CD as the reconstruction loss. EMD is still the metric of choice for evaluating point cloud generative models [1,10,11,13,15,21]. Another issue is disparity between performance measures (minimum matching distance, coverage, etc.) computed with EMD and CD, the comparisons are contradictory and often inconsistent across measures.

CD is usually insensitive to mismatched local density while EMD is dominated by global distribution [19] and overlooks the fidelity of detailed structures. [19] proposes a new similarity metric called Density-aware Chamfer distance (DCD) to tackle these issues. DCD is derived from CD and can also be computed in $O(N^2)$ time complexity. [16] proposed Deep Point Cloud Distance (DPDist) which measures the distance between the points in one cloud and the estimated continuous surface from which the other point cloud is sampled. The surface is estimated locally by a network using the 3D modified Fisher vector representation.

[14] proposed a linear time algorithm for approximating the EMD by exploiting the Hölder continuity constraint in its dual form to convert it into a simple optimization problem with an explicit solution in the wavelet domain and computed as the sum of absolute values of the weighted wavelet coefficients of the difference histogram. However, their approach is limited to low dimensional histograms. In the optimal transport literature, several efforts have been taken towards improving the statistical and computational properties. Recently, [5] proposed Information Maximizing Optimal Transport (InfoOT) which is an information-theoretic extension of optimal transport based on kernel density estimation of the mutual information which introduces global structure into OT maps. The resulting solution maximizes the mutual information between domains while minimizing geometric distance and improves the capability for handling data clusters and outliers.

Other approaches focus on regularizing the OT problem for making it smooth and strictly convex [3,6,7,9]. *Sinkhorn distances* [6] smooth the classic OT problem with an entropic regularization term and can be computed through Sinkhorn's matrix scaling algorithm at a speed that is several orders of magnitude faster than that of transport solvers. We provide more details in the appendix. Meta OT [2] proposes a meta model to predict the solution to the optimal transport problem which is then used to initialize a standard Sinkhorn solver to further refine the predicted solution. The architecture of the meta model depends on the data domain, and DeepEMD can be utilized when working with point clouds. The choice of meta model architecture is contingent upon the specific data domain, and DeepEMD demonstrates its exceptional utility in point cloud processing.

In this paper, our goal is to approximate the EMD using a deep network in a learning based paradigm where each sample represents two distributions and the target for regression is either the true metric i.e. EMD or the optimal matching ϕ^*, or both. Existing point cloud datasets can serve as an interesting learning problem, where we can interpret each point cloud as a 2D or 3D distribution of points on a shape (manifold). It is posed as a supervised learning problem where the task is to estimate the true EMD, or the true bipartite matching, or both, between a pair of input point clouds.

3 Method

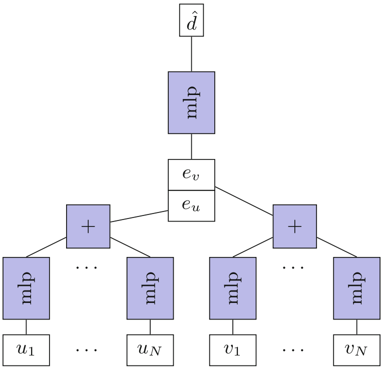


Fig. 3. The MLP model (see Sect. 3.1) predicts directly an estimate $\hat{d}$ of the EMD.

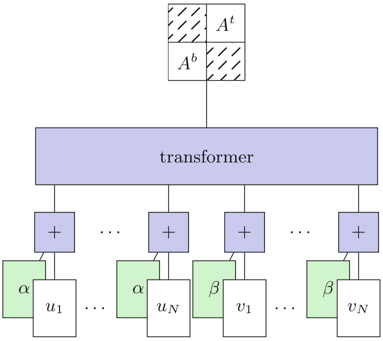


Fig. 4. The transformer model for DeepEMD (see Sect. 3.2) predicts directly the bipartite graph as an attention matrix.

We are interested in building a model which operates on a pair of point clouds (U, V) as input, where $U, V \in \mathcal{D}^N$, $U = \{u_i\}_{i=1}^N$, $V = \{v_j\}_{j=1}^N$, $u_i, v_j \in \mathbb{R}^D$, and N is the cardinality of the point clouds. Note that the points are unordered and the indexing is arbitrary. We denote the earth mover's distance between them as $d = \text{EMD}(U, V)$ where $d \in \mathbb{R}$. The goal of the model is to predict d and ∇d. Also, let $M \in \{0,1\}^{N \times N}$ denote the ground truth bipartite matching from EMD, where $M_{i,j} = 1$ indicates that u_i is matched to v_j and vice-versa. Bipartite-ness implies $\forall j, \sum_i M_{i,j} = 1$ and $\forall i, \sum_j M_{i,j} = 1$.

Since point clouds are unordered and invariant to elementwise permutation, we seek mappings $f : \mathcal{D}^N \times \mathcal{D}^N \to \mathbb{R}$ which are permutation invariant for any permutations π and π', i.e.,

$$f(U, V) = f(\pi(U), \pi'(V)), \tag{3}$$

In the following sections, we first introduce a simple MLP based baseline, followed by our transformer-based model, *DeepEMD*.

3.1 Predicting the Distance

We propose a simple MLP baseline composed of a point-wise MLP backbone, followed by a prediction head which is also a MLP (see Fig. 3). The backbone MLP takes a point cloud and returns an embedding $e \in \mathbb{R}^d$ as,

$$e_u = \sum_{i=1}^{n} g(u_i), \quad e_v = \sum_{j=1}^{n} g(v_i) \tag{4}$$

The prediction head then produces the final prediction as,

$$\hat{d} = h\left(e_u \oplus e_v\right) + h\left(e_v \oplus e_u\right), \tag{5}$$

where $\oplus$ denotes vector concatenation. Both g and h are composed of sequential linear layers with ReLU non-linearity between layers. The embeddings are permutation equivariant because of the sum aggregation which does not depend on the ordering of points. Further, we concatenate the embeddings both ways as in Eq. (5), which makes the mapping symmetric. We train the model with mean-squared error loss, $l = (d - \hat{d})^2$. Since the model does not predict the matching, we can interpret it from the direction of the gradient of a point $\delta v_j = \left[\frac{\partial \hat{d}}{\partial V}\right]_j$, e.g., by taking cosine similarity between δv_j and $u_i - v_j$, where u_i is the point matching to v_j from EMD.

3.2 Predicting Bipartite Matching

The transformer [18] intuitively seems to be a very good model for reasoning with point clouds and matching. Moreover, by considering that the output is the last layer's attention matrix, we can use it to directly predict the bipartite matching. Since the EMD–and consequently its gradient with respect to the point positions–is a function of the point positions and the matching array, predicting the latter leads to an accurate estimate of the former, that in particular is shielded from the issue of a possible decoupling between matching a functional point-wise (e.g. for MSE) and matching the gradients. While it is viable to directly predict the distance with a transformer-like architecture, we chose to only predict the matching since it is straightforward to estimate the distance using the predicted matching.

We propose *DeepEMD* composed of a sequence of multi-head full attention layers, followed by a prediction head which is also a full attention layer, but with a single head. Given two point clouds U and V, add a learned cloud-specific positional embedding to indicate if a point originates from U or V, we concatenate the points sequence and feed the resulting $I = U \cup V$ as input to the model. The group-id embedding helps the model in modulating attention locally within a point cloud as well as globally across both point clouds. We tried other variants with self-attention layers, cross attention layers, and an alternating mixture of both, but found full attention over both point clouds to be best performing.

For our problem, we get the input X for the transformer by adding positional embeddings to I. Let $\boldsymbol{0}^n$ and $\boldsymbol{1}^n$ denote a vector of n zeros and ones, respectively. X is then obtained as,

$$P = \boldsymbol{0}^n \cup \boldsymbol{1}^n, \quad X = I + W^P[P] \quad \text{(indexing)} \tag{6}$$

The intermediate feature $t(X)$ from the transformer encoder (see appendix for details) has the same number of elements as X with each element now being a contextualized representation for the corresponding point in the input. Further, these intermediate representation are fed into a single-head attention layer which outputs the attention matrix as,

$$K = t(X)W^K, \quad Q = t(X)W^Q, \quad A = \frac{QK^\top}{d_k} \tag{7}$$

$$A^t = A_{:n,n:} \quad A^b = A_{n:,:n} \tag{8}$$

Here, A is a $2N \times 2N$ matrix and we slice the top-right block (first N rows and last N columns) as A^t and bottom-left block (last N rows and first N columns) as A^b. $A^t_{i,j}$ can be interpreted as the relatedness of u_i with v_j. Similarly, $A^b_{i,j}$ can be interpreted as the relatedness of v_i with u_j.

Given M, the ground truth bipartite matching from EMD, we define the loss as average of the cross-entropies (CE) as,

$$l(U, V) = \frac{1}{N}\sum_{i=1}^{N} \mathrm{CE}(A^t_{i,.}, M_{i,.}) + \frac{1}{N}\sum_{i=1}^{N} \mathrm{CE}(A^b_{i,.}, M_{.,i}) \tag{9}$$

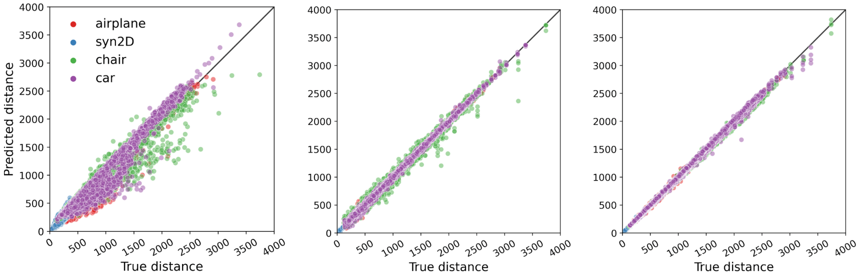
The EMD is then estimated with the predicted matching as,

$$\phi^b(i) = \arg\max_j A^b_{i,j}, \quad \phi^t(i) = \arg\max_j A^t_{i,j} \tag{10}$$

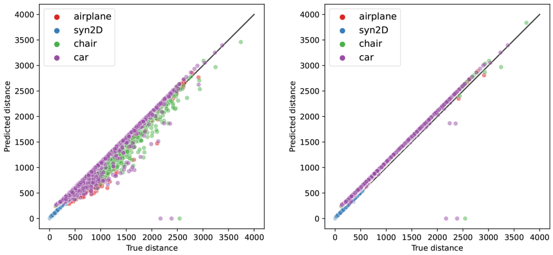
$$\hat{d} = \frac{1}{2}\left(\sum_i \|u_i - v_{\phi^t(i)}\| + \sum_i \|v_i - u_{\phi^b(i)}\|\right) \tag{11}$$

4 Experiments

In this section, we present the overall experimental setup, performance results and comparisons of DeepEMD across various tasks.



(a) Chamfer (b) MLP (c) DeepEMD



(d) Sinkhorn (iter=10) (e) Sinkhorn (iter=100)

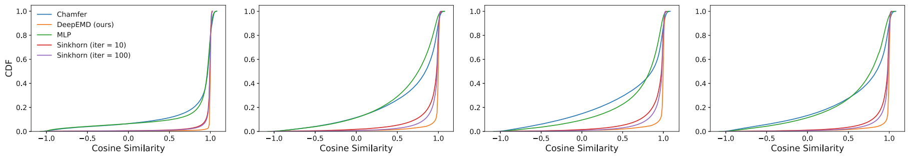
Fig. 5. Scatter plot for true vs. approximate EMD from different models/metrics on validation splits for Syn2D and ShapeNet datasets. DeepEMD (ours) consistently performs better across different categories as it has less dispersion. Sinkhorn algorithm becomes more accurate with more iterations. Also note that it encounters numerical errors for some examples.

4.1 Datasets

We consider different datasets for our experiments - *Syn2D*, ShapeNet [4], ModelNet40 [20] and ScanObjectNN [17]. *Syn2D* consists of 2D point clouds generated synthetically by sampling points on squares and circles (see Fig. 2). ShapeNet and ModelNet40 are datasets of 3D point clouds derived from 3D CAD models for different real world objects like chairs, cars, airplanes, etc. ScanObjectNN is a relatively new real-world point cloud object dataset based on scanned indoor scene data. In order to improve and assess generalization, we augment the train and test splits with synthetic perturbations. We provide more details about the datasets and these augmentations in the Appendix Sect A.

4.2 Performance Measures

We consider various measures to assess EMD approximation methods for distance as well as matching estimation. We compare accuracy and computation time to that of Sinkhorn and CD (see Sect. 2).



(a) Syn2D (b) ShapeNet Plane (c) ShapeNet Car (d) ShapeNet Chair

Fig. 6. CDF of cosine similarity between true and estimated gradients for all points across all point clouds collected together on validation splits for Syn2D and ShapeNet datasets. The ideal cdf curve should have all the mass at cosine similarity 1. DeepEMD (ours) consistently outperforms all the other methods across different datasets.

Distance Estimation. We visualize the true vs. predicted distance through scatter plots (Fig. 5), we expect the data points to be close to $x = y$ line. We compare various correlation measures : linear correlation (r), Spearman correlation (ρ) and Kendall-Tau correlation (τ), to assess the quality of distance estimation. The Spearman and Kendall-Tau are rank-statistic based correlation measures, indicative of the correspondence between two rankings. Note that, correlation measures are useful metrics as they indicate appropriateness of the predicted metric as a distance measure, irrespective of their absolute values. Additionally we look at different quantiles (RE_n) of relative approximation error, which penalizes the difference between absolute values of the predicted and true distance.

Matching Estimation. In order to assess quality of the matching, we consider the cosine similarity between the true and predicted gradient. The true gradient of EMD is always along the matched point. We visualize the cumulative distribution function (cdf) of cosine similarities (Fig. 6), where we expect all the mass to be close to 1. We also look at different quantiles (CS_n) of the cosine similarity. We also consider accuracy which is computed as the average accuracy of matching source points to target points and vice-versa, bipartiteness (B) which is fraction of points with bipartite matching, and also bipartiteness-correctness (B_{corr}) which is fraction of points which are bipartite as well as matched correctly.

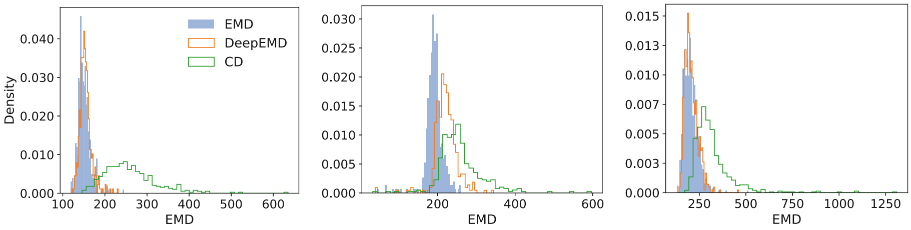
4.3 Results

EMD Prediction. Figure 5 shows the scatter plot of the true EMD vs. approximate EMD predicted from our trained models on the validation split for *Syn2D* and ShapeNet datasets. Note that the validation split also contains the augmentations as discussed in Sect. 4.1. We also validate on specific splits for which the results are shown in the appendix. The plots indicate that both DeepEMD (Fig. 5c) and MLP baseline (Fig. 5b) approximate the EMD faithfully. The MLP baseline seems to struggle a bit on ShapeNet Chair dataset. The higher dispersion in Chamfer (Fig. 5a) and Sinkhorn with 10 iterations (Fig. 5d) indicates poor EMD estimation. The approximation with Sinkhorn algorithm becomes more accurate with higher number of iterations (Fig. 5e), as expected.

We summarize various metrics in Tables 5, 6 and 7 the appendix. DeepEMD and MLP baseline both achieve linear correlation higher than 0.99 in each case. The models achieve Kendall-Tau correlation close to 0.99 and 0.96, respectively, and Spearman correlation close to 0.99 in each case, indicating that ordering of the samples based on approximate distances and true distances are very similar, and monotonocity of samples are preserved. It can be observed that DeepEMD is best except for relative error where it may do worse than our method with MLP. Also it is interesting to note that Sinkhorn performs worse than DeepEMD on correlation- or relative error-based measures.

Matching/Gradient Prediction. Estimating the matching and gradient of the distance is particularly important for training models with DeepEMD as a surrogate distance function. Note that the gradient of a point from true EMD is always along the matched point in the other point cloud. Figure 6a shows the cdf of cosine similarity between the true and estimated gradient for all the points across all point clouds collected together for *Syn2D*, while Figs. 6b, 6c, and 6d for ShapeNet. The cdf has most mass at cosine similarity close to 1 with a very short tail and is never negative indicating that the estimated gradient is aligned with the true gradient for DeepEMD. This is particularly important when the model is used as a surrogate reconstruction loss. Ideally, the model should provide good estimate of the true gradient throughout training and more particularly in the very beginning when the reconstructions are very noisy, and also towards the end when reconstructions likely become very similar to the training distribution. We discuss more on this in the next sections and the appendix. The MLP baseline usually did not perform well in this regard and also in generalizing to unseen examples and thus was not useful for training generative models. The same issue can be observed with Chamfer distance as well. Table 5 (appendix) shows the cosine similarity quantiles (CS_n), as well as accuracy, bipartiteness (B) and bipartiteness-correctness (B_{corr}). DeepEMD performs better than other models and metrics in each of these measures and indicates better matching approximation.

Out-of-Distribution Generalization. The generalization of the prediction to a novel distribution is particularly important for a surrogate metric. We test the out-of-distribution behaviour of our models in two different settings : Table 1 shows the generalization performance of the model trained on a single category of ShapeNet and tested on validation split of multi-category ModelNet40 dataset, while Tables 8, 9 and 10 in the appendix show the performance when tested on different ShapeNet categories. The results indicate that DeepEMD generalizes well when test and train data differ without any adaptation or fine-tuning. Further, the validation performance on a category of a model trained on another category (see Appendix for details) is very similar to the performance of the model trained on the same category. These quite remarkable behaviors point towards the network "meta-learning" in some way the matching algorithm. This



(a) ShapeNet Airplane (b) ShapeNet Car (c) ShapeNet Chair

Fig. 7. Comparison of EMD between input and reconstructed point clouds from SetVAE trained with different reconstruction losses. The better model should have smaller reconstruction loss and thus mass close to zero in the histograms. DeepEMD (ours) is consistently better as compared to Chamfer loss and very similar to EMD loss.

is further strengthened by the results on scaling to different number of points during test time as shown in Table 2.

Scaling Number of Points. Remarkably, the size of point clouds during testing can differ greatly from those during training without degrading performance. Table 2 shows performance of the model for test point cloud sizes ranging from 256 to 8196, while training was done with only 1024 points. Prediction of the metric itself (top 6 rows) does not degrade for all practical purposes. Regard-

Table 1. Out-of-distribution (dataset) generalization for our models and comparison with other metrics (Chamfer and Sinkhorn), tested on full validation split for ModelNet40 (with 40 categories) and ScanObjectNN (with 15 categories). The models are trained on a single ShapeNet category. The reported numbers are averaged over these categories as well as four training seeds. The first six rows show distance estimation metrics (see Sect. 4.2), while the last six rows correspond to matching estimation metrics. The arrows next to the metrics indicate whether higher (↑) values are better or lower (↓). Chamfer and Sinkhorn are deterministic, thus variances are not reported. Further, MLP does not provide accuracy and bipartiteness metrics.

DATASET	ModelNet40				ScanObjectNN			
MODEL	CHAMFER	SINKHORN	MLP	DEEPEMD	CHAMFER	SINKHORN	MLP	DEEPEMD
r (↑)	0.951	0.971	0.959 ± 0.011	$\mathbf{0.999} \pm 0.0$	0.971	0.929	0.965 ± 0.005	$\mathbf{0.997} \pm 0.001$
ρ (↑)	0.935	0.988	0.945 ± 0.017	$\mathbf{0.999} \pm 0.0$	0.979	0.965	0.963 ± 0.007	$\mathbf{0.999} \pm 0.0$
τ (↑)	0.792	**0.983**	0.819 ± 0.024	0.974 ± 0.002	0.882	0.968	0.855 ± 0.011	$\mathbf{0.973} \pm 0.002$
$\mathrm{RE}_{0.1}$ (↓)	0.03	0.057	0.009 ± 0.001	$\mathbf{0.005} \pm 0.002$	0.025	0.038	0.013 ± 0.001	$\mathbf{0.004} \pm 0.001$
$\mathrm{RE}_{0.5}$ (↓)	0.129	0.102	0.062 ± 0.005	$\mathbf{0.019} \pm 0.004$	0.094	0.078	0.076 ± 0.005	$\mathbf{0.019} \pm 0.004$
$\mathrm{RE}_{0.9}$ (↓)	0.321	0.2	0.257 ± 0.03	$\mathbf{0.04} \pm 0.004$	0.282	0.244	0.299 ± 0.025	$\mathbf{0.051} \pm 0.005$
$\mathrm{CS}_{0.1}$ (↑)	-0.067	0.824	-0.293 ± 0.047	$\mathbf{0.927} \pm 0.003$	0.138	0.879	-0.208 ± 0.042	$\mathbf{0.946} \pm 0.002$
$\mathrm{CS}_{0.5}$ (↑)	0.834	0.986	0.684 ± 0.023	$\mathbf{1.0} \pm 0.0$	0.917	0.992	0.719 ± 0.02	$\mathbf{0.999} \pm 0.0$
$\mathrm{CS}_{0.9}$ (↑)	0.997	0.999	0.96 ± 0.003	$\mathbf{1.0} \pm 0.0$	0.998	1.0	0.965 ± 0.003	$\mathbf{1.0} \pm 0.0$
Accuracy (↑)	12.651	31.91	-	$\mathbf{56.38} \pm 0.604$	7.673	20.04	-	$\mathbf{40.671} \pm 0.62$
B (↑)	17.045	33.458	-	$\mathbf{70.401} \pm 0.672$	9.474	19.43	-	$\mathbf{56.269} \pm 0.71$
B_{corr} (↑)	6.544	19.615	-	$\mathbf{47.084} \pm 0.741$	3.38	9.961	-	$\mathbf{30.055} \pm 0.678$

ing the matching estimation, directional measure of performance related to the cosine similarity (rows CS_n) do not degrade neither. We can notice degradation in accuracy based measures (last 3 rows) which is natural since the problem becomes difficult with increasing number of points N because of its combinatorial nature. For training when memory requirement is much higher due to backprop, we can use smaller number of points, and scale it up during inference without any fine-tuning.

Table 2. Scaling number of points and out-of-distribution (scale) generalization for DeepEMD. The models are trained on a single ShapeNet category with 1024 points and tested on validation split of same category but with different number of points. Reported values are averaged over 4 training seeds. DeepEMD generalizes well to unseen number of points at test time without fine-tuning.

	⟵ Less # points than training ⟶			Trained	⟵ More # points than training ⟶		
# points	256	512	768	1024	2048	4096	8192
r	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	0.999 ± 0.0	0.999 ± 0.001
ρ	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	0.999 ± 0.0	0.998 ± 0.0
τ	0.985 ± 0.0	0.987 ± 0.0	0.988 ± 0.0	0.988 ± 0.001	0.986 ± 0.001	0.981 ± 0.002	0.974 ± 0.004
$\mathrm{RE}_{0.1}$	0.002 ± 0.001	0.002 ± 0.001	0.004 ± 0.002	0.007 ± 0.003	0.012 ± 0.005	0.013 ± 0.007	0.014 ± 0.008
$\mathrm{RE}_{0.5}$	0.01 ± 0.002	0.011 ± 0.002	0.014 ± 0.003	0.017 ± 0.005	0.027 ± 0.009	0.034 ± 0.013	0.04 ± 0.016
$\mathrm{RE}_{0.9}$	0.026 ± 0.003	0.026 ± 0.003	0.029 ± 0.004	0.032 ± 0.005	0.042 ± 0.009	0.054 ± 0.013	0.066 ± 0.018
$\mathrm{CS}_{0.1}$	0.94 ± 0.002	0.955 ± 0.002	0.961 ± 0.001	0.964 ± 0.001	0.967 ± 0.001	0.967 ± 0.001	–
$\mathrm{CS}_{0.5}$	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	–
$\mathrm{CS}_{0.9}$	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	–
Accuracy	72.348 ± 0.44	69.384 ± 0.383	66.901 ± 0.379	64.648 ± 0.404	57.588 ± 0.464	47.78 ± 0.51	35.274 ± 0.483
B	81.857 ± 0.755	80.101 ± 0.547	78.013 ± 0.507	75.896 ± 0.521	68.658 ± 0.584	58.109 ± 0.597	44.603 ± 0.734
B_{corr}	64.838 ± 0.756	61.558 ± 0.587	58.545 ± 0.547	55.719 ± 0.568	46.831 ± 0.618	35.053 ± 0.6	21.606 ± 0.469

Computational Time and Complexity. Figure 8 compares the evaluation time for different models and metrics. DeepEMD achieves a significant speedup of about 100× as compared to EMD and 40× as compared to Sinkhorn with 100 iterations. This speedup becomes more pronounced on bigger point clouds as hungarian algorithm takes $O(N^3)$ time vs. $O(N^2)$ for DeepEMD.

4.4 DeepEMD Used as a Loss

Training a SetVAE, as for any auto-encoder, requires a reconstruction loss to assess the quality of the learned representation. While the eventual goal would be to minimize the EMD, standard approach uses Chamfer Distance due to the prohibitive computation cost of calculating the EMD. Instead of Chamfer Distance we propose to use DeepEMD and demonstrate its utility as a reconstruction loss as compared to Chamfer Distance.

DeepEMD was trained separately on each category of ShapeNet dataset and the trained model was then used as a surrogate reconstruction loss for training a variational auto-encoder. We use SetVAE [11], a transformer based VAE adapted for point clouds and set-structured data. The parameters of DeepEMD module are frozen during training of the SetVAE. We follow exactly the same protocol as in SetVAE and train using ShapeNet categories of airplane, chair, and car and also the same hyper-parameters for training. Figure 1 and Fig. 9 (appendix) shows the reconstruction on validation data achieved by SetVAE models trained with different reconstruction losses. DeepEMD consistently achieves lower reconstruction EMD as compared to CD. This is further verified from Fig. 7 which shows the distribution of true EMD between a point cloud and its reconstruction.

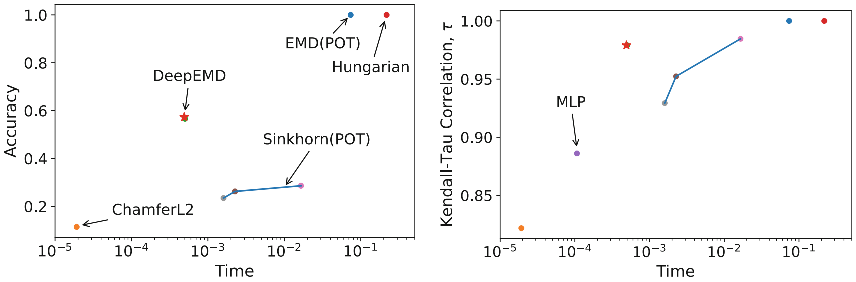


Fig. 8. Comparison of empirical evaluation time and different performance measures: Accuracy (left) and Kendall-Tau correlation τ (right). We use Python Optimal Transport (POT) library for computing Sinkhorn distances, and show metrics at different iterations (5, 10 and 100). DeepEMD is $\sim 100\times$ and $\sim 40\times$ faster than Hungarian algorithm and Sinkhorn (100 iterations), respectively.

5 Conclusion and Future Work

We propose DeepEMD, a method for fast approximation of EMD, improving time complexity from $O(N^3)$ to $O(N^2)$. It is composed of a multi-head multi-layer transformer, followed by a single-head full attention layer as the final output layer. It operates on two point clouds and outputs an attention matrix which is interpreted as the matching matrix and optimized to match the ground turh matching obtained from the hungarian algorithm. We demonstrated the effectiveness of DeepEMD in approximating the true EMD for synthetic 2D point clouds as well as real world datasets like ShapeNet, ModelNet40 and ScanObjectNN. It achieves a speed-up of $\times 100$ with 1024 points. Further, we show that it estimates the gradients well, generalizes well for unseen point clouds (or distributions), and can be used for end-to-end training of point cloud autoencoders achieving faster convergence than Chamfer distance surrogate. It would be interesting to explore fast transformer variants to further improve from the quadratic time complexity for future work. In terms of architecture various pooling/un-pooling strategies can be explored which can help with both, better time complexity and improved feature learning. In this work, we estimate the Wasserstein−2 metric, and extension to other Wasserstein−p metrics and other optimal transport problems could also be interesting for various applications. Lastly, extension to general probability distributions with fractional assignments (i.e. mass splitting) can also be very useful and valuable for some applications.

References

1. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Learning representations and generative models for 3d point clouds. In: International Conference on Machine Learning, pp. 40–49. PMLR (2018)
2. Amos, B., Cohen, S., Luise, G., Redko, I.: Meta optimal transport. arXiv preprint arXiv:2206.05262 (2022)
3. Blondel, M., Seguy, V., Rolet, A.: Smooth and sparse optimal transport. In: International Conference on Artificial Intelligence and Statistics, pp. 880–889. PMLR (2018)
4. Chang, A.X., et al.: Shapenet: an information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015)
5. Chuang, C.Y., Jegelka, S., Alvarez-Melis, D.: Infoot: information maximizing optimal transport. arXiv preprint arXiv:2210.03164 (2022)
6. Cuturi, M.: Sinkhorn distances: lightspeed computation of optimal transport. Adv. Neural Inf. Pocess. Syst. **26** (2013)
7. Flamary, R., Courty, N., Tuia, D., Rakotomamonjy, A.: Optimal transport for domain adaptation. IEEE Trans. Pattern Anal. Mach. Intell. **1** (2016)
8. Flamary, R., et al.: Pot: python optimal transport. J. Mach. Learn. Res. **22**(1), 3571–3578 (2021)
9. Genevay, A., Peyré, G., Cuturi, M.: Learning generative models with Sinkhorn divergences. In: International Conference on Artificial Intelligence and Statistics, pp. 1608–1617. PMLR (2018)
10. Huang, T., et al.: Learning to train a point cloud reconstruction network without matching. In: Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, 23–27 October 2022, Proceedings, Part I, pp. 179–194. Springer (2022)
11. Kim, J., Yoo, J., Lee, J., Hong, S.: Setvae: learning hierarchical composition for generative modeling of set-structured data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 15059–15068 (2021)
12. Kuhn, H.W.: The Hungarian method for the assignment problem. Naval Res. Logist. Quart. **2**(1–2), 83–97 (1955)
13. Luo, S., Hu, W.: Diffusion probabilistic models for 3d point cloud generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2837–2845 (2021)
14. Shirdhonkar, S., Jacobs, D.W.: Approximate earth Mover's distance in linear time. In: 2008 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8. IEEE (2008)
15. Shu, D.W., Park, S.W., Kwon, J.: 3d point cloud generative adversarial network based on tree structured graph convolutions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 3859–3868 (2019)
16. Urbach, D., Ben-Shabat, Y., Lindenbaum, M.: Dpdist: comparing point clouds using deep point cloud distance. In: European Conference on Computer Vision, pp. 545–560. Springer (2020)

17. Uy, M.A., Pham, Q.H., Hua, B.S., Nguyen, T., Yeung, S.K.: Revisiting point cloud classification: a new benchmark dataset and classification model on real-world data. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 1588–1597 (2019)
18. Vaswani, A., et al.: Attention is all you need. Adv. Neural Inf. Process. Syst. **30** (2017)
19. Wu, T., Pan, L., Zhang, J., Wang, T., Liu, Z., Lin, D.: Density-aware chamfer distance as a comprehensive metric for point cloud completion. arXiv preprint arXiv:2111.12702 (2021)
20. Wu, Z., et al.: 3d shapenets: a deep representation for volumetric shapes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1912–1920 (2015)
21. Yang, G., Huang, X., Hao, Z., Liu, M.Y., Belongie, S., Hariharan, B.: Pointflow: 3d point cloud generation with continuous normalizing flows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 4541–4550 (2019)

Equivariant Neural Networks for TEM Virus Images Improves Data Efficiency

Karl Bylander[1], Ingela Nyström[2], and Karl Bengtsson Bernander[2](✉)

[1] Swedish Competition Authority, Stockholm, Sweden
[2] Centre for Image Analysis, Department of Information Technology,
Uppsala University, Uppsala, Sweden
`karl.bengtsson_bernander@it.uu.se`

Abstract. While deep learning has seen consistent improvements in medical imaging applications, one of the biggest hurdles for further gains is the need for large amounts of data. This data is not always readily available for different reasons, e.g., due to a lack of experts to annotate the samples. Equivariant neural networks have proven to be a way to increase data efficiency by, e.g., forgoing the need for data augmentation. This study extends previous research on group-equivariant networks applied to transmission electron microscopy (TEM) images of different types of viruses. It is shown that group equivariant networks, when compared to baseline convolutional networks, obtain higher accuracies by 1.4 to 2.3% when increasing the training epochs from 300 to 2400. Furthermore, data augmentation strategies by rotations and reflections, as well as pre-training on ImageNet, can be skipped. When turning off batch normalization, the performance of the equivariant networks drops about 25% while the baseline fails to converge, implying that equivariance and batch normalization extract improved information from data by different mechanisms. The VGG16 architecture outperforms a smaller custom architecture by 3.9 to 4.4%, while the choice of symmetry group does not impact performance significantly. This study contributes to predictable scaling, which is already of great importance in, e.g., the training of large language models. Predictable scaling is expected to increase in significance in the field of biomedical image analysis since the model and dataset sizes continue to grow.

1 Introduction

This paper provides an overview of deep learning in biomedical image analysis, recent advances in equivariant neural networks, and promising strategies based on empirical deep learning.

The success of deep learning in biomedical image analysis is owed to several factors. Two of the most important ones are the availability of large amounts of annotated data and computational power. Larger datasets tend to need larger models to correctly discriminate between the fine-scale differences between the classes of the data [11]. Increasing the architecture size can however increase the

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 16–31, 2025.
https://doi.org/10.1007/978-3-031-78128-5_2

risk of overfitting to the training data as more parameters are used to distinguish between the details of the data.

Equivariant neural networks have shown to be more efficient than baseline convolutional neural networks (CNNs) in several respects, including accuracy on the test set, convergence speed during training and reducing the need for data augmentation [2–4]. These improvements follow from the equivariant networks' capacity to learn symmetries in the data while decreasing the number of parameters, reducing the risk of overfitting.

When developing deep learning classifiers, in biomedical image analysis, or similar problem settings, one of the main goals is usually a model that yields the highest possible accuracy on the test set [6]. Also important is the training techniques involved during the development process [19], such as transfer learning. This is relevant not only during research and development in new problem settings but also if older models need retraining to adapt to new settings, such as if a different microscope is used to acquire new images.

Since one of the main driving forces behind deep learning is the increasing amount of training data, training time is expected to increase as the years go by. Even if more powerful computational hardware becomes available, such as better Graphical Processing Units (GPU) [15] or more specialised Application Specific Integrated Circuits (ASIC) [17], the cost of training is still expected to go up in terms of electricity consumption or time spent waiting for the model to finish training. Therefore, progress towards reducing the amount of training data needed, and by extension the amount of computations, is highly valued. Equivariant neural networks have shown to be more data efficient than their baseline equivalents in biomedical image analysis contexts [16].

Characterizing the behaviour during training while varying the amount of data or the architecture size is called predictable scaling [13]. This strategy is used in the development of, e.g., large language models [22] when it is not feasible to tune models comprehensively. Typically, the error on the test set is plotted against varying sizes of data of number of model parameters in a log-log plot, yielding a power law. In the field of molecular dynamics, the power law exponents have shown steeper for equivariant networks, meaning it is more data efficient than the baseline models [1]. However, it is unclear how general these effects are.

The purpose of this study is to dive deeper into equivariant neural networks by varying the components of the trained models. Varying datasets is highly interesting for investigating how general the results are, but data variation is not the focus of this study. Here we instead vary other factors such as training times and model architectures. The role of optimizations such as batch normalization is also investigated. In addition, training times and test set accuracies are measured. The experiments are carried out on a dataset of Transmission Electron Microscopy (TEM) images, containing different types of viruses [20]. This dataset is highly relevant for characterizing equivariant neural networks, as the orientation under the microscope does not matter, making the images globally invariant. Still, local structures could convey important information for classification, which equivariant networks could exploit more efficiently than baseline architectures.

The rest of the paper is organized as follows. In Sect. 2, related work on equivariant neural networks is presented. Section 3 introduces the data, designs and experimental setups. Section 4 demonstrates the results, which are discussed in Sect. 5. Finally, Sect. 6 presents the conclusions of the paper.

2 Related Work

In Sect. 2.1, we introduce theoretical concepts of equivariant neural networks. Empirical results of previous works are then presented in Sect. 2.2.

2.1 Theory

While Visual Transformers (ViT) have recently become one of the most successful deep learning architectures [10], they lack some of the advantages of their forerunners, the convolutional neural networks (CNNs). A consequence of the design of CNN architectures is their equivariance to translations, or shifts, in the input image. This means that any shift of an object in an image yields a feature map after convolution that is identical to a shift applied to the feature map directly. The convolution operator (C) and the shift transformation (T) is commutative. This can be expressed in the following equation:

$$T(C(x)) = C(T(x)) \tag{1}$$

where x is the input image. This makes it possible to construct translation invariant classifiers by combining convolutional layers with pooling and fully connected layers.

This property does not hold for other distance-preserving transformations such as reflections and rotations. The field of equivariant neural networks extend these properties to both continuous and discrete symmetries. One of the most used ways of achieving this in practice is group-equivariant convolutional networks [8]. At the core of such networks is the G-convolution, which generalizes the convolution operator. First, a symmetry group, i.e. symmetries of an object, is chosen. An example is the P4 group consisting of multiples of 90 degree rotations in addition to translations. Then, the kernel for each filter is transformed by the components of the group. For the P4 group, the kernel is rotated by 0, 90, 180 and 270 degrees, and each output after convolution is stored in a stack of feature maps. Invariant classification follows from a special group pooling layer. This is useful for e.g. rotation-invariant classification of cells.

Several other methods for achieving equivariance to various transformations exist. Attentive Group Equivariant Convolutional Networks combine the attention mechanisms of transformers with group convolutions [24]. The method of learning steerable filters uses learnt basis filters for continuous resolution in orientation [28]. CFNET uses conic convolutions combined with the 2D discrete

Fourier transform to achieve equivariance to the transformations of the P4 symmetry group [7]. Another approach is to transform the filters themselves directly, which has been done for rotations [9]. The framework of E(2)-equivariant steerable CNNs implements many of these methods on the $\mathbb{R}^2$ Euclidean space [27] in a pytorch package. The E(N)-equivariant steerable CNNs framework works by a similar principle on the $\mathbb{R}^3$ Euclidean space [5].

2.2 Empirical Results

Equivariant neural networks have shown several improvements over baseline CNNs, including higher accuracy on the test set with an intact accuracy on the training set, meaning less overfitting. Furthermore, data augmentation steps can be skipped, as shown for biomedical image analysis applications including diagnosis of oral cancer based on cytology, mitosis detection, cell boundary segmentation and vessel segmentation [2,3]. This holds for various imaging modalities, including reflected light microscopy, transmission light electroscopy, and TEM. Similar improvements have been seen in 3D for synthetic datasets [12].

Another improvement is faster convergence during training. This has been demonstrated in different problem settings and modalities, including rotation-equivariant instance segmentation, reinforcement learning, and facial classification based on Zernike moments [4,18,23]. Finally, in contrast to CNNs, equivariant networks have shown to be more data efficient. This means that they learn faster when adding more training data, i.e. their gains in terms of test set accuracy is higher. This has been seen in various domains across biomedical image analysis, e.g., tumor classification [16]. This effect has been shown quantitatively in the following manner. In a log-log plot of test set error versus the amount of data, the data was fitted to a power law. It was then observed that the slope was steeper for the equivariant neural networks than for the CNNs, which has been demonstrated in molecular dynamics [1].

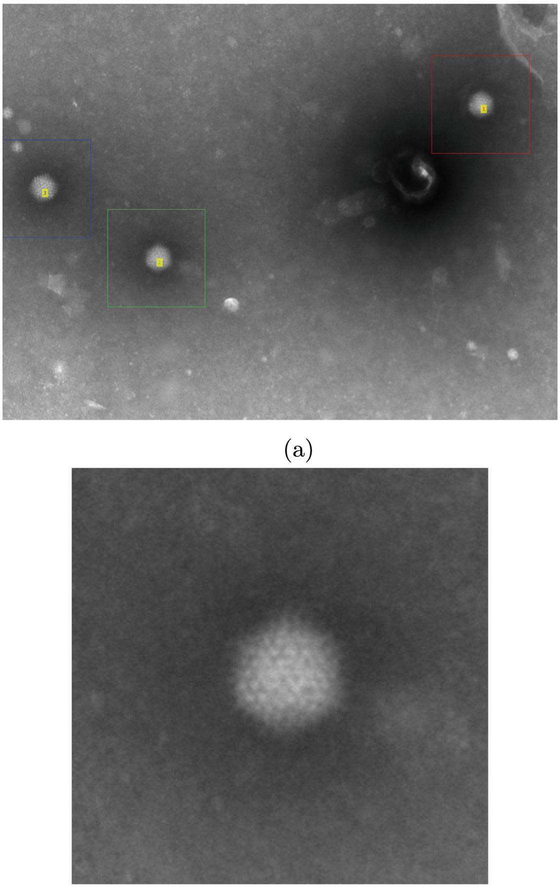
3 Methods

We introduce the TEM virus dataset in Sect. 3.1. Then, the architectures and how to extend their equivariant properties are explained in Sects. 3.2 and 3.3. Details about the symmetry groups, metrics and experimental setups are outlined in Sects. 3.4, 3.5 and 3.6, respectively.

3.1 Data

The dataset consists of Transmission Electron Microscopy (TEM) images containing different types of viruses described in detail in [21]. In brief, samples were prepared by treatment of 10% phosphate-buffered saline, before added to carbon-coated TEM grids and stained with 2% phosphotungstic acid (PTA) following standard negative stain sample preparation procedures. Images were acquired at different magnifications giving pixel sizes ranging from 0.26 to 5.57 nm, with either a LEO (Zeiss, Oberkochen, Germany) microscope mounted with a Morada (Olympus) camera or a Tecnai 10 (FEI, Hillsboro, OR, USA) with a MegaView III (Olympus, Münster, Germany) camera. All images were resized using a Lanczos-3 kernel to the common pixel size 1 nm. Each image is annotated with its corresponding virus class, of which there are 22 in total. The eight least populous classes are excluded, which yields a total of 14 classes.

For the experiments, every image is extracted from larger images containing multiple virus particles, where the cutout images are of size 256×256 pixels, in grayscale and centered around one virus particle. Figure 1 presents Adenoviruses, where global rotation invariance is exhibited in both the raw image and cutout.



(a)

(b)

Fig. 1. Raw image of three Adenoviruses. (a) The virus particles with the cutouts highlighted. (b) Cutout image of the bottom Adenovirus particle from the raw image.

The training data was rebalanced to contain 93 instances for each class, the lowest number found in any original class, for a total of 1302 images. Augmented copies of instances were first removed. After that, the remaining images were randomly removed until 93 instances remained. The training data was then augmented again by rotating the images by 0, 90, 180 and 270 degrees, in addition to a reflection over one axis, resulting in eight images from each original image. No instance originated from the same original raw image between the training, validation and test sets and there was no overlap between images to suppress data leakage. The validation set consisted of 2249 images and the test set of 1900 images. The number of instances per class was varied for both these partitions.

3.2 Architectures

The dataset has previously been classified using several architectures [20]. Top performing models in terms of test set accuracy were chosen for this study. The number was limited to two to strike a balance between collecting statistics and ensuring high quality experiments. The first architecture was VGG16 [26], which scored a test set accuracy of 92.3% when pre-training on the ImageNet dataset. The second was the Custom architecture with a test set accuracy of 90.1% when training from scratch.

The VGG16 architecture was chosen for its proven track record in the biomedical image classification community. It has also previously been modified to be equivariant to the P4 symmetry group [3], making it a prime candidate for further studies using other symmetry groups. The second architecture was the Custom network, which is similar to ResNet but uses about one tenth of the number of parameters. This was selected since the performance of a much smaller architecture would be interesting to investigate under further equivariant constraints.

3.3 Construction of the Networks

For both the VGG16 and Custom architectures, similar approaches were used to construct the networks. First, a baseline network was designed in pytorch using standard convolutions and pooling layers. Secondly, equivariant versions of the same networks were constructed. This was done using the e2cnn package on top of pytorch [27]. Since this allows for flexible changes of symmetry groups, it was simple to use the same model for different experiments with only the symmetry group changed. Layer by layer, the convolutions were replaced with group equivariant counterparts. Group pooling layers were added at the interface between the convolutional layers and the fully connected parts of the networks to extract the signals from any orientation or reflection, making the classifier invariant to the transformations of the chosen symmetry group.

For the Custom network, an additional pre-convolution layer was added at the input side for technical reasons, to avoid concatenating trivial and regular types after the Conv2 layer. The number of trainable parameters was the same for the different versions of the same architecture. The details of the architectures are seen in Fig. 2. Additional settings such as hyper-parameters are listed in Table 1.

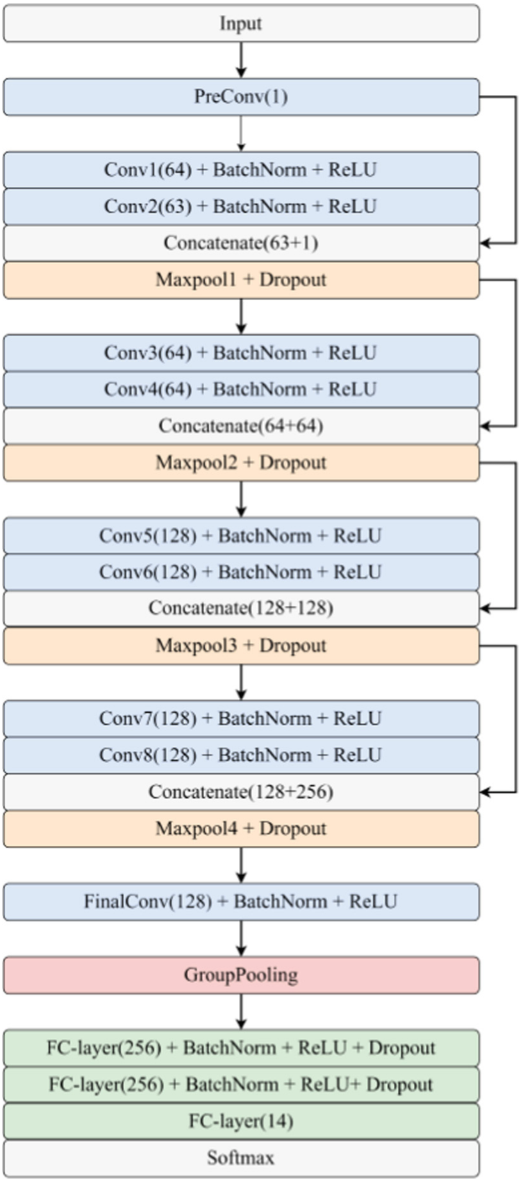

Input
PreConv(1)
Conv1(64) + BatchNorm + ReLU
Conv2(63) + BatchNorm + ReLU
Concatenate(63+1)
Maxpool1 + Dropout
Conv3(64) + BatchNorm + ReLU
Conv4(64) + BatchNorm + ReLU
Concatenate(64+64)
Maxpool2 + Dropout
Conv5(128) + BatchNorm + ReLU
Conv6(128) + BatchNorm + ReLU
Concatenate(128+128)
Maxpool3 + Dropout
Conv7(128) + BatchNorm + ReLU
Conv8(128) + BatchNorm + ReLU
Concatenate(128+256)
Maxpool4 + Dropout
FinalConv(128) + BatchNorm + ReLU
GroupPooling
FC-layer(256) + BatchNorm + ReLU + Dropout
FC-layer(256) + BatchNorm + ReLU+ Dropout
FC-layer(14)
Softmax

Fig. 2. The equivariant version of the Custom architecture used in this study. Dropout starts at a frequency of 0.2 and increases by 0.05 in every convolutional layer. The fully connected layers use the dropout rate from the last convolutional layer. The Custom network uses 63 instead of 64 channels in the second convolutional layer, which is then concatenated with the one channel from the PreConv layer.

Table 1. Settings for the architectures and training procedures.

Parameter	Setting
Loss function	*Cross entropy*
Weight initialization	*He*
Optimizer	*Adam*
Learning rate	*0.00001*
Batch size rate	*32*
Weight decay	*0.1*
No. of epochs	*300*
Activation functions	*ReLu*
Convolution layers - VGG16 (size, stride, padding)	*Layers 1–12: (3,1,1)* *Layer 13: (4,2,0)*
Maxpooling layers - VGG16 (size, stride, padding)	*Layers 1–3: (2,2,0)* *Layer 4: (4,4,0)* *Layer 5: (3,1,0)*
Convolution layers - Custom (size, stride, padding)	*PreConv: (1,1,1)* *Layers 1–8: (3,1,1)* *FinalConv: (4,2,0)*
Maxpooling layers - Custom (size, stride, padding)	*Layers 1–2: (2,2,0)* *Layers 3–4: (4,4,0)*

3.4 Symmetry Groups

Two symmetry groups were selected for the experiments: D4 and C8. The D4 group consists of the rotations by 0, 90, 180, and 270 degrees, in addition to translations and no reflection or reflection over one axis. This group was selected because it matches the transformations in the augmented training data. This implies that an equivariant network using the D4 group does not need to learn each separate transformation, but can learn more useful features of the data instead.

The C8 group consists of the rotations by 0, 45, 90, 135, 180, 225, 270, and 315 degrees in addition to translations. While this does not exactly match the transformations in the augmented data, a finer angular resolution could be useful for the problem setting of rotation-invariant classification. In previous studies, the angular resolution of the C8 group has been the point at which the model's performance saturates [16], which was also observed in this study's early stages of experimentation.

3.5 Metrics

Several metrics are used to evaluate the hypotheses through the experiments. Accuracy on the test set is defined as the percentage of correctly classified samples divided by the total number of samples. Training set accuracy is defined similarly. Both the top test accuracy attained during any part of the training and the accuracy after the final epoch are measured. Stable accuracy is defined as 95% of the top test accuracy. Time until convergence is defined as the time from the start of training until the time when stable accuracy first occurs.

3.6 Experiments

Our experiments involved comparing the performance of the baseline and equivariant networks for the VGG16 and Custom models. The unaugmented and augmented training sets were used to see if the baseline networks with data augmentation could be replaced with equivariant versions without data augmentation. For the equivariant networks, the D4 and C8 symmetry groups were tested.

For the augmented training set, since the transformations in the D4 equivariant group match the augmentations, these models are unlikely to get any additional information in comparison with using the unaugmented training data. Instead, they get eight times the amount of opportunities to train. To control for this effect, the baseline, D4 and C8 models were trained on the unaugmented training set for eight times as long, 2400 epochs, for both architectures.

The impact of batch normalization was assessed by creating additional models without batch normalization. The batch normalization was kept in the fully connected layers. Both the baseline and D4 equivariant models were trained on the unaugmented and augmented training sets.

To investigate any difference in data efficiency between the baseline and the equivariant models, both architectures were additionally trained using 25%, 50%, 75% of the unaugmented training set. Samples were randomly removed without replacement. For the equivariant networks, the D4 symmetry group was used. The experiments were averaged over three runs.

All experiments were run in a national high-performance computing centre. All development and job scheduling was performed by remote access by SSH from an integrated development environment. This provided flexibility since the research and development process was abstracted away from system maintenance. The hardware system that ran the computations consisted of an A40 GPU with 48 GB VRAM, 64 GB System memory and 16 CPU cores per GPU.

4 Results

The results from the main experiments are presented in Table 2. For the Custom architecture on the unaugmented training set, the equivariant model with the D4 symmetry group performed the best at 72.90%, followed by the baseline (69.90%), and finally the C8 symmetry group (64.10%). In contrast, when using the augmented training set, the equivariant models, with scores of 88.4% and 88.7%, respectively, for the D4 and C8 groups, performed slightly better than the baseline, which scored 87.8%.

When the VGG16 architecture was used on the unaugmented training set, the D4 equivariant model performed the best with a score of 85.5%, followed by the baseline (81.9%) and finally the C8 equivariant model (78.2%). However, when using the augmented training set, the C8 equivariant model performed the best with a score of 92.0%, followed by the baseline (91.1%) and finally the D4 model (91.0%).

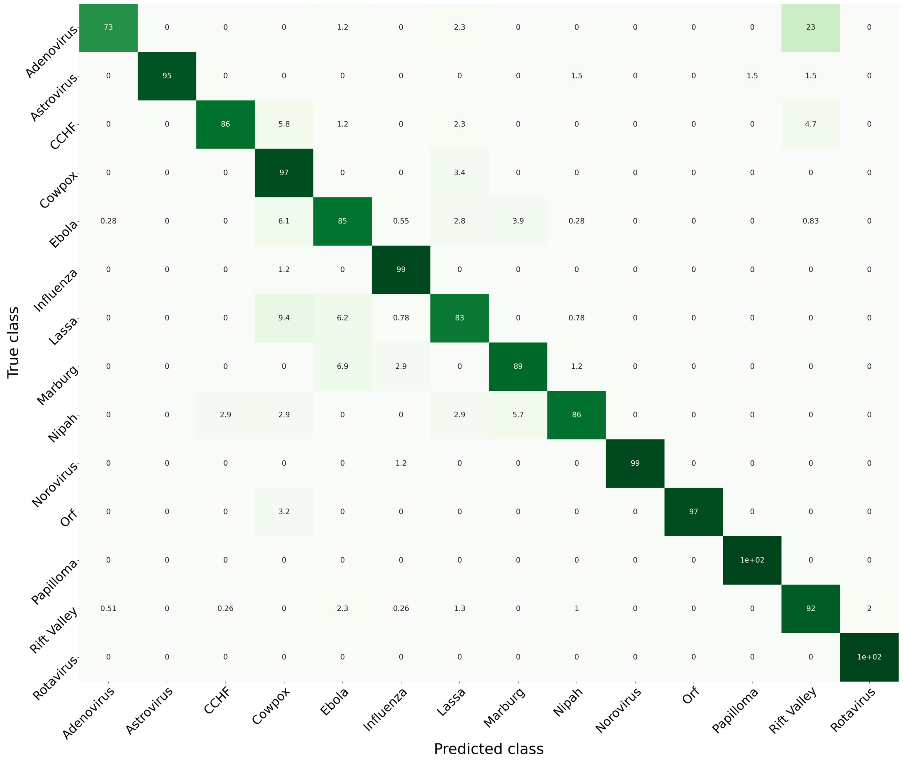
Table 2. Results from the main experiments for the Custom and VGG16 architectures. *Best accuracy* is the percentage correctly classified on the test set and *Time to stability* is in hours.

Custom network	**Baseline**	**D4**	**C8**
No augmentation			
300 epochs			
Best accuracy	**69.90**	**72.90**	**64.10**
Epochs to stability	**274**	**275**	**234**
Time to stability	**0.85**	**7.32**	**3.90**
2400 epochs			
Best accuracy	**83.30**	**89.80**	**89.00**
Epochs to stability	**2260**	**1886**	**2126**
Time to stability	**7.06**	**48.72**	**55.61**
Augmented training			
300 epochs			
Best accuracy	**87.80**	**88.40**	**88.70**
Epochs to stability	**244**	**212**	**2 56**
Time to stability	**4.46**	**33.54**	**40.54**

VGG16 network	**Baseline**	**D4**	**C8**
No augmentation			
300 epochs			
Best accuracy	**81.90**	**85.50**	**78.20**
Epochs to stability	**266**	**262**	**184**
Time to stability	**1.33**	**12.80**	**9.04**
2400 epochs			
Best accuracy	**87.00**	**92.40**	**92.50**
Epochs to stability	**1478**	**1474**	**1667**
Time to stability	**7.57**	**73.67**	**84.01**
Augmented training			
300 epochs			
Best accuracy	**91.10**	**91.00**	**92.00**
Epochs to stability	**145**	**253.4**	**275**
Time to stability	**4.28**	**75.09**	**81.28**

When the number of training epochs increased from 300 to 2400 on the unaugmented training set, for both the equivariant models and both the architectures, higher accuracy was reached. The same was true for the baseline model which reached 83.3% and 87.0% on the Custom and VGG16 architectures, respectively. However, the final accuracy reached was not comparable to the equivariant models'. On the Custom architecture, the scores were 89.8% and 89.0% for the D4 and C8 groups, respectively. For VGG16, the scores were 92.4% and 92.5%.

When comparing the accuracy of specific classes in Fig. 3, only Adenovirus showed an accuracy below 80%, where most mispredicted instances were instead classified as Rift Valley. The prediction of the Papilloma and Rotavirus were fully accurate.



	Adenovirus	Astrovirus	CCHF	Cowpox	Ebola	Influenza	Lassa	Marburg	Nipah	Norovirus	Orf	Papilloma	Rift Valley	Rotavirus
Adenovirus	73	0	0	0	1.2	0	2.3	0	0	0	0	0	23	0
Astrovirus	0	95	0	0	0	0	0	0	1.5	0	0	1.5	1.5	0
CCHF	0	0	86	5.8	1.2	0	2.3	0	0	0	0	0	4.7	0
Cowpox	0	0	0	97	0	0	3.4	0	0	0	0	0	0	0
Ebola	0.28	0	0	6.1	85	0.55	2.8	3.9	0.28	0	0	0	0.83	0
Influenza	0	0	0	1.2	0	99	0	0	0	0	0	0	0	0
Lassa	0	0	0	9.4	6.2	0.78	83	0	0.78	0	0	0	0	0
Marburg	0	0	0	0	6.9	2.9	0	89	1.2	0	0	0	0	0
Nipah	0	0	2.9	2.9	0	0	2.9	5.7	86	0	0	0	0	0
Norovirus	0	0	0	0	0	1.2	0	0	0	99	0	0	0	0
Orf	0	0	0	3.2	0	0	0	0	0	0	97	0	0	0
Papilloma	0	0	0	0	0	0	0	0	0	0	0	1e+02	0	0
Rift Valley	0.51	0	0.26	0	2.3	0.26	1.3	0	1	0	0	0	92	2
Rotavirus	0	0	0	0	0	0	0	0	0	0	0	0	0	1e+02

Fig. 3. The D4 equivariant VGG16 model's confusion matrix of the final epoch on the test data, trained on the unaugmented train set for 2400 epochs.

For the Custom architecture, training on the augmented training set for 300 epochs yielded faster convergence than when training on the unaugmented training set for 2400 epochs. This was true for both the baseline (4.46 vs 7.06 h) D4 equivariant (33.54 vs 48.72 h) and C8 models (40.54 vs 55.61 h). Also, for the

VGG16 architecture, this held for the baseline model (4.28 vs 7.57 h). For the equivariant models, the convergence times were similar for both training sets; 75.09 vs 73.67 h for the D4 model, and 81.28 vs 84.01 h for the C8 model. The D4 equivariant models generally converged faster than the C8 models, except when using unaugmented training data for 300 epochs.

When batch normalization was removed, the baseline networks failed to converge, attaining top accuracies of around 20%. This was true for both the Custom and VGG16 architectures. In contrast, the D4 equivariant network converged and attained fair accuracies of 64.10% using the Custom architecture with augmented data. For the VGG16 architecture with augmented data, the top accuracy was 72.6%. Still, this was significantly lower than with batch normalization added. The D4 model converged earlier when trained on the augmented training set compared to the unaugmented training set, which after 300 epochs had not attained full accuracy on the training set.

When measuring the error on the test set versus varying amounts of unaugmented training data and modelling a power law in a log-log plot, the results were different depending on the underlying architecture, see chart in Fig. 4. For the Custom network, there is no discernible difference between the slope of the baseline (−0.67) and D4 equivariant (−0.66) models. However, for the VGG16 architecture, the D4 equivariant model has a slope of −1.05 compared to the baseline's −0.75.

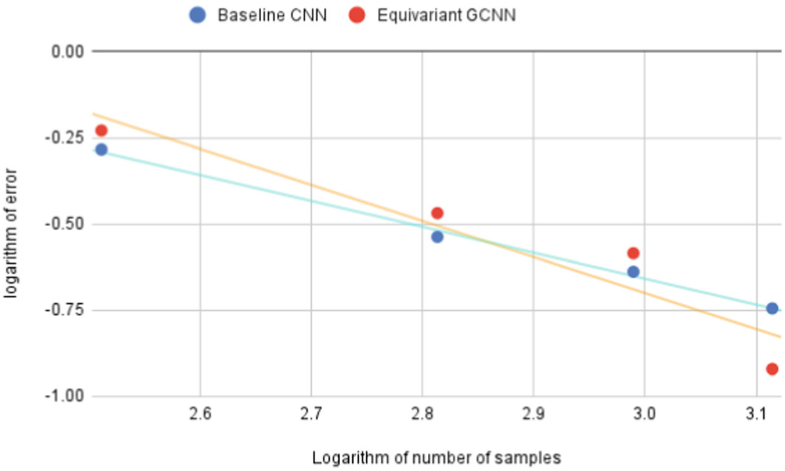


Fig. 4. Modelling a power law for the VGG16 network and varying number of training samples for both the baseline and D4 equivariant models. 25%, 50%, 75%, and 100% of the training data were used, and the error on the test set was measured. Taking the logarithm of both the data and the error resulted in the chart. Simple linear regression resulted in regression slopes of −0.75 and −1.05 for the baseline and D4 equivariant models, respectively.

5 Discussion

When training on unaugmented data for 300 epochs, the choice of symmetry group impacts the test set accuracy. In fact, the D4 equivariant network outperforms the baseline network, which in turn outperforms the C8 equivariant network. This might be due to the C8 group having too fine angular resolution for the problem setting, while the D4 group offers more significant variation to the training data. Also, it is possible that larger filter kernels would be needed to maximize the impact of the finer rotations in the C8 group.

On the other hand, when training on augmented data, all networks achieve similar top accuracy. Equivariant networks are thus able to adapt better to skipping data augmentations, and still benefit from them. Generally, the equivariant and baseline networks converge in a similar number of epochs.

However, the time to convergence is higher for equivariant networks since the feature maps are stacked with the component transformations. The longer training times could be reduced in the future with hardware improvements or by distributing the computations across multiple GPUs, i.e., using distributed deep learning. The results are similar for both architectures, except for the data efficiency experiments, where the VGG16 D4 equivariant model is more data efficient than the baseline network. This follows from the steeper slope in a log-log plot of test set error versus amount of data, while the same does not hold for the Custom model.

The results from turning off batch normalization show that this optimization method is critical for high accuracy. However, the equivariant networks suffer less than baseline models, which fail to converge at all. The mechanism by which batch normalization improves accuracy in general is unclear, but reducing internal covariate shift [14] and smoothing the objective function [25] are two main hypotheses. Hence, our experimental results support the notion that batch normalization and equivariant networks capture variations in the training data more efficiently by different mechanisms, since both model components improve accuracy. By the same reasoning, data augmentation and equivariant networks extract improved information from the data by different means.

The experiments suggest that increasing the number of training epochs considerably is crucial for achieving high accuracy. When increasing the number of epochs from 300 to 2400, both the baseline and the equivariant models increased the test set accuracy significantly. The equivariant models reached consistently higher accuracies than the baseline. For both architectures, the top accuracies were comparable to the top performing models in previous research on the same dataset [20].

6 Conclusion

This study contributes to a deeper understanding of the characteristics of equivariant neural networks during training and testing. By increasing the number of epochs significantly, we show that equivariant models trained on unaugmented data for more epochs could replace pre-trained baseline models using augmented training data since their accuracies on the test set were similar. Thereby, performance engineering is a good choice when the computational resources are available.

Further research should focus on more advanced equivariant architectures such as steerable convolutional neural networks or equivariant transformer networks which are becoming modern and step by step entering the field of biomedical image analysis. Designing networks with different symmetry groups in different layers depending on the scale of the symmetries in the data could also lead to additional improvements when classifying objects of interest in biomedical images.

Acknowledgments. The project is funded by WASP (Wallenberg AI, Autonomous Systems and Software Program). The computations were enabled by resources provided by the National Academic Infrastructure for Super-computing in Sweden (NAISS) at Chalmers University of Technology partially funded by the Swedish Research Council through grant agreement no. 2022-06725.

References

1. Batzner, S., et al.: E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. Nat. Commun. **13**(1), 2453 (2022). https://doi.org/10.1038/s41467-022-29939-5
2. Bekkers, E.J., et al.: Roto-translation covariant convolutional networks for medical image analysis. In: Frangi, A.F., Schnabel, J.A., Davatzikos, C., Alberola-López, C., Fichtinger, G. (eds.) MICCAI 2018. LNCS, vol. 11070, pp. 440–448. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00928-1_50
3. Bernander, K.B., Lindblad, J., Strand, R., Nyström, I.: Replacing data augmentation with rotation-equivariant CNNs in image-based classification of oral cancer. In: Tavares, J.M.R.S., Papa, J.P., González Hidalgo, M. (eds.) Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, pp. 24–33. Springer, Cham (2021)
4. Bernander, K.B., Lindblad, J., Strand, R., Nyström, I.: Rotation-equivariant semantic instance segmentation on biomedical images. In: Yang, G., Aviles-Rivero, A., Roberts, M., Schönlieb, C.B. (eds.) Medical Image Understanding and Analysis, pp. 283–297. Springer, Cham (2022)
5. Cesa, G., Lang, L., Weiler, M.: A program to build e(n)-equivariant steerable CNNs. In: International Conference on Learning Representations (2022). https://openreview.net/forum?id=WE4qe9xlnQw
6. Chakraborty, S., Mali, K.: An overview of biomedical image analysis from the deep learning perspective. In: Research Anthology on Improving Medical Imaging Techniques for Analysis and Intervention, pp. 43–59 (2023)
7. Chidester, B., Zhou, T., Do, M.N., Ma, J.: Rotation equivariant and invariant neural networks for microscopy image analysis. Bioinformatics **35**(14), i530–i537 (2019). https://doi.org/10.1093/bioinformatics/btz353
8. Cohen, T., Welling, M.: Group equivariant convolutional networks. In: Balcan, M.F., Weinberger, K.Q. (eds.) Proceedings of the 33rd International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 48, pp. 2990–2999. PMLR, New York (2016). https://proceedings.mlr.press/v48/cohenc16.html

9. Dieleman, S., Fauw, J.D., Kavukcuoglu, K.: Exploiting cyclic symmetry in convolutional neural networks. In: Balcan, M.F., Weinberger, K.Q. (eds.) Proceedings of The 33rd International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 48, pp. 1889–1898. PMLR, New York (2016). https://proceedings.mlr.press/v48/dieleman16.html
10. Dosovitskiy, A., et al.: An image is worth 16x16 words: transformers for image recognition at scale. In: International Conference on Learning Representations (2021). https://openreview.net/forum?id=YicbFdNTTy
11. Dziugaite, G., Roy, D.: Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. ArXiv (2017)
12. Gerken, J., Carlsson, O., Linander, H., Ohlsson, F., Petersson, C., Persson, D.: Equivariance versus augmentation for spherical images. In: International Conference on Machine Learning, pp. 7404–7421. PMLR (2022)
13. Hestness, J., et al.: Deep learning scaling is predictable, empirically. arXiv preprint arXiv:1712.00409 (2017)
14. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift (2015)
15. Kim, H., Nam, H., Jung, W., Lee, J.: Performance analysis of CNN frameworks for GPUS. In: 2017 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), pp. 55–64 (2017). https://doi.org/10.1109/ISPASS.2017.7975270
16. Lafarge, M.W., Bekkers, E.J., Pluim, J.P., Duits, R., Veta, M.: Roto-translation equivariant convolutional networks: application to histopathology image analysis. Med. Image Anal. **68**, 101849 (2021)
17. LeCun, Y.: 1.1 deep learning hardware: past, present, and future. In: 2019 IEEE International Solid- State Circuits Conference - (ISSCC), pp. 12–19 (2019). https://doi.org/10.1109/ISSCC.2019.8662396
18. Mahesh, V.G., Raj, A.N.J., Fan, Z.: Invariant moments based convolutional neural networks for image analysis. Int. J. Comput. Intell. Syst. **10**, 936–950 (2017). https://doi.org/10.2991/ijcis.2017.10.1.62
19. Mathew, A., Amudha, P., Sivakumari, S.: Deep learning techniques: an overview. In: Hassanien, A.E., Bhatnagar, R., Darwish, A. (eds.) AMLTA 2020. AISC, vol. 1141, pp. 599–608. Springer, Singapore (2021). https://doi.org/10.1007/978-981-15-3383-9_54
20. Matuszewski, D.J., Sintorn, I.M.: TEM virus images: benchmark dataset and deep learning classification. Comput. Methods Prog. Biomed. **209**, 106318 (2021)
21. Matuszewski, D., Sintorn, I.M.: Tem virus dataset. Mendeley Data (2021). https://data.mendeley.com/datasets/x4dwwfwtw3
22. OpenAI. Gpt-4 technical report (2023)
23. van der Pol, E., Worrall, D., van Hoof, H., Oliehoek, F., Welling, M.: MDP homomorphic networks: group symmetries in reinforcement learning. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) Advances in Neural Information Processing Systems, vol. 33, pp. 4199–4210. Curran Associates, Inc. (2020). https://proceedings.neurips.cc/paper/2020/file/2be5f9c2e3620eb73c2972d7552b6cb5-Paper.pdf
24. Romero, D., Bekkers, E., Tomczak, J., Hoogendoorn, M.: Attentive group equivariant convolutional networks. In: ICML, pp. 8188–8199 (2020)
25. Santurkar, S., Tsipras, D., Ilyas, A., Madry, A.: How does batch normalization help optimization? (2018)

26. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations. OpenReview (2015)
27. Weiler, M., Cesa, G.: General e(2)-equivariant steerable CNNs. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 32. Curran Associates, Inc. (2019). https://proceedings.neurips.cc/paper/2019/file/45d6637b718d0f24a237069fe41b0db4-Paper.pdf
28. Weiler, M., Hamprecht, F., Storath, M.: Learning steerable filters for rotation equivariant CNNs. In: Conference on Computer Vision and Pattern Recognition (CVPR), pp. 849–858 (2018). https://doi.org/10.1109/CVPR.2018.00095

AI Based Story Generation

Neha Sharma(✉), Prashant Karwasra, Paritosh Sharma,
and Md Arbaz Tahir

Indian Institute of Information Technology Una, Una 177209, India
neha724@gmail.com

Abstract. We utilized natural language processing (NLP) to craft a new story inspired by the iconic Harry Potter and Lord of the Rings series. Initially, we deployed LSTM for text generation that gave us limited set of results, then the capabilities of GPT-2 were explored, revealing challenges with grammatical correctness in the generated sentences. Subsequently, we integrated GPT-2, which significantly improved the quality of the story generation. GPT-2 not only produced logical narratives but also breathed life into the characters and plotlines, offering a rich source of inspiration for writers seeking novel ideas and fresh perspectives. We further used a stable diffusion model to create images that matched the generated story. Thus we attempted a well-rounded method to bring together both words and pictures, creating a storytelling sequence for writers and creators. The paper highlights how AI story generation can effectively help support the creation of stories in gaming, education and entertainment.

Keywords: Story Generation · Natural Language Processing (NLP) · GPT-2 · LSTM · Visual Storytelling

1 Introduction

Stories play a vital role in every culture, captivating people of all ages and serving as sources of entertainment, moral lessons, and wisdom. Recently, they've become tools for educating children [1]. Computational creativity, based on the interplay of psychology and artificial intelligence, involves teaching computers to generate original and valuable stories. To achieve this, understanding human creativity is essential, allowing computer scientists to replicate the human thought process in storytelling algorithms. In this paper, AI delves into an exciting and evolving field - using AI for creative tasks like story generation.

Automatic story generation has wide-ranging applications, including entertainment [1], education, and gaming. It involves mechanically selecting events to form narratives that meet specific criteria. These stories incorporate elements like characters, objects, and author goals, representing the intended message. In gaming, interactive stories enhance player engagement, while in education, stories can be tailored to individual learner needs. Despite the inception of story

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 32–47, 2025.
https://doi.org/10.1007/978-3-031-78128-5_3

generation systems in the early 1960s [2], they are still considered weak AI systems, as they have not achieved creativity comparable to humans [3]. Efforts to make computer systems creative in story generation involve crafting narratives that are unique and encompass various elements like settings, character motivations, and conflicts. Open story generation, without predefined models, adds complexity in the construction and evaluation of stories. Computational narrative research has focused on categorizing systems based on their automation levels and addressing challenges in generating engaging stories. In 2009, Gervás [4] scrutinized how story generation systems imitate human creativity and the extent to which they incorporate essential features of computational creativity. Since then, the field of computational narrative has garnered increased attention from the research community, resulting in the introduction of numerous automatic story generators. Young et al. [5] carried out a survey focusing on planning and reasoning within computational narratives. A more recent survey by Kybartas and Bidarra [6] categorized story generation systems based on the extent of automation in plot and space generation into four primary categories: manual authoring, plot generation, space generation, and a hybrid form of story generation that automates elements of both plot and space.

The text generation process involves training a language model on extensive text data, enabling it to predict the next characters in a sequence based on the input provided. For instance, given the sequence 'Cat likes mil', the model predicts the next character 'k'. In a referenced example [7], an LSTM model is trained on conditioning text, and the Softmax activation function is applied to predict the probability of the next character. The key to text generation lies in the sampling strategy, where introducing randomness through a temperature value ensures novelty and creativity in the generated text [8].

Creating a coherent narrative is a challenge for writers, impacted by factors like text type and readability (Begeny & Greene, 2014; Sáenz & Fuchs, 2002) [9]. In education, storytelling is integral to children's language development and reading skills (Agosto, 2016; Miller & Pennycuff, 2008; Peck, 1989) [10]. Fluency and comprehension are vital for learning, influencing academic success (Bigozzi et al., 2017) [11]. Teachers use a variety of literature to improve these skills, adapting to online resources and tools for assessing comprehension (Bulut et al., 2022) [12], reflecting the ongoing need for supporting online reading growth.

In this project we combined Natural Language Processing (NLP) with Books Dataset to inspire writers with fresh ideas. We used models like LSTM and GPT-2, with the purpose of generating captivating and unique stories. GPT-2 proved particularly effective in achieving this goal. Post-processing ensured clarity and error correction in the generated text. Additionally, a stable diffusion model was used to generate visuals to complement the narrative, providing writers with a comprehensive storytelling experience.

2 Models

In this section, we detail the working of model architectures based on Long Short-Term Memory (LSTM) [13] and Generative Pre-trained Transformer-2 (GPT-2) [15].

2.1 LSTM

The model accepts a sequence of words or tokens as input, with each word typically represented using word embeddings to convert discrete words into continuous vector representations. These embeddings capture semantic relationships between words. The input sequences then pass through an embedding layer, which transforms the discrete word representations into dense vectors of fixed size which enables the model to learn meaningful and continuous representations for words, taking into account their context in the input sequence. The core of the architecture comprises Long Short-Term Memory (LSTM) layers, which are recurrent neural networks designed to capture long-term dependencies in sequential data. Each LSTM unit possesses memory cells capable of storing and retrieving information over extended sequences, along with gates (input, forget, output) to regulate the flow of information through the network (Fig. 1).

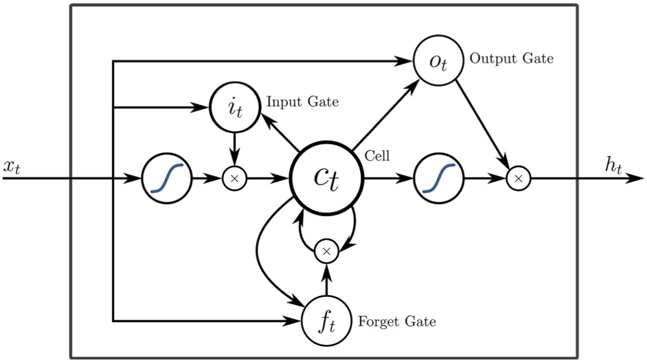


Fig. 1. Lstm Single Unit Architecture

The LSTM layers maintain both hidden states and cell states. Hidden states capture the context or memory of the sequence up to a certain point, while cell states store information over longer periods. These states are updated at each time step as the model processes the input sequence. Dropout layers, if included, are optional and may be added between LSTM layers for regularization. During training, dropout randomly sets a fraction of input units to zero, aiding in preventing the model from overly relying on specific features and thereby improving generalization. The final layer (output layer) is responsible for generating predictions that typically consists of a dense layer with a softmax activation function.

This layer produces a probability distribution over the vocabulary, indicating the likelihood of the next word in the sequence. During the training process, the input sequences are fed into the model and the predicted output is compared to the actual target sequence. The model is optimized using backpropagation and gradient descent techniques to minimize the difference between the predicted and true outputs.

2.2 GPT-2

The key components of GPT-2's architecture include:

GPT-2 utilizes the transformer architecture, known for its effectiveness in processing sequential data, enabling it to capture intricate dependencies. An attention mechanism is employed to weigh the significance of input sequence parts, crucial for comprehensive context consideration, especially in text generation tasks. Each transformer layer includes layer normalization and feedforward networks, enhancing stability and introducing non-linearity for discerning intricate patterns. Positional embeddings are incorporated to provide token position information, essential for understanding sequential order and relationships between tokens. Multi-head self-attention allows GPT-2 to attend to various positions in the input sequence simultaneously, enhancing its ability to capture diverse dependencies and relationships within the text (Fig. 2).

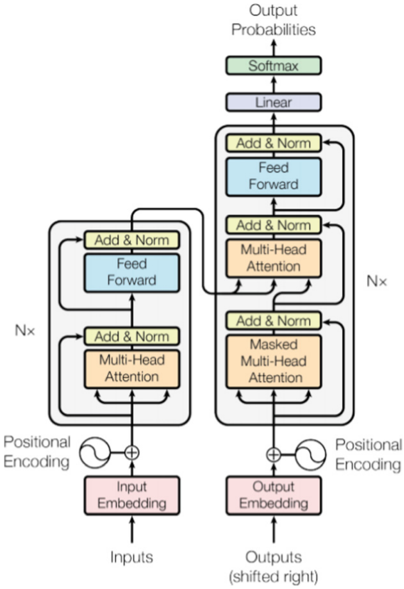


Fig. 2. GPT Architecture

PT-2 employs multiple layers of the transformer architecture stacked on top of each other, enhancing its capacity to capture intricate patterns. Within each layer, position-wise feedforward networks operate independently at each sequence position, contributing to the model's ability to understand diverse patterns. During training, GPT-2 uses masked self-attention to ensure each input position only attends to preceding positions, preventing access to future token information. Trained with an autoregressive language modeling objective, GPT-2 predicts the next token based on preceding ones, capturing statistical patterns in the training data. With its large scale, featuring numerous parameters and attention heads, GPT-2 excels in generating high-quality and diverse text, albeit requiring substantial computational resources.

2.3 Stable Diffusion

The stable diffusion model begins with initializing a noise image drawn from a simple distribution, such as Gaussian, serving as the starting point. It then proceeds with a series of diffusion steps, gradually adding controlled noise to the current image to transform it into samples resembling the target distribution. At each step, noise addition increases gradually, crucial for learning intricate data distribution details. The transformation of the image during diffusion steps is guided by a neural network, typically a generative model like a Variational Autoencoder (VAE) or Generative Adversarial Network (GAN), which captures patterns and features present in real data (Fig. 3).

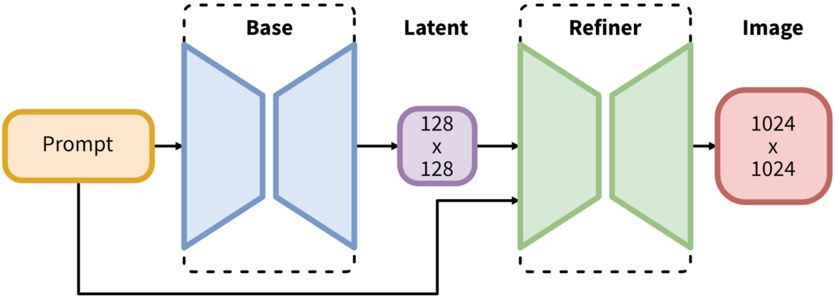


Fig. 3. Stable Diffusion Architecture

The stable diffusion [16] model is trained to minimize the difference between generated images and the real data distribution, often using maximum likelihood estimation. This aims to generate samples statistically similar to the true data. Designed hierarchically, the diffusion process allows the model to learn global and local features of the data distribution, aiding in capturing complex structures and dependencies in the images. Numerical stability ensures robust computations, preventing instability or divergence and ensuring convergence to realistic images during training. Upon completion of the diffusion process, the final generated image is obtained, with the option to sample from different stages to observe the progression from the initial noise image to the final generated image.

3 Dataset

For this study, we needed extensive data and diverse fantasy narratives. To foster innovation and creativity, we selected multiple well-known books from various fantasy genres and distinct universes, aiming to enhance the model's creative capabilities. In this approach, the Books Dataset was compiled by bringing together different volumes of literary works, including Harry Potter [17], The Hobbit [18], The Lord of the Rings [19], and The Silmarillion [20]. Additionally, this dataset was utilized to train GPT-2 model.

To train the base LSTM model, we leverage a dataset comprising the text of "The Jungle Book" [21] due to architectural constraints. The LSTM architecture faces limitations in accommodating extensive textual corpora such as complete book datasets.

3.1 Data Preprocessing

The initial step in data preprocessing involved cleaning the text data to enhance its consistency and readability. This entailed removing non-alphanumeric characters and addressing any inconsistencies in whitespace by replacing multiple consecutive spaces with a single space. The objective here was to establish a more uniform and cohesive representation of the textual content.

Following the cleaning process, the cleaned text from all scripts was concatenated and any leading or trailing whitespaces were removed. This ensured that the text data remained concise and free from unnecessary spacing issues.

Finally, the processed text data was stored in a file named "cleaned_text_data.txt" for subsequent analysis and further processing. This step facilitated easy access to the cleaned data for various analytical tasks and helped streamline the subsequent stages of the data analysis pipeline.

4 Results

4.1 LSTM

Input: Mowgli and Bagheera were playing in the forest, and then suddenly Sher Khan appeared.

Output Generated: *Mowgli and bagheera were playing in the forest and then suddenly sher khan appeared begged pots stare pronounced augrh arm dashed rid turn grew thinking whistled buildings show animal blind outward advice ts elders raw hill afterward point moons beating mao naked dig whistled forgot anxious sister boldly flung pot telling companions regard blossomed when brown struggle h year leaves lash india freed corners wishing hilltop escort drive marks tending dewanee wise names giant gate dimpled days milk mean she clung regard org hand revenged field sort piteous describe coals cleverly akela swayed slowly inspection folk sitting noiseless animal killing cobras shape pity remind there yawned freed* *copy pits rain rises sought circled roughly abreast rumbling fur blocked hauled overhear trot relief lumps withdraw spoiled therefore raw helped rolled coils louder right won another part bushy threatened firmly temper ah roost change gasped on set suspicious ignorance unlike temper entered himself loud thy belonged night khanhiwara bound fright starved me shower ruins over touched finding against curiously shame nearly doubt busy listened re wise games sides ran steel hurried sort accidents feeling and especially order give becomes curve suffered bounds push son knowing brilliant living covered frogs bad forest roots date heads shoulder belled curves pawpaws courtyards at give accepted milk built stream louder toward psshaw check knowest plucked speaker lose indian platform herself fallen squarely canes gorge swept mouths ask hanging kite cool jacket trot these whirled licker talked forefoot keep roused unlike besides stump ignorance crowds idea listener bricks boldly changed charge faded bring identify quarry knew why image indeed just glimpses teach beneath oodeypore beaten hillside repeated am stream behave bamboos gift whined regard phew coughing overtaking marrow drink dangerous biting nuts called child bark sweeping climb musket dressed dreaming whither envied circle dealt hole screen illo 2023 joy remembering before barber scattered hopefully lash baloo leaders heard halfway call bristles sorts ahai whoo sign part lazy fairs inspection causeways buck against everything empty yet pet hugged doubt journey suffer worse rage own white twelve merely be happened rudyard newly friend our sixty deserted altogether club path plot bold needs possible shower orders tails frog scumfish thundered january along barked affairs stillness beneath satisfied tearing worse come whine reached lowered rolled held collect waingunga cleverly crops honor amounted boast unlucky panted our swash carrying hunting winter have could beckoned pits out thief tribe lapis entrance hurrying heads yellow porcupine content mouths forgets odds sorely get afoot shadow heap grow things neither little becomes bull taking sorrowfully clung toomai humming afraid farewell bats upon morrow lamer yard swim chuckle chattering trapped breaking striking noble sleep mountain more wouldst stuck whooping breed lord ye goest worked dry swung proud cost amounted rains khanhiwara ram fastenings customs disappeared re fin pickers weakness gets hunts monkeys feet as blow roosted notion lessons loved goer squarely wound ago crowds lying mean book quarters accepted bristling night wild no feather misses wonderful this feather history canst behave stung swam honey me peacock work stung sprang blow woodcutter villagers grass branches fear move brother toomai lied proper skin tabaqui challenge footless summerhouse allowed five future miserable neither prettily honey owned threshold squarely crossing majesty anxious contents blind sided empty forgets ways anxiously day branch called olds throat roar cost feebler broke grass speed off melted roar promised winds customs avoid animals tonight beauties stupid enemies dancing dog pariah licked softly cobra whirled believed coughing twos splashed today.*

4.2 GPT-2

Input: The story began with Harry Potter, Hermione Granger, and Ron Weasley inadvertently stumbling upon an ancient portal while exploring the

Forbidden Forest during their final year at Hogwarts. Upon emerging from the portal, they found themselves in a vast and dense forest, surrounded by towering trees. To their surprise, they were greeted by none other than Frodo, Sam, Merry, and Pippin, the beloved hobbits from the Shire. Harry was surprised to see hobbits and then decided to befriend them.

Output Generated: *The story began with Harry Potter, Hermione Granger, and Ron Weasley inadvertently stumbling upon an ancient portal while exploring the Forbidden Forest during their final year at Hogwarts. Upon emerging from the portal, they found themselves in a vast and dense forest, surrounded by towering trees. To their surprise, they were greeted by none other than Frodo, Sam, Merry, and Pippin, the beloved hobbits from the Shire. Harry was surprised to see him, but he was pleased to see that he had a good sense of humor. He was also pleased that they had found a place to sit down together, because they felt that the time was ripe for a serious discussion. The forest was full of trees, which made it ideal for the discussion, as it was so large and full that it seemed to be full even for an Ent. They were not alone in this, however. There were many more Ents in the forest than there were Entwives, who were the descendants of the ancient Ent, descendants from ancient trees that lived in and around the Forest. Some of them were even taller than Harry, though they looked much younger, with long, pointed hair and long noses. One of these was called the Big One, for he looked like a tall, balding, hairy man with a long beard and a mustache that was longer than his hair. His hair was long and curly, too, so that his face looked rather like the face of a large, bearded man. It was a pleasant surprise to find that Frodo's face was not quite as round as the others, except that there was no mustache. Frodo was tall and handsome, very handsome. I don't know what you mean, said Sam. Well, I mean that, he said, looking at them. You know, you know that I am not a hobbit, or a wizard. But I have never been in it. And I've never seen it before. What do you think you're talking about said Merry. A forest said Pompously. That's not the place you are talking of, is it said Frodon. Not the one I was in, not that one. No, no, it is not, Mr. Baggins. Its the other one, I'm sorry. I'm not sorry, Fro, its the old Ent that you've got to go and see. We've come back, we've gone and seen the Ent before, yes, that's right. Yes, well, Ill see you, if you don't mind. Merry and I went down to the Great Hall and sat down in chairs, talking to each other. Then we sat on the floor, thinking about the past and the present. At last they sat up and began to talk. After a while they talked for some time, until they stopped. When they spoke of their past, there came a knock on their door. Mr, Baggin came in. and knocked. Come in Come into the hall, come in said Mr Frobin's, coming in We have a present, a nice little present for you. This is a big present from Mr Bilbo BaggINS. Bilbo's friend. Very nice, sir, answered the knocker. Good gracious, good gracious. So you have come into my house, my dear Frodo, just now. How nice said the landlord. Thank you very much, thank you Very good, Master Bilbob. Now, now, what is this I want to know about you I cant tell you what I meant, nor can I tell the truth. Why don't you come and sit in my room, while I talk to you said Bilbo. Because I think I* *must be getting very tired. If you want a drink, then you must go to bed. Master, please, sit now and think of me, before you go back to your room. Chapter 2 The Shrieking Shack The next day they went to breakfast. Sam was sitting in his chair, listening to all the talk, when suddenly he heard a noise. Something was coming from a corner of his head. Someone was shouting. Somebody was calling. Whats the matter with you Its coming, whats the trouble Its all right with us, all ruffians, rags, gollum, lembas. Gollum's going to get us. Ill get him. Lets get out of here. All right, go on, quick, hurry, Gently Sam crept out. As he crept nearer, his master saw that Gildor was standing by the door, holding a knife. Quick, master, quickly, let us go, lets go Galdor came running. With a cry he sprang out and ran back. Before he could catch up with him, Bilbo had fallen to his knees beside him and was bleeding. Suddenly he fell to a dead end.*

4.3 Stable Diffusion

The textual outcomes from previous models was further used as input to the publicly accessible stable diffusion model [22] to seamlessly convert the textual outputs into visually engaging images. The publicly accessible stable diffusion model available. Enable to introduce a creative visual element to the generated stories. Despite encountering computational resource constraints on the system, the model was able to effortlessly translate textual content into aesthetically appealing images (Figs. 4 and 5).

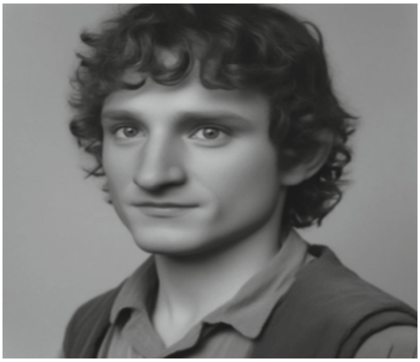


Fig. 4. It was a pleasant surprise to find that Frodo's face was not quite as round as the others, except that there was no mustache. Frodo was tall and handsome, very handsome.

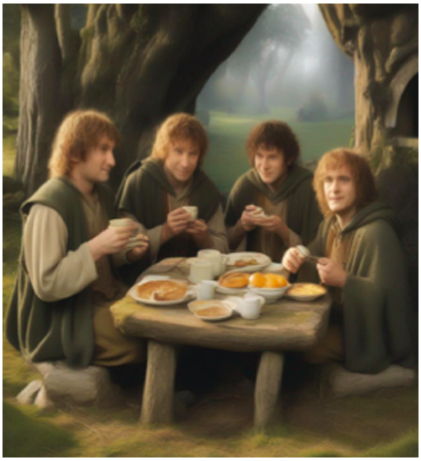


Fig. 5. The next day hobbits went to breakfast.

5 Evaluation Metrics

5.1 AI Evaluation (GPT-4o)

To evaluate the quality of the essays, we utilized a comprehensive scoring system based on the following criteria (Tables 1, 2, 3, 4, and 5):

- **Grammar (20 points)**
  - This criterion assesses the correctness of grammatical structures, including sentence construction, subject-verb agreement, and punctuation.
  - Score Range: 0–20 points
- **Spelling (20 points)**
  - This criterion evaluates the accuracy of spelling throughout the essay.
  - Score Range: 0–20 points
- **Coherence (20 points)**
  - This criterion examines the logical flow and connectivity of ideas within the essay.
  - Score Range: 0–20 points
- **Context (20 points)**
  - This criterion evaluates the clarity and relevance of the context within which the essay is written.
  - Score Range: 0–20 points
- **Syntax (20 points)**
  - This criterion assesses the correctness of sentence structure and word order.
  - Score Range: 0–20 points

Table 1. Grammar (20 points) Grammar Evaluation Matrix

Number of errors:	Points Range
0–5 errors:	18–20 points
6–10 errors:	15–17 points
11–15 errors:	12–14 points
16–20 errors:	8–11 points
21+ errors:	0–7 points

Table 2. Spelling (20 points) Spelling Evaluation Matrix

Number of errors:	Points Range
0–2 errors:	18–20 points
3–5 errors:	12–17 points
6–8 errors:	6–11 points
9+ errors:	0–5 points

The total score for each essay is calculated by summing the points awarded in each criterion, with a maximum possible score of 100 points. Essays are then classified based on their total score to determine their overall quality.

Table 3. Coherence and Clarity Evaluation Matrix

Evaluation Criteria	Points Range
Excellent logical flow and clarity:	18–20 points
Good with minor lapses:	15–17 points
Fair but with noticeable issues:	12–14 points
Poor with significant issues:	8–11 points
Very poor and hard to follow:	0–7 points

Table 4. Context Evaluation Matrix

Evaluation Criteria	Points Range
Highly relevant and on-topic:	18–20 points
Mostly relevant with minor deviations:	15–17 points
Some relevance but with major deviations:	12–14 points
Little relevance to the topic:	8–11 points
Irrelevant to the topic:	0–7 points

5.2 Lexical Coherence

Measures how well the words, phrases, and sentences in a piece of writing connect logically and smoothly, ensuring that the ideas flow cohesively from one point to the next. For example, the Lexical Coherence Score of text is 2.67 represents the average number of words that overlap between consecutive sentences in the given text.

5.3 Grammatical Error Rate

Grammatical Error Rate (GER) is a metric used to evaluate the grammatical accuracy of generated text, typically in the context of natural language processing tasks such as machine translation, text generation, or automated essay scoring. It measures the percentage of grammatical errors present in a piece of text, often normalized by the total number of words or sentences.

Table 5. Sentence Structure and Word Choice Evaluation Matrix

Evaluation Criteria	Points Range
Excellent sentence structure and word choice:	18–20 points
Good with minor issues:	15–17 points
Fair but with noticeable problems:	12–14 points
Poor with significant problems:	8–11 points
Very poor sentence structure and word choice:	0–7 points

Key Points to Note About GER Include:

- **Definition:** GER quantifies the number of grammatical errors per unit of text, often normalized to facilitate comparison across different texts or systems.
- **Types of Errors:** It encompasses errors such as subject-verb agreement issues, tense inconsistencies, incorrect word forms, and syntactic errors.
- **Evaluation:** GER can be calculated automatically using computational tools that analyze syntax and grammar, or manually by human annotators.
- **Application:** GER is used in research and development to assess the performance of language models, translation systems, and other NLP applications. Lower GER indicates higher grammatical accuracy and thus better performance.
- **Limitations:** GER may not capture all aspects of grammatical correctness, as it focuses primarily on surface-level errors and may not account for semantic or contextual inaccuracies.

5.4 Flesch Reading Ease

The Flesch Reading Ease score and Flesch-Kincaid Grade Level are metrics used to assess the readability of texts. The Flesch Reading Ease score rates text on a 100-point scale; higher scores indicate easier readability (e.g., a score of 90–100 is easily understood by an average 11-year-old), while lower scores indicate more difficult texts (e.g., 0–30 is best understood by university graduates). The Flesch-Kincaid Grade Level translates the score into a U.S. school grade level.

5.5 Human Evaluation

We conducted a survey wherein participants were requested to evaluate our responses based on several criteria, including grammar, spelling, coherence, context, and syntax. Each of these parameters was to be rated on a scale from 1 to 10. The average result of all responses are presented below (Tables 6 and 7)

Table 6. Human Evaluation Matrix

Evaluation Criteria	Lstm Rating	GPT - 2 Rating
Grammar	3.67	7.34
Spelling	5.35	6.32
Coherence	3,13	6.98
Context	4.98	5.76
Syntax	4.56	8.43

Table 7. Results of Lstm and GPT-2

Evaluation Metrics	Lstm	GPT-2
Lexical Coherence	0.0000	3.4590
Grammatical Error Rate	43	32
Flesch Reading Ease	−949.28	92.22
AI Evaluation (GPT-4)	**Grammar:** 2/20 **Spelling:** 4/20 **Coherence:** 1/20 **Context:** 1/20 **Syntax:** 1/20 **Total Score:** 7/100 **Review:** Needs significant improvement in all aspects. The text is mostly nonsensical and requires a complete overhaul for coherence and clarity.	**Grammar:** 12/20 **Spelling:** 14/20 **Coherence:** 10/20 **Context:** 10/20 **Syntax:** 10/20 **Total Score:** 56/100 **Review:** Shows potential with an interesting crossover story but needs substantial revisions in grammar, coherence, context, and syntax to enhance clarity and maintain a coherent narrative throughout.

6 Discussion and Conclusion

6.1 Discussion

Pre-trained transformer models have demonstrated remarkable proficiency in generating high-quality texts and items owing to their extensive training on vast corpora [23]. In this research, fine-tuning pre-trained transformer models for novel story generation has led to a nuanced understanding for crafting human-like narratives and items. However, it's important to acknowledge persistent challenges within the NLP community, such as repetitive words and awkward topic transitions [24]. The hybrid sampling approach employed in story generation has shown promising outcomes, yielding fluent, coherent, grammatically correct, logical, and human-sounding stories. These language models offer educators the ability to spontaneously generate authentic content, alleviating the need for extensive material search. Nevertheless, the generated items may require deeper human evaluation than the one presented in this paper and adjustments due to potential semantic errors or lack of coherence. Furthermore, they may not be optimal for evaluating complex reading skills such as inference, analysis, and critique.

AI-generated content, particularly in storytelling, raises important ethical considerations regarding authorship, copyright, and cultural sensitivity. One way to tackle the authorship issue is to view both humans and AI as co-creators. This method highlights the human's part in directing and polishing the AI's work, while also crediting the AI's role in the creative process [25] To address copyright concerns, we could update the laws to cover AI-generated content. This might mean giving copyright to the people or organizations that own or run the AI, similar to how companies hold copyrights for work done by their employees. Alternatively, we could establish a new category of copyright specifically for AI-generated works, making sure the rights and responsibilities are clearly outlined [26] For cultural sensitivity, it's crucial to use diverse and representative datasets when training AI models. Additionally, setting strong ethical guidelines and review processes can help make sure AI-generated stories are respectful and culturally considerate. Including human reviewers from varied backgrounds in the review process can also provide valuable perspectives and help reduce biases [27]

6.2 Conclusion

The study has limitations, including the use of LSTM for story and item generation due to the constraints of computing resources. Exploring more advanced versions of GPT-2 could potentially yield higher-quality outcomes. Moreover, the study's training dataset, drawn from freely available reading materials like Harry Potter and Lord of The Rings on the Internet, could be expanded with a larger and more diverse dataset to enhance the fine-tuning of transformer models and ensure more consistent results in story and item generation. Finally, an additional future direction involves implementing a more robust, stable diffusion model to generate high-quality images.

The AI based story generation can find applications in education, entertainment, or even assisting human writers. By including multiple transformer models (LSTM, GPT-2), the research compares their effectiveness in story generation. This helps identify GPT-2 as a strong candidate for this specific task. The paper highlights the possibility of using these AI-generated stories for educational purposes. This could provide educators with a new and engaging way to deliver content. The research implements the story generation approach using different models and presents results (stories and images) as concrete evidence of its functionality.

In our future work, we aim to explore cutting-edge transformer models, such as GPT-3 or its recent iterations, to bolster language modeling capabilities. By leveraging these advanced models, we seek to address challenges such as repetitive language and abrupt topic transitions, thus enhancing the overall quality of generated content. Additionally, we intend to diversify our training data by incorporating a broader range of reading materials. This expanded dataset will encompass various genres, subjects, and writing styles, enabling our models to generalize more effectively and produce more versatile outputs. Furthermore, we plan to investigate and implement stable diffusion models for image generation. These probabilistic generative models capture complex patterns in data, thus facilitating the production of high-quality images. We aim to build a tool that takes text as input and generate story. From each line of text, an image to be generated. Just like a story book.

References

1. Aylett, R., Louchart, S., Weallans, A.: Research in interactive drama environments, role-play and story-telling. In: Si, M., Thue, D., André, E., Lester, J.C., Tanenbaum, T.J., Zammitto, V. (eds.) ICIDS 2011. LNCS, vol. 7069, pp. 1–12. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25289-1_1
2. Ryan, J.: Grimes' fairy tales: a 1960s story generator. In: Nunes, N., Oakley, I., Nisi, V. (eds.) ICIDS 2017. LNCS, vol. 10690, pp. 89–103. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-71027-3_8
3. Jain, P., et al.: Story generation from sequence of independent short descriptions. arXiv preprint arXiv:1707.05501 (2017)
4. Gervás, P.: Computational approaches to storytelling and creativity. AI Magazine **30**(3), 49 (2009). https://doi.org/10.1609/aimag.v30i3.2250
5. Young, R.M., et al.: Plans and planning in narrative generation: a review of plan-based approaches to the generation of story, discourse and interactivity in narratives. Sprache und Datenverarbeitung, Spec. Issue Formal Comput. Model. Narrat. **37**(1–2), 41–64 (2013)
6. Kybartas, B., Rafael B.: A survey on story generation techniques for authoring computational narratives. IEEE Trans. Comput. Intell. AI Games **9**(3), 239–253 (2016). https://doi.org/10.1109/TCIAIG.2016.2546063
7. Shaikh, S., et al.: Towards improved classification accuracy on highly imbalanced text dataset using deep neural language models. Appl. Sci. **11**(2), 869 (2021). https://doi.org/10.3390/app11020869
8. Fatima, N., et al.: A systematic literature review on text generation using deep neural network models. IEEE Access **10**, 53490–53503 (2022). https://doi.org/10.1109/ACCESS.2022.3174108
9. Begeny, J.C., Greene, D.J.: Can readability formulas be used to successfully gauge difficulty of reading materials? Psychol. Schools **51**(2), 198–215 (2014). https://doi.org/10.1002/pits.21740
10. Agosto, D.E.: Why storytelling matters: unveiling the literacy benefits of storytelling. Child. Libraries **14**(2), 21–26 (2016). https://doi.org/10.5860/cal.14n2.21
11. Bigozzi, L., et al.: Reading fluency as a predictor of school outcomes across grades 4–9. Front. Psychol. **8**, 204198 (2017). https://doi.org/10.3389/fpsyg.2017.00200
12. Bulut, H.C., Bulut, O., Arikan, S.: Evaluating group differences in online reading comprehension: the impact of item properties. Int. J. Test. **23**(1), 10–33 (2023). https://doi.org/10.1080/15305058.2022.2044821
13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997). https://doi.org/10.1162/neco.1997.9.8.1735
14. Raffel, C., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv preprint arXiv:1910.10683 (2023)
15. Yenduri, G., et al.: Generative pre-trained transformer: a comprehensive review on enabling technologies, potential applications, emerging challenges, and future directions. arXiv preprint arXiv:2305.10435 (2023)
16. Balaji, Y., et al.: eDiff-I: text-to-image diffusion models with an ensemble of expert denoisers. arXiv preprint arXiv:2211.01324 (2023)
17. Rowling, J.K.: The Harry Potter. Bloomsburry, United Kingdom (26 June 1997–21 July 2007)
18. Tolkien, J.R.R.: The Hobbit. George Allen & Unwin, United Kingdom (1937)
19. Tolkien, J.R.R.: The Lord of the Rings. HarperCollins, United Kingdom (29 July 1954–20 October 1955)

20. Tolkien, J.R.R.: The Silmarillion . George Allen & Unwin, United Kingdom (5 September 1977)
21. Kipling , R.: The Jungle Book. Macmillan, United Kingdom (1894)
22. Hugging Face. https://huggingface.co/stabilityai/stable-diffusion-xl-base-1.0. Accessed 20 Mar 2024
23. Radford, A., et al.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)
24. See, A., et al.: Do massively pretrained language models make better storytellers? arXiv preprint arXiv:1909.10705 (2019)
25. Elgammal, A., Liu, B., Elhoseiny, M., Mazzone, M.: CAN: creative adversarial networks, generating "art" by learning about styles and deviating from style norms. arXiv preprint arXiv:1706.07068 (2017)
26. Samuelson, P.: Allocating ownership rights in computer-generated works. U. Pitt. L. Rev. **79**, 173 (2017)
27. Buolamwini, J., Gebru, T.: Gender shades: intersectional accuracy disparities in commercial gender classification. Proc. Mach. Learn. Res. **81**, 1–15 (2018)

Deep Learning Models for Inference on Compressed Signals with Known or Unknown Measurement Matrix

Huiyuan Yu and Maggie Cheng(✉)

Illinois Institute of Technology, Chicago, IL 60616, USA
hyu47@hawk.iit.edu, maggie.cheng@iit.edu

Abstract. In this paper we propose two sets of compressed learning methods. First, for the setting where the matrix that is used to compress the signal is known, then for the setting where the matrix is unknown. With a known measurement matrix, compressed learning can benefit from the success of compressed sensing by utilizing the intermediate results of signal recovery. In particular, we use a compressed sensing algorithm Blocked Successive Regression (BSR) to generate enhanced features, then apply some widely used machine learning models (such as multilayer perceptron and convolutional neural network) on the enhanced features. It is observed that the machine learning results are significantly improved. With an unknown measurement matrix, deep models are developed to learn the matrix. We develop two compressed learning models based on the variational autoencoders (VAE) framework, reducing the problem of learning the elements of the matrix to learning the distributional parameters of the matrix. This method significantly reduces the number of parameters to learn, and also offers robustness to small perturbation in data.

Keywords: Compressed Learning · Compressed Sensing · Variational Autoencoder · Image Classification

1 Introduction

Signal acquisition and processing methods have undergone significant developments in recent years, transitioning from traditional analog techniques to advanced digital methods to meet the growing demand for robust and cost-effective sensing systems in various fields such as imaging, telecommunications, and medical diagnostics. Two recent advances in this evolution are compressed sensing (CS) and compressed learning.

While compressed sensing enables signal acquisition at sub-Nyquist rates while achieving near-perfect reconstruction, many inference and learning tasks do not require a fully reconstructed signal. Compressed learning (CL) is an emerging paradigm in which learning and inference are performed directly in the measurement domain, bypassing the need for signal reconstruction [3,6,29].

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 48–63, 2025.
https://doi.org/10.1007/978-3-031-78128-5_4

Compressed learning was introduced to advocate performing inference tasks directly on CS measurements [7,8,25]. However, CL faces challenges such as information loss during the sampling process and adapting to different compression ratios. Efforts to overcome these challenges include incorporating image priors [16], leveraging Transformer networks for correlation modeling [24], and employing block-based compressed sensing techniques for large-scale image processing [2]. Methods to exploit the multidimensional properties of CL are proposed, highlighting ongoing efforts to enhance its capabilities [27]. Innovative frameworks such as TransCL combine block-based compressed sensing with Transformer networks, aiming to bridge the gap between CS and image domain methods [20].

The advent of compressed sensing and compressed learning has opened new research topics in signal processing and machine learning, particularly in handling high-dimensional data. This paper focuses on pushing the boundaries of these fields through innovative algorithmic contributions. The first part of the research builds on the success of compressed sensing to enhance the performance of deep learning models by integrating intermediate results as input features. This novel approach is rooted in the hypothesis that intermediate results from CS contain valuable information that can significantly improve learning efficiency and model accuracy. The second part of the research addresses the challenges of compressed learning in absence of a measurement matrix, and propose computationally efficient methods for learning measurement matrices. By employing a variational autoencoder framework, we propose a method to train only the parameters of the generative model for the measurement matrix instead of learning the matrix itself, thereby improves the model robustness to small perturbations and improves model efficiency. This research not only contributes to the theoretical understanding of compressed learning but also demonstrates practical applications in learning and inference on compressed data.

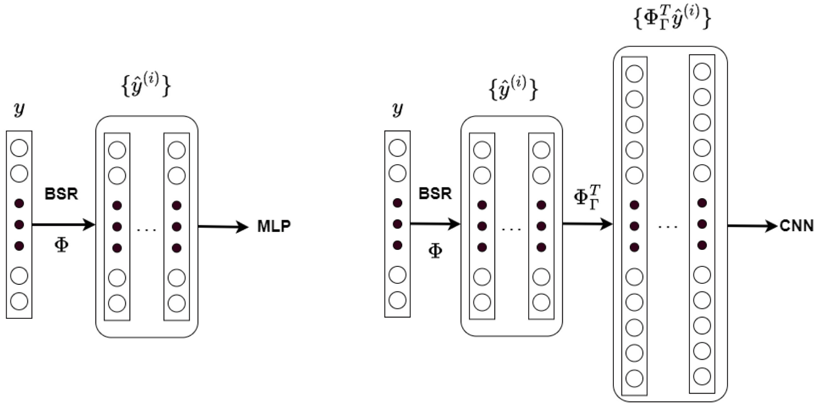
2 Inference on Compressed Signals with Known Measurement Matrix

Compressed sensing is a technique for recovering sparse signals with fewer measurements than traditionally required, while compressed learning is a technique for performing learning and inference on compressed signals. A typical approach in compressed learning is to first recover the signal by using a compressed sensing algorithm, then to apply a machine learning algorithm on the reconstructed signal. Sparse Bayesian Learning (SBL) [17,28] implements CS by applying a probabilistic model with sparse priors, enhancing signal recovery through iterative updates. Compressed sensing neural network (CSNN) [18,19,22,23] integrates deep learning to directly map measurement results to the original signal, optimizing the measurement matrix and neural network weights. Compression squeeze filtering algorithms and architectures (e.g., squeeze filters) are designed to infer directly from compression measurements, focusing on salient feature preservation. Known for its self-attention mechanism, transformer models are good at capturing long-range dependencies, making them suitable for a variety of applications, including image classification via Vision Transformers (ViT) [11,13,24]. Together, these methods improve the efficiency and applicability of computer science in various fields.

A major challenge in previous compressed learning work involves efficient and effective reconstruction of original signal. Sparse signal recovery is challenging at low measurement rates, since in the process critical information could be lost. On the other hand, the subsequent machine learning algorithms only require the key features to perform machine learning tasks, e.g., classification, and it does not require the signal to be reconstructed exactly. This means the full recovery of signals by a compressed sensing algorithm is neither sufficient to meet the requirement of machine learning nor necessary for the machine learning tasks. The observation motivates a different compressed learning framework, in which a compressed sensing algorithm is involved but not implemented to the end towards full signal recovery.

We propose to use the intermediate results generated from compressed sensing algorithms instead of the final product of compressed sensing as input to the machine learning algorithms, which can potentially address the aforementioned limitations by providing a richer, more informative set of features derived from compressed data.

In this paper, we leverage the intermediate results from the iterative sparse signal recovery algorithm, Blocked Successive Regression (BSR) from [26]), to generate the enhanced features for machine learning.



(a) MLP on $\{\hat{y}^{(i)}\}$. (b) CNN on $\{\Phi_\Gamma^T\hat{y}^{(i)}\}$.

Fig. 1. Features computed from the sparse signal recovery algorithm BSR are used as input to (a) MLP, and (b) CNN.

Consider the task of image classification. Three machine learning models are used to make inference on the enhanced features: Multi-Layer Perceptron

(MLP), Convolutional Neural Network (CNN), and Vision Transformer (ViT). The three methods are not specifically designed to classify compressed signals. The performance of classifying compressed signals is expected to drop sharply as the compression ratio increases. We will show that by using the enhanced features generated from CS, the performance degradation is slowed down.

The architectures of these models are in Fig. 1. The method *MLP on* $\{\hat{y}^{(i)}\}$ will use $\{\hat{y}^{(i)}\}$ as input to an MLP model and output the class label; the method *CNN on* $\{\Phi_\Gamma^T\hat{y}^{(i)}\}$ will use $\{\Phi_\Gamma^T\hat{y}^{(i)}\}$ as input to a CNN model and output the class lable, where Γ is the signal support at the i-th iteration; similarly, the method *ViT on* $\{\Phi_\Gamma^T\hat{y}^{(i)}\}$ will use $\{\Phi_\Gamma^T\hat{y}^{(i)}\}$ as input to a ViT model and output the class label.

The measurement matrix $\Phi \in \mathbb{R}^{M\times N}$ is a random Gaussian matrix, with each element Φ_{ij} generated by $\mathcal{N}(0, 1/N)$ [4]. Let $y \in \mathbb{R}^M$ represent the compressed signal, and $x \in \mathbb{R}^N$ is the original signal. The measurement rate (MR) is then defined as MR $= M/N$. Let $\hat{y}^{(i)} \in \mathbb{R}^M$ represent the intermediate result generated by the i-th iteration of the BSR algorithm. Figure 2 shows a sequence of $\Phi_\Gamma^T\hat{y}^{(i)}$ for the digits 3 and 5.

As iteration continues, the images rendered from $\Phi_\Gamma^T\hat{y}^{(i)}$ became clearer and easier to identify. It shows that with every iteration, non-essential features that do not contribute to image recognition are filtered out, and the key features that define the numbers are preserved. Image classification based on $\{\Phi_\Gamma^T\hat{y}^{(i)}\}$ achieves better result than based on $\Phi^T y$.

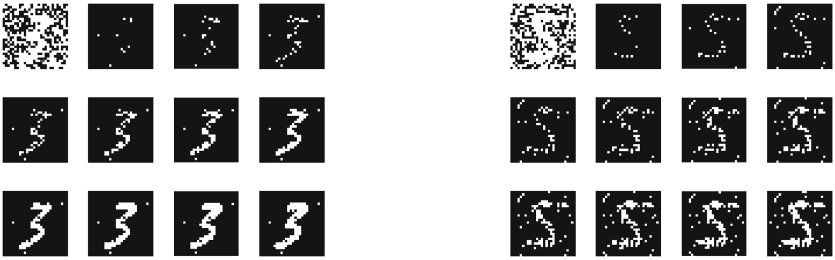


Fig. 2. Transformed images for digits 3 (left) and 5 (right). The first image in the upper left corner of each panel is $\Phi^T y$, other images are $\Phi_\Gamma^T\hat{y}^{(i)}$, with the iteration number i increasing from left to right and from top to bottom. MR $= 0.5$ is used.

2.1 Experiments

In this section, we test our models on real world datasets: MNIST, Imagenet2012, CIFAR-10 and CIFAR-100. Baseline compressive learning methods used for comparison are SF ([22]), DI ([18]), and TransCL ([21]).

ViT ([13]) is a transformer-based model, but it is not specifically designed to work with compressed signals. Similarly, MLP is also not a compressie learning method. We will show that by using our proposed methods, how much improvement they can achieve—comparing MLP on y with MLP on $\{\hat{y}^{(i)}\}$, and comparing ViT on $\Phi^T y$ with ViT on $\{\Phi_\Gamma^T \hat{y}^{(i)}\}$. We compare their classification accuracy and training time.

Table 1 shows the results of baseline methods. Table 2 shows the results of our models using intermediate results from BSR as input. In this experiment, the number of channels is 4, meaning, the intermediate results from four iterations of BSR are used when using $\{\Phi_\Gamma^T \hat{y}^{(i)}\}$ as input.

Accuracy: Overall, our models in Table 2 have better results than the baseline models in Table 1, especially at smaller MR values. Comparing MLP on y with MLP on $\{\hat{y}^{(i)}\}$, and ViT on $\Phi^T y$ with ViT on $\{\Phi_\Gamma^T \hat{y}^{(i)}\}$, we can see point-wise improvement in accuracy across all MR values.

At high MR(> 0.25), all methods perform well. As MR decreases, there's a noticeable decrease in model accuracy across all tests, which is expected, since there is more information loss in the compressed signal. CNN and ViT on $\{\Phi_\Gamma^T \hat{y}^{(i)}\}$, especially ViT, show a graceful performance degradation as MR decreases. CNN and ViT can still perform well when the signal is compressed at large ratios. ViT model still achieves more than 90% accuracy even when MR $=$ 0.01, which is remarkable. The robustness to large-ratio data compression highlights the advantage of convolutional and attention-based models in extracting useful features from sparse data.

Running Time: Since MLP and SF have fewer parameters than CNN and ViT, MLP and SF have shorter running time compared to CNN and ViT.

ViT on $\{\Phi_\Gamma^T \hat{y}^{(i)}\}$ in Table 2 has a little longer training time than ViT in Table 1, but it achieved improvements in accuracy at all MR levels, especially at low MR levels. Compared with MLP and CNN, the running time of ViT gradually increases with the increase in the number of input channels, which is attributed to the architectural efficiency of ViT in processing high-dimensional data. ViT utilizes a self-attention mechanism that scales more advantageously with input size, allowing them to handle additional channels without proportionally increasing computational complexity. In contrast, MLP and CNN rely on dense or convolutional layers, respectively, whose computational requirements grow significantly with the number of input channels, resulting in increased running time.

Table 3 shows the classification results on a subset of Imagenet2012 dataset (including 10274 images from 100 classes). Although the ViT model is large with training time more than 30000 s per epoch, the result from ViT is no better than the smaller models such as MLP or CNN. With MR $= 0.01$, the method *ViT on* $\{\Phi_\Gamma^T \hat{y}^{(i)}\}$ only attained 10.65% accuracy, while the method *CNN on* $\{\Phi_\Gamma^T \hat{y}^{(i)}\}$ attained 18.94%, which is the best result among all.

TransCL is a transformer-based compressive learning method, representing the recent top-performing compressive learning method. TransCL uses ViT-B as its backbone and significantly improves the performance of ViT-B. The reported results from [21] show that TransCL-32 is better than TransCL-16. So we compare our models with TransCL-32 for CIFAR-10 and CIFAR-100 datasets. The

Table 1. Classification results on MNIST dataset using baseline methods: Direct Inference (DI) [18], Smashed Filter (SF) [22], Vision Transformer (ViT) [13], as well as MLP on y.

	SF		MLP on y		DI		ViT on $\Phi^T y$	
	Accuracy	Time (s)	Accuracy	Time (s)	Accuracy	Time (s)	Accuracy	Time (s)
MR = 1	86.14%	88	97.29%	79.36	97.18%	95	94.94%	1554
MR = 0.5	84.61%	81	96.52%	72.52	96.82%	91	94.03%	1559
MR = 0.25	75.58%	74	96.44%	67.10	96.05%	94	93.74%	1569
MR = 0.1	56.45%	69	96.27%	65.56	93.77%	94	93.22%	1577
MR = 0.05	46.79%	64	91.62%	62.05	88.61%	92	91.05%	1518
MR = 0.02	40.07%	59	70.89%	60.11	61.44%	94	72.08%	1548
MR = 0.01	36.97%	57	53.23%	59.94	47.57%	92	67.38%	1503

Table 2. Classification results on MNIST dataset using our models. The choice of CNN is LeNet-5 for MNIST.

	MLP on $\{\hat{y}^{(i)}\}$		CNN on $\{\Phi_\Gamma^T \hat{y}^{(i)}\}$		ViT on $\{\Phi_\Gamma^T \hat{y}^{(i)}\}$	
	Accuracy	Time (s)	Accuracy	Time (s)	Accuracy	Time (s)
MR = 1	97.79%	179	98.31%	155	98.81%	1993
MR = 0.5	97.58%	134	97.60%	153	96.47%	1989
MR = 0.25	97.21%	122	97.63%	153	96.22%	1994
MR = 0.1	96.43%	111	96.91%	152	95.79%	1938
MR = 0.05	94.17%	90	93.82%	155	95.59%	1989
MR = 0.02	80.41%	75	92.21%	151	94.03%	1966
MR = 0.01	60.43%	77	82.83%	152	90.83%	1951

Table 3. Classification results on a subset of Imagenet2012 dataset obtained under different Measurement Rates (MR). Our methods are shown at the bottom of the table. The choice of CNN is AlexNet for Imagenet dataset.

	Accuracy			
	MR = 1	MR = 0.25	MR = 0.1	MR = 0.01
SF	3.44%	3.16%	2.83%	1.67%
MLP on y	93.85%	89.22%	44.43%	11.63%
DI	83.31%	59.48%	34.90%	12.05%
MLP on $\{r^{(i)}\}$	92.52%	87.86%	44.52%	12.14%
MLP on $\{\hat{y}^{(i)}\}$	94.01%	90.76%	45.13%	13.02%
CNN on $\{\Phi_\Gamma^T \hat{y}^{(i)}\}$	94.84%	92.23%	47.77%	18.94%
ViT on $\{\Phi_\Gamma^T \hat{y}^{(i)}\}$	96.93%	90.04%	47.28%	10.65%

classification results are presented in Table 4. TransCL-32 results are obtained by using the github code provided at https://github.com/MC-E/TransCL. The results show that CNN on $\{\Phi_\Gamma^T \hat{y}^{(i)}\}$ achieves better accuracy and uses much less training time than TransCL. Thus, it proves that by integrating compressive sensing with a lighter model such as CNN, which might not be specifically designed

Table 4. Image classification performance on CIFAR datasets. The baseline method is TransCL-32 from [21]. The top panel is for CIFAR-10 and the bottom panel is for CIFAR-100.

	TransCL-32		MLP on $\{\hat{y}^{(i)}\}$		CNN on $\{\Phi_\Gamma^T \hat{y}^{(i)}\}$	
	Accuracy	Time	Accuracy	Time	Accuracy	Time
MR = 1	79.96%	24h	70.61%	35m26s	91.36%	56m
MR = 0.25	76.93%	24h	61.95%	22m54s	88.39%	55m
MR = 0.10	72.73%	24h	57.36%	15m29s	84.56%	55m
MR = 0.01	44.90%	24h	16.62%	10m48s	63.09%	52m
MR = 1	52.56%	25h	33.39%	1h27m	94.31%	1h16m
MR = 0.25	46.77%	25h	27.94%	53m18s	91.67%	1h13m
MR = 0.10	38.83%	25h	23.24%	29m36s	88.05%	1h2m
MR = 0.01	33.82%	25h	11.20%	18m51s	77.24%	1h11m

for compressed signals, it can achieve comparable or even better results than the compressive learning methods based on heavy models such as Transformer.

3 Inference on Compressed Signals with Unknown Measurement Matrix

The compressed learning methods in Sect. 2 rely on a known measurement matrix, Φ. If Φ is unknown, and all we have is the compressed signals and their class labels, can we build a learning model to classify the compressed signals? This section will focus on the new task of learning the measurement matrix and using it for classifying compressed signals.

3.1 Preliminaries

Inference Based on the AE Model. When the true measurement matrix Φ is unknown, neural network models are used to learn it. In [1], Adler et al. introduced a compressed learning algorithm based on the Autoencoder model. It first uses a fully connected layer (FC) to embed the original signal X to the latent vector Z, and then use another fully connected layer to reconstruct the signal $\hat{X}$. The reconstructed signal is fed into CNN for classification:

$$
\begin{aligned}
Z &= \tilde{\Phi} X, \\
\hat{X} &= \Psi Z, \\
\text{label} &\leftarrow \text{CNN}(\hat{X}),
\end{aligned}
$$

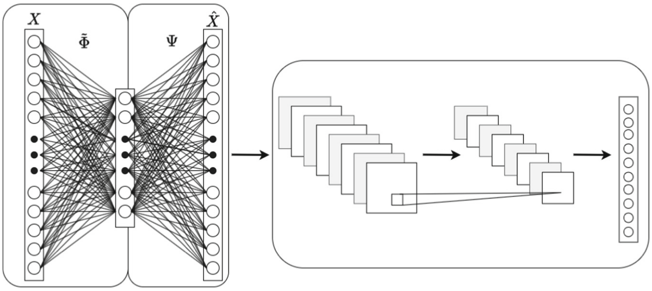


Fig. 3. The architecture of AE model [1].

where $\tilde{\Phi}$ and Ψ are parameterized by two fully connected layers, as shown in Fig. 3. Once the matrix Ψ is learned, it can be used to classify a compressed signal y:

$$\hat{X} = \Psi y,$$
$$\text{label} \leftarrow \text{CNN}(\hat{X}).$$

Variational Autoencoders. The variational autoencoder (VAE) framework, proposed by Kingma and Welling in [15], is designed to perform approximate inference with continuous latent variables. It combines reconstruction loss and regularization through an objective function to balance data fidelity and model complexity ([9,14]). They are based on Bayesian inference [5], using variational inference to approximate latent variable posteriors. Due to its interpretability and generative capabilities, VAE have applications in various fields, including image synthesis and drug discovery [12].

In the VAE framework, it is assumed that the data X are generated from a random process from a generative model. The generative model is described by $p_\theta(X, Z) = p_\theta(X|Z)p_\theta(Z)$, where X is the i.i.d. data points, and Z is the continuous latent variable, and both true parameters θ and the latent variables Z are hidden from us.

In case the true posterior $p_\theta(Z|X)$ is intractable, a variational approximation to it is used, denoted as $q_\Phi(Z|X)$. The variational parameters Φ are learned jointly with the generative model parameters θ using the Auto-Encoding variational Bayesian (AEVB) approach.

Using AEVB to jointly learn the true parameter θ and variation parameters Φ involves maximizing the variational lower bound, which contains a term that is called Kullback-Leibler (KL) divergence between the variational approximate posterior distribution $q_\Phi(Z|X)$ and the prior distribution $p_\theta(Z)$. Often times, the KL term can be integrated analytically.

When the variational approximate posterior is assumed to be a multivariate Gaussian with a diagonal covariance structure, i.e.,

$$\log q_\Phi(Z|X) = \log \mathcal{N}(Z; \mu, \sigma^2 I), \tag{1}$$

where μ and σ^2 represent the mean and variance of the approximate posterior, respectively, and the prior is a standard Gaussian, the Kullback-Leibler (KL) divergence is given as the following without estimation,

$$-D_{KL}(q_{\Phi}(Z)\|p_{\theta}(Z)) = \frac{1}{2}\sum_{j=1}^{J}\left(1+\log((\sigma_j)^2)-(\mu_j)^2-(\sigma_j)^2\right), \qquad (2)$$

where J denotes the dimensionality of the latent variable Z.

3.2 Proposed Models with Variational Measurement Matrix (VMM)

The AE model in [1] is simple and works well if we only consider the classification accuracy. However, in compressive sensing, the matrix Φ is large, so are the learned matrices $\tilde{\Phi}$ and Ψ. The AE model would learn each and every element in the matrices, leading to increased computational complexity and potentially overfitting. This is especially true in scenarios with limited training data or high-dimensional signals.

Additionally, the elements in the matrices are fixed once the training is completed, therefore the AE model is not robust to small perturbation on the compressed signal. The VAE framework, on the other hand, uses a generative model to generate the matrix instead of using a fixed matrix. It is naturally fit to overcome the lack of robustness issue. We hereby propose a compressed learning method based on the VAE framework.

The generative model for the measurement matrix is governed by a probability density function, as the matrix is supposed to be a random Gaussian matrix. In contrast to previous methods, our method involves learning the parameters of the probability density function instead of learning all the elements in the matrix. This approach reduces the number of learnable parameters and provides a direct way to specify the properties of the matrix, thus striking a balance between flexibility and efficiency as it allows the model to adapt the measurement matrix to the input data while maintaining a compact parameter space. Furthermore, by incorporating statistical properties such as mean and variance into the parameterization, we can capture important features of the signal and improve overall compression performance.

Let Φ represent the unknown measurement matrix. We denote $\Phi_{i;.}$ as the i-th row of Φ, which has mean mu_i, and variance σ_i^2. Each element of the matrix is then generated by using the following generative model: $\Phi_{ij} \sim \mathcal{N}(\mu_i, \sigma_i^2)$, $\forall j$. Let $\eta_i = (\mu_i, \sigma_i^2)$ represent the parameters in the generative model of Φ.

The objective of this model is twofold: first, to reduce the number of learnable parameters and thus the computational complexity; second, to provide a direct way to control the properties of the measurement matrix, making the model easier to adapt to practical applications.

Suppose there are L data points in the training set. Let X represent the l-th data point $X^{(l)}$, then the calculation of each hidden layer element Z_i involves multiplying the measurement matrix Φ with the input data X,

$$Z_i = \sum_{j=1}^{N} \Phi_{ij} X_j, \tag{3}$$

$$E(\sum_{j=1}^{N} \Phi_{ij} X_j) = \sum_{j=1}^{N} E(\Phi_{ij} X_j) = \mu_i \sum_{j=1}^{N} X_j, \tag{4}$$

$$Var(\sum_{j=1}^{N} \Phi_{ij} X_j) = \sum_{j=1}^{N} Var(\Phi_{ij} X_j) = \sum_{j=1}^{N} \sigma_i^2 X_j^2. \tag{5}$$

The loss function is computed over the entire batch, averaging over all data points to obtain a mean loss. The mean μ_{Z_i} and variance $\sigma_{Z_i}^2$ of the i-th element of Z can be calculated as follows,

$$\begin{aligned} E(Z_i) &\approx \frac{1}{L} \sum_{l=1}^{L} \left(\mu_i \sum_{j=1}^{N} X_j^{(l)} \right) \\ &= \mu_i \frac{1}{L} \sum_{l=1}^{L} \left(\sum_{j=1}^{N} X_j^{(l)} \right), \end{aligned} \tag{6}$$

$$\begin{aligned} Var(Z_i) &\approx \frac{1}{L} \sum_{l=1}^{L} \left(\sum_{j=1}^{N} \sigma_i^2 \left(X_j^{(l)} \right)^2 \right) \\ &= \sigma_i^2 \frac{1}{L} \sum_{l=1}^{L} \sum_{j=1}^{N} \left(X_j^{(l)} \right)^2. \end{aligned} \tag{7}$$

With an abuse of notation, we use $E_X(h(X))$ to represent the average value of $h(X)$ computed over all L data points. Thus,

$$\mu_{Z_i} = E(Z_i) \approx \mu_i \sum_{j=1}^{N} E_X(X_j), \tag{8}$$

$$\sigma_{Z_i}^2 = Var(Z_i) \approx \sigma_i^2 \sum_{j=1}^{N} E_X(X_j^2). \tag{9}$$

The objective is to infer $\eta_i = (\mu_i, \sigma_i^2)$ in order to generate the matrix Φ, as $\Phi_{ij} \sim \mathcal{N}(\mu_i, \sigma_i^2)$. This can be done by using an MLP to compute μ_{Z_i} and $\sigma_{Z_i}^2$, and then using Eqs. (10) and (11) to get (μ_i, σ_i^2).

$$\mu_i = \frac{\mu_{Z_i}}{\sum_{j=1}^{N} E_X(X_j)}, \tag{10}$$

$$\sigma_i^2 = \frac{\sigma_{Z_i}^2}{\sum_{j=1}^{N} E_X(X_j^2)}. \tag{11}$$

Therefore, we can generate the measurement matrix by Eq. (12),

$$\Phi_{ij} = \mu_i + \sigma_i \epsilon, \ \forall j, \quad \text{where } \epsilon \sim N(0,1). \tag{12}$$

Loss Function. The loss function consists of two parts: the KL divergence between the variational approximate posterior and the prior, and the cross-entropy loss for classification of images [10]. The prior distribution is standard Gaussian. Reparameterization is performed following the standard "reparameterization trick" in [15].

For the KL divergence part, we can either use the KL divergence on the parameters of Φ, or on the parameters of Z:

$$\begin{aligned} -D_{KL}\left(q_\theta(Z) \| \mathcal{N}(\mathbf{0},\mathbf{I})\right) &= \int q_\theta(Z)\left(\log \mathcal{N}(Z;\mathbf{0},\mathbf{I}) - \log \mathcal{N}\left(Z;\mu_Z,\sigma_Z^2\mathbf{I}\right)\right) dZ \\ &= \frac{1}{2}\sum_{j=1}^{J}\left(1+\log\left(\sigma_{Z_j}^2\right) - \mu_{Z_j}^2 - \sigma_{Z_j}^2\right), \end{aligned} \tag{13}$$

$$\begin{aligned} -D_{KL}\left(q_\eta(\Phi) \| \mathcal{N}(\mathbf{0},\mathbf{I})\right) &= \int q_\eta(\Phi)\left(\log \mathcal{N}(\Phi;\mathbf{0},\mathbf{I}) - \log \mathcal{N}\left(\Phi;\mu,\sigma^2\mathbf{I}\right)\right) d\Phi \\ &= \frac{1}{2}\sum_{j=1}^{J}\left(1+\log\left(\sigma_j^2\right) - \mu_j^2 - \sigma_j^2\right), \end{aligned} \tag{14}$$

where $J = M$ is the dimension of the latent variable Z, $\theta_i = (\mu_{Z_i}, \sigma_{Z_i}^2)$ and $\eta_i = (\mu_i, \sigma_i^2)$ follow the relationship of Eq. (8) to Eq. (11).

The experiment results show that using the KL divergence on the variational parameters of Φ gives better classification accuracy. This is no surprise since μ_i and σ_i are directly used to generate the matrix Φ.

Two Variants of the VMM Model. The overarching approach is to learn the parameters that are used to generate the measurement matrix Φ instead of learning the elements of the matrix as the matrix can be very large.

let $Z = g(X)$. g is the embedding function that maps from input X to latent vector Z. From Z, we can compute the parameters $\theta_i = (\mu_{Z_i}, \sigma_{Z_i}^2)$. Then from Eqs. (10) and (11) we derived the paramters $\eta_i = (\mu_i, \sigma_i^2)$. Using η_i, we generate the measurement matrix $\Phi_{ij} \sim \mathcal{N}(\mu_i, \sigma_i^2)$.

Once the measurement matrix Φ is generated based on the learned parameters μ_i and σ_i^2, the input data X is compressed by multiplying it with Φ. Subsequently, the compressed data is reconstructed by applying the transpose of Φ. The resulting reconstructions of the images are then input to a CNN model to address the practical task at hand.

We propose two variants for the embedding function g. The first one is parameterized by a fully connected layer, and the second one is a linear model by using the estimated measurement matrix. We denote the end-to-end neural network models as VMM-a and VMM-b, respectively.

VMM-a with $Z = FC(X)$: In this model, we train a fully connected linear layer (FC) to map the input data X to the latent vector Z, with $Z = FC(X)$. As the

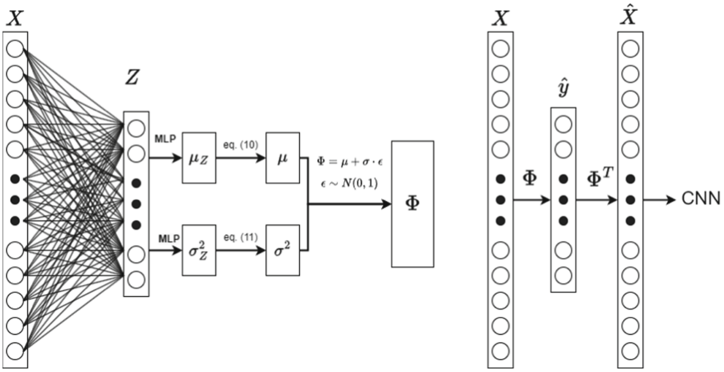


Fig. 4. The architecture of model VMM-a.

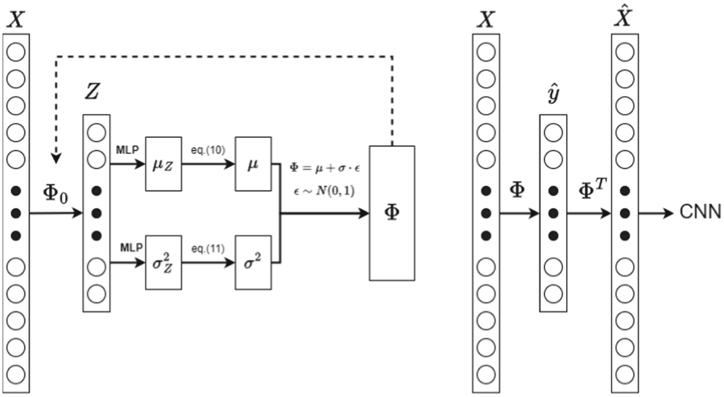


Fig. 5. The architecture of model VMM-b.

training progresses, the parameters of the fully connected layers are iteratively improved, the estimates of μ_i and σ_i^2 get improved as well. After training stops, (μ_i, σ_i^2) are used to generate the matrix Φ. For a new compressed signal y, $\Phi^T y$ is then used as input to the CNN model for classification. Figure 4 shows the architecture of VMM-a.

VMM-b with $Z = \Phi X$: We initialize the estimates of μ_i and σ_i as $\mu_i^{(0)}$ and $\sigma_i^{(0)}$, and generate the initial $\Phi^{(0)}$ using $\mu_i^{(0)}$ and $\sigma_i^{(0)}$. As μ_i and σ_i are updated during the training stage, the new measurement matrix Φ is generated based on the updated parameters $\eta_i = (\mu_i, \sigma_i^2)$. Then the embedded vector $Z = \Phi X$ is updated by using the new Φ. At the end of the training stage, we have the learned parameters $\eta_i = (\mu_i, \sigma_i^2)$, which will be used to generate the matrix Φ. For a new compressed signal y, $\Phi^T y$ is then used as input to the CNN model for classification. The architecture of VMM-b is shown in Fig. 5.

3.3 Experiments on Signals Compressed with the Same Matrix

We test our models VMM-a and VMM-b on MNIST dataset and compare them with the baseline method AE in [1].

Table 5 and Table 6 show the experiment results of VMM-a and VMM-b with two different loss functions. Compared to the AE model, both VMM-a and VMM-b show a significant reduction in the number of training parameters. In particular, VMM-b has more significant reduction, requiring only 2% to 15% of the parameters required by the AE model. As a result, both VMM-a and VMM-b have much shorter running time, indicating that they are more computationally efficient than AE models.

However, a trade-off between parameter reduction and accuracy must be acknowledged. The accuracy of VMM-a and VMM-b start to suffer when the measurement rate gets below 5% (i.e., when the signal is compressed with a larger ratio) although their accuracies are comparable to AE for larger MRs. This result is expected since AE model fully parameterized the matrix, as long as the matrix is the same matrix used for training data and testing data, AE has excellent performance. However, when the testing data are compressed with dif-

Table 5. Model VMM-a results on MNIST dataset, showing the classification accuracy, number of parameters, and training time (in seconds). Baseline method is AE from [1].

	VMM-a with D_{KL} Eq. (14)			VMM-a with D_{KL} Eq. (13)			AE		
	Accuracy	n_Param	Time	Accuracy	n_Param	Time	Accuracy	n_Param	Time
MR = 1	99.24%	1220880	257	98.91%	1220880	256	97.78%	1652650	328
MR = 0.5	98.61%	615832	202	97.94%	615832	197	97.12%	1250850	307
MR = 0.25	97.56%	308308	173	95.48%	308308	180	96.81%	1049950	294
MR = 0.1	91.17%	123116	139	91.71%	123116	154	96.40%	929000	286
MR = 0.05	88.36%	61975	112	89.22%	61975	129	95.09%	131595	233
MR = 0.02	85.88%	24319	83	11.35%	24319	79	94.88%	49638	226
MR = 0.01	83.86%	15700	76	11.35%	15700	74	93.12%	24319	222

Table 6. Model VMM-b results on MNIST dataset

	VMM-b with D_{KL} Eq. (14)			VMM-b with D_{KL} Eq. (13)			AE		
	Accuracy	n_Param	Time	Accuracy	n_Param	Time	Accuracy	n_Param	Time
MR = 1	98.90%	252608	184	99.21%	252608	197	97.78%	1652650	328
MR = 0.5	97.81%	126384	132	97.77%	126384	128	97.12%	1250850	307
MR = 0.25	96.82%	63272	92	96.74%	63272	101	96.81%	1049950	294
MR = 0.1	95.16%	12324	79	95.34%	12324	72	96.40%	929000	286
MR = 0.05	89.77%	3120	52	89.76%	3120	45	95.09%	131595	233
MR = 0.02	60.57%	1490	32	58.82%	1490	31	94.88%	49638	226
MR = 0.01	45.61%	414	26	44.98%	414	28	93.12%	24319	222

Table 7. Classification results on a subset of the MNIST dataset when images are compressed with different measurement matrices. Our models are at the bottom. For our models and AE, it is assumed the matrix $\varPhi$ is unknown.

Models	MR = 1	MR = 0.5	MR = 0.25	MR = 0.1	MR = 0.05	MR = 0.02	MR = 0.01
$\varPhi$ is known	98.56%	97.92%	94.93%	92.62%	79.36%	70.85%	67.80%
AE	94.55%	94.97%	91.78%	86.92%	73.55%	52.00%	34.81%
VMM-a	97.83%	97.52%	95.67%	89.90%	80.30%	64.73%	54.05%
VMM-b	97.76%	96.54%	93.61%	90.16%	82.09%	53.77%	41.82%

ferent matrix from the training data, the advantage of our model, i.e., robustness to small perturbation, starts to show.

3.4 Experiments on Signals Compressed with Different Matrices

We generate the compressed signals with different matrices, and then randomly split the data into a training set and a testing set. A total of 16 different measurement matrices are used on the MNIST dataset.

The compressed signal is denoted by y. In the baseline method, the true measurement matrix $\varPhi$ is given to the compressed learning algorithm, then the reconstructed signal is obtained by $\hat{X} = \varPhi^T y$; In AE, matrix $\varPsi$ is learned by a fully connected layer, then $\hat{X} = \varPsi^T y$ is obtained; in our models, $\varPhi$ is generated by the learned parameters μ and σ^2, then the reconstructed signal $\hat{X} = \varPhi^T y$ is obtained. For all models, $\hat{X}$ is fed to a CNN (LeNet-5) for classification. Table 7 shows the accuracy result of classification. Our models achieved overall better accuracy results than AE, and close to the baseline, while our models and AE both deal with unknown measurement matrix. The robustness of the variational approach compared to the autoencoder approach is once again proved.

4 Conclusion and Future Work

The proposed methods for inference on compressed signals with known measurement matrices have noticeable improvement over previous work. The improvement is more significant when the measurement rate (MR) is small.

For unknown measurement matrix, previous work based on AE model has strong performance when the sparse signals are all compressed with the same measurement matrix. Our models have shorter training time than AE, but the classification accuracy starts to deteriorate when the measurement rate gets below 5%. However, when the signals are not compressed with the same matrix, our models VMM-a and VMM-b both demonstrate better applicability than AE.

The results also showed that model VMM-a with loss function including the KL divergence on matrix parameters worked better than other variants. Future work will focus on formal methods to verify this observation, and the optimal neural network model versus loss function combination.

References

1. Adler, A., Elad, M., Zibulevsky, M.: Compressed learning: a deep neural network approach. arXiv preprint arXiv:1610.09615 (2016)
2. An, Y., Zhang, Y., Guo, H., Wang, J.: Compressive sensing-based three-dimensional laser imaging with dual illumination. IEEE Access **7**, 25708–25717 (2019)
3. Banitalebi-Dehkordi, M., Banitalebi-Dehkordi, A., Abouei, J., Plataniotis, K.N.: Face recognition using a new compressive sensing-based feature extraction method. Multim. Tools Appl. **77**, 14007–14027 (2018)
4. Baraniuk, R., Davenport, M., DeVore, R., Wakin, M.: A simple proof of the restricted isometry property for random matrices. Constr. Approx. **28**, 253–263 (2008)
5. Blei, D.M., Kucukelbir, A., McAuliffe, J.D.: Variational inference: a review for statisticians. J. Am. Stat. Assoc. **112**(518), 859–877 (2017)
6. Caiafa, C.F., Cichocki, A.: Multidimensional compressed sensing and their applications. Wiley Interdiscip. Rev.: Data Mining Knowl. Discov. **3**(6), 355–380 (2013)
7. Calderbank, R., Jafarpour, S., Schapire, R.: Compressed learning: universal sparse dimensionality reduction and learning in the measurement domain. preprint (2009)
8. Davenport, M.A., Romberg, J.: An overview of low-rank matrix recovery from incomplete observations. IEEE J. Select. Topics Signal Process. **10**(4), 608–622 (2016)
9. Doersch, C.: Tutorial on variational autoencoders. arXiv preprint arXiv:1606.05908 (2016)
10. Domingos, P.: A few useful things to know about machine learning. Commun. ACM **55**(10), 78–87 (2012)
11. Dosovitskiy, A., et al.: An image is worth 16x16 words: transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
12. Gómez-Bombarelli, R., et al.: Automatic chemical design using a data-driven continuous representation of molecules. ACS Cent. Sci. **4**(2), 268–276 (2018)
13. Han, K., et al.: A survey on vision transformer. IEEE Trans. Pattern Anal. Mach. Intell. **45**(1), 87–110 (2022)
14. Heaton, J., Goodfellow, I., Bengio, Y., Courville, A.: Deep learning. Genet. Prog. Evol. Mach. (11), 305–307 (2017). https://doi.org/10.1007/s10710-017-9314-z
15. Kingma, D.P., Welling, M.: Auto-encoding variational Bayes. In: ICLR 2014. arXiv preprint arXiv:1312.6114 (2014)
16. Li, L., Wang, X., Wang, G., et al.: Alternating direction method of multipliers for separable convex optimization of real functions in complex variables. Math. Prob. Eng. **2015**, 1–14 (2015)
17. Lin, J., Nassar, M., Evans, B.L.: Impulsive noise mitigation in powerline communications using sparse bayesian learning. IEEE J. Sel. Areas Commun. **31**(7), 1172–1183 (2013)
18. Lohit, S., Kulkarni, K., Turaga, P.: Direct inference on compressive measurements using convolutional neural networks. In: 2016 IEEE International Conference on Image Processing (ICIP), pp. 1913–1917. IEEE (2016)
19. Mardani, M., et al.: Deep generative adversarial neural networks for compressive sensing MRI. IEEE Trans. Med. Imaging **38**(1), 167–179 (2018)
20. Mou, C., Zhang, J.: Transcl: transformer makes strong and flexible compressive learning. IEEE Trans. Pattern Anal. Mach. Intell. **45**(4), 5236–5251 (2022)

21. Mou, C., Zhang, J.: Transcl: transformer makes strong and flexible compressive learning. IEEE Trans. Pattern Anal. Mach. Intell. **45**(4), 5236–5251 (2023). https://doi.org/10.1109/TPAMI.2022.3194001
22. Rodriguez, A., Boddeti, V.N., Kumar, B.V., Mahalanobis, A.: Maximum margin correlation filter: a new approach for localization and classification. IEEE Trans. Image Process. **22**(2), 631–643 (2012)
23. Shi, W., Jiang, F., Liu, S., Zhao, D.: Image compressed sensing using convolutional neural network. IEEE Trans. Image Process. **29**, 375–388 (2019)
24. Vaswani, A., et al.: Attention is all you need. Adv. Neural Inf. Process. Syst. **30** (2017)
25. Wimalajeewa, T., Varshney, P.K.: Application of compressive sensing techniques in distributed sensor networks: a survey. arXiv preprint arXiv:1709.10401 (2017)
26. Yu, H., He, J., Cheng, M.: Fast orthogonal matching pursuit for exact recovery and sparse approximation. arXiv preprint arXiv:2404.00146 (2024)
27. Zeng, J., Lau, T.T.K., Lin, S., Yao, Y.: Global convergence of block coordinate descent in deep learning. In: International Conference on Machine Learning, pp. 7313–7323. PMLR (2019)
28. Zhang, Z., Rao, B.D.: Sparse signal recovery with temporally correlated source vectors using sparse bayesian learning. IEEE J. Select. Topics Signal Process. **5**(5), 912–926 (2011)
29. Zhao, W., Gao, L., Zhai, A., Wang, D.: Comparison of common algorithms for single-pixel imaging via compressed sensing. Sensors **23**(10), 4678 (2023)

Training Point-Based Deep Learning Networks for Forest Segmentation with Synthetic Data

Francisco Raverta Capua[1,2](✉), Juan Schandin[1], and Pablo De Cristóforis[1,2]

[1] Facultad de Ciencias Exactas y Naturales, Departamento de Computación,
Universidad de Buenos Aires, Buenos Aires, Argentina
{jschandin,pdecris}@dc.uba.ar
[2] Instituto de Ciencias de la Computación (ICC), CONICET-Universidad de Buenos Aires, Buenos Aires, Argentina
fraverta@icc.fcen.uba.ar

Abstract. Remote sensing through unmanned aerial systems (UAS) has been increasing in forestry in recent years, along with using machine learning for data processing. Deep learning architectures, extensively applied in natural language and image processing, have recently been extended to the point cloud domain. However, the availability of point cloud datasets for training and testing remains limited. Creating forested environment point cloud datasets is expensive, requires high-precision sensors, and is time-consuming as manual point classification is required. Moreover, forest areas could be inaccessible or dangerous for humans, further complicating data collection. Then, a question arises whether it is possible to use synthetic data to train deep learning networks without the need to rely on large volumes of real forest data. To answer this question, we developed a realistic simulator that procedurally generates synthetic forest scenes. Thanks to this, we have conducted a comparative study of different state-of-the-art point-based deep learning networks for forest segmentation. Using created datasets, we determined the feasibility of using synthetic data to train deep learning networks to classify point clouds from real forest datasets. Both the simulator and the datasets are released as part of this work.

Keywords: Deep Learning · Point Cloud Segmentation · Forest Simulator

1 Introduction

The use of remote sensing for environmental monitoring has grown significantly in recent years, thanks to the development of Terrestrial Laser Scanning (TLS), Aerial Laser Scanning (ALS), and Aerial Photogrammetry, techniques widely used in precision forestry [1]. Both LiDAR and camera sensors made it possible

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 64–80, 2025.
https://doi.org/10.1007/978-3-031-78128-5_5

to easily acquire three-dimensional data of the studied environment, accurately representing it with high precision level point clouds. Both have been widely used in forest environments for health monitoring, species classification, tree parameter estimation, and even illegal logging detection amidst other applications [2]. Laser scanning, while more expensive, heavier, and energy-consuming, is considered the most accurate method for estimating the forest structure, as both the canopy and the ground can be detected [3].

Deep learning architectures, popular in natural language and image processing nowadays, have recently been extended to point cloud processing, and multiple techniques have already been adapted to the goals of classification, segmentation, and point completion [4].

Unfortunately, few point cloud datasets are publicly available for training, validating, and testing deep learning architectures, among which we can mention ScanNet [5], ScanObjectNN [6] and ModelNet40 [7] for object classification and ShapeNetPart [8] and SemanticKITTI [9] for part segmentation and scene completion. Generally speaking, none of them are designed for specific environments, such as forests. Therefore, if training a deep learning architecture for forested environment is needed, a new dataset for this purpose has to be generated. For example, [10] developed a dataset from the regions of the Southern Sierra Nevada Mountains, USA, [11] from Australia and New Zealand, and [12] from Evo, Finland. All these works were conducted for forest segmentation. Of the three, only the latter dataset is publicly available, limiting the repeatability of the experiments and the comparison between the cited works.

Creating a specific point cloud dataset for forested environments is expensive, as high-end equipment, including UAVs and high-precision sensors, is required to survey the studied area. It is also time-consuming, as it implies labeling the points manually. Moreover, forest areas could be inaccessible or dangerous for humans, further complicating data collection.

This work aims to answer whether synthetic data for training point-based deep learning networks is suitable for segmenting real forest point clouds generated from LiDAR or camera sensors. For this purpose, we developed a forest simulator based on Unity [13] that allows us to generate several forested scenes with high realism procedurally. We extract the point clouds from the synthetic scenes, which are then used to train the deep learning architectures instead of using real forest data. The main contributions of this work are as follows:

1. The development of a novel open-source forest simulator based on Unity that procedurally generates forest scenes. It also includes a configurable survey mission planner mode that takes pictures of the scene like a camera from an up-down UAV view.
2. Public-domain synthetic datasets of forest scenes that can be used to train or test different deep-learning networks.
3. A comparative study of state-of-the-art point-based deep learning networks to determine whether training with synthetic data is suitable for segmenting point clouds from real forest datasets.

This paper is organized as follows: Sect. 2 overviews related works. Section 3 briefly explains the deep-learning architectures selected for this work, and presents the developed forest simulator and the dataset generation. Section 4 shows and discusses the experimental results, and Sect. 5 ends with conclusions and future work.

2 Related Work

The development of TLS, ALS and aerial photogrammetry, aided with computer vision techniques, has made acquiring an accurate 3D reconstruction of the studied environment possible. This allows to extract valuable information for the characterization, monitoring, and conservation of forests.

Thanks to the ability of the deep learning networks to adapt to several tasks, such as classification, segmentation, and completion, its use has expanded to several applications, including the study of different natural environments. Specifically in point cloud processing, [10,14–16] use deep learning for the extraction of digital terrain models (DTM) from LiDAR captured data. In a forest environment, [11] segments the LiDAR captured point cloud into different categories: terrain, vegetation, coarse woody debris and stems. It is noticeable that segmenting the terrain from the rest of the points results equivalent to finding the DTM of the studied environment. Similarly, [12] uses segmentation techniques to classify the points into terrain, understorey, tree trunk, or foliage categories; the labeled point cloud dataset used in this work is one of the few publicly available. To the best of our knowledge, no prior research has utilized purely synthetic data to train networks for classifying forest strata. This approach could be beneficial, considering the complexity of creating real forest datasets.

Numerous deep learning architectures designed for point cloud processing were developed in the last years, such as PointNet++ [17], built mostly with multi-layer perceptrons, and PointNeXt [18], that updates PointNet++ to the state-of-the-art using more modern training strategies, such as newer optimization techniques, data augmentation, and an updated architecture. Besides, transformer technology, based entirely on self-attention layers [19], presents advantages over the use of convolutional layers and multi-layer perceptrons in encoder-decoder architectures. This translates into more accurate results with lower training time but at the cost of using a greater number of parameters, which requires a greater volume of input data for training. Some of the architectures that were based on this technology are PointBERT [20], PointMAE [21] and PointGPT [22]. The first one divides the point cloud into several smaller local clouds, and codifies each of them into tokens using a Discrete Variational AutoEncoder (DVAE) [23]. A proportion of them are masked and input to the transformer blocks which is trained to recover the original tokens of the masked clouds. On the other hand, PointMAE masks the small cloud without the need for a DVAE, and PointGPT orders the smaller local clouds using a Morton curve by spatial proximity [24], and uses a dual mask strategy for masking. In this work PointBERT, PointMAE, PointGPT and PointNeXt were selected to study if training with synthetic forest point clouds will give accurate results in real forest data, as they have state-of-the-art performance in point-cloud segmentation tasks.

Regarding the use of synthetic datasets in forest environments, in [25], a simulator is presented using procedural techniques, where sensors like LiDAR, an RGB camera, and a depth camera are also simulated for data extraction, all of them with the capacity of segmenting the scene in the following categories: background, terrain, traversable, trunks, canopy, shrubs, herbaceous plants, and rocks. A synthetic dataset using this simulator is available in [26]. It includes RGB images, semantic segmentation maps, depth maps and the projection of LiDAR point clouds on the RGB field of view for two different LiDAR scanning patterns. However, this simulator is not publicly available, and the dataset does not include the 3D reconstruction neither the point clouds dataset of the environment. Finally, [27] uses this dataset's images for training networks to detect fuel for preventing spread of forest fires. This work concludes that the synthetic data fails to generalize to real data. In a similar way, for this work an open-source forest simulator with the ability to export point clouds of the generated scenes was developed. Two publicly available datasets were built with these point clouds to train the aforementioned networks.

3 Materials and Methods

3.1 Point Cloud Deep Learning Networks

Following [12], we aim to train different deep learning networks to segment the forest point clouds into trunks, canopy, understorey, and terrain. This lets us differentiate forest strata and the DTM that corresponds with the terrain points. We selected four well-known state-of-the-art architectures, PointNeXt, Point-BERT, PointMAP, and PointGPT, and trained them with synthetic data generated by our forest simulator. Of these four selected architectures, the last three are built using transformers, and pre-trained versions are available with the ShapeNetPart dataset, so these networks can be fine-tuned specializing them in the respective study area. On the other hand, PointNext is not built using transformers but multi-layer perceptrons, and as we do not count on pre-trained versions of the network, we trained it from scratch.

3.2 Forest Simulator

For this work, a forest simulator based on the Unity engine was developed, from which synthetic data with a similar appearance to real-world forests was extracted. We aim to train the mentioned architectures with synthetic data and test their performance with real forest data. As [12] notices, the manual labeling of the extracted point cloud is a very demanding task, and in several cases, it is impossible to human experts to discern which category each point belongs to. Using a simulator for dataset generation overcomes this problem, as the point cloud labeling can be carried out automatically. Moreover, as transformer technology requires a large volume of data for training, generating synthetic data procedurally becomes even more relevant. Below we detail the most important modules of the presented simulator.

Terrain Generation. The simulator first generates a terrain mesh using fractal noise to build a heightmap for all its vertices. These noise samples are from Perlin noise layers at different scales or octaves. This permits controlling the amount of detail and the general aspect of the terrain. As the implementation is easily parallelizable, we took advantage of the Perlin noise function provided by Unity and the Unity Jobs framework for parallel execution. The random appearance and realistic aspect of this method's results are notorious and well-regarded in the video game developers community.

Trees, Bushes, and Plants Generation. Vegetation (excluding grass) is generated via pipelines. Each pipeline is a Directed Acyclic Graph (DAG) that links prefabs (i.e. reusable pre-generated game objects, like individual trees or bushes) with their position over the terrain. This is done through textures that determine the spawn probability and density over the terrain mesh. Each pipeline is built using different nodes:

- Source: imports a texture from a file or another pipeline, or generates a new one via a Voronoi diagram or sampling noise.
- Logic: applies logic operations over textures
- Sampling: variants of Poisson disk sampling method.
- Placement: generates instancing parameters for the assigned prefabs.

Regarding the sampling process, an implementation of Bridson's Poisson disk sampling method [28] and a variation of it were implemented. In Bridson's original algorithm, given an object a_0 with radius r, new objects of the same radius are added in an annulus of size $[r, 2r]$ without overlapping, until it reaches a maximum quantity, or until there can not be placed any more, and then this process is repeated with the next object. Using a cell size of $r/\sqrt{n}$, where n is the dimension of the background grid for storing samples, each cell can contain only one placed object. This process is fast for object placement, but it produces a distribution of points that may appear equidistant, especially for small values of r giving an unrealistic point distribution. To face this issue we propose a variation to the method: new points are seeked in an annulus of size $[r_{min}, r_{max}]$, where r_{min} and r_{max} are a given minimum radius and a maximum radius respectively, interpolating the distance linearly using the value of a greyscale texture at each point. This means that for values near 0 (where the texture is black), points at a distance r_{min} from a_0 are generated, and the inverse holds for values near 1. To do that, we tweaked Bridson's algorithm to use a spatial cell of size $r_{min}/\sqrt{n}$, but instead of each cell holding the index of only one placed object, it holds a list of indices for the objects that shadow that cell and remains at an acceptable distance between themselves.

Fig. 1. Sample models of trees used in the simulator, generated with software TreeIt [29].

The texture of spawn probability acts as the probability of effectively instantiating an object at a given point. Having this as a separate node from the sampling process is useful since the number of objects spawned in an area can effectively be reduced, even to 0, making clearings of arbitrary shapes possible.

There can be created as many pipelines as required. The simulator already counts with basic pipelines for trees and bushes. The prefab models of trees, bushes, and other plants were generated with the free-to-use TreeIt software [29]. A small sample of them can be seen in Fig. 1. More models can be easily added if needed. Before instantiating these models, some transformations are applied to include more variability to the scene: a random spin around its up axis, a random twist to bend the up direction with regards to the world's up direction, and a random scale.

Grass Generation. None of the sampling methods for vegetation distribution could scale to generate millions of points while keeping the frame rate manageable. Thus, a parallelizable method was devised for placing grass. It is generated by indirect instancing, where the geometry is produced via a compute shader (a program that run on the GPU, outside of the normal rendering pipeline) and sent to the graphics pipeline through a shared memory buffer. The sampling algorithm for the grass leaves is depicted in Algorithm 1. As the tiles are non-overlapping, the processing for each tile can be parallelized over the number of tiles. The generated points are then transformed into the terrain coordinates using parallel raycasting. For a 256 × 256 pixels texture, a tile size of 4 pixels and a maximum of 1024 points per tile, the shader takes less than one second to run approximately 4 million points, running in an Intel Core i9-10900 processor, with 32 GB RAM and a NVIDIA GTX 1060 board.

The points are then fed into a compute shader that generates the geometry for a single blade. The number of blade segments can be customized. To add a realistic feeling to the grass, the following transformations are also applied to each blade: a random jitter to the anchor point because of the grid-like pattern of the sampling algorithm; a random rotation that sets which direction the blade is oriented; a random bend for the tip of the blade and a random scaling. After these transformations, the points are returned to world space coordinates to be

Algorithm 1. Grass Sampling

```
Input: texture $T$, tile size $t_s$, maximum number of points for a tile $P$
Output: points for grass leaf instancing $G$
Divide the texture $T$ into non-overlapping tiles according to a tile size $t_s$
Define an empty list $G$ for the points for grass instancing
for each tile $t$ do
    Define density $d = \frac{1}{t_s^2} \sum_{(i,j) \in t} p_{i,j}$, where $p_{i,j}$ is the texture's value in pixel $(i, j)$
    Define the number of sampling points in $t$ by $p = d \times P$
    Define points $G_t$ by distributing $p$ in a grid-like pattern in the tile
    Append $G_t$ to $G$
end for
return generated points $G$
```

placed over the terrain. Finally, a shader applies a grass texture to each leaf to add volume and color.

Approximately 4 million blades of grass, each composed of 9 points, can be generated and updated at $20 \sim 30$ fps in the mentioned hardware, displaying all grass blades simultaneously. It is worth noting that this instancing method is not used with trees, bushes, and other plants because the random jitter prevents us from enforcing a minimum radius distance between instanced objects to avoid collisions.

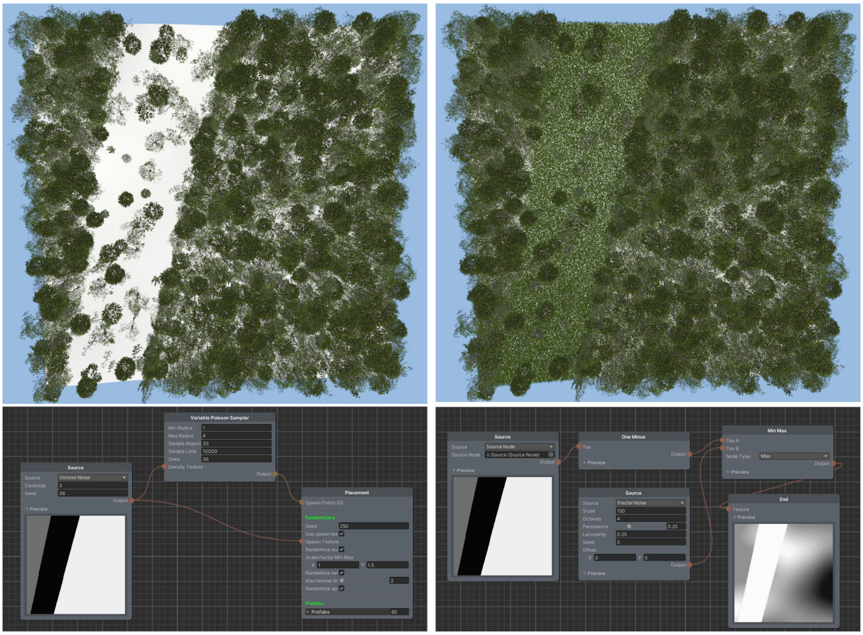
Repeatability. Each scene is generated using a seed to ensure repeatability. This seed is transmitted to every vegetation pipeline and to the grass and terrain generators. Figure 2 shows an example of the pipelines for generating trees and grass and a top-down view of the resulting forest scene, and Fig. 3 shows the same scene from a front view and a closer view.

Point Cloud Extraction. The point cloud of the generated scene can be extracted directly from the Unity Editor as a .csv file. By tagging the instanced objects with meshes, they can be exported as various categories, including but not limited to terrain, canopy, trunk, branches, bushes, understorey, grass, cacti, and deadwood, assigning the corresponding label to each point of the point cloud. The size of the scene's point cloud can be altered by adding more points to the terrain mesh, by generating more grass leaves or changing their number of segments, or by importing other vegetation prefabs with the desired quantity. This customization helps generating scenes where its point cloud can vary in size, and thus be adapted to specific needs, such as training large deep learning networks.

As one of the contributions of this work, the code of the presented forest simulator was released[1].

---

[1] https://github.com/lrse/forest-simulator.



Variable Poisson Sampler
Min Radius 1
Max Radius 4
Sample Rejecti 30
Sample Limit 10000
Seed 36
Density Texture
Output

Source
Source Voronoi Noise
Centroids 3
Seed 36
Output
Preview

Placement
Spawn Points OS
Randomizers
Seed 250
Use spawn tex ☑
Spawn Texture
Randomize sca ☑
Scale Factor Min Max
X 1 Y 1.5
Randomize twi ☑
Max Normal Ar 2
Randomize spi ☑
Prefabs
Prefabs 60

Source
Source Source Node
Source Node Source (Source Node)
Output
Preview

One Minus
Tex
Output
Preview

Min Max
Tex A
Tex B
Node Type Max
Output
Preview

Source
Source Fractal Noise
Scale 150
Octaves 4
Persistance 0.25
Lacunarity 0.25
Seed 0
Offset
X 0 Y 0
Output
Preview

End
Texture
Preview

Fig. 2. Above: Forest scene with trees (left), and trees, bushes, and grass (right). Below: the correspondence pipelines for instancing trees (left), and grass (right).

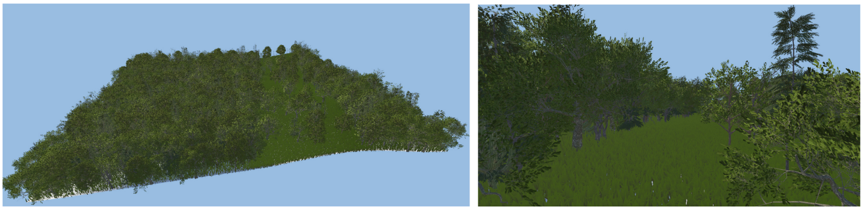


Fig. 3. Left: Frontal view of a generated forest scene. Right: Close up view.

3.3 Dataset Assembling

Two datasets were created to train the selected deep learning architectures, simulating the point clouds obtained by LiDAR and applying structure from motion algorithm to synthetic camera images, both from a top-down view. For the camera-like dataset, a method to include occlusion to the point cloud was used [30], as several points should not be visible from a top-down view of the forest. Then, random noise with zero mean is added to give variability to the point clouds, and it is partitioned in the xy plane using K-Means clustering to assemble subclouds with which to train the networks. Both datasets, LiDAR-like and Camera-like, and the seeds used to generate the scenes are publicly available[2] to ensure this work's results are reproducible and to increase the point cloud datasets available to the scientific community.

The Evo Dataset, given by [12], was employed to test the trained architectures. We used an occluded version of this dataset to test the architectures trained with the Camera-like dataset, using the same occlusion method [30].

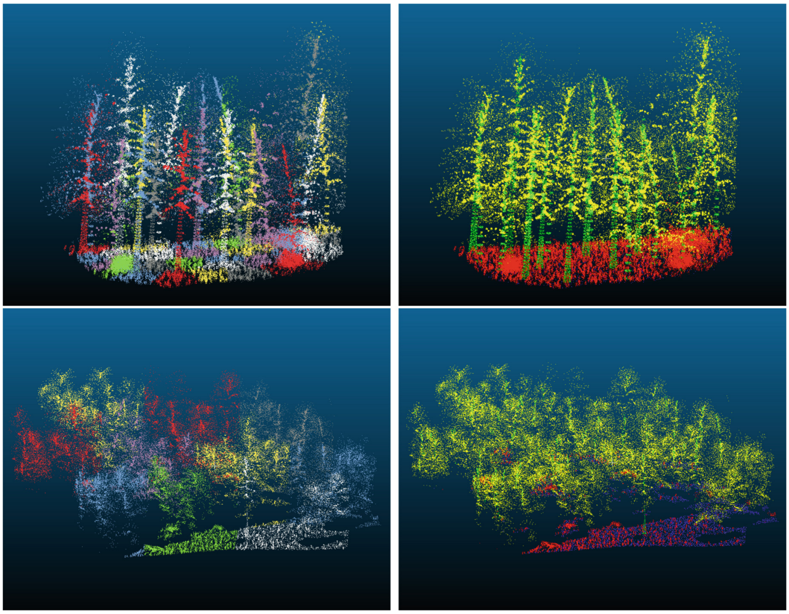


Fig. 4. Above: Example scene without occlusion segmented using K-Means (left) and segmented via labels: blue, green, yellow, and red corresponding to terrain, trunks, canopy, and understorey, respectively (right). Below: Same example for another scene, but using occlusion from a top-view point. (Color figure online)

To create a synthetic dataset that resembles the Evo Dataset, we have extracted four categories from the simulator: terrain, trunks, canopy, and understorey, the latter including grass, bushes, and all other vegetation that are not trees. In Fig. 4, an example of both datasets segmented using K-means and segmented into the studied categories can be seen. In Table 1 there is a summary of some statistics of both datasets. Note that the percentages of each category are not uniformly distributed, similarly as in the Evo Dataset.

4 Results and Discussion

For experimental results, the four selected architectures were trained with an AMD Phenom II X6 1075T Processor CPU, with 32GB RAM, and two NVIDIA

---

[2] https://github.com/lrse/synthetic-forest-datasets.

Table 1. LiDAR-like and Camera-like dataset statistics.

Dataset	Number of scenes used	Total Points	Terrain Points [%]	Trunk Points [%]	Canopy Points [%]	Understorey Points [%]
LiDAR-like	6	120,460,000	0.0171	0.1066	0.3599	0.5164
Camera-like	20	31,230,000	0.0767	0.03654	0.4454	0.4414

RTX 3090 connected with an SLI bridge. 75 epochs were used for every network, with an average training time of 472 s per epoch for the LiDAR-like dataset and 239 s for the Camera-like dataset. Further increasing the number of epochs did not increase significantly the results while testing with real forest data. In Fig. 5 we can see that the accuracy of the networks converge within the scope of the selected number of epochs for both datasets.

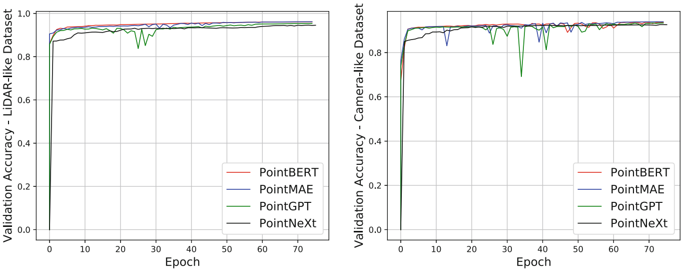
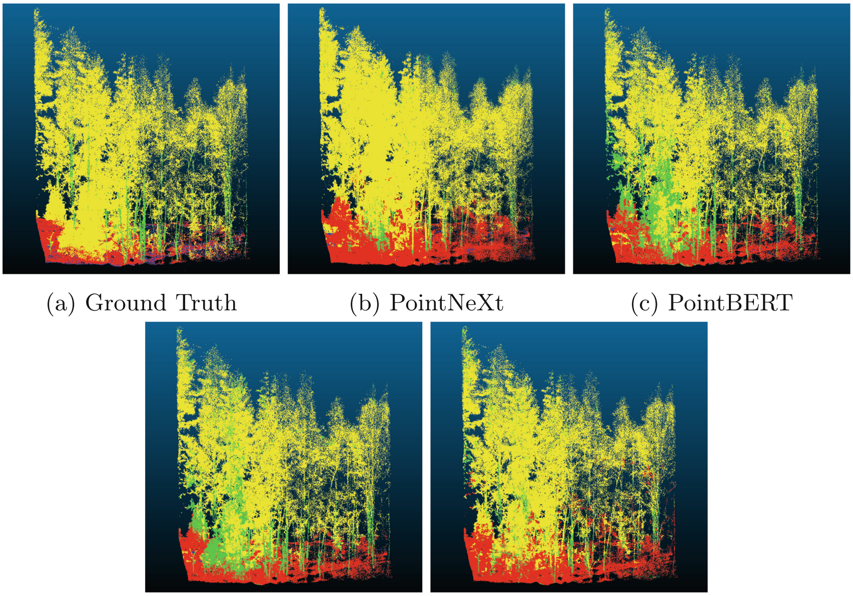


Fig. 5. Overall accuracy obtained for the studied networks while training the LiDAR-like (left) and Camera-like (right) datasets.

4.1 LiDAR-Like Experiment

Table 2 shows the results of testing the networks trained with the LiDAR-like dataset with the Evo dataset. The confusion matrix of each network is presented in Table 3. It can be seen that, regardless of having an overall good accuracy, the networks still struggle to differentiate understorey from terrain, especially when the vegetation is on a near-ground level. Trunk and canopy points are also confused, especially in branch points that are classified as canopy. This may be related to the structure of the assets used in the simulator. Assets that relate more closely to the Evo trees may throw better results. However, using more specifically generated assets may result in a loss of generalization over other forest environments. PointNeXt had a better result in classifying the points in the Evo dataset, and therefore it seems more suitable for LiDAR datasets, although PointMAE has better class average accuracy. It is also noticeable that the other three networks performed better in the Synthetic Forest Dataset than PointNeXt, but worse in the Evo Dataset. This behaviour may be due to a strong learning of the synthetic features, but a lack of ability to generalize it when applied to real data. Despite this, in Sect. 4.3 we show that the networks perform well in segmenting the trees from the rest of the categories. In Fig. 6 we can see a small fraction of the Evo dataset classified with each network.



(a) Ground Truth (b) PointNeXt (c) PointBERT

(d) PointMAE (e) PointGPT

Fig. 6. Fraction of the Evo Dataset classified with the selected networks: terrain (blue), trunks (green), canopy (yellow) and understorey (red). (Color figure online)

4.2 Camera-Like Experiment

Table 4 shows the results of testing the networks trained with the Camera-like dataset with the Evo dataset with occlusion. The confusion matrices of each network can be seen in Table 5. Similar to the previous case, the networks struggle to differentiate terrain points from understorey points, and as few trunk points remain visible, especially the ones at the base of the trunk, it is also often confused with understorey points. We can see that the overall performance is significantly lower than in the case without occlusion. PointBERT and PointMAE obtains slightly better accuracy than the other networks, although all have similar responses. In Fig. 7 we can see a small fraction of the Evo dataset classified with each network. Again, when considering only two categories, Tree and Non-tree, the overall accuracy achieves better results (see Sect. 4.3).

Table 2. Results obtained testing both on the created Synthetic Forest Dataset and in the Evo dataset, using the LiDAR-like dataset.

Network	**Sythetic Forest Dataset**			**Evo Dataset**		
	Overall Accuracy	Class Avg. Accuracy	Class Avg. mIoU	Overall Accuracy	Class Avg. Accuracy	Class Avg. mIoU
PointNeXt	0.9373	0.8055	0.7291	**0.7878**	0.5695	**0.4858**
PointBERT	0.9592	0.8867	0.8180	0.7010	0.6336	0.4093
PointMAE	**0.96077**	**0.8868**	**0.8185**	0.7269	**0.6342**	0.4266
PointGPT	0.9537	0.8536	0.7759	0.6528	0.5316	0.3616

Table 3. Confusion matrix for the four studied networks.

PointNeXt				
	Terrain	Trunk	Canopy	Understorey
Terrain	369303	1659	99640	**976022**
Trunk	4	105801	**218950**	62017
Canopy	403	235029	**8469286**	1091116
Understorey	79625	8288	172479	**1995818**

PointBERT				
	Terrain	Trunk	Canopy	Understorey
Terrain	331318	114867	78758	**1568793**
Trunk	42	**464666**	148591	43254
Canopy	4922	1228512	**6649283**	686987
Understorey	51466	129174	95209	**2289598**

PointMAE				
	Terrain	Trunk	Canopy	Understorey
Terrain	272741	26422	24020	**1769680**
Trunk	9	**426090**	179322	51339
Canopy	6392	1038688	**6971328**	553048
Understorey	52880	50075	40230	**2423176**

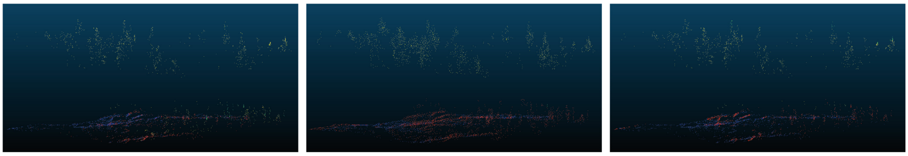
PointGPT				
	Terrain	Trunk	Canopy	Understorey
Terrain	187120	12496	39142	**1856968**
Trunk	146	231938	**247044**	176434
Canopy	5801	729701	**6177700**	1656671
Understorey	15953	23934	55922	**2468470**

4.3 Tree and Non-Tree Segmentation

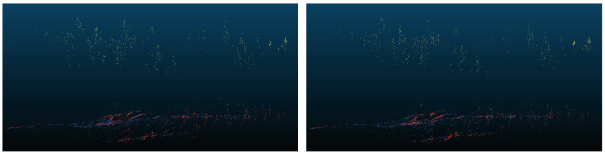
Table 6 shows the results obtained considering only Tree and Non-Tree categories, the first one including trunk and canopy points, and the latter including terrain and understorey points, training with both datasets, LiDAR-like and Camera-like, and testing with the Evo Dataset. It can be seen that the tree segmentation obtains a high accuracy percentage with all the considered networks, being PointMAE the one with slightly better results.

Table 4. Results obtained testing both on the created Synthetic Forest Dataset and in the Evo dataset, using the Camera-like dataset.

Network	Sythetic Forest Dataset			Evo Dataset		
	OverallAccuracy	Class Avg. Accuracy	Class Avg. mIoU	Overall Accuracy	Class Avg.Accuracy	Class Avg. mIoU
PointNeXt	**0.9491**	0.6829	0.5877	0.6885	0.5144	0.4069
PointBERT	0.9368	0.7405	0.6445	**0.7126**	0.4963	0.4452
PointMAE	0.9402	**0.7496**	**0.6596**	0.7109	**0.5669**	**0.4613**
PointGPT	0.9341	0.7188	0.6174	0.6008	0.5364	0.3944



(a) Ground Truth (b) PointNeXt (c) PointBERT



(d) PointMAE (e) PointGPT

Fig. 7. Fraction of the Evo Dataset classified with the selected networks: terrain (blue), trunks (green), canopy (yellow) and understorey (red). (Color figure online)

Table 5. Confusion matrix for the four studied networks.

PointNeXt					**PointBERT**				
	Terrain	Trunk	Canopy	Understorey		Terrain	Trunk	Canopy	Understorey
Terrain	4911	0	0	**7884**	Terrain	**11444**	0	0	2792
Trunk	32	4	26	**329**	Trunk	126	12	30	**434**
Canopy	38	0	**6991**	970	Canopy	173	99	**6050**	856
Understorey	1144	0	0	**4295**	Understorey	**3140**	0	0	1468

PointMAE					**PointGPT**				
	Terrain	Trunk	Canopy	Understorey		Terrain	Trunk	Canopy	Understorey
Terrain	**10573**	0	0	3828	Terrain	**8758**	3	0	5587
Trunk	71	120	27	**339**	Trunk	80	224	18	**284**
Canopy	132	188	**6041**	615	Canopy	121	405	**4717**	1752
Understorey	**2431**	2	0	2194	Understorey	**2346**	32	0	2297

4.4 Fine-Tuning with Real Data

Table 7 shows the results obtained after applying an additional fine-tuning training phase to the previously trained networks with the LiDAR-like dataset (Sect. 4.1) with a small fraction of the Evo Dataset (approximately 10% of the dataset), and testing with the remaining points. Table 8 shows the associated confusion matrix. It can be seen that both the accuracy and mIoU significantly increases in all cases compared to the obtained training only with synthetic data. All the networks performed similarly, although PointBERT and PointMAE give the best results.

This process could not be repeated with the networks trained with the Camera-like dataset (Sect. 4.2), as very few points remains in the Evo Dataset after applying the algorithms to simulate occlusion. This reinforces the advan-

Table 6. Results obtained considering the categories Tree and Non-Tree, testing in the Evo dataset and training in both the LiDAR-like and the Camera-like datasets.

Network	LiDAR-like Dataset			Camera-like Dataset		
	Overall Accuracy	Class Avg.Accuracy	Class Avg.mIoU	Overall Accuracy	Class Avg.Accuracy	Class Avg. mIoU
PointNeXt	0.8966	0.9052	0.7836	0.9485	0.9184	0.8835
PointBERT	0.9169	0.9153	0.8333	0.9470	0.9077	0.8732
PointMAE	**0.9460**	**0.9519**	**0.8890**	**0.9514**	**0.9196**	**0.8894**
PointGPT	0.8680	0.8863	0.7563	0.9146	0.8519	0.7377

tage of using synthetic data over the lack or difficulty of obtaining real data from forest environments.

Table 7. Results obtained considering the four categories, after a fine-tuning with a fraction of the Evo Dataset.

Network	LiDAR-like Dataset		
	Overall Accuracy	Class Avg.Accuracy	Class Avg.mIoU
PointNeXt	0.8669	0.5916	0.6132
PointBERT	**0.8755**	**0.7804**	**0.6670**
PointMAE	**0.8755**	0.7802	0.6658
PointGPT	0.8302	0.6776	0.5926

5 Conclusions and Future Work

In this work, we developed an open-source simulator based on Unity Engine that generates realistic forest scenes procedurally. With it, we have created synthetic point-based datasets, with each point labeled into one of the categories: terrain, trunk, canopy, and understorey (including grass, bushes, and other vegetation that are not trees). We then employed these datasets to train four state-of-the-art deep-learning point-based networks. Finally, we tested and compared them in the real forest Evo dataset. The results show that synthetic point cloud data can be used to train deep-learning networks for posterior forest segmentation with real data. Among the tested networks, PointNeXt seemed to give better overall accuracy when trained with the LiDAR-like dataset, although PointMAE had a better average accuracy. PointMAE also obtained slightly better results when trained with the Camera-Like dataset. When considering only two categories, Tree and Non-tree, a higher accuracy percentage is obtained, and PointMAE also achieved the best results. Moreover, after applying a fine-tuning training with a small proportion of the Evo Dataset, the results significantly increase, giving PointBERT and PointMAE the best results, and reinforcing the notion of pre-training with synthetic data and fine-tuning with a small amount of real data to achieve a better performance without the need of creating larger datasets

Table 8. Confusion matrix for the four studied networks after a fine-tuning with a fraction of the Evo Dataset.

	PointNeXt					**PointBERT**			
	Terrain	Trunk	Canopy	Understorey		Terrain	Trunk	Canopy	Understorey
Terrain	**1037663**	1477	8657	254311	Terrain	**1350273**	5152	9192	486200
Trunk	1622	5209	**307303**	24976	Trunk	813	**420985**	168130	12326
Canopy	4306	18880	**8481171**	145389	Canopy	1782	128501	**7396903**	94541
Understorey	762910	3858	109044	**1178568**	Understorey	488835	13861	126933	**1640917**

	PointMAE					**PointGPT**			
	Terrain	Trunk	Canopy	Understorey		Terrain	Trunk	Canopy	Understorey
Terrain	**1382793**	2485	9002	455813	Terrain	**1123618**	24613	58181	644310
Trunk	765	**417884**	167733	15351	Trunk	1418	282205	**293743**	24981
Canopy	1834	106894	**7399689**	114228	Canopy	57133	89410	**7078084**	397312
Understorey	546291	8966	106817	**1608799**	Understorey	450261	27989	188923	**1603163**

for the training phase. Taking all the experiences into account, PointMAE seems the most suitable network to work with the proposed approach.

As future work, we aim to test this approach on other datasets, especially the Camera-like experiment with fine-tuning with real data. We will also perform experiments with simulated flight surveys, using Structure from Motion to extract point clouds from images to ensure a more realistic scenario.

References

1. Murtiyoso, A., et al.: Virtual forests: a review on emerging questions in the use and application of 3D data in forestry. Int. J. For. Eng. **35**(1), 29–42 (2024)
2. Guimarães, N., Pádua, L., Marques, P., Silva, N., Peres, E., Sousa, J.J.: Forestry remote sensing from unmanned aerial vehicles: a review focusing on the data, processing and potentialities. Remote Sens. **12**(6), 1046 (2020)
3. Pessacg, F., et al.: Simplifying UAV-based photogrammetry in forestry: how to generate accurate digital terrain model and assess flight mission settings. Forests **13**(2), 173 (2022)
4. Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., Bennamoun, M.: Deep learning for 3D point clouds: a survey. IEEE Trans. Pattern Anal. Mach. Intell. **43**(12), 4338–4364 (2020)
5. Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: ScanNet: Richly-annotated 3d reconstructions of indoor scenes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5828–5839. IEEE Xplore, Honolulu, United States of America (2017)
6. Uy, M. A., Pham, Q. H., Hua, B. S., Nguyen, T., Yeung, S. K.: Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 1588–1597. IEEE Xplore, Seoul, Korea (2019)
7. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3D ShapeNets: a deep representation for volumetric shapes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1912–1920. IEEE Xplore. Boston, United States of America (2015)

8. Yi, L., et al.: A scalable active framework for region annotation in 3D shape collections. ACM Trans. Graph. **35**(6), 1–12 (2016)
9. Behley, J., et al.: SemanticKITTI: a dataset for semantic scene understanding of lidar sequences. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9297–9307. IEEE Xplore, Seoul, Korea (2019)
10. Jin, S., Su, Y., Zhao, X., Hu, T., Guo, Q.: A point-based fully convolutional neural network for airborne lidar ground point filtering in forested environments. IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens. **13**, 3958–3974 (2020)
11. Krisanski, S., Taskhiri, M.S., Gonzalez Aracil, S., Herries, D., Turner, P.: Sensor agnostic semantic segmentation of structurally diverse and complex forest point clouds using deep learning. Remote Sens. **13**(8), 1413 (2021)
12. Kaijaluoto, R., Kukko, A., El Issaoui, A., Hyyppä, J., Kaartinen, H.: Semantic segmentation of point cloud data using raw laser scanner measurements and deep neural networks. ISPRS Open J. Photogramm. Remote Sens. **3**, 100011 (2022)
13. Unity Homepage. https://unity.com/. Accessed 20 Feb 2024
14. Hu, X., Yuan, Y.: Deep-learning-based classification for DTM extraction from ALS point cloud. Remote Sens. **8**(9), 730 (2016)
15. Lê, H.Â., Guiotte, F., Pham, M.T., Lefèvre, S., Corpetti, T.: Learning digital terrain models from point clouds: ALS2DTM dataset and rasterization-based GAN. IEEE J. Sel. Top. Appli. Earth Observ. Remote Sens. **15**, 4980–4989 (2022)
16. Li, B., et al.: Terrain-Net: a highly-efficient, parameter-free, and easy-to-use deep neural network for ground filtering of UAV LiDAR data in forested environments. Remote Sens. **14**(22), 5798 (2022)
17. Qi, C. R., Yi, L., Su, H., Guibas, L. J.: PointNet++: deep hierarchical feature learning on point sets in a metric space. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
18. Qian, G., et al.: PointNext: revisiting PointNet++ with improved training and scaling strategies. Adv. Neural. Inf. Process. Syst. **35**, 23192–23204 (2022)
19. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, 30 (2017)
20. Yu, X., Tang, L., Rao, Y., Huang, T., Zhou, J., Lu, J.: Point-BERT: pre-training 3D point cloud transformers with masked point modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 19313–19322. IEEE Xplore, New Orleans, United States of America (2022)
21. Pang, Y., Wang, W., Tay, F.E.H., Liu, W., Tian, Y., Yuan, L.: Masked autoencoders for point cloud self-supervised learning. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part II, pp. 604–621. Springer Nature Switzerland, Cham (2022). https://doi.org/10.1007/978-3-031-20086-1_35
22. Chen, G., Wang, M., Yang, Y., Yu, K., Yuan, L., Yue, Y.: PointGPT: auto-regressively Generative Pre-training from Point Clouds. In: Advances in Neural Information Processing Systems, vol. 36 (2024)
23. Rolfe, J. T.: Discrete variational autoencoders. In: 9th International Conference on Learning Representations (2017)
24. Morton, G.M.: A Computer Oriented Geodetic Data Base and A New Technique in File Sequencing. International Business Machines Company, New York (1966)
25. Nunes, R., Ferreira, J. F., Peixoto, P.: Procedural generation of synthetic forest environments to train machine learning algorithms. In: ICRA 2022 Workshop in Innovation in Forestry Robotics: Research and Industry Adoption (2022)

26. SynPhoRest Dataset Homepage. https://zenodo.org/records/6369446. Accessed 21 Feb 2024
27. Russell, D.J., et al.: UAV mapping with semantic and traversability metrics for forest fire mitigation. In: ICRA 2022 Workshop in Innovation in Forestry Robotics: Research and Industry Adoption (2022)
28. Bridson, R.: Fast Poisson disk sampling in arbitrary dimensions. SIGGRAPH Sketches **10**(1), 1 (2007)
29. TreeIt Homepage. http://www.evolved-software.com/treeit/treeit. Accessed 27 Feb 2024
30. Katz, S., Tal, A., Basri, R.: Direct visibility of point sets. ACM Trans. Graph. **26**(3), 24 (2007)

Brain Age Estimation of Alzheimer's and Parkinson's Affected Individuals Using Self-Attention Based Convolutional Neural Network

Raveendra Pilli[1], Tripti Goel[1](✉), R. Murugan[1], and M. Tanveer[2]

[1] Biomedical Imaging Lab, National Institute of Technology Silchar, Silchar, Assam, India
`triptigoel@ece.nits.ac.in`
[2] Department of Mathematics, Indian Institute of Technology Simrol, Indore, India

Abstract. In recent years, the efficacy of deep learning models in accurately estimating brain age using structural magnetic resonance imaging (MRI) images has been extensively utilized. This study employs a self-attention-based convolutional neural network (CNN) to extract features from preprocessed MRI slices. While CNNs have shown remarkable performance in brain age prediction, they often fail to capture global dependencies within images. A self-attention mechanism is integrated into the CNN architecture to discern long-range relationships within the MRI images and enhance feature extraction robustness. Subsequently, a single hidden layered random vector functional link (RVFL) network is employed to predict the age of healthy individuals. The discrepancy between true age and predicted age, termed the brain age gap, serves as a biomarker for the early diagnosis of neurological disorders. Furthermore, the proposed age estimation framework is evaluated using an Alzheimer's and Parkinson-affected dataset, demonstrating its versatility and potential for clinical applications.

Keywords: Brain age · Magnetic Resonance Imaging · Random vector functional link network · Regression · Self-attention

1 Introduction

As people age, the structure and functions of their brains change, and on the other hand, neurological disorders like Alzheimer's and Parkinson's disease cause significant changes to the brain's architecture [1,2]. Neurodegenerative disorders primarily target neurons in the brain, resulting in cognitive decline, memory loss, and diminished mental function in Alzheimers disease (AD), as well as movement disorders, loss of motor function, and stiffness in Parkinson's disease (PD) [3]. The pathological characteristics of PD encompass the degeneration of neurons in the substantia nigra and the accumulation of ubiquitinated protein deposits

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 81–96, 2025.
https://doi.org/10.1007/978-3-031-78128-5_6

within the cytoplasm of neurons [4]. The brain age estimation (BAE) model is a potential strategy for facilitating the early detection and diagnosis of various neurological illnesses. The BAE model estimates an individual's brain's real (chronological) age by analyzing various brain datasets, encompassing functional and structural brain pictures acquired from neuroimaging modalities. Most prediction models are developed using data from healthy individuals, suggesting that they reflect the natural aging progression. As such, they offer a platform for investigating how different diseases impact the aging process [5].

Neuroimaging methods have been developed to forecast an individual's brain age using morphological alterations in brain areas.T1-weighted Magnetic resonance imaging (MRI) data has performed well in determining brain age among other neuroimaging modalities. This modality successfully shows age-related alterations in morphology and captures the brain structure. As a result, most of the research has used T1-weighted MRI data to estimate brain age [6,7]. The disparity between the predicted and chronological age is known as the brain age gap (BAG), which can used as a biomarker for the early detection of neurological disorders. A high BAG represents accelerating aging due to the effect of neurological ailments, and a low BAG represents normal aging or delayed aging due to physical activities and meditations.

Machine learning (ML) techniques have played a significant role in brain age classification and estimation tasks in recent years. Some common ML approaches like Gaussian process regression (GPR), support vector regression (SVR), Random Forest (RF), and Ridge regression (RR) networks have been widely used to develop brain age estimation frameworks [8,9]. Traditional ML methods effectively predict brain age, but they require feature extraction strategies, and manually extracted features impact model performance. Deep learning (DL) models are widely used in computer vision tasks such as classification, regression, and segmentation. They excel by combining feature extraction, reduction, and prediction, surpassing traditional ML methods, particularly with large datasets [10]. Convolutional Neural Network (CNN) is a powerful data-driven model known for its automatic feature extraction capability from complex patterns in input data by expanding the gradient backpropagation method with layers that preserve spatial correlations in the input data given to the network [11]. However, CNN-based algorithms neglect long-range dependencies in images and instead concentrate on the local information in an image, reducing the value of the overall relationship. A self-attention mechanism is incorporated in the CNN design to detect long-range relationships in images to extract robust features and improve the efficacy of the brain age prediction model. In this context, the signal at a particular position is determined by taking a weighted sum of features from all positions within the same image.

Backpropagation (BP) based methods encounter several issues, including slow convergence, learning rate sensitivity, and local minima susceptibility. Single-layer feed-forward Networks (SLFNs) overcome these limitations of BP models, offering faster training and superior generalization results. The random vector functional link (RVFL) network, proposed by Pao et al. [12], stands out as one of the most notable SLFNs. It initializes input weights and biases randomly while determining the output weights analytically. In RVFL, input and hidden layer output features are forwarded to the output layer. An in-depth evaluation of RVFL networks is conducted by Zhang and Suganthan [13]. They experimented with 121 UCI datasets and concluded that direct links exhibit better performance and no significant effect of bias in the output layer. The RVFL network finds application in diverse fields, such as breast cancer detection [14], emotion recognition [15], prediction of wastewater in tree plantation [16], and wind power prediction [17].

The automatic feature extraction capabilities of CNN enhanced with self-attention and the randomized RVFL network, serve as motivation for implementing a brain age estimation model. Such a model is pivotal for the early detection of neurological disorders and for analyzing the status of brain structure. The major contributions of the present work are

- images are preprocessed using the statistical parametric mapping (SPM-12) toolbox before conducting experiments.
- The computed BAG is a valuable tool for clinicians to effectively assess the health status of individuals.
- A CNN model with a self-attention mechanism is employed to extract robust features from the 2D slices, aiming to enhance the reliability of feature extraction crucial for accurate age prediction.
- The study utilized a randomized RVFL network to predict brain age, comparing its performance against various conventional regression networks.
- Additionally, the age prediction model is validated using AD and PD-affected data to estimate the brain age of unhealthy controls.
- The computed Brain Age Gap (BAG) serves as a valuable tool for clinicians in effectively assessing the health status of individuals.

The rest of the paper is organized as age prediction-related, preliminary works are discussed in Sects. 2 and 3, and the methodology is explained in Sect. 4. Section 5 demonstrates the results and discussion, and finally, the conclusion is discussed in Sect. 6.

2 Related Works

Aging impacts both the structure and function of the brain, and this can be identified through neuroimaging modalities. The brain ages due to advancements in cell growth and myelination, but it also regresses due to cell death and shrinkage. These changes don't occur randomly; certain brain areas undergo more significant changes than others. Valizadeh et al. [18] conducted a study to assess the effectiveness of six ML algorithms like multiple linear regression (MLR), deep neural network (DNN), K-nearest neighbors, SVR, RR, and RF networks on healthy individuals ages 7 to 96 years across different cohorts. Their investigation revealed that the MLR method, employing concise morphological measures from various brain regions, achieved notable prediction accuracy, yielding a correlation coefficient R^2 value of 0.73. Furthermore, they demonstrated that the DNN approach outperformed other methods through various combinations of diverse morphological features. Cole et al. [19] segmented the T1-weighted MRI images into gray matter (GM) and white matter (WM) maps and concatenated them into a unified feature matrix based on vector product. The GPR network is used to predict the chronological age and achieved 5.02 years of mean absolute error (MAE) and 6.31 years of root mean square error (RMSE) values. Similarly, Baecker et al. [20] assessed the performance of SVR, RVR, and GPR ML algorithms using morphological features of HC subjects. Their results suggested minimal disparities in accuracy, with 3.77 to 3.81 years of MAEs across the three algorithms examined.

Huang et al. [21] employed a VGG-based 2D-CNN model to forecast the brain age of 1099 HC subjects ranging from 20 to 80 years old, using T1-weighted MRI images. Their CNN architecture comprises five convolutional blocks and three fully connected layers, achieving an MAE of 4 years. In another study [22], a 3D-CNN is devised to predict the age of HC subjects, yielding an MAE of 3.67 years. While the 3D-CNN demonstrates superior performance to the 2D-CNN, it has a trade-off in computational cost and necessitates high-end hardware resources. Lam et al. [23] emphasized the drawbacks of 3D-CNN compared to 2D-CNN for brain age estimation, such as the extensive parameter requirements and computational complexity during training. The authors found that 2D-CNN efficiently extracts important features, utilizing a sagittal slice-based recurrent neural network for age prediction.

Preprocessing MRI images is essential to optimize the brain age estimation model's effectiveness. ML models necessitate manual feature extraction methods, whereas DL models autonomously derive robust features from input MRI images. Specifically, 3D-CNN models are specialized for predicting and classifying brain age, but they come with the drawback of higher computational demands. 2D models have been developed to reduce the computational cost. This section provides an overview of previous research in the domain, offering inspiration and essential insights for developing the brain age estimation framework. The subsequent section outlines the methodology to be employed.

3 Preliminary Works

The current section demonstrates preliminary works like self-attention mechanisms and the RVFL network.

3.1 Self-Attention Mechanism

A self-attention mechanism [24] enables the model to prioritize different parts of the input sequence with varying weights, aiding in capturing long-range dependencies and enhancing performance. Within self-attention, each input feature map is associated with key (K), query (Q), and value (V) vectors. The key vectors encode information from the input sequence to compute attention scores. The query vectors question the key vectors to assess their relevance in generating the output. The value vectors hold the actual content used to produce the output, influenced by the attention weights derived from key and query vectors. The output vector is calculated by combining values through a weighted sum, where each weight is determined by a Softmax compatibility function based on the query and key vectors. The architecture of the self-attention mechanism is shown in Fig. 1. The attention function can be expressed as

$$\text{Attention(Q,K,V)} = Softmax\left(\frac{QK^T}{\sqrt{d}}\right)V, \tag{1}$$

where d is the dimensions of the key vector.

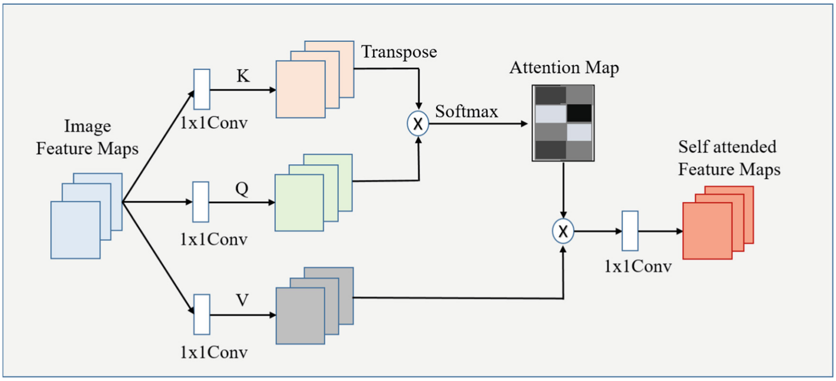


Fig. 1. Self-attention mechanism architecture

3.2 Random Vector Functional Link (RVFL) Network

In the RVFL regression network, weights w and biases b between input and hidden nodes are randomly initialized and fixed during the training and testing. The direct connections between input and output improve the generalization of the RVFL network. The graphical representation of the RVFL network is shown in Fig 2.

The RVFL network consists of only one enhancement layer (hidden layer). RVFL is supplied with X training dataset with M samples, each containing p features. The enhancement layer weights are structured as a matrix with dimensions of $p \times Q$, where Q denotes hidden layer neurons. The input samples are multiplied by the generated weights to produce the input for the "radbas" activation function $k()$ within the neurons of the enhancement layer. The output

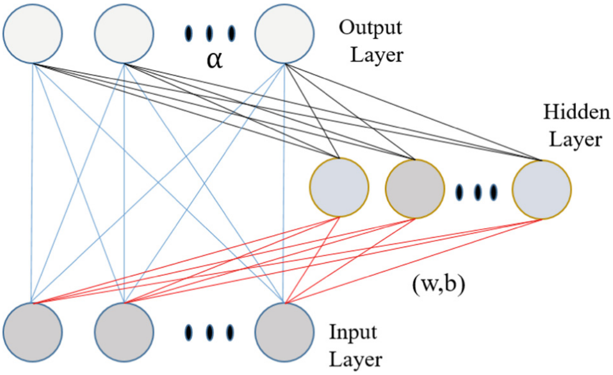


Fig. 2. Graphical representation of the RVFL network

layer then receives a combination of the hidden layer output H and the input X, resulting in the concatenated matrix $Z = [H\ X]$. X and H are shown as

$$X = \begin{bmatrix} x_{11} & \dots & x_{1p} \\ \dots & \dots & \dots \\ \dots & \dots & \dots \\ x_{M1} & \dots & x_{Mp} \end{bmatrix}, H = \begin{bmatrix} h_1(x_1) & \dots & h_Q(x_1) \\ \dots & \dots & \dots \\ \dots & \dots & \dots \\ h_1(x_M) & \dots & h_Q(x_M) \end{bmatrix}.$$

Here, $h_1(x_1) = k(w_1.x_1 + b_1)$ and so on. The output weights α needed to be calculated, and the optimal solution expressed as

$$\min_{\alpha} \|Z\alpha - Y\|^2 + \lambda \|\alpha\|^2, \tag{2}$$

where $\lambda = \frac{1}{C}$ is the regularization parameter, and Y is the output vector.

The output weights α can be calculated using ridge regression or matrix inversion with Moore-Penrose pseudoinverse (MPPI) methods. For MPPI method $\alpha = Z^{+}Y$ and for ridge regression α value can be expressed as

$$Primal\ space : \alpha = (Z^T Z + \lambda I)^{-1} Z^T Y, \tag{3}$$

$$Dual\ space : \alpha = Z^T (Z Z^T + \lambda I)^{-1} Y. \tag{4}$$

4 Methodology

The current section describes the dataset, preprocessing, feature extraction, and regression network used to predict the age. The architecture of the proposed age estimation model is shown in Fig. 3.

4.1 Dataset

In this study, 1150 structural MRI scans are acquired from Information Extraction from Images (IXI, https://brain-development.org), Southwest University Adult Lifespan Dataset (SALD, http://nitrc.org/indi/sald), Parkinson's

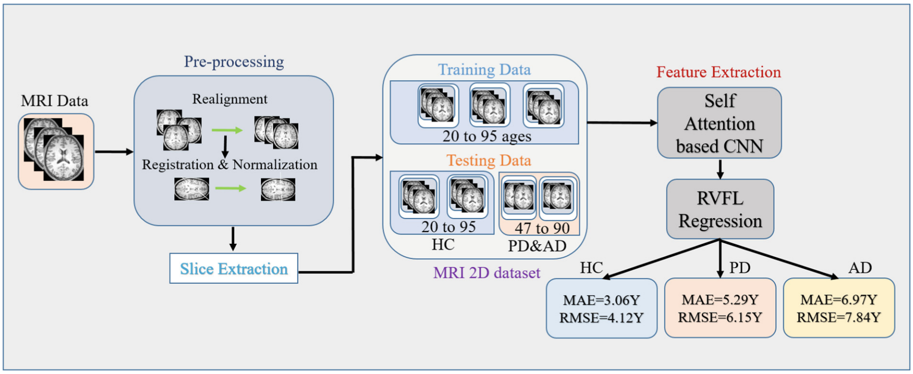


Fig. 3. Architecture of the brain age prediction model

Progression Makers Initiative (PPMI, https://www.ppmi.com), and Alzheimer's Disease Neuroimaging Initiative (ADNI, https://www.adni.loni.usc.edu/) open-source neuroimaging databases. In the IXI database, HC subjects are captured by a 3T Phillips scanner with 1.2 mm slice thickness and dimensions $256 \times 256 \times 150$. In SALD, all MRI scans are HC subjects collected from a 3T scanner with a slice thickness of 1 mm and $256 \times 256 \times 176$ dimensions. From the PPMI database, 100 samples of PD subjects are collected, and the dimensions are $256 \times 256 \times 191$ (Table 1).

Table 1. Demographical information of the Dataset

Dataset	Subjects	Mean Age	SD	TR	TE
IXI	560	48.67	16.47	9 Sec	4 $mSec$
SALD	490	45.07	17.39	1.9 Sec	2.5 $mSec$
PPMI	100	64.29	14.76	3.9 $mSec$	8.4 $mSec$
ADNI	100	74.29	14.76	2.4 $mSec$	7.9 $mSec$

SD: Standard Deviation, TE: Echo Time, TR: Repetition Time

4.2 Preprocessing

The preprocessing of 3D T1-weighted MRI images has been done by utilization of the statistical parametric mapping (SPM12) toolbox. This preprocessing is essential for illness detection as it identifies defects, outliers, and missing information. During preprocessing, image realignment corrects any misalignment or motion, image normalization standardizes intensity values using the Montreal Neurological Institute (MNI)-152 template, and image registration ensures uniformity across scans. Following preprocessing, MRI images are resized to dimensions of $256 \times 256 \times 150$, with each scan containing 150 slices. Three middle slices from the sagittal plane of each MRI scan are then extracted, resized to 224 × 224, and inputted into the self-attention-based CNN framework.

4.3 Proposed Model Architecture

The feature extraction model comprises four convolutional blocks, each containing a convolutional layer followed by the ReLU activation function, batch normalization (BN), and maxpooling (MP) layer. Convolutional layers extract features from the input image via filters, ReLU functions enable learning of complex patterns, BN accelerates training, and MP layers reduce the spatial dimensionality of feature maps over a 2 × 2 pooling window. The convolutional layers employ [32, 64, 128, 256] filters, with a kernel size of 3 × 3. MRI images contain hierarchical structures of features, from edges and textures to more intricate patterns like anatomical structures or pathological abnormalities. The increasing number of filters accommodates this hierarchy, ensuring the model can discern and accurately capture diverse features. After each convolutional block, a self-attention layer is incorporated to enhance the extraction of significant features. The self-attention-based CNN model hyperparameters are as follows: learning rate: 0.001, batch size (BZ): 26, optimizer: Adam, and number of epochs: 16 (Fig. 4).

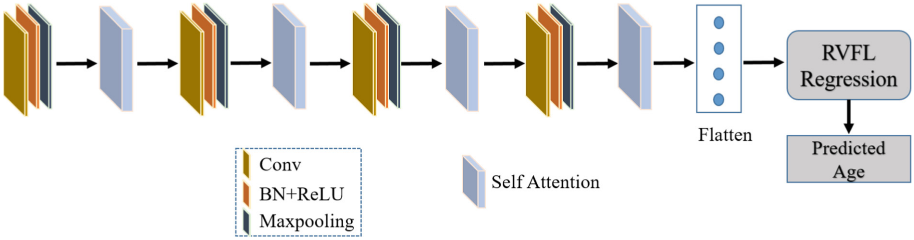


Fig. 4. Architecture of Self attention based CNN for feature extraction

The self-attention mechanism can focus on various parts of the input MRI image while determining the representation of each element within the image, and it enables the model to capture long-range dependencies effectively. Finally, the flattened layer provides the single feature matrix fed to the RVFL regression network for accurate age prediction. RVFL networks are utilized significantly in diverse environments due to reduced training time and strong performance outcomes. They operate with a single hidden layer where weights and biases are randomly initialized within a predefined range and remain fixed throughout the training phase. The output weights are computed using a closed-form solution. An advantageous feature of RVFL is its direct linking of input features to the output layer, which acts as a regularization method, enhancing overall network performance.

Depending on the input dataset, Eq. (3) or (4) can be chosen, and by multiplying α with $Z = [H\ X]$, the model's final prediction can be achieved. The prediction model output at the node is determined as

$$f(x) = [h(x)\ x]\alpha, \tag{5}$$

$$f(x) = [h(x)\ x]Z^T(ZZ^T + \lambda I)^{-1}Y. \tag{6}$$

The next section will discuss the experiments and results of the self-attention-based CNN for robust feature extraction and RVFL regression network for brain age estimation using HC, PD and AD datasets.

5 Results and Discussion

The following section demonstrates the results and discussion of the present work.

5.1 Experimental Implementation Details

The evaluations are conducted using a computer with an Intel(R) i7 processor and 16GB of RAM loaded with MATLAB R2023a and Google Colab. The HC brain dataset is divided into a 70:30 for training and testing the brain age prediction framework. In the RVFL network, the number of hidden units is chosen from the set $\{256, 512, 1024\}$, and the regularization parameter, denoted as λ, is selected from 2^k, where k=$\{-4, -2, \dots 10\}$. The regularization parameter C value is taken as 0.5, and hidden nodes Q are taken as 500. The "radbas" activation function determines an optimal hyperparameter configuration for training and evaluating the RVFL model.

The evaluation of the regression networks is assessed by metrics such as MAE, RMSE, and Pearson's correlation coefficient (PCC).

$$\text{MAE} = \frac{1}{M}\sum_{i=1}^{M} |o_i - x_i|, \tag{7}$$

$$\text{RMSE} = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(o_i - x_i)^2}, \tag{8}$$

$$\text{PCC} = \frac{\sum_{i=1}^{M}(o_i - \overline{o}_i)(x_i - \overline{x}_i)}{\sqrt{\sum_{i=1}^{M}(o_i - \overline{o}_i)^2}\sqrt{\sum_{i=1}^{M}(x_i - \overline{x}_i)^2}}. \tag{9}$$

where o_i is predicted age and x_i is real age, and $\overline{o}_i$ is mean value of o_i and $\overline{x}_i$ is mean value of x_i.

5.2 Comparison Between the State-of-the-Art Regression Networks on HC Training Dataset

The performance evaluation of the proposed brain age estimation framework on the 2205 MRI 2D samples of the HC training dataset is summarized in Table 2. The proposed model is compared with several conventional regression networks, including SVR, GPR, RF, RR, and Linear-R, based on metrics such as MAE, RMSE, BAG, and PCC. On the HC training dataset, the RVFL regression model achieved 2.62 years of MAE, 3.57 years of RMSE, and a PCC of 0.98. The SVR model yielded an MAE of 5.19 years and a PCC of 0.90, while the GPR model obtained an RMSE of 5.94 years. The RF and RR models provided prediction accuracy of 4.97 years and 5.91 years for MAE and RMSE, respectively. The suggested model achieved lower error outcomes than the conventional regression networks on the HC training dataset.

Table 2. Various regression networks assessment on HC training dataset

Regression model	MAE (years)	RMSE (years)	BAG (years)	PCC
SVR [25]	5.19	6.08	0.97	0.90
GPR [26]	5.06	5.94	0.92	0.90
RF [27]	4.97	5.86	0.83	0.91
RR [28]	4.99	5.91	0.86	0.91
Linear-R [29]	4.96	5.89	0.83	0.91
Proposed	**2.62**	**3.57**	**0.072**	**0.98**

5.3 Comparison with State-of-the-Art Regression Networks on HC, PD, and AD Testing Dataset

Table 3 presents a comparative analysis of regression models on 945 HC MRI slices and 300 AD, 300 PD MRI slices from testing datasets. The proposed regression model on the HC testing dataset achieved an MAE of 3.06 years and 0.12 years of BAG. The Linear-R model yielded an RMSE of 6.35 years and a PCC of 0.89, while the GPR regression model obtained an MAE of 5.73 years. Additionally, the RR model achieved a BAG of 1.183 years. The RVFL regression model yielded an MAE of 5.29 years, 6.15 years of RMSE, and a BAG of 3.13 years on the PD testing dataset. The SVR model obtained a BAG of 7.13 years and a PCC of 0.41, while the RF model obtained an MAE of 7.46 years and a BAG of 7.01 years.

On the AD testing dataset, the suggested model yielded an MAE of 6.97 years, and a BAG of 5.03 years. It is noteworthy that the error outcomes for AD subjects are higher than those for HC and PD subjects, indicating that AD-affected brains appear older than their actual age, with abnormal structural brain variations in AD subjects. Figure 5 shows the scatter plot between the predicted age and chronological age of HC, PD, and AD testing datasets.

Table 3. Various regression networks assessment on HC, PD and AD testing dataset

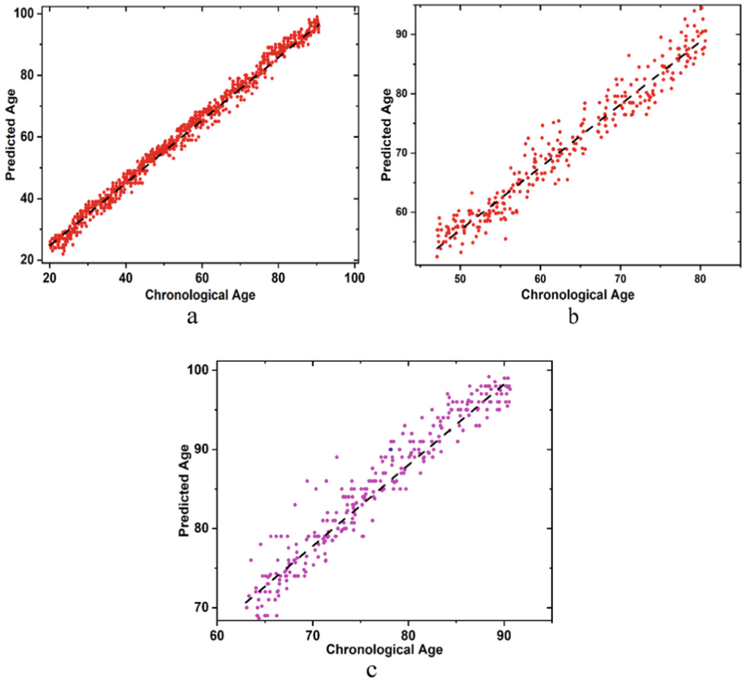
Regression Model	Group	MAE (years)	RMSE (years)	BAG (years)	PCC
SVR [24]	HC	5.89	6.93	1.23	0.87
	PD	7.73	8.66	7.13	0.41
	AD	9.67	10.58	8.41	0.37
GPR [25]	HC	5.73	6.81	1.16	0.88
	PD	7.52	8.62	7.06	0.42
	AD	9.57	10.49	8.25	0.39
RF [26]	HC	5.52	6.49	1.19	0.88
	PD	7.46	8.38	7.01	0.44
	AD	9.66	10.43	8.36	0.39
RR [27]	HC	5.50	6.43	1.183	0.89
	PD	7.49	8.32	7.03	0.44
	AD	9.58	10.78	8.40	0.39
Linear-R [28]	HC	5.41	6.35	1.16	0.89
	PD	7.45	8.52	7.02	0.44
	AD	9.23	10.38	8.95	0.39
Proposed	**HC**	**3.06**	**4.12**	**0.12**	**0.96**
	PD	**5.29**	**6.15**	**3.13**	**0.62**
	AD	**6.97**	**7.84**	**5.03**	**0.53**

5.4 Comparison with Different DL Models

The features from the MRI scans are extracted using different deep neural networks (DNNs) like AlexNet, InceptionV3, GoogleNet, and SqueezeNet individually. The extracted features of each DNN are fed to the RVFL model separately to estimate the brain age. Table 4 shows the comparison between the suggested model and various DL models on testing datasets.

5.5 Computational Complexity

The current study includes the computational complexity (CC) of self-attention-based CNN and RVFL networks. The CC of each convolutional layer is $O(BZ \times H \times W \times C_{in} \times C_{out} \times K \times K)$, and the CC of BN layer is $O(BZ \times H \times W \times K \times K \times C_{out})$. The MP layer's CC is $O(BZ \times H \times W \times C_{out})$ The self-attention layer adds CC of $O(BZ \times H \times W^2 \times d)$. The RVFL output weights are determined by matrix inversion, the CC of RVFL is $O((x+Q)^3)$. C_{in}, C_{out} are input and output channels, BZ is batch size, $H \times W$ spatial dimensions, and $K \times K$ kernel size.



a

b

c

Fig. 5. Scatter plot between chronological age and predicted age a) HC b) PD c) AD testing datasets

5.6 Discussion

Brain age estimation techniques are vital in identifying early symptoms of neurological disorders and cognitive decline. Determining the brain age gap helps assess individuals' brain health status, facilitating timely interventions and medication to prevent the onset of diseases [30]. In this study, 1050 HC subjects are acquired from publicly available databases to train and test the age prediction model. MRI images underwent preprocessing using SPM-12 MATLAB-based software, with three middle slices extracted from 3D images to simplify model complexity. A self-attention-based CNN with four convolutional blocks is employed to extract features, capturing both local and global dependencies from the images. We utilized an RVFL regression network with a "radbas" activation function in the hidden layers for accurate age prediction. The direct connections between input and output layers in the RVFL network enhance model accuracy and reliability. This study introduces novel advancements in brain age estimation by integrating state-of-the-art methodologies. It employs a CNN enhanced with a self-attention mechanism for robust feature extraction from MRI slices, significantly improving accuracy in predicting brain age. Furthermore, pioneering the application of a randomized RVFL network for regression, the model surpasses traditional methods, offering a unique and practical approach to pre-

Table 4. Evaluation of various DL models on HC, PD, and AD testing dataset

Regression Model	Group	MAE (years)	RMSE (years)	BAG (years)	PCC
AlexNet	HC	4.65	5.71	0.15	0.90
	PD	5.92	6.23	6.35	0.69
	AD	7.15	8.34	7.23	0.54
InceptionV3	HC	4.08	5.12	0.09	0.92
	PD	5.42	6.36	6.19	0.70
	AD	6.44	7.74	7.09	0.54
GoogleNet	HC	4.12	5.38	0.11	0.92
	PD	5.45	6.43	6.22	0.70
	AD	6.21	7.74	6.37	0.54
SqueezeNet	HC	3.91	4.98	0.085	0.92
	PD	5.07	6.12	5.94	0.70
	AD	6.47	7.92	7.09	0.54
Proposed	**HC**	**3.06**	**4.12**	**0.12**	**0.96**
	PD	**5.29**	**6.15**	**3.13**	**0.62**
	AD	**6.97**	**7.84**	**5.03**	**0.53**

cise brain age estimation. The comparison between existing brain age estimation models and the current estimation model is summarized in Table 5. Furthermore, when assessed with AD and PD testing subjects, our model yielded an MAE of 5.29 years and 6.15 years of RMSE on the PD dataset, 6.97 years of MAE, and 5.03 years of BAG on the AD dataset.

The suggested model shows a close alignment in RMSE values between training (3.57 years) and testing (4.12 years) datasets, suggesting it does not excessively fit the training data. It indicates strong generalization capabilities, mini-

Table 5. Comparison between proposed model and existing models

Existing Model	Regression Model	Group	MAE (years)	RMSE (years)	BAG (years)	R^2	PCC
Valizadeh et al.	SVR	HC	–	–	–	0.84	–
	RF		–	–	–	0.81	–
Cole et al.	GPR	HC	5.02	6.31	–	–	0.94
Baecker et al.	SVR	HC	3.77	4.65		0.51	0.74
	RVR		3.82	4.65	–	0.51	0.74
	GPR		3.81	4.64	–	0.51	0.74
Huang et al.	2D-CNN	HC	4.04	5.13	–	–	0.94
Ueda et al.	3D-CNN	HC	3.67	4.71	–	–	0.96
Proposed	**CNN+ RVFL**	**HC**	**3.06**	**4.12**	**0.12**	-	**0.96**
		PD	**5.29**	**6.15**	**3.13**	-	**0.62**
		AD	**6.97**	**7.84**	**5.03**	-	**0.53**

mizing the need for additional regularization or complexity reduction techniques to mitigate overfitting.

The present study demonstrates promising results with HC subjects; however, further validation with larger and more diverse datasets is crucial. This includes incorporating longitudinal studies to assess the model's stability and ability to accurately track age-related changes. Moreover, exploring the model's performance across diverse demographic groups and clinical conditions will enhance its applicability and reliability in real-world settings. Our proposed brain age estimation model holds significant potential for clinical applications. By precisely determining the BAG, clinicians can effectively evaluate brain health conditions and identify early indicators of cognitive decline or neurological disorders.

6 Conclusion

Age-related diseases and neurological disorders are increasingly burdening society, highlighting the need for biomarkers for the underlying biological aging process to examine individuals at higher risk of age-related physical and cognitive decline. This study focuses on estimating the brain age of healthy individuals, Parkinson's disease (PD), and Alzheimer's disease (AD) affected subjects using structural MRI images and deep learning models. By assessing whether a person's brain appears younger or older than their actual age, this approach offers insights into deviations from typical healthy brain aging, as observed in PD and AD subjects. In AD subjects, the affected brain structure appears older than the individual's actual age among healthy and PD subjects.

Future research can expand upon this by incorporating additional neuroimaging modalities such as functional MRI (fMRI), positron emission tomography (PET) images, and diffusion-weighted imaging (DWI) images. The model's hyperparameters can be optimized by suitable techniques to improve the prediction accuracy. This would enable a more comprehensive understanding of functional, metabolic, and chemical variations in the aging brain, both in normal and abnormal aging.

Acknowledgment. This work is supported by a Core Research Grant provided by the Science and Engineering Research Board (SERB) for financial assistance under Grant No. CRG/2022/006866.

References

1. Franke, K., Gaser. C.: Longitudinal changes in individual BrainAGE in healthy aging, mild cognitive impairment, and Alzheimer's disease. GeroPsych (2012)
2. Chen, Z., Wu, B., Li, G., Zhou, L., Zhang, L., Liu, J.: Age and sex differentially shape brain networks in Parkinson's disease. CNS Neurosci. Ther. **29**(7), 1907–1922 (2023)
3. Tokuchi, R., et al.: Differences between the behavioral and psychological symptoms of Alzheimer's disease and Parkinson's disease. J. Neurol. Sci. **369**, 278–282 (2016)

4. Kalia, L.V., Lang, A.E.: Evolving basic, pathological and clinical concepts in PD. Nat. Rev. Neurol. **12**(2), 65–66 (2016)
5. Tanveer, M., et al.: Deep learning for brain age estimation: a systematic review. Inf. Fus. (2023)
6. Mishra, S., Beheshti, I., Khanna, P.: A review of neuroimaging-driven brain age estimation for identification of brain disorders and health conditions. IEEE Rev. Biomed. Eng. **16**, 371–385 (2021)
7. Pilli, R., Goel, T., Murugan, R., Tanveer, M.: Association of white matter volume with brain age classification using deep learning network and region wise analysis. Eng. Appl. Artif. Intell. **125**, 106596 (2023)
8. Beheshti, I., Ganaie, M.A., Paliwal, V., Rastogi, A., Razzak, I., Tanveer, M.: Predicting brain age using machine learning algorithms: a comprehensive evaluation. IEEE J. Biomed. Health Inform. **26**(4), 1432–1440 (2021)
9. Baecker, L., Garcia-Dias, R., Vieira, S., Scarpazza, C., Mechelli, A.: Machine learning for brain age prediction: introduction to methods and clinical applications. EBioMedicine, 72 (2021)
10. Zhou, S.K., Greenspan, H., Shen, D. (Eds.). Deep learning for medical image analysis. Academic Press (2023)
11. Jiang, H., Chen, K., Yao, L., Li, K., Zhang, J., Guo, X.: Predicting brain age of healthy adults based on structural MRI parcellation using convolutional neural networks. Front. Neurol. **10**, 494386 (2020)
12. Pao, Y.H., Park, G.H., Sobajic, D.J.: Learning and generalization characteristics of the random vector functional-link net. Neurocomputing **6**(2), 163–180 (1994)
13. Zhang, L., Suganthan, P.N.: A comprehensive evaluation of random vector functional link networks. Inf. Sci. **367**, 1094–1105 (2016)
14. Nayak, D.R., Dash, R., Majhi, B., Pachori, R.B., Zhang, Y.: A deep stacked random vector functional link network autoencoder for diagnosis of brain abnormalities and breast cancer. Biomed. Signal Process. Control **58**, 101860 (2020)
15. Cheng, W.X., Gao, R., Suganthan, P.N., Yuen, K.F.: EEG-based emotion recognition using random convolutional neural networks. Eng. Appl. Artif. Intell. **116**, 105349 (2022)
16. Elmaadawy, K., Elaziz, M.A., Elsheikh, A.H., Moawad, A., Liu, B., Lu, S.: Utilization of random vector functional link integrated with manta ray foraging optimization for effluent prediction of wastewater treatment plant. J. Environ. Manage. **298**, 113520 (2021)
17. Al-qaness, M.A.A., Ewees, A.A., Fan, H., Abualigah, L., Elsheikh, A.H., Elaziz, M.A.: Wind power prediction using random vector functional link network with capuchin search algorithm. Ain Shams Eng. J. **14**(9), 102095 (2023)
18. Valizadeh, S.A., Hänggi, J., Mérillat, S., Jäncke, L.: Age prediction on the basis of brain anatomical measures. Hum. Brain Mapping **38**(2), 997–1008 (2017)
19. Cole, J.H., et al.: Brain age predicts mortality. Molecular Psychiatry **23**(5), 1385–1392 (2018)
20. Baecker, L., et al.: Brain age prediction: a comparison between machine learning models using region-and voxel-based morphometric data. Hum. Brain Mapping **42**(8), 2332–2346 (2021)
21. Huang, T.-W., et al.: Age estimation from brain MRI images using deep learning. In: 2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017), pp. 849–852. IEEE (2017)
22. Ueda, M., et al.: An age estimation method using 3D-CNN from brain MRI images. In: 2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019), pp. 380–383. IEEE (2019)

23. Lam, P.K., et al.: Accurate brain age prediction using recurrent slice-based networks. In: 16th International Symposium on Medical Information Processing and Analysis, vol. 11583, pp. 11–20. SPIE (2020)
24. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
25. Zhang, F., O'Donnell, L.J.: Support vector regression. In: Machine learning, pp. 123–140. Academic Press (2020)
26. Wilson, A.G., Knowles, D.A., Ghahramani, Z.: Gaussian process regression networks. arXiv preprint arXiv:1110.4411 (2011)
27. Liu, Y., Wang, Y., Zhang, J.: New machine learning algorithm: random forest. In: Liu, B., Ma, M., Chang, J. (eds.) ICICA 2012. LNCS, vol. 7473, pp. 246–252. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34062-8_32
28. Salim, I., Hamza, A.B.: Ridge regression neural network for pediatric bone age assessment. Multimedia Tools Appl. **80**(20), 30461–30478 (2021). https://doi.org/10.1007/s11042-021-10935-8
29. Lee, K-Y., et al.: Comparison and analysis of linear regression and artificial neural network (2017)
30. Pilli, R., Goel, T., Murugan, R., Tanveer, M., Suganthan, P.N.: Kernel ridge regression-based randomized network for brain age classification and estimation. IEEE Trans. Cogn. Dev. Syst. (2024)

IFSENet: Harnessing Sparse Iterations for Interactive Few-Shot Segmentation Excellence

Shreyas Chandgothia(✉), Ardhendu Sekhar, and Amit Sethi

Department of Electrical Engineering, Indian Institute of Technology Bombay,
Mumbai, India
`shreyaschandgothia@gmail.com`

Abstract. Training a computer vision system to segment a novel class typically requires collecting and painstakingly annotating lots of images with objects from that class. Few-shot segmentation techniques reduce the required number of images to learn to segment a new class, but careful annotations of object boundaries are still required. On the other hand, interactive segmentation techniques only focus on incrementally improving the segmentation of one object at a time (typically, using clicks given by an expert) in a class-agnostic manner. We combine the two concepts to drastically reduce the effort required to train segmentation models for novel classes. Instead of trivially feeding interactive segmentation masks as ground truth to a few-shot segmentation model, we propose IFSENet, which can accept sparse supervision on a single or few support images in the form of clicks to generate masks on support (training, at least clicked upon once) as well as query (test, never clicked upon) images. To trade-off effort for accuracy flexibly, the number of images and clicks can be incrementally added to the support set to further improve the segmentation of support as well as query images. The proposed model approaches the accuracy of previous state-of-the-art few-shot segmentation models with considerably lower annotation effort (clicks instead of maps), when tested on Pascal and SBD datasets on query images. It also works well as an interactive segmentation method on support images. The code will be made available at Github post paper acceptance.

Keywords: Few-shot segmentation · Interactive segmentation · Deep Learning

1 Introduction

Image segmentation tasks, which aim to label each pixel in an image, can be categorized into three main types: (i) Semantic segmentation assigns unique labels to each class, (ii) Instance segmentation assigns unique labels to separate objects, and (iii) Panoptic segmentation assigns both semantic class labels and instance IDs to each pixel. Unlike classification and detection, segmentation tasks require extensive manual effort to collect pixel-level annotated training data. However, even with ample high-quality data, segmentation models struggle with predicting novel classes with fewer samples effectively.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 97–113, 2025.
https://doi.org/10.1007/978-3-031-78128-5_7

Two frameworks, few-shot segmentation and interactive segmentation, address these issues. Few-shot segmentation uses a small support set with annotated masks for an unseen class. Typically, the aim is to segment pixels of the same class in a separate query image by learning semantic feature similarity, disregarding class-specific information. Interactive segmentation involves human-in-the-loop, allowing the model to refine segmentation masks through manual clicks, scribbles, and bounding boxes.

Both Few-Shot and Interactive approaches have drawbacks. The former focuses on predicting novel target classes but relies heavily on high-quality support masks. The latter excels at accurate masks for seen classes but is limited to segmenting one image at a time, requiring a significant number of clicks for satisfactory results on *every* test image.

We propose a semantic segmentation model that overcomes the limitations of few-shot and interactive learning methods while leveraging their strengths. Our model achieves simultaneous segmentation of unannotated (query set) images without pre-existing masks by accepting sparse user annotations in the form of clicks on a subset of images (support set). It generates dense segmentation masks for both the sparsely annotated images and the remaining unannotated images. Furthermore, the model can continuously improve results by iteratively incorporating more images or corrective clicks from the user on the support set.

Through iterative refinement, our model achieves query prediction results comparable to few-shot segmentation techniques without requiring densely annotated masks as support supervision. Additionally, our model achieves support prediction results on par with state-of-the-art interactive segmentation models.
Contributions of This Work Include:

1. **Integration of Few-Shot and Interactive Segmentation:** We combine few-shot and interactive segmentation techniques to leverage their respective strengths and address their limitations.
2. **Iterative Improvement with User Input:** Our method iteratively incorporates user-provided clicks to refine segmentation results, making it robust to user input variability.
3. **Efficient Use of Sparse Annotations:** IFSENet effectively uses sparse annotations to generate dense segmentation masks, reducing the overall annotation effort.
4. **State-of-the-Art Performance:** We achieve state-of-the-art results on Pascal VOC and SBD datasets, demonstrating the effectiveness of our approach.
5. **Applicability to Diverse Domains:** While we focus on general image segmentation, our method can be extended to other domains such as medical imaging and satellite imagery.

By directly connecting our contributions to prior work and providing an elaborative discussion, we aim to offer a comprehensive understanding of the problem foundation and the advancements made through IFSENet. This enhanced introduction benefits a wide range of readers by clearly highlighting the novelty and significance of our work in the context of existing segmentation techniques.

2 Related Work

2.1 Semantic Segmentation

Semantic segmentation involves assigning a category to each pixel in an image. The Fully Convolutional Network (FCN) [9] introduced a paradigm shift by replacing fully-connected dense layers with convolution layers, enabling end-to-end segmentation. FCN utilizes an encoder-decoder architecture with downsampling and upsampling operations. Subsequently, improved models such as U-Net [10], DeepLab [1], PSPNet [23], and HRNet [15] emerged to enhance the performance of semantic segmentation.

U-Net [10] utilizes a 'U'-shaped network architecture with systematic skip-connections. It consists of two major parts: a contracting path with convolution and pooling, and an expansive path with convolution and upsampling layers.

DeepLab [1] is a semantic segmentation model developed by Google, with progressive advancements across generations. The latest version, DeepLabv3+, introduces an enhanced ASPP module that employs parallel atrous convolutions to capture multi-scale context using multiple atrous rates. The DenseCRF post-processing module in earlier models is replaced by a more streamlined decoder module, resulting in improved segmentation refinement.

HRNet [15] introduces the concept of high-resolution representations for dense prediction tasks, such as detection and segmentation. The network incorporates multiple high-to-low-resolution convolution streams and connects them in parallel. This approach ensures the utilization of high-resolution information throughout the network.

2.2 Few-Shot Segmentation

Few-shot segmentation involves predicting pixel-level labels on an image, where the target classes differ from those seen during training. SG-One [22] addresses one-shot segmentation using masked average pooling on the support image and cosine similarity for query image feature relationships. PL [3] generates prototypes for each support class and employs cosine similarity to segment query images. PANet [16] introduces prototype alignment regularization for compatible embedding prototypes. CANet [19] employs a two-branch dense comparison module to compare multi-level features between support and query images.

PFENet [14] is a few-shot semantic segmentation model that enhances generalization on unseen classes. It achieves spatial consistency between query and support targets and effectively utilizes mid-level semantic information of training classes. The authors propose a training-free prior mask generation method, improving generalization. They also introduce a multi-scale architecture that addresses spatial inconsistency by enriching query features with support features and prior masks in an adaptive manner.

2.3 Interactive Segmentation

Interactive segmentation allows users to segment images by providing object cues, facilitating human-computer interaction. It offers a faster and more convenient alternative to manual pixel labeling for generating ground-truth masks. Grabcut [11] introduced an iterative energy minimization approach using a Gaussian mixture cost function. The first CNN-based interactive segmentation model, DIOS [18], proposed a click simulation strategy later utilized by other authors. CNN-based architectures such as [5–8], emphasized attention mechanisms. Other interactive feedback forms, like scribbles and bounding boxes, were also explored in works like [2,11,17].

RITM [13] is a click-based interactive segmentation model that incorporates segmentation masks from previous steps. Its architecture is similar to semantic segmentation models, using DeepLabv3+ with Resnet backbone and HRNet+OCR semantic segmentation architectures. Additional input channels encode spatial information of positive and negative clicks, which are combined with the backbone's output using element-wise addition. RITM samples points from the largest mislabelled region between ground truth and prediction, supporting iterative training. It also allows the optional inclusion of the output mask from a previous iteration, enhancing the power of iterative learning.

2.4 Fusion Methods in Image Segmentation

Fusion networks stand out due to their ability to integrate the high adaptability of few-shot learning with the detailed, user-driven corrections characteristic of interactive segmentation. The Support-Query Prototype Fusion Network (SQPFNet) [22] employs a two-branch structure, where one branch processes support images to generate class-specific prototypes, while the other branch handles the query image. By incorporating interactive elements, SQPFNet can adapt quickly to new tasks with limited data, ensuring more precise and reliable results. Order Learning Vision Transformer (Order-ViT) [23], designed for cancer classification in pathology images, incorporates interactive elements to refine the segmentation based on user inputs, thus demonstrating the potential of transformers in handling complex medical image segmentation tasks with minimal supervision, effectively bridging few-shot learning and interactive refinement. RheumaVIT [24], a transformer-based model, focuses on automated scoring of hand joints in rheumatoid arthritis. It combines few-shot learning principles with interactive capabilities, allowing for detailed segmentation and scoring with minimal examples. The model's design ensures that it can handle new, unseen cases effectively by leveraging prior knowledge and refining outputs based on user feedback. These examples illustrate the recent advancements in integrating few-shot learning and interactive segmentation, providing robust solutions for complex tasks in various domains, particularly medical imaging.

3 Method, Training, and Validation

Given a dataset of images, IFSENet generates binary segmentation masks, separating pixels belonging to a specific semantic class (e.g., person, car, horse) as foreground and the remaining image as background. It can handle novel classes unseen during training. The user provides information about the class to segment by giving positive and negative clicks on a subset of the images.

The images on which the user provides clicks become part of the support set S and the rest of the images constitute the query set Q. The model generates segmentation masks for both sets. Through an iterative process, the user can refine the masks by providing clicks and incorporating the predicted masks from previous iterations. This loop allows for continuous improvement of the segmentation results (Table 1).

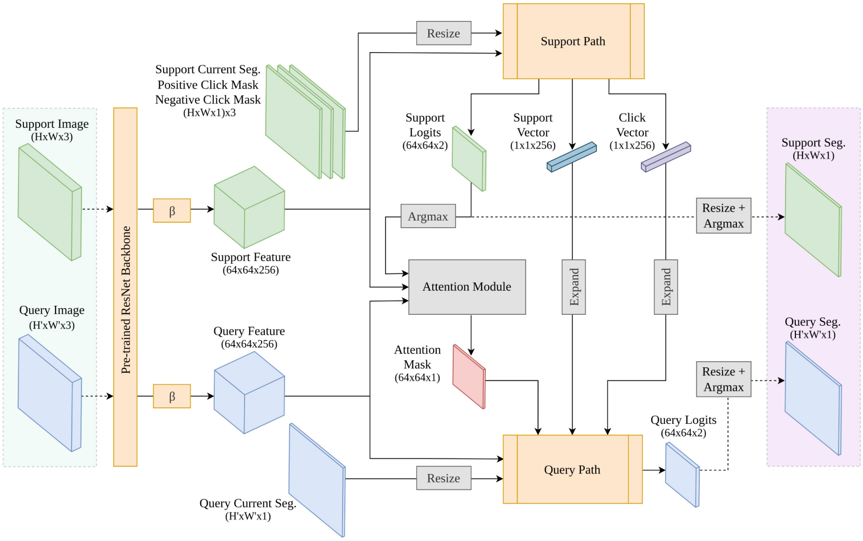


Fig. 1. Architecture of IFSENet: Notion-wise yellow blocks are operations with learnable parameters, grey blocks are training-free operations, β block is 1×1conv+RELU, resize block is spatial bilinear interpolation, argmax block operates along the channel dimension, expand block makes multiple copies of a $1 \times 1 \times$ C vector and stacks them to make the desired spatial dimension. (Color figure online)

The model's architecture, depicted in Fig. 1, focuses on learning the similarities between support and query images and leveraging the importance of clicks on the support image for supervision. During training, the model avoids learning class-specific information as it aims to predict novel classes during validation, relying on user clicks.

Table 1. Comparison of methods of General Few Shot Segmentation Models(GFSSM) with our proposed IFSENet. Genreally, few shot segmentation models use support images and their dense support masks to predict the query masks for query images. But IFSENet needs support clicks instead of support masks.

Method	Input			Output
	Support Image	Support Mask	Support Clicks	Query Mask Prediction
GFSSM	✓	✓	X	✓
IFSENet (Ours)	✓	X	✓	✓

Both the support and query images undergo feature extraction using a pre-trained ResNet backbone, originally trained on ImageNet classes. The support feature is then processed through the support path, incorporating click masks and an optional previous prediction mask, to produce a new segmentation mask for the support image. This part of the network resembles an interactive segmentation architecture. Similarly, the query feature goes through the query path, accompanied by the optional previous prediction mask, to generate an updated mask. Since direct supervision is absent for the query image, information from the support path needs to be propagated. Thus, the query path receives three additional inputs: a support vector and a click vector from the support path, as well as an attention mask from the attention module.

It should be noted that the network is capable of handling multiple support and query images. Handling multiple query images is straightforward as each prediction is independent; so the model processes them one by one to generate their respective outputs. Similarly, for multiple support images, predicting their outputs from the support path is trivial. However, it is necessary to propagate the cumulative information from the multiple support images to the query path. If there are k support images, the support path produces a total of k support vectors and k click vectors, while the attention module generates k attention masks. To combine these outputs, we simply average across the k of them and pass them to the query path.

3.1 Support Path

The support path architecture of our network is shown in Fig. 2. It follows a U-Net styled structure, having a contracting half that shrinks the feature spatial dimension 8x times, and an expanding half that restores the original spatial dimensions. We augment the feature map with three additional channels, i.e. two for click masks and one for the previously available segmentation mask, if available. Apart from the support image prediction in the form of logits, this part of the network produces two additional outputs:

Support Vector: Contains the information of the foreground pixels in the support image, which later serves as supervision for the query. To obtain this vector, the binary support prediction, calculated from the support logits, is broad-

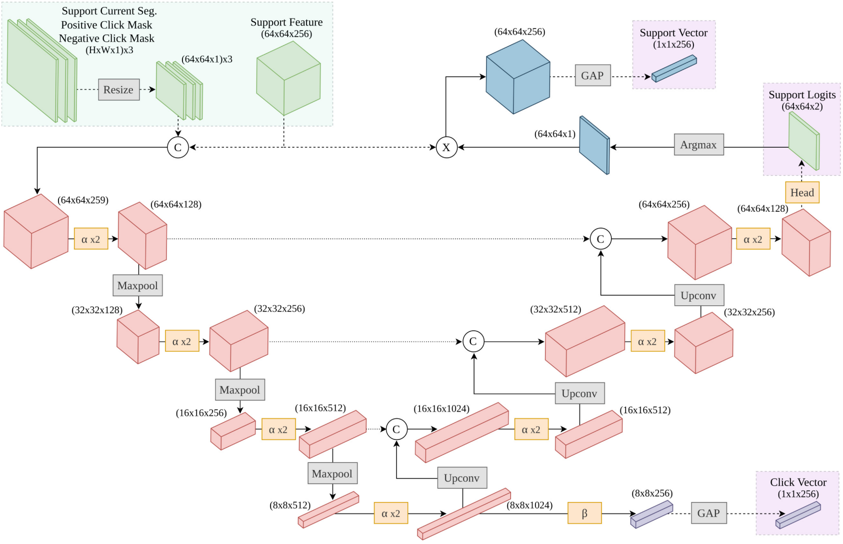


Fig. 2. Architecture of the support (see Fig. 1): Notation-wise 'C' is channel concatenation, α block is 3×3conv+RELU, β block is 1×1conv+RELU, max pool halves the spatial dimensions, upconv block doubles the spatial dimensions but also halves the channel dimension, GAP block is spatial global average pooling operation, head block is 1×1conv with 2-channel output.

casted and multiplied channel-wise with the support feature, followed by a GAP operation.

Click Vector: To propagate click information from the support to the query path, we cannot pass the click masks alone as they lack the context of the support image features. Instead, we utilize the output of the last layer of the encoder, as it contains both the support image and click information. To align the dimensions with the support vector, we reduce the channel dimensions by applying a 1×1 convolution followed by a GAP operation.

3.2 Attention Module

The attention module takes the support features, query features, and the output logits from the support path as inputs. Its primary function is to generate an attention mask for the query image, which signifies the pixel-wise correspondence between the support and query images. In other words, it determines which pixels in the query image are most similar to the foreground pixels in the support image. This attention mask is obtained by calculating the cosine similarity between the support and query features on a pixel-wise basis. A higher value in the attention mask indicates a stronger similarity between the query pixel feature and at least one foreground support pixel feature. By leveraging this attention mechanism, the model can effectively align and relate the relevant information between the support and query images, facilitating accurate segmentation.

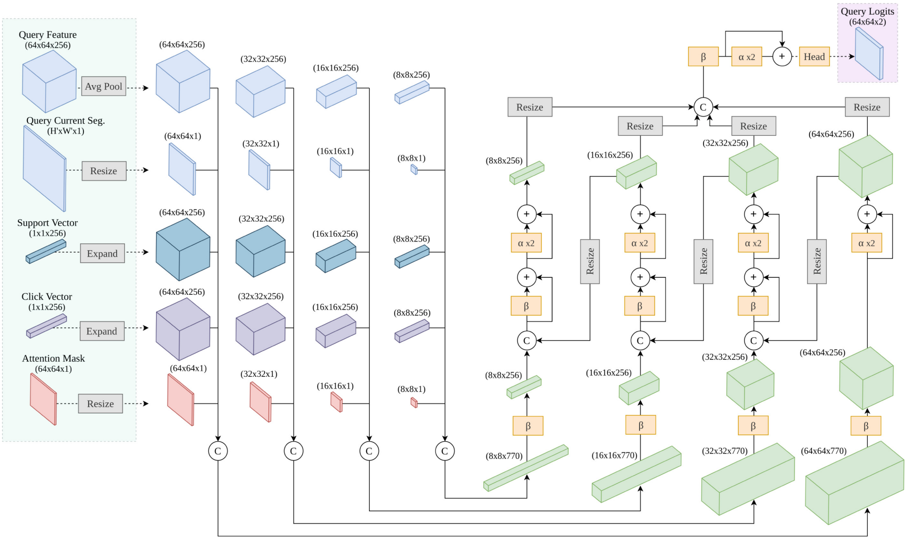


Fig. 3. Architecture of the query path (see Fig. 1): Notation-wise 'C' is channel concatenation, α block is 3×3conv+RELU operations, β block is 1×1conv+RELU operations, head block is 1×1conv operation with 2-channel output.

3.3 Query Path

The query path architecture of our network (Fig. 3) utilizes a multi-scale parallel processing approach inspired by PFENet's Feature Enrichment Module [14]. This allows for adaptive enrichment of query features with support vectors, click vectors, and attention masks. Inputs are resized, expanded, and concatenated before being processed by convolutional layers. Parallel processing layers at different scales produce multiple outputs, which are then fused and post-processed to obtain the final query prediction.

3.4 Dataset: Pascal + SBD

The Pascal Visual Object Classes (VOC) dataset is widely used in object detection, semantic segmentation, and classification tasks. It comprises 17,125 images, with 2,913 having segmentation masks for 20 object classes. The Semantic Boundaries Dataset (SBD) is a complementary dataset with 11,355 images that share the same 20 class labels as Pascal VOC. We merge the images and annotations from both datasets, prioritizing the higher-quality segmentation masks from SBD for common images. The merged dataset consists of 12,031 image-mask pairs. For few-shot segmentation, we split the dataset into training and validation classes, with 5 classes reserved for validation and 15 classes used for training. The dataset is denoted as Pascal-5^i having four folds in total.

3.5 Training Regime

To begin with, we select the subset of images from our dataset, which contain at least 1 training class object within them. We also select the number of shots ***k***. Then we run a training loop for a chosen number of epochs, where each epoch involves iterating over all the images once. Below are the steps followed for each image in each epoch:

1. Read the current image as query: (***q_image***)
2. Randomly select a target class from the query image, in case multiple classes are present. (***class_chosen***)
3. Obtain the binary ground truth mask corresponding to the ***class_chosen*** for the ***q_image***. (***q_mask***)
4. Obtain the stored predicted query mask on this ***q_image*** and this ***class_chosen*** from a previous iteration, if it exists, else initialize as a blank mask. (***q_currseg***)
5. Randomly select ***k*** images having an object from the ***class_chosen*** present within them and make them the support. (***s_images***)
6. Obtain their binary ground truth masks corresponding to the ***class_chosen***. (***s_masks***)
7. Obtain the stored predicted support masks on these ***s_images*** and this ***class_chosen*** from previous iterations, if they exist, else initialize with blank masks. (***s_currsegs***)
8. Obtain the stored positive and negative click masks for these ***s_images*** and this ***class_chosen*** from previous iterations, if they exist, else initialize as blank masks. (***s_posclicks*** and ***s_negclicks***)
9. Provide ***s_images***, ***s_currsegs***, ***s_posclicks***, ***s_negclicks*** ***q_image*** and ***q_currseg*** as input to the model.
10. Obtain the segmentation outputs and update the stored versions of ***q_currseg*** and ***s_currsegs*** with the new predictions for future iterations.

Implementation Details: Our models are trained on Pascal-5^i for 100 epochs with learning rate 0.0025 and batch size 4. We use SGD as our optimizer. Momentum and weight decay are set to 0.9 and 0.0001 respectively. We adopt the 'poly' policy to decay the learning rate. During training, samples are processed with mirror operation, random rotation from -10 to $10°$C and random crop of size 512×512. During evaluation, each input sample is resized to the training patch size but with respect to its original aspect ratio by padding zeros. We continuously save the query and support predictions as well as click masks. During the next iteration of the same training image, we use the previously saved masks with probability 0.9 or reset those to blank masks with probability 0.1. Note that the ground truth support and query masks are only required for loss calculation purposes and are not given as inputs to the model.

3.6 Validation Regime

For this part, we work with only those images, which contain at least 1 validation class object within them. The validation proceeds in an episodic fashion, where each episode begins by selecting a ***class_chosen***, number of support images ***s***, and number of query images ***q***. The episode involves iteratively providing clicks on the support images and updating all the predicted segmentation. Below are the steps followed for each episode:

1. Randomly select ***s*** images as support, having an object from the ***class_chosen*** present within them (***s_images***)
2. Randomly select ***q*** images as query, having an object from the ***class_chosen*** present within them (***q_images***)
3. Obtain their binary ground truth mask corresponding to the ***class_chosen*** (***s_masks*** and ***q_masks***)
4. Initialize ***s_currsegs***, ***s_posclicks***, ***s_negclicks*** and ***q_currsegs*** as blank masks
5. Iterate for 20 clicks:
    i. Add one positive or negative click to each support image from ***s_images***, based on the largest error region between the respective ***s_mask*** and ***s_currseg***.
    ii. Store the new click for each support image in the respective click masks ***s_posclicks*** or ***s_negclicks***.
    iii. Provide ***s_images***, ***s_currsegs***, ***s_posclicks***, ***s_negclicks*** ***q_images*** and ***q_currsegs*** as input to the model.
    iv. Obtain the segmentation outputs on all the ***s+q*** images and update their stored versions ***s_currsegs*** and ***q_currsegs*** with the new predictions.

Implementation Details: For generating the validation results, we conduct 100 episodes for each of the classes available in the dataset. We fix the number of query images per episode to $\boldsymbol{q} = 5$ and use two different values for the number of support images i.e. $\boldsymbol{s} = 1$ and $\boldsymbol{s} = 5$. Note that the ground truth support and query masks are required for metric evaluation purposes and automating the clicking process, but are not provided as inputs to the model.

3.7 Click Sampling Strategy

Training: Positive clicks are sampled randomly from the ground truth foreground (0.2) and false negative regions (0.8). Negative clicks are sampled randomly from the ground truth background (0.04), other class objects in the background (0.06), border region around the ground truth foreground (0.1), and false positive regions (0.8). If any mask is blank, probabilities are redistributed. Higher weight is given to false positive and false negative regions to simulate validation behavior.

Validation: We simulate user click behavior by identifying the largest mislabeled region between the ground truth and current prediction. A click is then provided near the center of this region. If the selected pixel is a false negative, a positive click is given; otherwise, a negative click is provided.

3.8 Evaluation Metrics

IoU is used to evaluate general semantic segmentation by measuring the overlap between masks. Values range from 0 to 1, with higher values indicating better overlap. For few-shot tasks, Class mIoU averages foreground IoU across validation classes. In interactive segmentation, mIoU is the average foreground IoU across validation images, regardless of classes.

3.9 Loss Function

We use simple pixel-level Binary Cross Entropy (BCE) loss. The total loss is the weighted sum of the losses over the k support image predictions ($\mathcal{L}_S^i$), losses over the n intermediate query predictions at the different scales ($\mathcal{L}_{Q,1}^i$) and the final query prediction ($\mathcal{L}_{Q,2}$).

$$\mathcal{L} = \frac{1}{k}\sum_{i=1}^{k} \mathcal{L}_S^i + \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_{Q,1}^i + \mathcal{L}_{Q,2}$$

4 Experiment and Results

The qualitative results for selected episodes on validation classes are shown in Fig. 4. In the below figures, the network initially mis-segments the ‘Person’ class for the ‘Potted Plant’ class but corrects it with additional support image clicks. No clicks are given on the query images in our test results, but users have the option to add clicks for poor-performing queries and transfer them to the support set. Demo videos of our results are hosted at https://drive.google.com/drive/folders/18P8vXBJVGbSHKCVFNcJDwncBMDxhXLV6?usp=sharingthis link.

Table 2. Main result: An unfair comparison of our model *using only 20 clicks per support image and no query clicks* with few-shot segmentation techniques *that use dense ground truth masks* on class mIoU for query images of validation classes from Pascal-5^i. Best results are bold, second-best results are underlined.

Method	1 Shot					5 Shot				
	Fold-0	Fold-1	Fold-2	Fold-3	Mean	Fold-0	Fold-1	Fold-2	Fold-3	Mean
SG-One [22]	40.2	58.4	48.4	38.4	46.3	41.9	58.6	48.6	39.4	47.1
PANet [16]	42.3	58.0	51.1	41.2	48.1	51.8	64.6	**59.8**	46.5	55.7
CANet [19]	52.5	65.9	51.3	51.9	55.4	55.5	67.8	51.9	53.2	57.1
PFENet [14]	**61.7**	**69.5**	**55.4**	**56.3**	**60.8**	63.1	**70.7**	55.8	**57.9**	**61.9**
Ours	59.4	66.5	50.3	51.5	56.9	**64.3**	70.1	55.9	55.7	61.5

We first present key results for query image predictions on the Pascal-5^i dataset in Fig. 5. The plot shows Class mIoU, indicating improved performance

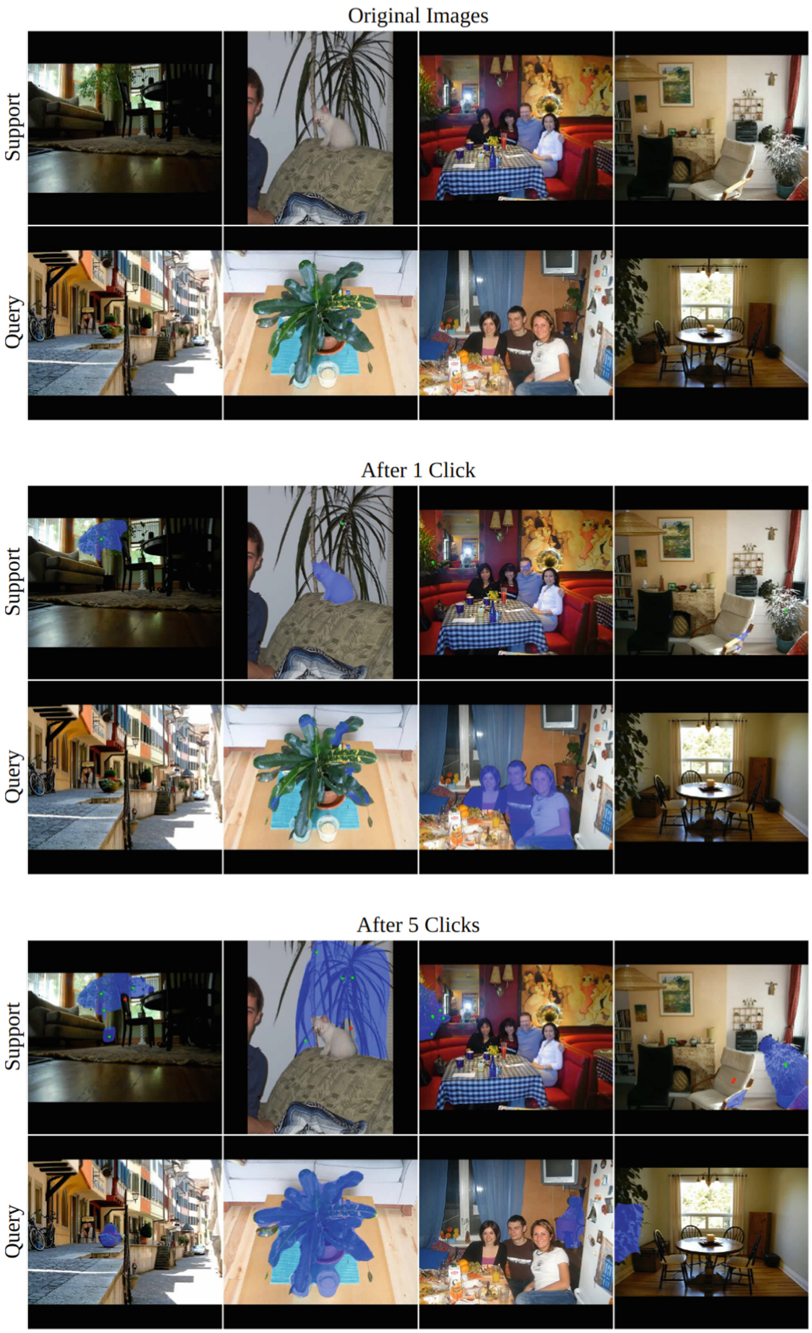


Fig. 4. Visualization of segmentation episodes on potted plants class: Positive clicks are green dots, negative clicks are red dots in the support images, and the segmentation masks are overlaid blue regions on both support and query images. (Color figure online)

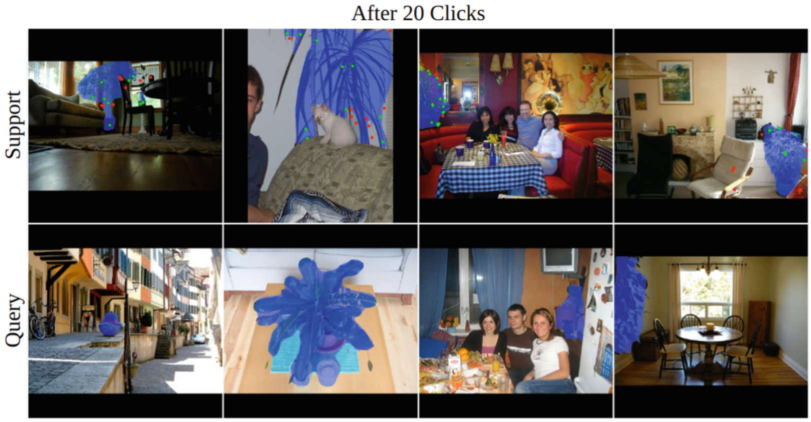


Fig. 4. (*continued*)

with support image clicks. Notably, Split 1 exhibits higher IoUs on validation classes due to easier class composition.

We compare the performance of our 1-shot and 5-shot models with previous works in few-shot segmentation on the validation classes of different folds. Table 2 demonstrates that our model achieves comparable results using sparse support information in the form of clicks, unlike previous works that rely on dense ground-truth support masks.

In Fig. 6, we present mIoU results for support image predictions on the Pascal-5^i dataset. Additional clicks have a more significant impact on the performance of support images compared to query images. The performance on validation classes eventually catches up with training classes, indicating that the model learns a meaningful understanding of clicks rather than relying solely on training class features.

We also compare our support module's mean performance on the training classes across different folds of Pascal-5^i with various previous works in interac-

Table 3. Comparison of our model with previous works on interactive segmentation using training classes of SBD. For RITM, the training dataset is mentioned in parenthesis. Best results are bold, second-best are underlined.

Method	NoC@85	NoC@90
Latent diversity [5]	7.41	10.78
BRS [4]	6.59	9.78
f-BRS-B [12]	5.06	8.08
RITM HRNet18 IT-M (SBD) [13]	**3.39**	**5.43**
RITM HRNet18 IT-M (C+L) [13]	3.80	<u>6.06</u>
Ours	<u>3.79</u>	7.08

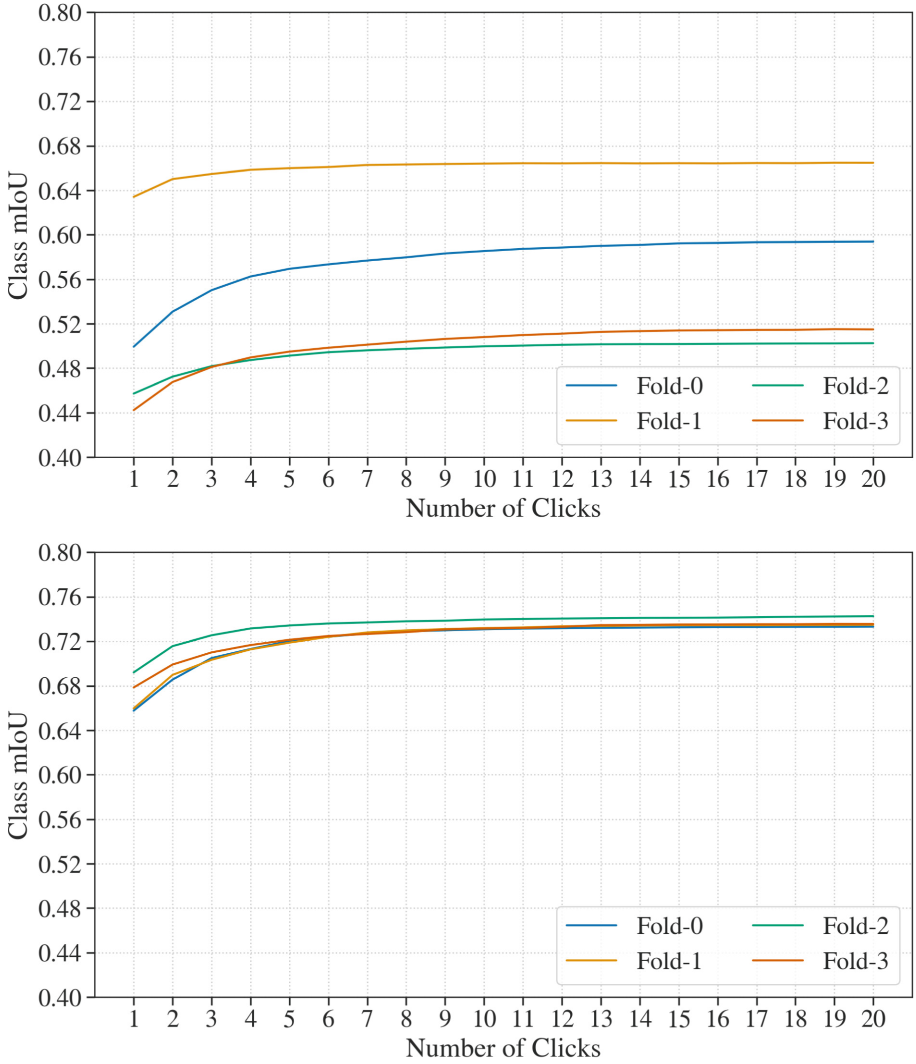


Fig. 5. Results on query predictions for our 1-shot model trained on Pascal-5^i for validation classes (upper panel) and training classes (lower panel).

tive segmentation in Table 3. The metric NoC@x denotes the mean number of clicks to achieve an image mIoU of x.

We have observed a few failure cases where the performance of the model tends to be sub-optimal. These largely include two scenarios: (1) as expected, when the object to be segmented has multiple spatially-separated fragments (for e.g. spokes of a wheel), the user needs to provide multiple clicks, (2) when the object to be segmented has another object on top of it or at close separations (for e.g. a cat sitting on a sofa), the second object has to be given at least one

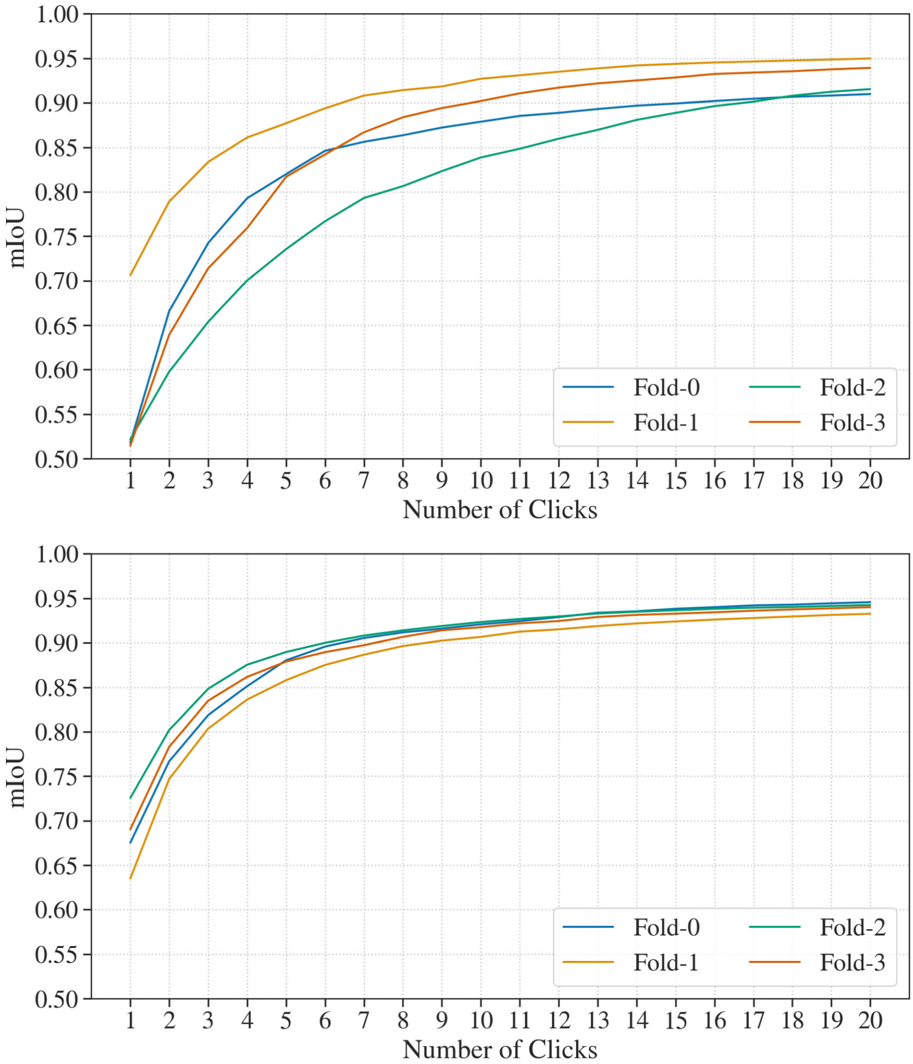


Fig. 6. Results on support predictions for our 1-shot model trained on Pascal-5^i for validation classes (upper panel) and training classes (lower panel).

negative click. In future work, we wish to extend our work to bigger datasets, such as MS COCO [25], to quantify such limiting scenarios better.

We have trained our model on the Nvidia A100 GPU having a RAM capacity of 80 GB. The model, with a size of 80 MB, was trained over 100 epochs using a batch size of 4 for individual updates and a total of 2048 images per epoch. The entire training process was completed in approximately 26 h. These benchmarks highlight the model's efficiency, making it suitable for real-time applications.

5 Conclusion and Future Work

We have presented IFSENet, a combined network that successfully makes predictions on support images with the help of clicks, and propagates the relevant information to segment query images without clicks for novel classes. The model shows significant iterative improvement with additional clicks provided. The support performance is comparable to previous state-of-the-art interactive segmentation architectures, even though we use a much simpler and lighter U-net styled architecture. Our query performance, using only sparse supervision in the form of support clicks, is comparable to previous few-shot architectures employing dense support masks.

As part of our future work, we aim to develop a high-quality GUI-based application, which aids the user in interacting with our model via clicks and runs our algorithm in the backend smoothly and efficiently. We also plan to extend our model to other image domains such as medical, satellite, etc.

References

1. Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. IEEE Trans. Pattern Anal. Mach. Intell. **40**(4), 834–848 (2017)
2. Cheng, M.-M., Prisacariu, V.A., Zheng, S., Torr, P.H.S., Rother, C.: Densecut: densely connected CRFs for realtime grabcut. In: Computer Graphics Forum, vol. 34, pp. 193–201. Wiley Online Library (2015)
3. Dong, N., Xing, E.P.: Few-shot semantic segmentation with prototype learning. In: British Machine Vision Conference (2018)
4. Jang, W.-D., Kim, C.-S.: Interactive image segmentation via backpropagating refinement scheme. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5297–5306 (2019)
5. Li, Z., Chen, Q., Koltun, V.: Interactive image segmentation with latent diversity. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 577–585 (2018)
6. Liew, J., Wei, Y., Xiong, W., Ong, S.-H., Feng, J.: Regional interactive image segmentation networks. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 2746–2754. IEEE (2017)
7. Liew, J.H., Cohen, S., Price, B., Mai, L., Ong, S., Feng, J.: Multiseg: semantically meaningful, scale-diverse segmentations from minimal user input. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 662–670 (2019)
8. Lin, Z., Zhang, Z., Chen, L.-Z., Cheng, M.-M., Lu, S.-P.: Interactive image segmentation with first click attention. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13339–13348 (2020)
9. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440 (2015)

10. 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
11. Rother, C., Kolmogorov, V., Blake, A.: "grabcut interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. **23**(3), 309–314 (2004)
12. Sofiiuk, K., Petrov, I., Barinova, O., Konushin, A.: f-brs: rethinking backpropagating refinement for interactive segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8623–8632 (2020)
13. Sofiiuk, K., Petrov, I.A., Konushin, A.: Reviving iterative training with mask guidance for interactive segmentation. In: 2022 IEEE International Conference on Image Processing (ICIP), pp. 3141–3145. IEEE (2022)
14. Tian, Z., Zhao, H., Shu, M., Yang, Z., Li, R., Jia, J.: Prior guided feature enrichment network for few-shot segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **44**(2), 1050–1065 (2020)
15. Wang, J., et al.: Deep high-resolution representation learning for visual recognition. IEEE Trans. Pattern Anal. Mach. Intell. **43**(10), 3349–3364 (2020)
16. Wang, K., Liew, J.H., Zou, Y., Zhou, D., Feng, J.: Panet: few-shot image semantic segmentation with prototype alignment. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9197–9206 (2019)
17. Wu, J., Zhao, Y., Zhu, J.-Y., Luo, S., Tu, Z.: Milcut: a sweeping line multiple instance learning paradigm for interactive image segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 256–263 (2014)
18. Xu, N., Price, B., Cohen, S., Yang, J., Huang, T.S.: Deep interactive object selection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 373–381 (2016)
19. Zhang, C., Lin, G., Liu, F., Yao, R., Shen, C.: Canet: class-agnostic segmentation networks with iterative refinement and attentive few-shot learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5217–5226 (2019)
20. Zhang, X., Wei, Y., Yang, Y., Huang, T.S.: Sg-one: similarity guidance network for one-shot semantic segmentation. IEEE Trans. Cybernet. **50**(9), 3855–3865 (2020)
21. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2881–2890 (2017)
22. Wu, X., Gao, Z., Chen, X., Wang, Y., Qu, S., Li, N.: Support-query prototype fusion network for few-shot medical image segmentation. arXiv preprint arXiv:2405.07516 (2024)
23. Lee, J.C., Kwak, J.T.: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops, pp. 2493–2502 (2023)
24. Stolpovsky, A., et al.: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops, pp. 2522–2531 (2023)
25. Lin, T.-Y.: Microsoft COCO: common objects in context. arXiv preprint arXiv:1405.0312 (2014)

Interpretable Deep Graph-Level Clustering: A Prototype-Based Approach

Jianbin Cui and Lingyang Chu(✉)

McMaster University, Hamilton, ON, Canada
`{cuij48,chul9}@mcmaster.ca`

Abstract. Many real-world data, such as chemical compounds and proteins, are naturally modelled as datasets of graphs. However, the labels of the graphs are often difficult to obtain due to the high relative cost to label graphs. To extract knowledge from a dataset of unlabelled graphs, we aim to conduct the task of interpretable graph-level clustering, which aims to find good clusters of graphs and also gain useful insights into the clustering result by interpreting why each graph is allocated to its corresponding cluster. To the best of our knowledge, this is a novel task that has not been systematically studied in the literature. In this paper, we successfully tackle this task by developing an interpretable deep graph-level clustering (IDGC) framework, which not only achieves good clustering performance, but also provides insightful interpretations on the clustering result. Extensive experiments on six benchmark datasets demonstrate the outstanding performance of our method. Our code is available at: https://github.com/cjbbb/IDGC-implementation.

Keywords: Graph-level clustering · Interpretability

1 Introduction

Many real-world data, such as chemical compounds [8] and proteins [37], are naturally modelled as graphs. However, the labels of the graphs are often difficult to obtain due to the high human labor cost and experimental cost to label chemical compounds and proteins. To extract knowledge from such a dataset of unlabelled graphs, we are interested in tackling a novel unsupervised learning task named **interpretable graph-level clustering**. The goal of the task is two-fold: 1) we aim to partition a dataset of graphs into multiple clusters, where the graphs in the same cluster are similar to each other and the graphs in different clusters are different; and 2) we want to provide additional insight into the clustering result by interpreting why each graph is allocated to its corresponding cluster.

Interpretable graph-level clustering enjoys many potential applications. As a solid example, protein clustering is a fundamental task in bioinformatics to identify meaningful and stable groups of proteins with similar structures and functionalities [37]. Conducting interpretable graph-level clustering on a dataset

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 114–129, 2025.
https://doi.org/10.1007/978-3-031-78128-5_8

of proteins not only finds such groups of proteins as clusters of graphs, but also provides useful insights into the clustering result by interpreting why each protein belongs to its corresponding group.

As discussed later in Sect. 2, some existing methods can be extended to conduct graph-level clustering. For example, the graph kernel methods [3,21,22,30], which compute the similarity between different graphs, can be coupled with spectral clustering [31] to produce clusters of graphs. The unsupervised graph representation learning methods [14,29,34,35], which learn graph embeddings to represent each graph, can also be coupled with either K-means [17] or spectral clustering to perform graph-level clustering. However, since the graph kernels and graph embeddings are designed and trained separately from the graph-level clustering task, the final clustering performance is often sub-optimal. Moreover, none of these methods can interpret why a graph is allocated to its corresponding cluster, which limits their potential to gain useful insights from the clustering results.

To the best of our knowledge, interpretable graph clustering is a novel task that has not been systematically studied before. In this paper, we tackle this challenging task by a well designed method named **interpretable deep graph-level clustering (IDGC)**. Our contributions are listed as follows. First, we propose the novel task of interpretable graph-level clustering, which aims to find good clusters of graphs while interpreting why each graph is allocated to its corresponding cluster. Second, we successfully tackle the task by the proposed IDGC method. IDGC achieves outstanding graph-level clustering performance by building an end-to-end clustering framework, which smoothly incorporates the training of graph embeddings and the clustering of graphs. It also achieves high-quality interpretations by introducing a prototype layer in the framework to learn prototypical graph structures for each cluster of graphs. Last, we conduct extensive experiments on six benchmark datasets to demonstrate the outstanding performance of our method.

2 Related Work

In this section, we discuss the relationship between our work and the existing works in the literature. Our work is generally related to the graph kernel methods [3,21,22,30] and the unsupervised graph-level representation learning methods [14,29,34,35]. We discuss these related works as follows.

2.1 Graph Kernel Methods

Graph kernel is a classic family of methods to evaluate the similarity between two graphs, thus they can be easily coupled with off-the-shelf clustering algorithms [6,31] to conduct unsupervised graph-level clustering. The clustering performance largely depends on the effectiveness of the graph kernel in accurately measuring the similarity between graphs.

Many existing graph kernels are well-developed in the literature. One famous family of graph kernels focus on identifying the optimal subgraphs and using them to compute the similarity between two graphs. These methods include: graphlet kernel [22], random walk kernel [30], shortest path kernel [3], subgraph matching kernel [15] and pyramid matching kernel [18]. Another famous family of graph kernels aims to extend the Weisfeiler-Lehman isomorphism test [32]. A typical work is the Weisfeiler-Lehman kernel [21], which is notable for its efficiency in handling large graphs. Based on the Weisfeiler-Lehman isomorphism test, this kernel enhances graph labels by combining the original node labels with an ordered set of neighboring node labels to encapsulate a subtree structure. Another typical approach is the shortest-path kernel [22], which focuses on paths rather than walks and cycles to evaluate the similarity between graphs.

Despite the elegant design of graph kernels, their effectiveness in graph-level clustering is often sub-optimal [10,25,35], because the descriptive power of hand-designed features is often limited [7] and many graph kernels are not specifically customized to facilitate graph-level clustering.

2.2 Unsupervised Graph Representation Learning

By leveraging the power of deep graph neural networks, unsupervised graph representation learning aims to learn a high-dimensional embedding vector (i.e., representation) for the node in a graph or for an entire graph. The well-learned representation is a general purpose feature, which can be coupled with off-the-shelf clustering methods [2,17,31] to conduct unsupervised clustering tasks.

Many effective graph representation learning approaches have been developed in the literature. For instance, Node2Vec [9] learns low-dimensional embeddings of nodes that preserve the neighborhood structure to the highest degree. DGI [29] advances the learning of node representations by maximizing the mutual information between local patch representations and the overall graph representations. In a similar way, InfoGraph [25] employs mutual information maximization and graph isomorphism networks [34] to learn general purpose representations for graph-level classification tasks. Drawing from the paradigm of self-supervised learning, GraphCL [35] first enhances graph data to form positive and negative sample pairs, and then adopts a contrastive learning strategy to learn good graph representations. Similarly, MVGRL [10] enhances contrastive learning by first generating an augmented structural view via graph diffusion, and then employing shared multi-layer perceptrons (MLP) to learn compact node and graph representations.

The graph-level representations learned by the above methods are good general purpose features for downstream tasks. However, since such representations are often not specifically customized for graph-level clustering, straight-forwardly coupling them with an off-the-shelf clustering method often cannot achieve optimal clustering performance. Moreover, the clustering results are not interpretable because the representations are learned through a black-box deep graph neural network.

3 The Task of Interpretable Graph-Level Clustering

In this section, we first introduce some necessary notations and then define the task of interpretable deep graph-level clustering.

Denote by $\mathcal{G} = \{G_1, G_2, \ldots, G_N\}$ a collection of N graphs, where each graph $G_i = (V_i, E_i)$ is associated with a set of node features $X_i = \{x_v^{(i)}\}_{v \in V_i}$. Here, V_i and E_i are the set of nodes and set of edges, respectively, of the graph G_i. Let $\chi = \{X_1, X_2, \ldots, X_N\}$ represent the ensemble of the node features across all graphs. We define the task of interpretable graph-level clustering as follows.

Definition 1. *Given a set of graphs $\mathcal{G} = \{G_1, G_2, \ldots, G_N\}$, the **goal of interpretable graph-level clustering** is to partition $\mathcal{G}$ into a number of K non-overlapping **clusters** of graphs, denoted by $\mathcal{G}^{(1)}, \ldots, \mathcal{G}^{(K)}$, in an unsupervised manner, such that*

1. *$\mathcal{G} = \mathcal{G}^{(1)} \cup \mathcal{G}^{(2)} \cup \ldots \cup \mathcal{G}^{(K)}$, and $\mathcal{G}^{(i)} \cap \mathcal{G}^{(j)} = \emptyset$ for any $i \neq j$;*
2. *the graphs in the same cluster are similar and the graphs in different clusters are not; and*
3. *for each graph G_i in a cluster $\mathcal{G}^{(k)}$, $k \in \{1, \ldots, K\}$, we can interpret why G_i is allocated to $\mathcal{G}^{(k)}$ by showing its resemblance with the other graphs in $\mathcal{G}^{(k)}$.*

4 Our Solution

In this section, we first introduce an overview of our entire framework; then we introduce how to train our model to conduct deep graph-level clustering; last, we introduce how to interpret why each graph is allocated to its corresponding cluster.

4.1 Framework Overview

Figure 1 shows an overview of the proposed **interpretable deep graph clustering (IDGC)** framework. It contains four major modules, such as the graph encoder in module 1, the prototype layer in module 2, the fully connected layer in module 3, and the pseudo label generation in module 4.

The **key idea** of IDGC is as follows. **First**, we use the modules 1, 2 and 3 to build a prototype-based graph-level classification network [38], which can only be trained in a supervised manner when the ground truth class label of each graph is available. **Second**, since we do not have ground truth class labels in the proposed unsupervised learning setting, we use the module 4 to generate pseudo labels according to the K clusters obtained by K-means on the graph embeddings produced by module 1. In this way, each cluster is assigned one class label, which means all the graphs in the same cluster belong to the same class. **Third**, by using the pseudo labels of clusters as the ground truth class labels for training, the prototype-based graph-level classification network is trained to predict the cluster label for an input graph, thus achieving the goal of allocating each graph $G_i \in \mathcal{G}$ to its corresponding cluster $\mathcal{G}^{(k)}$. **Last**, the prototype layer

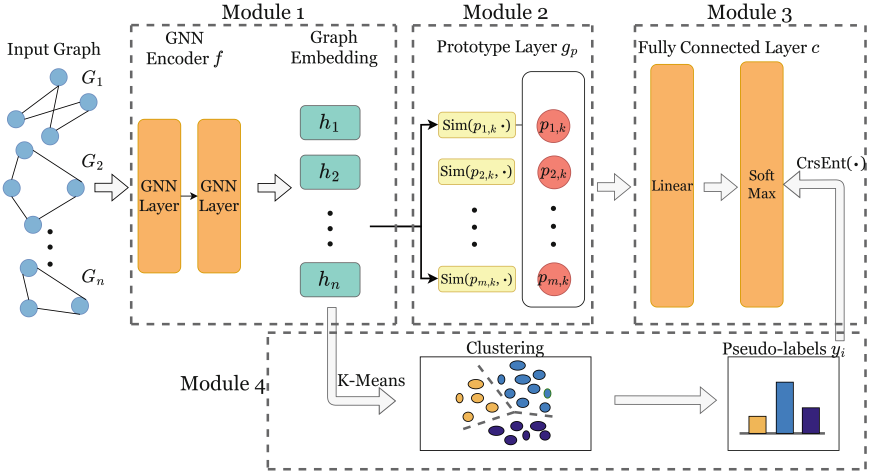


Fig. 1. An overview of the proposed IDGC framework. Due to the limit of space, we only draw the prototype embeddings for the k-th cluster in module 2, where $p_{j,k}$ is the j-th prototype embedding of the k-th cluster, $j \in \{1, \ldots, m\}$, $k \in \{1, \ldots, K\}$. Module 2 is used during training and for post-training explaining.

in module 2 can be further utilized to interpret why G_i is allocated to $\mathcal{G}^{(k)}$ by showing its resemblance with the other graphs in $\mathcal{G}^{(k)}$.

Next, we introduce each module in detail as follows.

Module 1 (Graph Encoder). Given a collection of graphs $\mathcal{G} = \{G_i\}_{i=1}^{N}$, a GNN encoder, denoted as f, maps each graph G_i to a corresponding graph embedding h_i. The encoder leverages a backbone GNN architecture, which can implemented by GCN [14], GIN [34], or GAT [28]. Without loss of generality, we implement the graph encoder by GIN, and use the output of the final layer of GIN as the graph embedding h_i. For an input graph G_i, we can write $h_i = f(G_i)$.

Module 2 (Prototype Layer). In the prototype layer, denoted by g_p, a fixed number of m prototypes are associated with each cluster (i.e., class[1]). Since we are using a total number of K clusters, there are in total $m * K$ prototypes. According to [38], when the prototypes are well trained, each cluster can be represented by a set of learned prototypes; and the prototypes should capture the most relevant graph patterns for identifying graphs of each cluster. Denote by $p_{j,k}$ the trainable **prototype embedding** to represent the j-th prototype for the k-th cluster, the prototype layer calculates the similarity score between the embedding $p_{j,k}$ of each prototype in each cluster and the embedding h_i of an input graph G_i by

---

[1] A cluster corresponds to a class because we are using cluster labels as pseudo class labels to train the prototype-based graph-level classification network consisting of the modules 1, 2 and 3.

$$\mathtt{sim}(p_{j,k}, h_i) = \log\left(\frac{\|p_{j,k} - h_i\|_2^2 + 1}{\|p_{j,k} - h_i\|_2^2 + \varepsilon}\right), \tag{1}$$

where $\|\cdot\|_2^2$ is the square of L2-norm and $\varepsilon \in [0, 1]$ is a small hyper-parameter. Obviously, $\mathtt{sim}(p_{j,k}, h_i)$ is a non-negative monotonic decreasing function with respect to $\|p_{j,k} - h_i\|_2^2$. It reaches its maximum value when $\|p_{j,k} - h_i\|_2^2 = 0$ and it approaches 0 when $\|p_{j,k} - h_i\|_2^2$ approaches $+\infty$.

Module 3 (Fully Connected Layer). The fully connected layer, denoted by c, implements a linear classifier that takes the similarity scores computed by the prototype layer as input and predicts the final cluster label as the output.

Module 4 (Pseudo Label Generation). The pseudo label generation module takes the graph embeddings generated by the graph encoder as input and uses K-means to cluster the graph embeddings into K clusters. The cluster labels graphs are used as the pseudo class labels to train the prototype-based graph-level classification network composed by Modules 1, 2 and 3.

4.2 Training the IDGC Model

In this subsection, we introduce how to train the proposed IDGC model to perform deep graph-level clustering. Inspired by [38], the **key idea** of IDGC training is to iteratively align the predicted class labels produced by module 3 with the pseudo labels generated by module 4; meanwhile, we also aim to preserve the interpretability of the prototype embeddings by imposing two clustering constraints to form meaningful clusters in the embedding space.

Following the above key idea, we design the loss function of IDGC as

$$\mathcal{L}(\theta, \mathcal{P}) = \frac{1}{N}\sum_{i=1}^{n} \mathtt{CrsEnt}(c \circ g_p \circ h_i, y_i) + \lambda_1 \mathtt{CC} + \lambda_2 \mathtt{SC}, \tag{2}$$

where λ_1 and λ_2 are hyper-parameters, $\circ$ means composition of functions, $h_i = f(G_i)$ is the embedding of graph $G_i \in \mathcal{G}$, g_p is the function of the module 2, c is the function of the module 3, $\mathcal{P} = \big\{\{p_{j,k}\} \mid j \in \{1, \ldots, m\}, k \in \{1, \ldots, K\}\big\}$ is the set of trainable prototype embeddings, and θ is the rest trainable model parameters except $\mathcal{P}$. The second term named clustering constraint (CC) is defined as

$$\mathtt{CC} = \frac{1}{n}\sum_{i=1}^{n} \min_{j \in \{1,\ldots,m\}, k = y_i} ||h_i - p_{j,k}||^2, \tag{3}$$

and the last term named separating constraint (SC) is defined as

$$\mathtt{SC} = -\frac{1}{n}\sum_{i=1}^{n} \min_{j \in \{1,\ldots,m\}, k \neq y_i} ||h_i - p_{j,k}||^2. \tag{4}$$

The **cross entropy term**, denoted by $\mathtt{CrsEnt}(c \circ g_p \circ h_i, y_i)$, computes the cross entropy between the pseudo label y_i of each input graph G_i and the predicted label $\hat{y}_i = c \circ g_p \circ h_i$, where $h_i = f(G_i)$ is the embedding of G_i. Minimizing

Algorithm 1. Training IDGC

```
 1: Input: a set of graphs $\mathcal{G} = \{G_1, G_2, \ldots, G_N\}$, and the periods $T_c$ and $T_p$ used for
    generating pseudo labels and updating prototype embeddings.
 2: Output: trained parameters $\theta$ and $\mathcal{P}$.
 3: Initialization: randomly initialize $\theta$ and $\mathcal{P}$.
 4: for each epoch $t \in \{1, 2, \ldots, T\}$ do
 5:     Compute graph embeddings: $\{h_1, h_2, \ldots, h_N\} \leftarrow f(\mathcal{G})$
 6:     if $(t \mod T_c) = 0$ then
 7:         Generate pseudo labels: $\{y_1, y_2, \ldots, y_N\} \leftarrow K\text{-means}(\{h_1, h_2, \ldots, h_N\})$
 8:     end if
 9:     if $(t \mod T_p) = 0$ then
10:         for each $p_{j,k}$ in $\mathcal{P}$ do
11:             Update prototype embedding by: $p_{j,k} \leftarrow f\big(\texttt{MCTS}(\mathcal{G}^{(k)}, p_{j,k})\big)$
12:         end for
13:     end if
14:     for each mini-batch in epoch $t$ do
15:         Update $\theta$ by: $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}(\theta, \mathcal{P})$, $\eta$ is the learning rate.
16:     end for
17: end for
18: return $\theta$ and $\mathcal{P}$.
```

the cross entropy term aligns the predicted label with the pseudo label, which trains the IDGC model to predict the corresponding cluster label for each input graph G_i.

The **CC term** requires the embedding h_i of each graph G_i to be close to the prototype embedding of one prototype in its corresponding cluster. Minimizing the CC term improves the tightness of each cluster around a prototype embedding, which helps the prototype embedding to capture the common similarity of the graphs allocated to the corresponding prototype. This improves the quality interpretation when we interpret why a graph is allocated to one cluster.

The **SC term** requires the embedding h_i of each graph G_i to be different to the prototype embedding of the other prototypes that is not associated with the cluster of G_i. This avoids possible overlaps between clusters in the embedding space and forces different clusters to effectively capture different categories of graphs. Minimizing the SC term also improves the quality of interpretation because less overlap between clusters allows each prototype to capture unique similarities between graphs.

Algorithm 1 summarizes the algorithm to train IDGC by minimizing the loss $\mathcal{L}(\theta, \mathcal{P})$.Line 5 computes the graph embeddings of all the graphs in $\mathcal{G}$. Line 7 generates the pseudo labels for every T_c epochs. Lines 9–13 update the prototype embeddings for every T_p epochs, such that each prototype embedding corresponds to a real subgraph in $\mathcal{G}$. Lines 14 -16 update the model parameter θ by standard stochastic gradient descent [1].

The $\texttt{MCTS}(\mathcal{G}^{(k)}, p_{j,k})$ in line 11 uses Monte Carlo Tree Search (MCTS) [23] to find a subgraph $\hat{G}_i$ contained in one of the graphs in the cluster $\mathcal{G}^{(k)}$, such that the embedding of $\hat{G}_i$ is the closest to $p_{j,k}$. By setting the reward function

Algorithm 2. Interpreting Cluster Allocation

```
Input: a graph $G_i$ and its allocated cluster $\mathcal{G}^{(k)}$.
Output: a subgraph $\hat{G}_i$ to interpret the cluster allocation.
Initialization: initialize the candidate set of interpretation by $\mathcal{I} = \emptyset$.
for each $j \in \{1, \ldots, m\}$ do
    Find the most similar subgraph: $\hat{G}_i \leftarrow \mathtt{MCTS}(G_i, p_{j,k})$
    Compute the similarity score: $\text{score}_j \leftarrow \mathtt{sim}\big(p_{j,k}, f(\hat{G}_i)\big)$.
    Buffer the candidate: $\mathcal{I} \leftarrow \mathcal{I} \cup \big(\hat{G}_i, \text{score}_j\big)$.
end for
return the subgraph $\hat{G}_i$ with the largest $\text{score}_j$ in $\mathcal{I}$.
```

of MCTS to the similarity score in Equation (1), the MCTS is better poised to identify the subgraph that closely approximates the prototype embedding. Specifically, we define the process of using MCTS to search the subgraphs of all the graphs in the cluster $\mathcal{G}^{(k)}$ as

$$\mathtt{MCTS}(\mathcal{G}^{(k)}, p_{j,k}) = \underset{\hat{G}_i \subseteq G_i \in \mathcal{G}^{(k)}}{\operatorname{argmax}} \ \mathtt{sim}\big(p_{j,k}, f(\hat{G}_i)\big), \tag{5}$$

where $\hat{G}_i$ is the subgraph to return, $f(\hat{G}_i)$ is the embedding of $\hat{G}_i$, and $\mathcal{G}^{(k)}$ is an intermediate cluster of graphs whose pseudo labels generated by module 4 in the corresponding epoch t are equal to k.

Updating the prototype embeddings in the above way achieves two benefits: 1) it updates the m centers (i.e., prototypes) of each cluster, which helps forming better clustering structures. Here, since each cluster is characterized by m prototypes, the prototypes are serving as the centers to describe the cluster. 2) it improves the interpretability of each prototype embedding, because every prototype embedding is equal to the embedding of a real subgraph, thus the prototype embedding does not fall outside the distribution of the embeddings of real graphs.

4.3 Cluster Allocation and Interpretation

In this section, we introduce how to allocate an input graph $G_i \in \mathcal{G}$ to a cluster $\mathcal{G}^{(k)}$, $k \in \{1, \ldots, K\}$, and interpret why G_i is allocated to $\mathcal{G}^{(k)}$ by showing its resemblance with the other graphs in $\mathcal{G}^{(k)}$.

Cluster Allocation. After training the IDGC model by Algorithm 1, we use the modules 1, 2 and 3 of the trained IDGC model to predict the cluster labels of each input graph $G_i \in \mathcal{G}$. This allocates each graph G_i to its corresponding cluster $\mathcal{G}^{(k)}$, $k \in \{1, \ldots, K\}$. Since we use the pseudo cluster labels to supervise the training of the modules 1, 2 and 3 of the IDGC model, and we also use the CC term and SC term to form tight and non-overlapping clustering structures in the embedding space, the quality of the final clusters $\mathcal{G}^{(1)}, \ldots, \mathcal{G}^{(K)}$ are often quite good. Moreover, since the modules 1, 2 and 3 of the IDGC model form a predictive model, they can be straight-forwardly used to allocate a new graph G_{new} that is not previously contained in $\mathcal{G}$ to one of the K clusters.

Interpretation. To interpret why an input graph G_i is allocated to a cluster $\mathcal{G}^{(k)}$, our key idea is to identify the subgraph $\hat{G}_i$ of G_i, whose embedding $f(\hat{G}_i)$ has the highest similarity score $\texttt{sim}\big(p_{j,k}, f(\hat{G}_i)\big)$ to a prototype embeddings $p_{j,k}$ of the cluster $\mathcal{G}^{(k)}$. The highest similarity score means $\hat{G}_i$ contributes the most to the similarity between G_i and $p_{j,k}$. Since the final cluster allocation is predicted by module 3 based on the similarity scores between G_i and each of the prototypes, the highest similarity score also implies $\hat{G}_i$ contributes the most to allocating G_i to the cluster $\mathcal{G}^{(k)}$. As a result, we use $\hat{G}_i$ to interpret why G_i is allocated to the cluster $\mathcal{G}^{(k)}$. We summarize the above interpretation process in Algorithm 2, where the MCTS function in line 5 is defined as

$$\texttt{MCTS}(G_i, p_{j,k}) = \underset{\hat{G}_i \subseteq G_i}{\operatorname{argmax}}\, \texttt{sim}\big(p_{j,k}, f(\hat{G}_i)\big), \tag{6}$$

which finds the subgraph $\hat{G}_i$ in G_i with the largest $\texttt{sim}\big(p_{j,k}, f(\hat{G}_i)\big)$.

5 Experiment

In this section, we compare the performance of the proposed method with six state-of-the-art baseline methods, such as shortest path kernel (SP) [3], Weisfeiler-Lehman kernel (WL) [21], random walk kernel (RW) [30][2], graph contrastive learning (GraphCL) [36][3], InfoGraph [25][4], and Multi-View Graph Representation Learning (MVGRL) [11][5].

For the graph kernel methods, such as SP, WL and RW, we employ spectral clustering (SC) [31] [6] on the similarity matrix computed by each kernel to produce K clusters of graphs. For the unsupervised graph representation methods, such as GraphCL and MVGRL, we produce K clusters of graphs by applying each of K-means (KM)[6] and SC on the graph embeddings learned by the methods. We use the publicly available source code of the baseline methods and we use their default parameter settings in our experiments. Our code is available at the following link[7]. For our method, we set $T_c = 1$, $T_p = 10$, $\lambda_1 = 0.05$, $\lambda_2 = 0.05$, and $\eta = 0.05$ in Algorithm 1. For all methods, the number of clusters K is set to the number of classes in each dataset.

Datasets. Table 1 summarizes the statistics of the six benchmark datasets. A brief introduction to each dataset is as follows. MUTAG [5] is a dataset of chemical compounds, each represented by a graph. Every compound is categorized

---

[2] Source code for SP, WL and RW: https://github.com/jajupmochi/py-graph.

[3] Source code: https://github.com/Shen-Lab/GraphCL.

[4] Source code: https://github.com/sunfanyunn/InfoGraph.

[5] Source code: https://github.com/kavehhassani/mvgrl.

[6] Source code for K-means and spectral clustering: https://scikit-learn.org.

[7] Source code for IDGC: https://github.com/cjbbb/IDGC-implementation.

Table 1. Statistics of Datasets.

Dataset	#Graphs	Avg. #nodes	Avg. #edges	#Classes (i.e., K)
MUTAG	188	17.93	19.79	2
BBBP	2039	24.06	25.95	2
BZR	405	35.75	38.36	2
ENZYMES	600	32.63	62.14	6
COX2	467	41.22	43.45	2
BA2Motifs	1000	25.00	51.00	2

into one of two distinct classes depending on their mutagenic impact on bacteria. BBBP [33] is derived from the research on blood-brain barrier permeability; it is also a dataset of chemical compounds represented by graphs. Each compound is associated with a binary label indicating a compound's ability to penetrate blood-brain barrier. BZR [19] consists of molecular structures of ligands for the benzodiazepine receptor, where each molecular structure is represented by a graph and classified into two categories (i.e., active and inactive) based on their binding affinities. ENZYMES [13] consists of 600 protein structures from the BRENDA enzyme database [20]. Each protein structure is represented by a graph and categorized into one of six different classes. COX2 [19] consists of chemical formulas of cyclooxygenase-2 inhibitors, where each chemical formula is represented by a graph and categorized into two categories (i.e., active and inactive) based on their efficacy. BA2Motifs [16] contains 1,000 synthetic graphs, where each graph is categorized based on whether it contains a "house" structure or a five-node cycle. We use this dataset for the experiments of interpreting cluster allocations.

Evaluation Metrics. We employ normalized mutual information (NMI) [24] and adjusted rand index (ARI) [12] to evaluate the clustering performance of each baseline method. Both NMI and ARI are widely used in the literature [26,27] to measure the similarity between the ground truth clustering and the clustering computed by a clustering method. A larger value of NMI and ARI means a better clustering performance.

5.1 Clustering Performance

In this subsection, we evaluate the clustering performance of all compared methods on six benchmark datasets. The NMI and ARI of each method are reported in Table 2, where the best results are highlighted in bold and the second best results are underlined.

We can see that the performance of the graph kernel methods is generally inferior to the other methods. Although WL+SC and RW+SC achieve the second best ARI and the second best NMI on COX2 and BZR, respectively, they cannot consistently achieve good clustering performance on all the datasets due

Table 2. The NMI and ARI of all compared methods.

Method	MUTAG		BBBP		BZR		ENZYMES		COX2		BA2Motifs	
	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI
GraphCL+SC	14.72	22.10	0.48	-0.67	2.29	1.55	5.63	0.17	2.26	0.06	73.97	82.79
GraphCL+KMeans	30.15	**36.39**	1.58	7.29	0.91	-3.24	7.45	4.25	1.86	-0.22	81.03	88.35
InfoGraph+SC	14.77	20.36	0.78	-2.56	2.09	3.62	4.95	0.64	1.01	0.00	62.67	73.25
InfoGraph+KMeans	29.35	35.12	0.75	-2.75	0.84	-2.79	6.17	2.95	0.61	-0.13	61.93	72.56
MVGRL+SC	13.67	18.49	1.42	1.78	2.58	4.40	7.40	3.38	0.33	-0.21	5.52	7.31
MVGRL+KMeans	26.94	33.81	**5.60**	11.58	0.53	-2.10	5.95	2.12	0.72	-0.24	10.77	14.05
SP+SC	21.06	31.80	1.63	0.73	2.77	-1.01	2.57	1.69	0.13	0.01	18.45	6.28
WL+SC	14.14	18.54	4.42	2.26	1.73	2.47	1.52	2.33	0.00	0.46	4.14	5.57
RW+SC	14.14	18.54	0.97	2.91	4.52	-3.49	4.64	0.61	0.04	-0.92	46.66	55.61
IDGC	**43.49**	36.28	4.88	**14.81**	**5.92**	**8.41**	**8.16**	**4.80**	**6.12**	**8.08**	**87.63**	**91.99**

to the limited descriptive power of hand-designed features [7]. The unsupervised graph representation methods generally achieve better performance than the graph kernel methods due to their strength of learning highly descriptive graph embeddings in a data-driven manner. However, the clustering performance of the unsupervised graph representation methods is still inferior to IDGC, because the graph embeddings are trained for general purpose downstream tasks, but they are not specifically customized to improve graph-level clustering performance. In comparison, IDGC achieves the best graph-level clustering performance on BZR, ENZYMES, COX2 and BA2Motifs. Its performance is also among the top on MUTAG in terms of NMI, and on BBBP in terms of ARI. These results demonstrate the outstanding clustering performance of IDGC, which is largely achieved by smoothly incorporating data driven feature learning and graph-level clustering in an end-to-end manner.

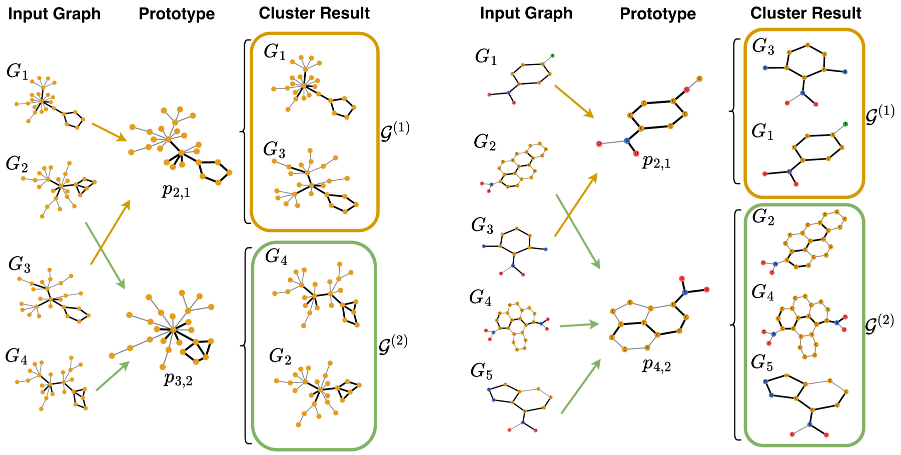
Besides the outstanding clustering performance, IDGC also produces surprisingly good interpretations to illustrate why an input graph is allocated to a cluster. We demonstrate this by conducting a comprehensive case study in Sect. 5.2.

5.2 Case Study: Interpreting Cluster Allocation

In this subsection, we conduct a case study on BA2Motifs and MUTAG to show IDGC's interpretations on cluster allocation. The interpretation results are shown in Fig. 2.

Take BA2Motifs in Fig. 2(a) as an example, there are four input graphs $G_1, \ldots, G_4$, where G_1 and G_3 are allocated to the cluster $\mathcal{G}^{(1)}$ due to their high similarity to the prototype $p_{2,1}$; G_2 and G_4 are allocated to the cluster $\mathcal{G}^{(2)}$ due to their high similarity to the prototype $p_{3,2}$. The graphs of the two prototypes are called **prototype graphs**, which are obtained by Eq. (5) and shown in the middle column of Fig. 2(a).

If we zoom in to take a closer look at each graph in Fig. 2(a), we will notice a subgraph shown by bold edges. This subgraph is obtained by Eq. (6) and it



(a) BA2Motifs (b) MUTAG

Fig. 2. A case study to show the interpretation on cluster allocations. Better viewed in color and zoomed in.

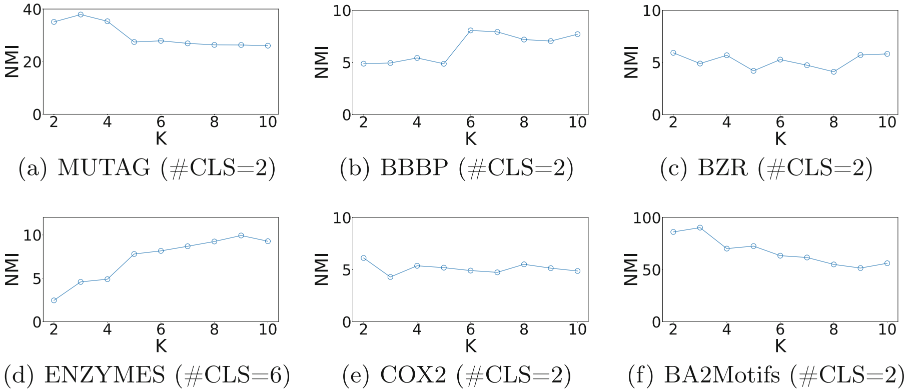
highlights the reason why an input graph has a high similarity score with its corresponding prototype graph, which further interprets why the input graph is allocated to its corresponding cluster. Specifically, in the input graphs G_1 and G_3 and the prototype graph of $p_{2,1}$, we can see that the subgraphs in bold edges successfully identify the five node circle. This accurately interprets why G_1 and G_3 are allocated to the cluster $\mathcal{G}^{(1)}$, because $\mathcal{G}^{(1)}$ corresponds to the ground truth class of graphs containing five node circles. The cluster $\mathcal{G}^{(2)}$ corresponds to the ground truth class of graphs containing a "house" structure. Thus, we can see the subgraphs in bold successfully identify the "house" structure in the input graphs G_2, G_4 and the prototype graph of $p_{3,2}$. Similar results can also be viewed in Fig. 2(b) on the MUTAG dataset.

In summary, as demonstrated by the case study in Fig. 2, the proposed IDGC method can produce good interpretations to illustrate why each input graph is allocated to its corresponding cluster.

5.3 Parameter Analysis

In this subsection, we analyze the effect of K on the NMI of IDGC. We do not evaluate the effect of K on the ARI of IDGC, because ARI is meaningless when K is not equal to the ground truth number of classes in each dataset.

Figure 3 shows the effect of K on the NMI of IDGC on each benchmark dataset. Like many well-known clustering tasks [6,17,31], graph-level clustering is also an NP-hard problem and the final clustering result is often affected by many factors, such as the distribution of data, the value of K, and the initialization of the clustering algorithm. Therefore, we could not observe a consistent trend of NMI on all the datasets.



(a) MUTAG (#CLS=2) (b) BBBP (#CLS=2) (c) BZR (#CLS=2)

(d) ENZYMES (#CLS=6) (e) COX2 (#CLS=2) (f) BA2Motifs (#CLS=2)

Fig. 3. Impact of cluster number k on NMI for IDGC on different datasets. #CLS is the ground truth number of classes in the corresponding dataset.

However, by taking a closer look at Fig. 3, we can still summarize some interesting patterns from the curves of NMI. For example, in Fig. 3(a), 3(c), 3(e) and 3(f) setting K to the ground truth number of classes yields near optimal NMI on the corresponding dataset. We can also observe in Fig. 3(b) and 3(d) that setting K to the ground truth number of classes yields sub-optimal NMI and increasing the value of K will further increase NMI. Such an improvement of NMI is likely due to the over-splitting effect caused by using a larger K than the ground truth number of classes. This effect has also been observed by some other deep clustering works [4]. However, the splitting-effect does not apply to all the datasets. We can see in Fig. 3(c) and 3(e) that setting a larger K than the ground truth number of classes does not increase NMI. Besides, in Fig. 3(a) and 3(f), increasing K from 2 to 3 slightly increases NMI, but further increasing the value of K drops NMI.

In summary, the optimal value of K is different on different datasets. Like many classic clustering methods [2,6,17,31], we can use cross validation to choose the optimal value of K in practice. However, for the fairness of comparison in the previous experiments, we set the value of K for all the compared methods to the same ground truth number of classes.

6 Conclusion

In this paper, we propose the novel unsupervised learning task of interpretable graph-level clustering, which aims to find good clusters of graphs and also gain useful insights into the clustering result by interpreting why each graph is allocated to its corresponding cluster. We successfully tackle this task by an effective method named interpretable deep graph-level clustering (IDGC). IDGC achieves superior graph-level clustering performance by incorporating the training of graph embeddings and the clustering of graphs in an end-to-end deep learning framework; it also achieves surprisingly good interpretation performance by introducing a prototype layer to learn prototypical graph structures for each cluster of graphs. Extensive experiments on six benchmark datasets demonstrate the outstanding performance of the proposed method.

Acknowledgements. This work was supported in part by the NSERC Discovery Grant program (RGPIN-2022-04977) and in part by the start-up funding of McMaster University. All opinions, findings, conclusions and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

References

1. Amari, S.I.: Backpropagation and stochastic gradient descent method. Neurocomputing **5**, 185–196 (1993)
2. Arthur, D., Vassilvitskii, S., et al.: k-means++: the advantages of careful seeding. In: SODA, vol. 7, pp. 1027–1035 (2007)
3. Borgwardt, K.M., Kriegel, H.P.: Shortest-path kernels on graphs. In: Fifth IEEE International Conference on Data Mining, pp. 74–81 (2005)
4. Caron, M., Bojanowski, P., Joulin, A., Douze, M.: Deep clustering for unsupervised learning of visual features. In: Proceedings of the European Conference on Computer Vision, pp. 132–149 (2018)
5. Debnath, A.K., Lopez de Compadre, R.L., Debnath, G., Shusterman, A.J., Hansch, C.: Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds, correlation with molecular orbital energies and hydrophobicity. J. Med. Chem. **34**, 786–797 (1991)
6. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: Density-based spatial clustering of applications with noise. In: The International Conference on Knowledge Discovery and Data Mining, vol. 240 (1996)
7. Feng, A., You, C., Wang, S., Tassiulas, L.: Kergnns: interpretable graph neural networks with graph kernels. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, pp. 6614–6622 (2022)
8. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: International Conference on Machine Learning, pp. 1263–1272 (2017)
9. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864 (2016)
10. Hassani, K., Khasahmadi, A.H.: Contrastive multi-view representation learning on graphs. In: International Conference on Machine Learning, pp. 4116–4126 (2020)
11. Hassani, K., Khasahmadi, A.H.: Contrastive multi-view representation learning on graphs. In: International Conference on Machine Learning, pp. 4116–4126 (2020)
12. Hubert, L., Arabie, P.: Comparing partitions. J. Classif. **2**, 193–218 (1985)
13. Kersting, K., Kriege, N.M., Morris, C., Mutzel, P., Neumann, M.: Benchmark data sets for graph kernels (2016). https://ls11-www.cs.tu-dortmund.de/staff/morris/graphkerneldatasets
14. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)

15. Kriege, N., Mutzel, P.: Subgraph matching kernels for attributed graphs. arXiv preprint arXiv:1206.6483 (2012)
16. Luo, D., et al.: Parameterized explainer for graph neural network. Adv. Neural. Inf. Process. Syst. **33**, 19620–19631 (2020)
17. MacQueen, J., et al.: 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)
18. Nikolentzos, G., Meladianos, P., Vazirgiannis, M.: Matching node embeddings for graph similarity. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31 (2017)
19. Rossi, R.A., Ahmed, N.K.: The network data repository with interactive graph analytics and visualization. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 4292–4293 (2015)
20. Schomburg, I., Chang, A., Schomburg, D.: Brenda, enzyme data and metabolic information. Nucleic Acids Res. **30**, 47–49 (2002)
21. Shervashidze, N., Schweitzer, P., Van Leeuwen, E.J., Mehlhorn, K., Borgwardt, K.M.: Weisfeiler-Lehman graph kernels. J. Mach. Learn. Res. **12** (2011)
22. Shervashidze, N., Vishwanathan, S., Petri, T., Mehlhorn, K., Borgwardt, K.: Efficient graphlet kernels for large graph comparison. In: International Conference on Artificial Intelligence and Statistics, pp. 488–495 (2009)
23. Silver, D., et al.: Mastering the game of go without human knowledge. Nature **550**, 354–359 (2017)
24. Strehl, A., Ghosh, J.: Cluster ensembles–a knowledge reuse framework for combining multiple partitions. J. Mach. Learn. Res. **3**, 583–617 (2002)
25. Sun, F.Y., Hoffmann, J., Verma, V., Tang, J.: Infograph: unsupervised and semi-supervised graph-level representation learning via mutual information maximization. arXiv preprint arXiv:1908.01000 (2019)
26. Tian, F., Gao, B., Cui, Q., Chen, E., Liu, T.Y.: Learning deep representations for graph clustering. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 28 (2014)
27. Tsitsulin, A., Palowitch, J., Perozzi, B., Müller, E.: Graph clustering with graph neural networks. J. Mach. Learn. Res. **24**, 1–21 (2023)
28. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
29. Veličković, P., Fedus, W., Hamilton, W.L., Liò, P., Bengio, Y., Hjelm, R.D.: Deep graph infomax. arXiv preprint arXiv:1809.10341 (2018)
30. Vishwanathan, S.V.N., Schraudolph, N.N., Kondor, R., Borgwardt, K.M.: Graph kernels. J. Mach. Learn. Res. **11**, 1201–1242 (2010)
31. Von Luxburg, U.: A tutorial on spectral clustering. Stat. Comput. **17**, 395–416 (2007)
32. Weisfeiler, B., Leman, A.: The reduction of a graph to canonical form and the algebra which appears therein. Nauchno-Technicheskaya Informatsiya **2**, 12–16 (1968)
33. Wu, Z., et al.: Moleculenet: a benchmark for molecular machine learning. Chem. Sci. **9**, 513–530 (2018)
34. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? arXiv preprint arXiv:1810.00826 (2018)
35. You, Y., Chen, T., Shen, Y., Wang, Z.: Graph contrastive learning automated. In: International Conference on Machine Learning, pp. 12121–12132 (2021)
36. You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., Shen, Y.: Graph contrastive learning with augmentations. Adv. Neural. Inf. Process. Syst. **33**, 5812–5823 (2020)

37. Zaslavsky, L., Ciufo, S., Fedorov, B., Tatusova, T.: Clustering analysis of proteins from microbial genomes at multiple levels of resolution. BMC Bioinformatics **17**, 545–552 (2016)
38. Zhang, Z., Liu, Q., Wang, H., Lu, C., Lee, C.: Protgnn: towards self-explaining graph neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, pp. 9127–9135 (2022)

A Saliency-Aware NR-IQA Method by Fusing Distortion Class Information

Menglong Chen[1], Jianming Wang[2], Zhitao Xiao[3], and Yukuan Sun[4(✉)]

[1] School of Computer Science and Technology, Tiangong University, Tianjin, China
[2] Tianjin Key Laboratory of Autonomous Intelligence Technology and Systems,
Tiangong University, Tianjin, China
`wangjianming@tiangong.edu.cn`
[3] Tianjin Key Laboratory of Optoelectronic Detection Technology and Systems,
Tiangong University, Tianjin, China
`xiaozhitao@tiangong.edu.cn`
[4] Center for Engineering Intership and Training, Tiangong University, Tianjin, China
`sunyukuan@tiangong.edu.cn`

Abstract. Blind image quality assessment (BIQA) is a crucial technique for selecting high quality images, which are vital for reliable diagnosis, effective algorithms, and ensuring an optimal visual experience for users. Due to full-reference and reduced-reference IQA methods require reference images, the demand for no-reference (NR-IQA) techniques is more pronounced in practical scenarios. In recent years, NR-IQA models based on Convolutional Neural Networks have achieved significant success. However, existing methods treat image regions uniformly, contradicting the trend of the human visual system to prioritize attention to salient areas. Inspired by the Human Visual System (HVS), we propose the hypothesis that different regions contribute differently to perceived image sharpness, with significant regions exerting a greater influence on quality assessment. To validate this hypothesis, we conducted subjective experiments, revealing that applying the same degree of blur to foreground and background regions results in different quality degradations. Based on this observation, we introduce A Saliency-Aware NR-IQA Method by Fusing Distortion Class Information(SADCIQA). A hierarchical saliency aware module is designed to obtain weighted fusion of features from different regions of the image. Additionally, we employ a self-supervised method to train a quality aware module to extract image quality features. SADCIQA emphasizes salient regions while reducing the contribution of background areas, aligning quality predictions with subjective judgments. We conducted experiments on six datasets, demonstrating that our approach achieved state-of-the-art performance in both synthetic and real distortions.

Supported by the National Natural Science Foundation of China (grant number 62072335) and the Introduction of Foreign Intelligence Project of Shijiazhuang Science and Technology Bureau (Project No. 20240024).

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 130–143, 2025.
https://doi.org/10.1007/978-3-031-78128-5_9

Keywords: Blind image quality assessment · Self-supervised learning · Salient region features · Human visual system

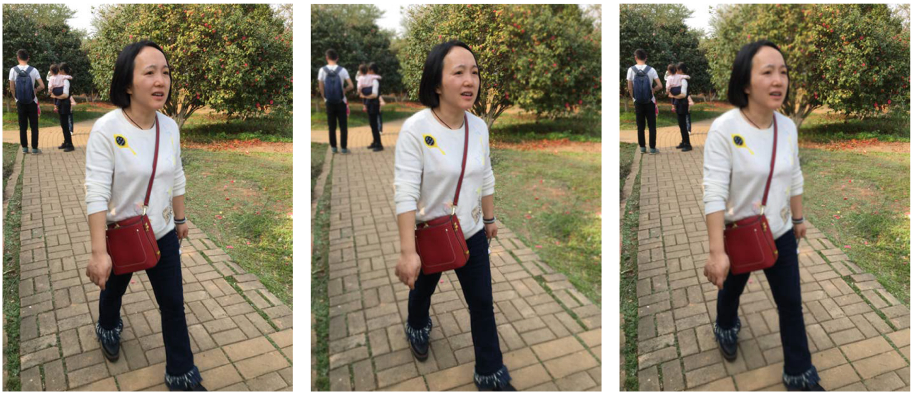
1 Introduction

As the demand for high-quality visual content rises, image quality assessment (IQA) becomes crucial in various applications. IQA aims to measure image perceptual quality aligned with human judgments. IQA methods are usually divided into subjective(relying on human evaluations) and objective (using computational models). Subjective methods are generally not in the option list of the computer vision community because of their requirements for high labor costs. Objective methods include full-reference IQA(FR-IQA), reduced-reference IQA(RR-IQA), and no-reference IQA(NR-IQA). In real-world scenarios, NR-IQA methods prove more desirable due to their adaptability in the absence of reference images.

With the advancement of Convolutional Neural Networks (CNNs) in recent years, numerous CNN-based data-driven IQA models (such as VCRNet [1], HyperNet [2], and DB-CNN [3]) have achieved notable success. However, we observed that all the existed methods deal with the whole image equally and indiscriminately, which is not consistent with that of human visual information processing. Biological vision, including that of humans, has evolved over a long period of time. To enhance the adaptability of species, human vision has to process received images selectively rather than indiscriminately. On the contrary, the visual system automatically distinguishes "critical" parts of the scene and prioritizes attention to them, such as areas containing animals like snakes or tigers. Therefore, we assume that different regions of an image contribute differently to the perceived sharpness by the human visual system. The higher the attention to a region in the image, the greater its impact on the evaluation of image quality.

To validate this hypothesis, we conducted a quick experiment as shown in Fig. 1. Figure 1(a) shows the original image, Fig. 1(b) applies Gaussian blur to the central human body region, and Fig. 1(c) applies the same degree of Gaussian blur to the background region outside the human body area. For these three images, we initially invited 20 individuals to observe image (a) along with its MOS(69.64) from the SPAQ dataset. Subsequently, they were asked to rate images (b) and (c) separately. The final MOS scores for images (b) and (c) were established as 64.4 and 58.2 respectively. Through analysis of the scoring results, the following conclusions can be inferred: (1) A decrease in image quality in either the foreground or background area results in lower subjective evaluation scores for the image. (2) When subjected to equal degrees of image blur, the impact of blurring on the foreground image region is significantly greater than that on the background region.

Based on the aforementioned observations, this paper proposes a saliency-aware nr-iqa method by fusing distortion class information(SADCIQA). The main idea is to amplify the influence of salient regions while diminishing the



(a) Original Image (b) Gaussian blur:1 (c) Gaussian blur:2

Fig. 1. Comparison of the Impact of Foreground and Background Blur on Image Quality: Image (a) was randomly selected from the SPAQ dataset. Gaussian blur was independently applied to the foreground and background areas, resulting in Images (b) and (c) respectively. This allows for a direct comparison of how foreground and background blur impact image quality perception.

influence of background regions on the assessment. Thus obtaining evaluation results that are more aligned with subjective judgments. To achieve differential treatment of salient and background regions, we modify a salient region segmentation model to make it applicable to our task. In addressing the issue of insufficient annotated data in NR-IQA, we make modifications based on the framework proposed in [4]. The primary contributions of the paper include:

1. We propose a hypothesis to impact image quality evaluation results: Different regions of an image contribute differently to the perceived sharpness by the human visual system. The higher the attention to a region in the image, the greater its impact on the evaluation of image quality.
2. Based on the above hypothesis, (SADCIQA) has been enhanced.
3. The proposed SADCIQA has conducted extensive experiments on six IQA databases, and achieved the state-of-the-art performance on the image quality prediction accuracy.

2 Related Work

2.1 Feature Extraction Methods for NR-IQA

Prior to the emergence of deep learning, the dominance of Natural Scene Statistics (NSS) theory in the field of Blind Image Quality Assessment (NR-IQA) was notable. NSS theory posits that pristine natural images adhere to specific statistical distributions, and various distortions disrupt this statistical regularity. [5–7]. Consequently, researchers proposed various hand-crafted features across different domains, including spatial [8,9], frequency [10], and gradient features [11]. Additionally, there were initial explorations into learning-based methods, such as support vector regression, and features learned by visual codebooks [12], to estimate subjective quality.

In recent years, NR-IQA (No-Reference Image Quality Assessment) models have transitioned from utilizing hand-crafted perceptual features and shallow learners to adopting Deep Learning-based approaches trained on extensive subjective databases [1–3].

However, these methods do not differentiate between different regions of the image as effectively as the human visual system. When processing an image, the human visual system tends to assign different levels of importance to various regions based on the image's content and context. For instance, in a portrait photograph, people are more likely to focus on the facial features of the subject, while details in the background may be less important and easily overlooked.

2.2 Solutions for Scarcity of Labeled Data in NR-IQA

Following the intricate model design, a limited number of studies have shifted their focus to addressing the fundamental challenge in NR-IQA: the scarcity of labeled data. In response, some researchers have explored strategies to maximize the utilization of existing supervisory signals, such as rank learning [13,14], multi-task learning [15,16], and mixed-dataset training [17,18].

PaQ-2-PiQ [16] stands out for its utilization of a specially designed dataset, where the authors not only collected subjective quality scores for entire images but also for a large number of image patches. This dataset is sufficiently large to train deep models in a supervised manner. However, despite incorporating patch-level and image-level quality information during training, the susceptibility to dataset sampling errors exists, as only a limited number of patches were extracted from each image and annotated with quality scores.

MUSIQ [19] employs a transformer-based architecture, pre-trained on the ImageNet classification dataset. The method significantly benefits from this transformer architecture and further enhances its performance by fine-tuning the transformer backbone on IQA test databases.

Re-IQA [4] aims to learn quality-aware representations in a self-supervised manner. Re-IQA achieves this by learning to group images with similar types and distortion degrees into classes using an independent dataset. We adopt the Re-IQA approach to train an image quality aware module.

3 Proposed Method

As mentioned in the conclusion of the related work section, existed methods fail to perceive the significant differences in various regions of the image. Addressing this challenge, In this paper, we introduce a novel approach named a saliency-aware NR-IQA method by fusing distortion class information(SADCIQA). SADCIQA utilizes a self-supervised approach to learn image distortion classes and leverages salient region segmentation techniques to extract features related to image saliency. The architecture of SADCIQA, illustrated in Fig. 2, consists of a Quality Features Aware module and a Hierarchical Saliency Features Aware module.

To specifically capture hierarchical saliency regions in images, we employed our Hierarchical Saliency Aware Module, implemented using U-Net architecture. Further, we referred to Re-IQA for the implementation of the Quality Features Aware method. This method employs a self-supervised approach, introducing diverse distortions to images and training a feature extractor. Further discussion on the use of the Hierarchical Saliency Aware Module and the Quality Features Aware method is provided in Sects. 3.1, 3.2.

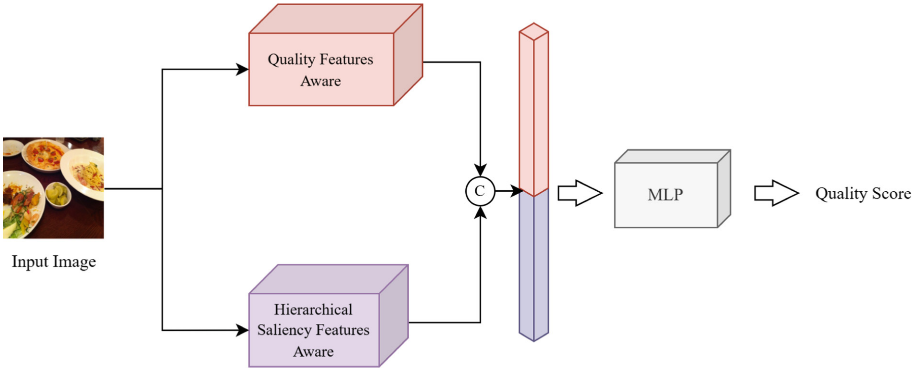


Fig. 2. Architecture of SADCIQA: For the input image, feature extraction was performed separately using the quality-aware module and the hierarchical-saliency aware module. Subsequently, the extracted features were fused and passed through an MLP head to obtain scores.

3.1 Hierarchical Saliency Aware Module

To distinguish and acquire features from different saliency regions of the image, we enhance the salient region segmentation model as our Hierarchical Saliency Feature Aware module. Through training, it focuses on capturing features from salient regions within the image.

RSU-L. Inspired by both U-Net [20] and U2-Net [21], we aim to better simulate the Human Visual System while extracting hierarchical saliency features. To achieve this, we introduce an RSU-L block, illustrated in Fig. 3, designed to capture intra-stage multi-scale features. The proposed model comprises three main components: Firstly, an input convolution layer transforms the input feature map $x(H \times W \times C_{\text{in}})$into an intermediate map F_1 with C_{out} channels, facilitating local feature extraction. a U-Net-like structure with height of L is employed to extract

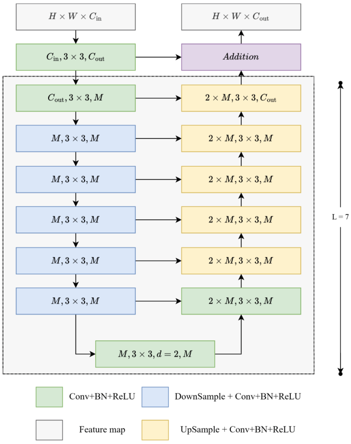

$H \times W \times C_{\text{in}}$
$H \times W \times C_{\text{out}}$
$C_{\text{in}}, 3 \times 3, C_{\text{out}}$
Addition
$C_{\text{out}}, 3 \times 3, M$
$2 \times M, 3 \times 3, C_{\text{out}}$
$M, 3 \times 3, M$
$2 \times M, 3 \times 3, M$
$M, 3 \times 3, M$
$2 \times M, 3 \times 3, M$
$M, 3 \times 3, M$
$2 \times M, 3 \times 3, M$
$M, 3 \times 3, M$
$2 \times M, 3 \times 3, M$
$M, 3 \times 3, M$
$2 \times M, 3 \times 3, M$
$M, 3 \times 3, d = 2, M$
L = 7
Conv+BN+ReLU
DownSample + Conv+BN+ReLU
Feature map
UpSample + Conv+BN+ReLU

Fig. 3. We introduce an RSU-L block to capture intra-stage multi-scale features, with the model comprising three main components

and encode multi-scale contextual information. This structure, adaptable with the parameter, enables the extraction of multi-scale features from input feature maps of various spatial resolutions. Lastly, a residual connection merges local and multi-scale features.

Architecture. Our hierarchical saliency-aware module primarily consists of encoder and decoder components, as illustrated in Fig. 4. In the encoder stages, En 1 to En 6, we utilize residual U-blocks (RSU) with varying heights, including RSU-7, RSU-6, RSU-5, RSU-4, and RSU-4F. The designation "F" in RSU-4F indicates that it is a dilated version, where pooling and upsampling operations are replaced with dilated convolutions. The selection of block height depends on the resolution of the input feature map, with taller blocks capturing more large-scale information. RSU-4F in stages En 5 and En 6 are dilated versions specifically introduced to prevent the loss of context during downsampling. Additionally, downsampling is applied between each encoder, while upsampling is employed between each decoder.

After successfully training the saliency region segmentation model, the Hierarchical Saliency Aware Module is used for feature extraction within the image quality assessment framework. As depicted in Fig. 4, we extract feature maps from different encoder components and weighted fusion them to obtain a final feature map that accentuates salient regions while attenuating background regions.

This approach capitalizes on the learned representations from the saliency region segmentation task to improve the performance of the image quality evaluation model. By integrating the encoder component, optimized for capturing salient features, into the image quality assessment pipeline, the resulting model gains a deeper understanding of crucial visual cues. This strategy highlights the

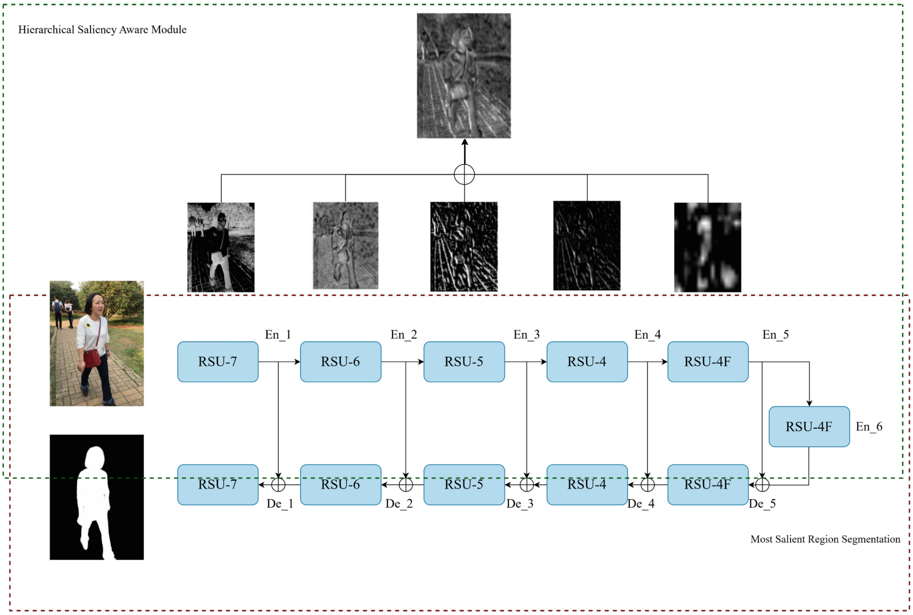


Fig. 4. We train a module for segmenting the most salient regions. Once trained, we freeze its parameters. Then, for each trained encoder, we compute a weighted sum of the generated features to obtain the overall hierarchical saliency features, which are used to calculate the final quality score.

adaptability and transferability of deep neural network architectures, enabling learned features to be repurposed across various tasks for enhanced performance and efficiency.

3.2 Quality Aware Module

In this section, we deploy an image quality perception module inspired by the Re-IQA method [4] as illustrated in Fig. 5, Firstly, we employ the Choosing Crop Locations technique to perform image cropping, ensuring comprehensive coverage of visual content. Subsequently, we apply Image Quality Distortion Augmentations to the cropped images, introducing various classes of distortions to simulate real-world image variations. Finally, we utilize an Encoder to extract image features, optimizing its parameters to minimize the distance between images with similar features while maximizing the distance between dissimilar features in the feature space.

Choosing Crop Locations: The selection of crop locations is critical for ensuring effective feature learning while minimizing redundancy. To achieve this, the algorithm selects crop locations in such a way that the overlapping area (OLA) between the two crops falls within predetermined bounds. This ensures that each

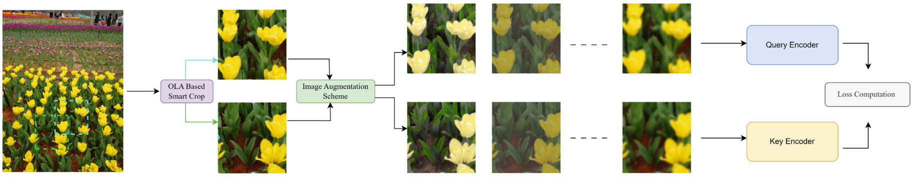


Fig. 5. By employing the OLA-based crop and an image augmentation scheme, we generate pairs of images with similar and dissimilar qualities. Subsequently, these paired images are fed into both the query encoder and key encoder for training purposes.

pair of cropped images provides sufficient variation for learning while avoiding excessive redundancy.

Image Quality Distortion Augmentations: A novel bank of image quality distortion augmentations is employed to introduce a diverse range of distortions into the image dataset. This augmentation bank comprises 25 distortion methods, each applied at five different severity levels. These distortions include, but are not limited to, noise addition, blur, compression artifacts, and color distortions. Through the application of these augmentations, the model learns to recognize and adapt to various classes and levels of image quality degradation.

Encoder Design and Training: The model utilizes two identical encoders: an Online Encoder (query encoder) and a Momentum Encoder (key encoder). Both encoders employ ResNet-50 backbones and an MLP head to generate final output embeddings from the ResNet features. After selecting pairs of images with corresponding distortions, one image from each pair is passed through the query encoder while the another is passed through the key encoder. The loss between the representations generated by the query and key encoders is calculated using the InfoNCE [22] loss function. This loss function encourages the model to bring representations of similar images closer together while pushing representations of dissimilar images apart, aiding in learning distortion classes effectively. Once the encoder pre-training is completed, the frozen ResNet-50 can be utilized as an encoder backbone for any subsequent task related to perceptual image quality.

4 Experiments

4.1 Training Datasets

In the Quality Aware Module, we integrate pristine images and authentically distorted images as training data. Our model ensures effective discrimination of features between two images with identical content but differing distortion classes. Our training data methodology parallels that of previous prominent approaches. For pristine images, we utilize 140,000 images from the KADID dataset [23]. Meanwhile, for authentically distorted images, we employ a dataset combination similar to that proposed in CONTRIQUE [24], comprising (a) 255K images from AVA [25], (b) 330K images from COCO [26], (c) 2,450 images from CERTH-Blur [27], and (d) 33K images from VOC [28]. In the Hierarchical Saliency Aware Module, similar to the approach adopted by U2-Net, our network is trained on DUTS-TR, a subset of the DUTS dataset [29]. DUTS-TR comprises a total of 10,553 images, To augment this dataset, we perform offline horizontal flipping, resulting in a doubled dataset size of 21,106 training images.

4.2 Evaluation Datasets

We evaluate our method on several widely used image quality assessment (IQA) datasets, including both synthetic and authentic distortions. The synthetic datasets include LIVE IQA [30], TID2013 [31], CSIQ [32], which provide images impaired by various classes and levels of single distortions. For authentic distortions, we utilize more recent “in the wild” datasets captured using mobile devices in real-world scenarios. These include KonIQ-10K [33], CLIVE [34], SPAQ [35]. KonIQ-10K contains 10,000 images from YFCC100M [36] covering a wide range of authentic quality issues. CLIVE has 1,162 mobile images exhibiting diverse distortion classes. SPAQ provides 11,000 mobile images along with quality scores, content labels, and EXIF data. By evaluating on both synthetic and authentic datasets, we aim to demonstrate the generalizability of our approach across distortion classes, levels, and realism. The diversity and size of these datasets enable thorough benchmarking of IQA methods.

4.3 Training Configurations

For the quality-aware module pre-training, we employed ResNet-50 as the primary feature extractor, accompanied by a 2-layer MLP head aimed at regressing contrastive features of dimensionality 128. The hidden dimension of the MLP head was configured with 2048 neurons. Training was conducted with a learning rate of 0.5, utilizing a cosine annealing scheduler, an InfoNCE temperature of 0.1, and a momentum coefficient of 0.99.

For the hierarchical saliency-aware module pre-training, We employed the Adam optimizer for network training, with hyperparameters of initial learning rate lr = 1e-4, betas = (0.95, 0.999), eps = 1e-8, weight decay = 0. Training was conducted over 40 epochs.

For fine-tuning, upon the completion of pre-training, we utilized the trained aforementioned modules as two distinct feature extractors, responsible for extracting saliency region features and distortion features respectively. Subsequently, we devised a dynamically updatable weight matrix to fuse the obtained features, followed by input to the final MLP head. We configured the learning rate as 5e-5 for the MLP head and 0.005 for the weight matrix updates, employing an L2 loss function, Adam optimizer, and a batch size of 128.

All implementations were realized in Python, leveraging the PyTorch deep learning framework. During pre-training, 2 NVIDIA A100 GPUs were utilized with mixed precision training, while subsequently, NVIDIA RTX4090 GPUs were employed for fine-tuning.

4.4 Evaluation Protocol

Pearson's Linear Correlation Coefficient (PLCC) and Spearman's Rank-Order Correlation Coefficient (SRCC) serve as the criteria for assessing accuracy and monotonicity, respectively, both ranging from 0 to 1. A higher PLCC indicates a more precise numerical fit with MOS scores, while a higher SRCC signifies a more accurate ranking between samples. Following the approach outlined in [19,37], we split the dataset into a 80% training set and a 20% testing set randomly. To avoid overlap of contents in datasets with synthetic distortions, splits were selected based on source images. We also prevented any bias towards the training set selection by repeating the train/test split operation 10 times and reporting the median performance.

4.5 Results

Our approach, by separately learning saliency region features and distortion classes features, aligns more closely with the human visual system by learning salient region-specific and distortion-specific features in a decoupled manner, our hierarchical approach is able to better capture quality degradations localized to salient image regions. As evidenced by the results in Table 1 and Fig. 6, this leads to performance improvements over existing general-purpose no-reference IQA methods like Re-IQA that do not explicitly model region saliency. We

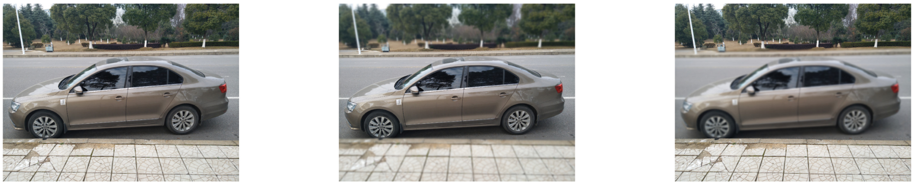

(a) Reference (b) Background Blur (c) Foreground Blur

Fig. 6. Randomly Selected Images from SPAQ Dataset: (a) Original Image, (b) Gaussian Blur Applied to Background, (c) Gaussian Blur Applied to Foreground

Table 1. Here are the scores obtained for images a, b, and c, respectively, along with the scores obtained using Re-IQA and SADCIQA for image quality assessment:

	Quality Score
Mos	71.43
Re-IQA	70.37
Ours	71.88

	Quality Score
Mos	64.4
Re-IQA	66.37
Ours	64.33

	Quality Score
Mos	54.8
Re-IQA	60.72
Ours	56.50

Table 2. Performance comparison of SADCIQA against various NR-IQA models on IQA databases with authentic and synthetic distortions. Presents the results of various methods, with VCRNet sourced from [1], and others sourced from [4].

Method	CLIVE		SPAQ		KONIQ		TID2013		CSIQ		LIVE	
	SRCC	PLCC	SRCC	PLCC	SRCC	PLCC	SRCC	PLCC	SRCC	PLCC	SRCC	PLCC
CORNIA	0.629	0.671	0.709	0.725	0.780	0.795	0.678	0.768	0.678	0.776	0.947	0.950
CONTRIQUE	0.845	0.857	0.914	0.919	0.894	0.906	0.843	0.857	0.942	0.955	0.960	0.961
HyperIQA	**0.859**	**0.882**	0.916	0.919	0.906	0.917	0.840	0.858	0.923	0.942	0.962	0.966
VCRNet	0.856	0.865	–	–	0.894	0.909	0.846	0.875	0.943	0.955	0.952	0.963
Re-IQA	0.820	0.834	0.900	0.918	**0.914**	**0.923**	0.824	0.861	0.947	0.960	0.970	0.971
BRISQUE	0.608	0.629	0809	0.817	0.665	0.681	0.604	0.694	0.746	0.829	0.939	0.935
DB-CNN	0.844	0.851	0.911	0.915	0.875	0.884	0.816	0.865	0.946	0.948	0.969	0.966
SADCIQA(Ours)	0.850	0.857	**0.916**	**0.920**	0.901	0.914	**0.856**	**0.882**	**0.957**	**0.965**	**0.969**	**0.978**

thus demonstrate the advantages of our proposed hierarchical framework for no-reference image quality assessment (Table 2).

Among these, the best-performing methods are denoted in bold, while the second and third best-performing methods are underscored. Notably, on the synthetic dataset, the third, first, and third positions were achieved, respectively. Conversely, on the authentic dataset, first place was attained uniformly. This observation underscores the efficacy of employing a method that combines distortion-specific feature extraction with salient region feature extraction for image quality assessment across diverse datasets.

5 Conclusion

In this paper, we proposed a novel approach to image quality assessment that combines self-supervised learning with salient region segmentation techniques. Our method adopts a hierarchical structure, separately learning quality features and salient region features to better mimic the human visual system. Through experiments on multiple datasets, we have demonstrated the effectiveness and robustness of our approach. Compared to existing methods, our approach shows superior performance in assessing image quality and can handle various classes and levels of quality distortions. This provides a new perspective and method for the field of image quality assessment, with significant theoretical and practical implications. Future work could explore further improvements to our method to enhance its performance and applicability.

Acknowledgements. This work was supported in part by the National Natural Science Foundation of China (No. 62072335) and the Introduction of Foreign Intelligence Project of Shijiazhuang Science and Technology Bureau (Project No. 20240024).

References

1. Pan, Z., Yuan, F., Lei, J., Fang, Y., Shao, X., Kwong, S.: Vcrnet: visual compensation restoration network for no-reference image quality assessment. IEEE Trans. Image Process. **31**, 1613–1627 (2022)
2. Su, S., et al.: Blindly assess image quality in the wild guided by a self-adaptive hyper network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3667–3676 (2020)
3. Zhang, W., Ma, K., Yan, J., Deng, D., Wang, Z.: Blind image quality assessment using a deep bilinear convolutional neural network. IEEE Trans. Circuits Syst. Video Technol. **30**(1), 36–47 (2018)
4. Saha, A., Mishra, S., Bovik, A.C.: Re-iqa: unsupervised learning for image quality assessment in the wild. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5846–5855 (2023)
5. Mittal, A., Moorthy, A.K., Bovik, A.C.: No-reference image quality assessment in the spatial domain. IEEE Trans. Image Process. **21**(12), 4695–4708 (2012)
6. Mittal, A., Soundararajan, R., Bovik, A.C.: Making a “completely blind" image quality analyzer. IEEE Signal Process. Lett. **20**(3), 209–212 (2012)
7. Moorthy, A.K., Bovik, A.C.: Blind image quality assessment: from natural scene statistics to perceptual quality. IEEE Trans. Image Process. **20**(12), 3350–3364 (2011)
8. Mittal, A., Moorthy, A.K., Bovik, A.C.: No-reference image quality assessment in the spatial domain. IEEE Trans. Image Process. **21**(12), 4695–4708 (2012)
9. Mittal, A., Soundararajan, R., Bovik, A.C.: Making a “completely blind" image quality analyzer. IEEE Signal Process. Lett. **20**(3), 209–212 (2012)
10. Moorthy, A.K., Bovik, A.C.: Blind image quality assessment: from natural scene statistics to perceptual quality. IEEE Trans. Image Process. **20**(12), 3350–3364 (2011)
11. Zhang, L., Zhang, L., Bovik, A.C.: A feature-enriched completely blind image quality evaluator. IEEE Trans. Image Process. **24**(8), 2579–2591 (2015)
12. Ye, P., Doermann, D.: No-reference image quality assessment using visual codebooks. IEEE Trans. Image Process. **21**(7), 3129–3138 (2012)
13. Li, D., Jiang, T., Jiang, M.: Norm-in-norm loss with faster convergence and better performance for image quality assessment. In: Proceedings of the 28th ACM International Conference on Multimedia, pp. 789–797 (2020)
14. Liu, X.: Learning from rankings for no-reference image quality assessment by siamese network. Computer VIsion Center, Master thesis, Universitat Autonoma de Barcelona, Barcelona (2016)
15. Fang, Y., Zhu, H., Zeng, Y., Ma, K., Wang, Z.: Perceptual quality assessment of smartphone photography. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3677–3686 (2020)
16. Ying, Z., Niu, H., Gupta, P., Mahajan, D., Ghadiyaram, D., Bovik, A.: From patches to pictures (paq-2-piq): mapping the perceptual space of picture quality. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3575–3585 (2020)
17. Sun, W., Min, X., Tu, D., Ma, S., Zhai, G.: Blind quality assessment for in-the-wild images via hierarchical feature fusion and iterative mixed database training. IEEE J. Select. Topics Signal Process. (2023)
18. Zhang, W., Ma, K., Zhai, G., Yang, X.: Uncertainty-aware blind image quality assessment in the laboratory and wild. IEEE Trans. Image Process. **30**, 3474–3486 (2021)

19. Ke, J., Wang, Q., Wang, Y., Milanfar, P., Yang, F.: Musiq: multi-scale image quality transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 5148–5157 (2021)
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. Qin, X., Zhang, Z., Huang, C., Dehghan, M., Zaiane, O.R., Jagersand, M.: U2-net: going deeper with nested u-structure for salient object detection. Pattern Recogn. **106**, 107404 (2020)
22. Oord, A.v.d., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 (2018)
23. Lin, H., Hosu, V., Saupe, D.: Kadid-10k: a large-scale artificially distorted IQA database. In: 2019 Eleventh International Conference on Quality of Multimedia Experience (QoMEX), pp. 1–3. IEEE (2019)
24. Madhusudana, P.C., Birkbeck, N., Wang, Y., Adsumilli, B., Bovik, A.C.: Image quality assessment using contrastive learning. IEEE Trans. Image Process. **31**, 4149–4161 (2022)
25. Murray, N., Marchesotti, L., Perronnin, F.: Ava: a large-scale database for aesthetic visual analysis. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2012, pp. 2408–2415. IEEE (2012)
26. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
27. Mavridaki, E., Mezaris, V.: No-reference blur assessment in natural images using Fourier transform and spatial pyramids. In: 2014 IEEE International Conference on Image Processing (ICIP), pp. 566–570. IEEE (2014)
28. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. Int. J. Comput. Vision **88**, 303–338 (2010)
29. Wang, L., et al.: Learning to detect salient objects with image-level supervision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 136–145 (2017)
30. Sheikh, H.R., Sabir, M.F., Bovik, A.C.: A statistical evaluation of recent full reference image quality assessment algorithms. IEEE Trans. Image Process. **15**(11), 3440–3451 (2006)
31. Ponomarenko, N., et al.: Color image database tid2013: peculiarities and preliminary results. In: European Workshop on Visual Information Processing (EUVIP), pp. 106–111. IEEE (2013)
32. Larson, E.C., Chandler, D.M.: Most apparent distortion: full-reference image quality assessment and the role of strategy. J. Electron. Imaging **19**(1), 011006 (2010)
33. Hosu, V., Lin, H., Sziranyi, T., Saupe, D.: Koniq-10k: an ecologically valid database for deep learning of blind image quality assessment. IEEE Trans. Image Process. **29**, 4041–4056 (2020)
34. Ghadiyaram, D., Bovik, A.C.: Massive online crowdsourced study of subjective and objective picture quality. IEEE Trans. Image Process. **25**(1), 372–387 (2015)
35. Fang, Y., Zhu, H., Zeng, Y., Ma, K., Wang, Z.: Perceptual quality assessment of smartphone photography. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3677–3686 (2020)

36. Thomee, B., et al.: Yfcc100m: the new data in multimedia research. Commun. ACM **59**(2), 64–73 (2016)
37. Golestaneh, S.A., Dadsetan, S., Kitani, K.M.: No-reference image quality assessment via transformers, relative ranking, and self-consistency. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 1220–1230 (2022)

A Guided Input Sampling-Based Perturbative Approach for Explainable AI in Image-Based Application

Arju Bano and Monidipa Das(✉)

Department of Computer Science and Engineering, Indian Institute of Technology
(ISM) Dhanbad, Dhanbad 826004, Jharkhand, India
`monidipa@iitism.ac.in`

Abstract. The role of artificial intelligence-based decision-making processes in the recent advancements of data-driven technology is inevitable, but the key challenge is embedded here in the opacity of their decision-making mechanisms, presenting difficulties in explaining these processes to the end users. The well-known RISE method (randomized input sampling for explanations) and its variants are widely used now-a-days for explainability with image data through perturbative approach. However, due to significantly large number of forward passes as required for increasing number of mask generation, RISE is heavy in computation. The issue is potentially addressed in this paper by intelligently sampling fewer number of masks through a guided scheme, instead of using large number of randomly generated masks. Our proposed approach of guided input sampling-based explanations (GuISE), introduces an innovative method for generating an importance map, illustrating the saliency of each pixel in the model's predictions. Unlike white-box explanation schemes that depend on gradients or internal network states for pixel importance estimation, GuISE functions as a black-box approach and outperforms particularly in its masking technique. To validate our approach, we compare it against the state-of-the-art importance extraction methods using both automatic deletion and insertion metrics. Extensive experiments on benchmark image datasets demonstrates comparable or superior performance of our proposed GuISE, even surpassing the white-box approaches. This highlights the effectiveness of GuISE in achieving explainability of deep neural networks for image-based applications.

Keywords: Explainable artificial intelligence · Perturbation method · Deep learning · Masking · Image classification

1 Introduction

The recent breakthroughs in deep neural networks (DNNs) have propelled substantial growth in Artificial Intelligence (AI) research. A large group of researchers is performing their research using deep neural networks, but there is an unclear and vague wave which is going on behind these research works due to

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 144–162, 2025.
https://doi.org/10.1007/978-3-031-78128-5_10

the black-box nature of the deep network models [6]. In domains where decisions carry significant consequences, like medical diagnosis, predicting diabetes [4], multispectral video classification [2], etc., the transparency of decision-making models becomes more important than the other models. The architectures of DNNs present a complex and convoluted configuration, characterized by layers of interconnected nodes and parameters. This complexity often results in a lack of transparency regarding the internal mechanisms dictating the classification outcomes produced by these networks. Such opacity in understanding the decision-making process within DNNs has led to the emergence of Explainable Artificial Intelligence (XAI), a field aimed at elucidating the inner workings of AI models, particularly in the context of image data processing which is also shown in Morphological Fragmental Perturbation Pyramid for Black-Box model explanations [21]. XAI strives to elucidate the specific features within an image that contribute to a given classification by a neural network.

A prominent method in XAI is the deployment of attribution techniques that generate saliency maps, that provide us with visual representations and indicate the relative importance or influence of different pixels in the model's decision-making process. Saliency maps have primarily been applied to the task of image classification, providing insights into the model's focus areas within an image. Various approaches have been developed for this purpose, including model-specific techniques, such as Class Activation Mapping (CAM) [24] and Gradient-weighted Class Activation Mapping (Grad-CAM) [15], as well as model-agnostic approaches, such as Shapley Additive Explanation (SHAP) [13], Randomized Input Sampling for Explanation (RISE) [10], and so on. While model-agnostic method like RISE offers the advantage of universal applicability across diverse image datasets without the prerequisite of understanding the model's internal architecture, it comes with certain drawbacks. A significant challenge with RISE is its high computational time. The generation of large number of masks substantially increases the computation load, making these less practical for real-world applications where computational efficiency is crucial.

We address this critical issue by proposing an optimized approach to reduce computational time in model-agnostic XAI techniques. We focus on optimizing the masking process in RISE and implementing a guided approach of mask generation. The strategy enhances computational efficiency while maintaining the power of explainability (refer Fig. 1), thereby making the model-agnostic XAI methods more feasible and effective for real-world applications where rapid decision-making is essential. Our contributions can be outlined as follows:

- We propose a novel method, GuISE (Guided Input Sampling-based Explanations), which enhances the explainability of deep networks in image classification, while intelligently sampling fewer masks for saliency map generation.
- GuISE improves upon well-known RISE method, reducing the computational overhead by minimizing the number of masks required for map generation.
- The paper presents experimental validation on benchmark image datasets (like ImageNet and PASCAL VOC07), demonstrating that GuISE outperforms the state-of-the-art XAI methods.

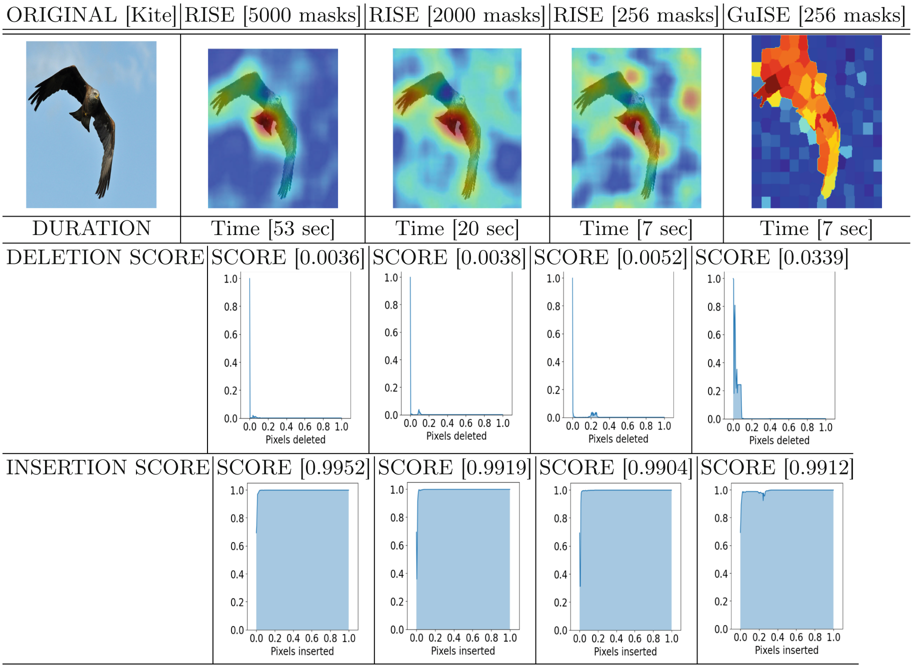


Fig. 1. This comparison highlights that the GuISE method outperforms RISE, offering more accurate saliency evaluations with fewer masks and less noise. The features highlighted within the saliency map that do not contribute to image classification by a given model, are termed here as the noise. The duration for generating saliency maps is also noted, providing insights into the efficiency of both the methods. This study is conducted on a Colab T4 GPU environment, with 72 GB disk space and 12 GB RAM.

- The paper extends the application of GuISE to medical image processing, specifically for brain tumor MRI scans, demonstrating the method's adaptability and effectiveness in medical domain.

The remaining manuscript is organized as follows: Sect. 2 discusses the related works. Section 3 illustrates the proposed GuISE method and its implementation. Section 4 presents the experimental study along with datasets details, baselines, evaluation metrics, and results. Finally, we conclude in Sect. 5.

2 Related Work

In this section, we begin by exploring classical methods aligned with Bayer et al.'s [1] classification of XAI strategies based on image evaluation, such as *Gradient-based* and *Perturbation-based* approach, both of which visualize their interpretations through heatmaps.

Gradient-based methods analyze how each input pixel contributes to a neural network's final decision by backpropagating importance scores from the output back to the input. Techniques like Gradients [16], Excitation Backprop [23], and Class Activation Mapping (CAM) [24] elucidate which parts of an image influence the model's predictions. However, these methods are often tailored to specific network architectures, requiring adjustments for new models. Grad-CAM [15], Grad-CAM++ [3], Score-CAM [20], Faster Score-CAM [20] are advancements in this area, which generate saliency maps by combining gradients and activations from a network's intermediate layers, offering more general approaches that can be applied across different Convolutional Neural Network (CNN) architectures. Grad-CAM's low-resolution saliency maps lack the precise detail, offered by perturbation-based approaches, potentially missing subtle but important features. Grad-CAM, Grad-CAM++ are predominantly applicable to convolutional layers, limiting its versatility compared to model-agnostic perturbation methods. In case of Score-CAM and Faster Score-CAM, a notable drawback is that if the model is already highly confident in its prediction, these may not effectively identify the most important regions of the image [17].

In the **perturbation-based approach** for image analysis, various techniques are used to modify regions of an image such as occlusion, noise addition, inpainting and blurring, to observe the effect on the model's output. These methods then assess the significance of the perturbed region based on changes in the model's performance. For example, the occlusion method [22] obscures square sections of the image sequentially, using a sliding window approach, and evaluates the impact on the classification score to ascertain the perturbed region's importance. LIME [12] simplifies the complex model into a linear classifier, which is trained near the input data point. This training uses samples with occluded superpixels, and the significance of superpixels is gauged by the weights assigned during the linear classifier's training. The drawback of LIME lies in its reliance on a linear approximation to explain the behavior of potentially highly nonlinear models within a local vicinity of the input being explained. Randomized input sampling for explanations (RISE) [10] creates an ensemble of random masks, applies them to the input image and calculates the class probabilities for each masked version. The saliency map is then generated as a weighted sum of these masks, with weights derived from the predicted class probabilities. However, it incurs high demand computational costs when numerous activation maps with positive slopes are present, impacting efficiency in the prediction output. Transitioning from the exploration of classical methods in XAI, in the paragraph below, we delve into the nuances of understanding the broader categories of methodology, namely ***white-box*** and ***black-box*** approaches. The categorization of the methods here is primarily based on their strategy of governing the application in practical settings. This also illuminates the importance of bridging the gap between specific analytical methods and their broader implementation strategies.

White-box and Black-box approaches serve slightly different purposes. White-box methods are faster as required for large-scale and real-time applications. The methods discussed in this context focus on explaining a scalar value in the model's output, typically the class probability score in image classification. In contrast, the black-box methods, though slower at runtime, offers higher generalizability and ease of application making them suitable for proprietary models or Application Programming Interface (API). They directly measure the impact of input ablations on the model without relying on backpropagation heuristics. In our efforts to enhance the black-box approach, we aimed for real-life applicability and reduced execution time by minimizing the number of masks employed in the existing method. Our approach aligns with the perturbation-based methodology, wherein we perturb the image to facilitate explainability.

3 Proposed Method: GuISE

In this study, we introduce GuISE (Guided Input Sampling for Explanations), a novel approach that enhances saliency output and accuracy in explainability compared to the existing RISE method. RISE requires a large number of masks to perturb the image, significantly increasing computational time. It relies on a random binary masking technique, further escalating computational demands. GuISE is designed to address these challenges by reducing the number of masks needed and optimizing the masking process for more efficient computation. Overall, our proposed GuISE method is inspired by the randomized perturbation masks generation approach by the RISE model applied on the image to explain object classifiers, except that GuISE leverages the random-masking concept to guide masks. The key idea is to measure the effect of masking guided regions on the predicted output, to determine the importance.

In our GuISE methodology, we have redefined the masking approach, shifting from the traditional random binary masking to a strategy that is more targeted and deliberate. Unlike the conventional method where each image segment has an equal probability of 0.5 for masking, GuISE assigns priority to segments based on the extent of the area they cover. Specifically, areas with a smaller coverage of similar pixels are perturbed with a probability higher than 0.5. This adjustment ensures that the smaller segments, which would otherwise receive minimal importance with a flat probability rate, are adequately emphasized in comparison to larger areas receiving the same probability. This innovative masking technique incorporates ***superpixel segmentation***, followed by ***fuzzy c-means clustering***, and ***histogram analysis*** to guide the masking process, which significantly enhances the specificity and the effectiveness of saliency detection. In the subsequent part of this section we discuss on each of these steps individually.

3.1 Guided Mask Generation

Superpixel Segmentation: For this purpose GuISE employs Simple Linear Iterative Clustering (SLIC) method. SLIC provides a straightforward approach for clustering pixels into superpixel segments. By setting a target number k, SLIC aims to produce a specified number of superpixels that are approximately equal in size (refer Algorithm 1). The parameter k is used to generate the number of superpixels, referred to as $n\_segments$. This value is computed by dividing the total number of pixels in the original image by 256. The choice of 256 as the divisor strikes a balance between computational efficiency and segmentation granularity. SLIC incorporates parameters like *sigma* for enhancing the smoothness of segmentation and *compactness* for ensuring that each segment achieves a proper shape. This approach allows meaningful segmentation by balancing between the spatial coherence and the appearance of superpixels.

Algorithm 1: Segmentation using SLIC

```
Data: $original\_img$, $n\_segments$, $compactness$, $sigma$
Result: $segments\_slice$
$segments\_slice \leftarrow slic(original\_img, n\_segments = k, compactness =$
    $10, sigma = 1)$;
```

Fuzzy C-Means Clustering: After using SLIC to generate superpixels, we grouped similar superpixels using clustering. These groups represent areas of similar characteristics within the image. Unlike binary classification, which restricts groups to either 0 or 1, our approach ensures variability within groupings. To achieve this nuanced classification, we employed fuzzy c-means clustering, which allows for more flexibility by assigning membership levels to each superpixel in multiple clusters. Fuzzy c-means is an unsupervised clustering algorithm that operates without the need for pre-labeled data, grouping pixels based on their similarity. The process begins by selecting a desired number of clusters ($n\_cluster$), and randomly assigning pixels to these clusters. During experimentation we chose to use $n\_cluster = 5$ as this number effectively groups similar features together, avoiding excessive fragmentation that can occur with a higher number of clusters. Our empirical testing showed that using 5 clusters provided interpretable and meaningful results in our dataset. The algorithm then iteratively adjusts the pixel assignments based on the distance of each pixel from the cluster centers, taking into account the level of fuzziness (m), which is controlled by setting $m = 2$.

$Fuzzy\_cmeans \leftarrow$ cmeans(Image, n_cluster=n_cluster, m=2, $maxiter = 1000$). The parameter *maxiter* specifies the maximum number of iterations allowed for optimizing the clusters to achieve the final segmentation. We are utilizing fuzzy c-means clustering to identify distinct partitions within the image. This facilitates the analysis of histograms for each unique area, enabling precise characterization of their individual properties.

Histogram Analysis: It serves the purpose of summarizing data, applicable to both discrete and continuous types measured on an interval scale. It is commonly used to visually illustrate the significant characteristics of data distribution. In our study, we utilized a histogram to calculate the composition of each cluster derived from fuzzy c-means. Formally it can be represented as: $bincount(clustered\_segments\_img.flatten(), minlength =$

$n\_cluster$). In this formula, $n\_cluster$ represents the number of clusters, while the $clustered\_segments\_img$ refers to the outcome after applying SLIC and fuzzy c-means clustering to the image. The histogram is utilized to determine area percentage per cluster and also for computing cluster-wise probability such that the small areas are prioritized. Subsequently, a binary scheme is applied to generate the mask while receiving guidance from these estimated probabilities. Figure 2 presents the overall workflow of the proposed guided mask generation approach. The approach is also elucidated through an algorithmic presentation below (refer Algorithm 2).

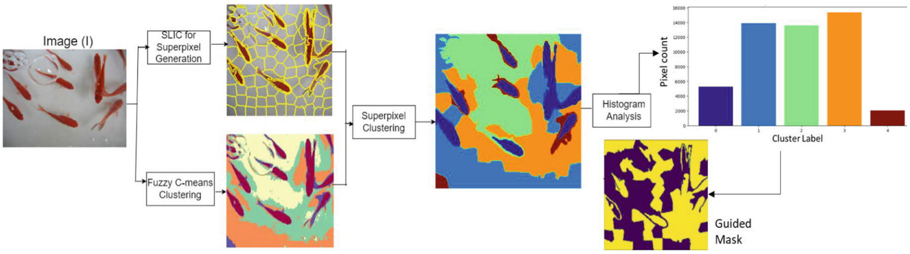


Fig. 2. Guided Mask Approach: Initiates with segmenting image I using SLIC, applies Fuzzy c-means to superpixels, and creates histograms for each superpixel cluster

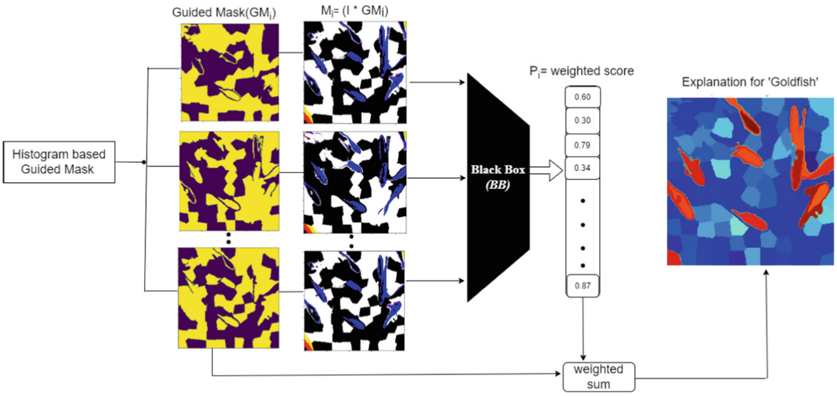


Fig. 3. GuISE Workflow: Begins with image I multiplied by guided masks GM_i, then processed through the base model; saliency map is generated through linear combination of guided masks, weighted by target class scores.

3.2 Guided Input Sampling-Based Explanation

After generating the guided mask we follow the basic perturbation approach of RISE method and subsequently the saliency map is generated using the guided mask. This enhanced RISE method with the added guided masking scheme is termed as GuISE. Figure 3 shows an overview of the GuISE approach, where the input image, denoted as I, undergoes element-wise multiplication with the guided mask GM_i and then the guided masked image (M_i) is fed into the base model (BB). The saliency map is formed by linearly combining masks with the weights being determined by the scores (P_i) of the target class associated with the corresponding masked inputs (M_i). The overall approach is also formally presented through the Algorithm 3.

Algorithm 2: Algorithm for Guided Mask

```
Data: Image ($I$), number of segment for superpixel ($n\_segment$), number of
      cluster for fuzzy c-means ($n\_cluster$), number of Guided Mask ($N$)
Result: Guided Mask ($GM$)
Procedure:
1. SI ← SLIC ($I$, $n\_segment$, compactness=10, sigma=1)/* Refer Algorithm 1
2. CI ← cmeans ($I$, $n\_cluster$, $m = 2$, $maxiter = 1000$) ;   /* Fuzzy C-Means
   Clustering applied to Image I with $n\_clusters$ */
3. foreach $segment \in SI$ do
       segments ← AssignCluster (segment, CI);   /* Compute and assign the
         most frequent cluster from $CI$ */
4. ClusterHistogram ← $\sum_{i=1}^{n\_cluster} CountSegments(CI, segments)$ /*For each
   cluster in CI, count segments in SI and update ClusterHistogram
5. ClusterAreaPercentage ← Area (ClusterHistogram) / Area ($I$) ;
   /* Calculate the area(number of pixel count) percentage of each
   cluster, for computing cluster probability */
6. MeanArea ← Area ($I$) / $n\_cluster$ /*Computing mean cluster area
7.foreach $c \in CI$ do
       if $Area(c) \geq MeanArea$ then
           Probability[$c$] ← 0.5;
       else
           Probability[$c$] ← 1 - ClusterAreaPercentage;
       /* For each cluster $c$, assign a probability based on its area
          relative to $MeanArea$                                        */
8. Mask ← ZeroMatrix ($I$.Height, $I$.Width) ;   /* Initialize a mask as a
   zero matrix with dimensions equal to the height and width of I */
9.foreach $cluster\ c \in CI$ do
       $n_{\text{assigned\_segments}}$ ← Round(ClusterHistogram[$c$] × Probability[$c$])
         $GM$ ← RandomAssignOne(Mask, $n_{\text{assigned\_segments}}$, $c$) /*For each cluster,
         determine segment count for Mask, then randomly assign segments to 1
10. Repeat step 9 and generate $N$ number of Guided Mask ($GM$).
```

4 Experimental Study

Datasets: Our research is conducted using two prominent publicly accessible datasets for object classification: ImageNet [14] and PASCAL VOC07 [5]. We employ the Resnet-50 architecture, leveraging its capabilities as a pre-trained model to facilitate efficient feature extraction and object classification tasks. To extend our research into the domain of medical imaging, we have also used a publicly available dataset comprising Magnetic Resonance Imaging (MRI) scans, specifically focused on brain tumors [9]. By applying the Resnet-50 model as a foundation and modifying its final layer, we adapt it for precise classification of brain tumors into four distinct categories, showcasing the effectiveness of transfer learning in adapting to specialized medical imaging tasks.

Algorithm 3: GuISE ALGORITHM

```
Data: Image ($I$), Guided Mask ($GM$), Pre-trained Model ($M$), number of
      guided mask ($N$)
Result: Saliency Map ($S$)
Procedure:
1. Select an Image $I$ /*Choose an input image I for generating saliency map
2. BlackBoxModel (BB) $\leftarrow$ ($M$)/*Select a pre-trained black-box model, such as
   ResNet-50 or VGG-16 which represents the decision-making process
3. foreach guidedmask $GM_i \in GM$ do
       $M_i \leftarrow I \odot GM_i$ ;    /* Apply the Guided Mask to the Input Image I to
         get the guided masked image $M_i$, where 1<=$i$<=N */
       $P_i \leftarrow$ BB($M_i$) /*Pass the modified masked image $M_i$ to the black-box
         model to predict the probability of the weighted score $P_i$
4. $SaliencyMap$(S) $\leftarrow \sum_{i=1}^{N}(P_i^T \cdot M_i)$ /*Generate Saliency Maps by weighted
   sum of modified masked images and their weighted scores
```

Baseline: Our methodology is benchmarked against the RISE method for explainability in neural networks. We also compare our saliency map generation approach with the white-box CAM variants, namely Grad-CAM++ [3], Score-CAM [20] and Faster Score-CAM [20]. For this comparative analysis, publicly available code implementations of CAM variants and RISE have been utilized. It is important to note that we do not compare with recent black-box methods, like Detector Randomized Input Sampling for Explanation (DRISE) [11] and Detector-Cascading multiple Levels of Segments to Explain (D-CLOSE) [19], as these are primarily focused on object detection tasks. Our research is concentrated on classification tasks and the GuISE method is an extension of the RISE concept, therefore, we have limited our comparison specifically to RISE for a focused assessment. The code utilized in this research is publicly accessible and can be found at the following GitHub repository: //github.com/aulphin/GuISE-XAI.

Evaluation Metrics: We perform quantitative and qualitative evaluations on our proposed method. The qualitative evaluation is done based on how good it is in generating *saliency maps* compared to existing methods. For the quantitative evaluation, we use the evaluation metrics proposed by RISE [10], that is deletion and insertion scores and the evaluation metric m_{GT}, as discussed below.

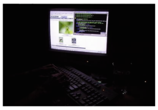
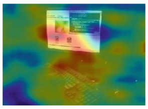
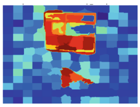
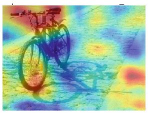
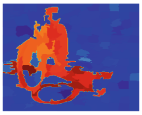
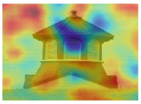
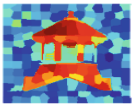
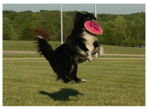
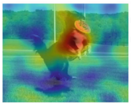
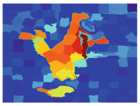
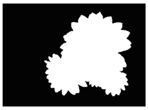
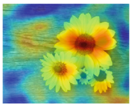
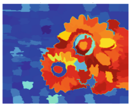
1. **Deletion score-** This metric quantifies the reduction in the probability of the predicted class as more important pixels are removed. The significance of each pixel is derived from the saliency map. This can be presented by:

$$Deletionscore = DeletePixels(I(empty), S, N)$$

   where $DeletePixels$ is a function that deletes N pixels from saliency map S and computes its score based on original image I till the important pixels get empty.
2. **Insertion score-** It quantifies the increase in the probability of the predicted class with the gradual introduction of important pixels, providing insights into how efficiently the model gains confidence in its predictions during the pixel insertion process. This can be presented by:

$$Insertionscore = InsertPixels(I(blurred), S, N)$$

Table 1. Comparison of RISE and GuISE generated Saliency Maps by Utilizing Ground Truth Masks on MSRA-B Image Dataset and Computed Evaluation Metrics

DESKTOP IMAGE 	GT mask 	RISE 	Insertion Score: **0.3215** Deletion Score: **0.0023** m_{GT}: 0.0946	GuISE 	Insertion Score: 0.2673 Deletion Score: 0.0246 m_{GT}: **0.2269**
MOUNTAIN BIKE 	GT mask 	RISE 	Insertion Score: 0.7138 Deletion Score: **0.0044** m_{GT}: 0.1536	GuISE 	Insertion Score: **0.7978** Deletion Score: 0.0571 m_{GT}: **0.6636**
BEACON HOUSE 	GT mask 	RISE 	Insertion Score: **0.5655** Deletion Score: **0.0019** m_{GT}: 0.3593	GuISE 	Insertion Score: 0.5009 Deletion Score: 0.0730 m_{GT}: **0.3890**
BORDER COLLIE 	GT mask 	RISE 	Insertion Score: **0.5396** Deletion Score: **0.0014** m_{GT}: 0.3759	GuISE 	Insertion Score: 0.5356 Deletion Score: **0.0014** m_{GT}: **0.5101**
DAISY (FLOWER) 	GT mask 	RISE 	Insertion Score: **0.9092** Deletion Score: **0.0100** m_{GT}: 0.4312	GuISE 	Insertion Score: 0.8231 Deletion Score: 0.1117 m_{GT}: **0.5779**

where $InsertPixels$ is a function that inserts N pixels into the blurred image I based on the saliency map S and computes its score.

3. m_{GT} **Metric-** The m_{GT} metric [18] is computed as the ratio of the number of overlapping significant pixels to the total number of significant pixels in the ground truth (GT) mask and is computed as follows: $m_{GT} = n/p$, where n is the number of pixels in the explainer model generated saliency map mask that fall within the positive regions of the GT mask, and p is the total number of positive pixels (binary ones) in the GT mask. Saliency map with higher m_{GT} value indicates a better XAI approach (refer Table 1).

Table 2. Insertion Score Comparison: GuISE outperforms RISE on ImageNet/PASCAL dataset, achieving higher scores with 256 masks against RISE's 2000; higher scores indicate better performance.

INSERTION SCORE		
IMAGE	RISE[2000 masks]	GuISE[256 masks]
GOOSE	0.9477	**0.9679**
GOLDFISH	0.8221	**0.8336**
BRAMBLING	**0.9976**	0.9910
BLACK-FOOTED FERRET	0.7081	**0.7247**
EUROPEAN GALLINULE	0.3514	**0.5790**
DRAKE	**0.6815**	0.5450
SPOTLIGHT	0.7942	**0.8234**
WHIPPET	**0.6020**	0.5660

Table 3. Deletion Scores Comparison: GuISE outperforms RISE on ImageNet/PASCAL dataset, showing better results with 256 masks versus RISE's 2000 masks; lower scores indicate better performance.

DELETION SCORE		
IMAGE	RISE [2000 masks]	GuISE [256 masks]
GOOSE	0.6078	**0.0167**
GOLDFISH	**0.0441**	0.0515
BRAMBLING	**0.0108**	0.0951
BLACK-FOOTED FERRET	**0.0021**	0.0039
EUROPEAN GALLINULE	**0.0008**	0.0009
DRAKE	**0.0015**	**0.0015**
SPOTLIGHT	**0.0063**	**0.0063**
WHIPPET	0.0016	**0.0015**

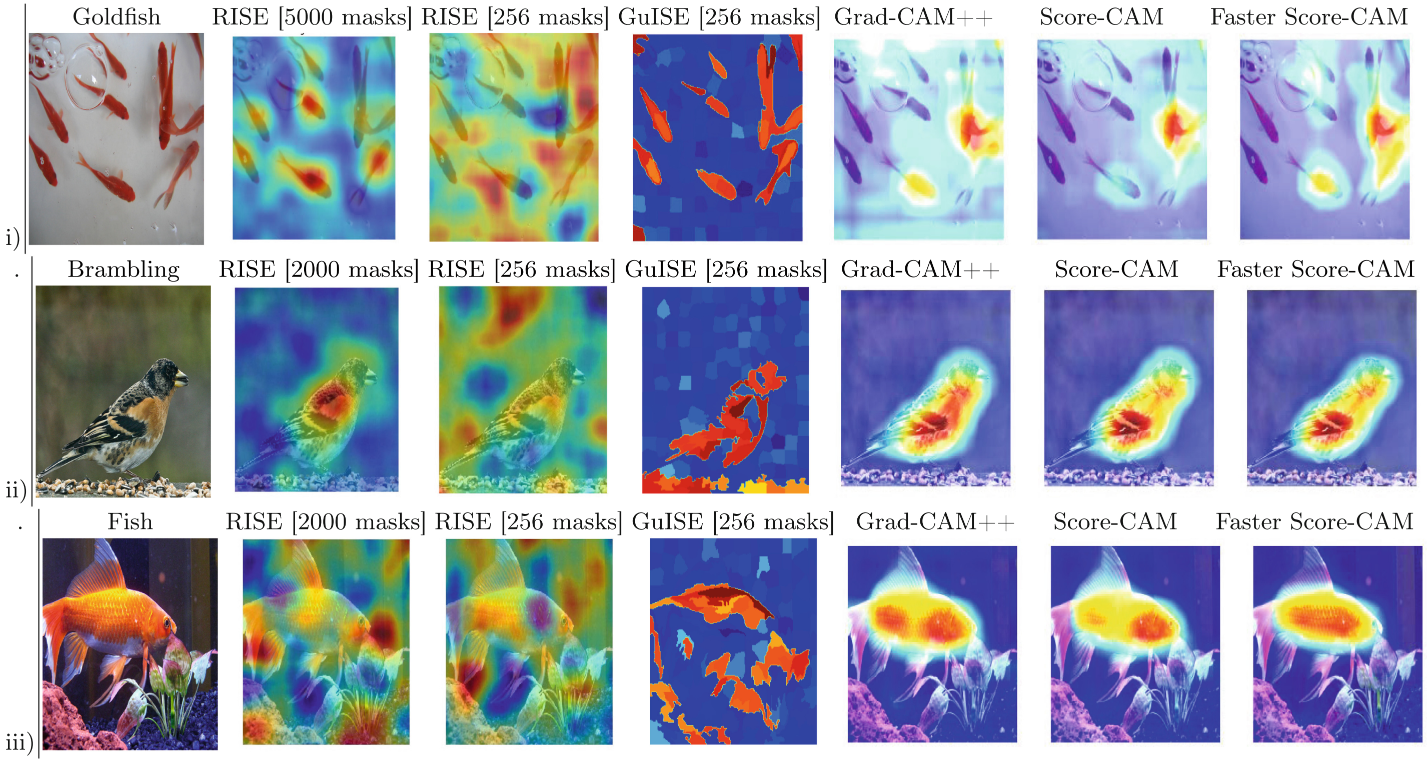


Fig. 4. Part-I: This Figure demonstrates GuISE's superiority in capturing critical features using state-of-the-art methods. It highlights images such as a Goldfish, a Brambling bird, and fish. The saliency maps show more accurate feature identification, with intensities ranging from blue to red. (Color figure online)

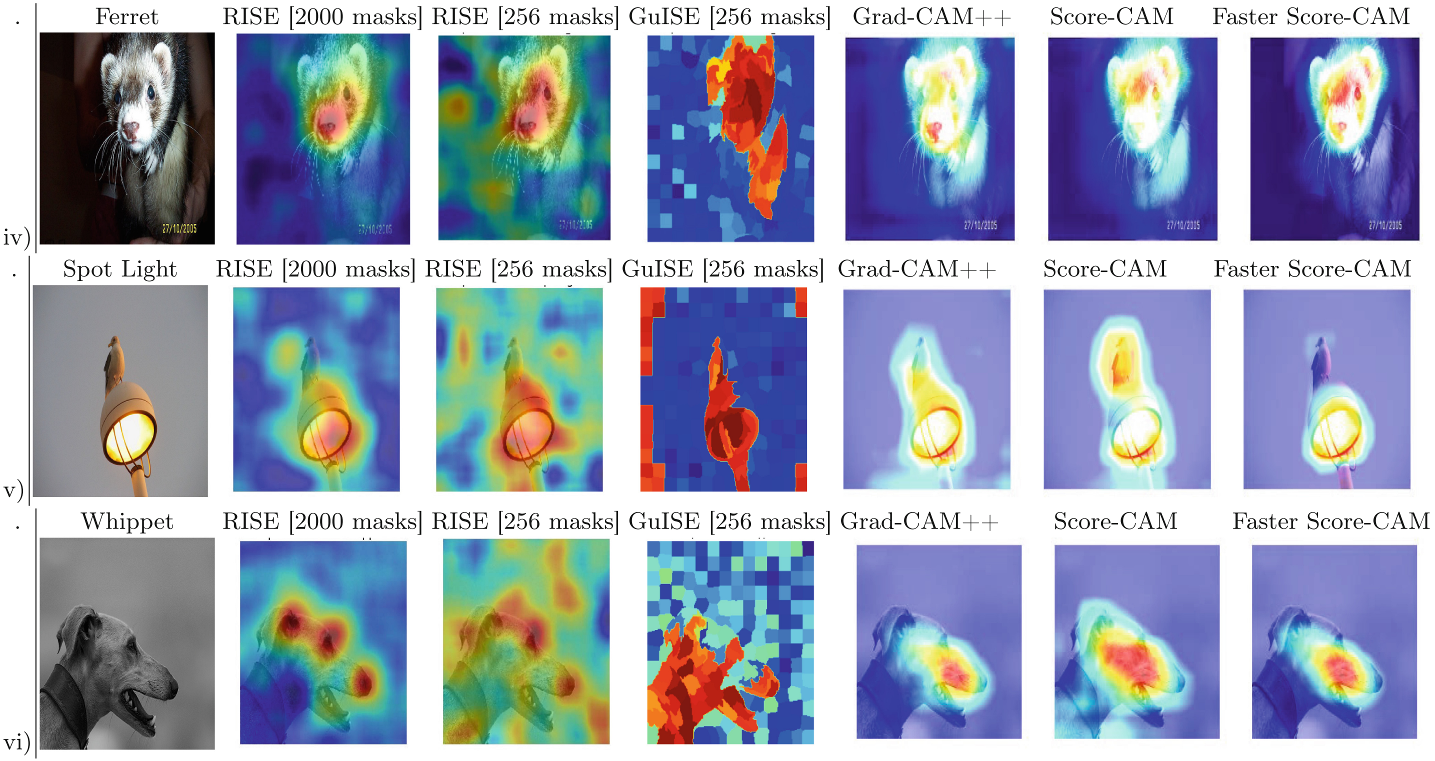


Fig. 5. Part-II: This Figure demonstrates GuISE's superiority in capturing critical features using state-of-the-art methods. It highlights images such as a Black-Footed Ferret, Spotlight, Whippet. The saliency maps show more accurate feature identification, with intensities ranging from blue to red. (Color figure online)

Table 4. Deletion Score Comparison: GuISE vs. RISE on Brain Tumor MRI dataset, with GuISE's 256 masks outperforming RISE's 2000 masks; lower scores are better.

DELETION SCORE		
BRAIN TUMOR	RISE [2000 masks]	GuISE [256 masks]
IMAGE1 [Pituitary]	**0.0187**	0.3031
IMAGE2 [Meniongima]	0.3563	**0.1538**
IMAGE3 [Glioma]	**0.1506**	0.2246
IMAGE4 [No Tumor]	0.4486	**0.3998**

Table 5. Insertion Score Comparison: On Brain Tumor MRI dataset, GuISE surpasses RISE, showing higher scores with 256 masks versus 2000; higher scores are preferable.

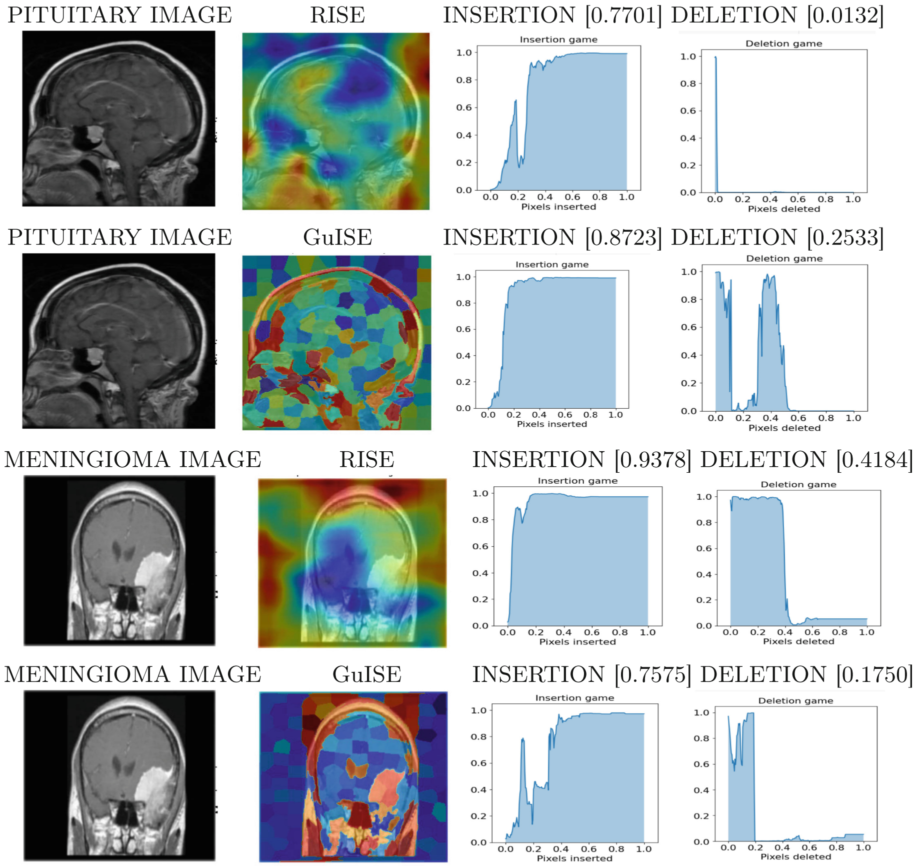
INSERTION SCORE		
BRAIN TUMOR	RISE [2000 masks]	GuISE [256 masks]
IMAGE1 [Pituitary]	0.6783	**0.8909**
IMAGE2 [Meniongima]	**0.8040**	0.7551
IMAGE3 [Glioma]	0.9854	**0.9900**
IMAGE4 [No Tumor]	0.7151	**0.8487**

Experimental Settings: Our study involved experiments on two categories of datasets: ImageNet [14] /PASCAL VOC07 [5] for general classification and the medical Brain Tumor MRI dataset [9], featuring grayscale images. For the generation of binary guided masks, a technique is employed where probabilities were assigned as 0 s and 1 s based on the area coverage within each image. To achieve this, we utilize a set of 256 masks. In addition, the fuzzy c-means clustering parameter $n\_cluster$ is set to 5. All images are standardized to 224×224 pixels in height and width.

4.1 Results and Discussions

Exploratory Analysis on Benchmark Image Dataset and Its Saliency: Under conditions where fewer masks are utilized, our method outperforms RISE by generating less noise and more effectively highlighting key features, demonstrating its efficiency in focus and clarity. The evaluation result is shown in Tables 2 and 3. GuISE is outperforming in most of the cases over the existing method RISE, in terms of insertion and deletion scores. For saliency map representation, here we have also compared our method with gradient-based (white-box) approaches including Grad-CAM++ [3], Score-CAM [20] and Faster Score-CAM [20], as shown in Figs. 4 and 5. While gradient-based approaches produce good saliency maps, their limitations include difficulty in capturing small pixel details, particularly when multiple objects are important in a single image, and a lack of user-friendliness.

Table 6. Saliency Map Comparison in Medical Imaging: GuISE provides more focused saliency representation with significantly reduced noise, compared to RISE.



In case of black-box methods, we report the mean values obtained from conducting five separate experimental runs. However, it is worth noting that RISE [2000 masks] incurs heavy computational costs due to the higher number of forward passes. In our saliency map generation process, we compared RISE and GuISE methods using ResNet-50 model. RISE, which uses 2000 masks, takes 25 s to generate the saliency map. In contrast, GuISE, which uses only 256 masks, takes 9 s, demonstrating significantly higher efficiency. These tests were conducted on a Colab T4 GPU with 72 GB disk space and 12 GB RAM.

We also experimented with RISE using 256 masks. In Figs. 4 and 5 the results show that RISE generates saliency maps with considerably more noise. Here, by means of noise we indicate the features which the deep learning model does not consider important, yet are highlighted by the explainer model. In these saliency maps, red indicates areas of higher importance, while blue signifies less important regions. However, the RISE method [256 masks] erroneously interprets noisy regions as important, resulting in a misleadingly enhanced deletion score. In contrast, our method generates saliency maps that effectively emphasize important features while substantially reducing noise. This leads to a more genuine deletion score or insertion score, even if numerically higher (which might appear less favorable) compared to RISE. This is because our method produces significantly less noise than the RISE method, offering a more accurate representation of performance in the evaluation metrics. In order to strengthen our claim regarding noisiness of the generated saliency maps, we did additional comparative study with respect to m_{GT} metric [18]. Since this metric requires a ground truth mask, for the saliency map evaluation purpose we utilized the publicly available MSRA-B dataset[1], which contains the original images as well as ground truth masks. Our results show that GuISE always performs better than RISE in terms of m_{GT} value, as shown in Table 1, demonstrating its superior capability to highlight important features which match well with the ground truth.

Exploratory Analysis on Medical Image and Its Saliency: We have also experimented on medical images to evaluate the efficiency of our method in medical data analysis. To adapt our approach for medical image processing, we have utilized transfer learning with a pre-trained ResNet-50 model. Since ResNet-50 is originally trained on the comprehensive ImageNet dataset, using this pre-trained model enables the precise classification of medical images. This is achieved by applying the extensive knowledge gathered from the ImageNet dataset to improve the performance of our specific task of classifying medical images. In our study, the brain MRI dataset consisted of 300 glioma images, 306 meningioma images, 300 pituitary tumor images, and 405 images with no tumor. After fine-tuning ResNet-50 on this dataset, we achieved a training accuracy of 98.20% and a validation accuracy of 95.35%, with a precision of 0.9570, recall of 0.9550, and an F1 score of 0.9537.

The outcomes, as presented in Tables 4 and 5, demonstrate that our method outperforms the RISE technique with respect to deletion and insertion scores. The experiments use the publicly available Brain Tumor MRI images dataset to test the effectiveness of our approach in the domain of medical imaging. Specifically, the medical image in Table 6 illustrates that the GuISE method produces saliency maps with less noise compared to those generated by the RISE method.

The key role of an XAI model in the medical domain can be to increase the physicians' confidence in prediction during the brain tumor MRI diagnosis process. However, the application of a number of XAI methods to brain tumor MRI datasets has not been discovered yet [8]. There is a pressing gap in our understanding regarding how XAI impacts the predictive accuracy, confidence levels, and overall trustworthiness of models among physicians and radiologists in simulated environments. Moreover, the real-world implications of XAI in clinical

---

[1] https://jianghz.me/drfi/.

settings are yet to be elucidated. In our study, we observe that RISE is effective for general image datasets, yet at the cost of increased computational demands. However, when applied to medical imaging data, RISE's efficiency diminishes, primarily because it tends to interpret noise as significant pixels. This misinterpretation inflate its performance metrics, such as insertion and deletion scores. In contrast, our proposed method, GuISE demonstrates a more targeted approach by focusing on the crucial aspects of MRI images that indicate the presence of tumors.

As shown in Table 6, in case of classified pituitary image, though RISE highlights salient features, the method also introduces substantial noise. In contrast, GuISE correctly focuses on the MRI pituitary gland region. Additionally, GuISE achieves a higher insertion score than RISE. Although RISE has a better deletion score, GuISE provides more meaningful information in its saliency. GuISE saliency indicates that the classification of pituitary tumors involves not only the pituitary gland but also other regions of the brain. This assertion is corroborated by the work of Kaltsas et al. [7]. This demonstrates that our approach performs well in the medical domain, offering deeper visualization and understanding. Similarly, in case of classified Meningioma image, we observe that the GuISE method focuses on the relevant MRI part of the brain, while RISE highlights areas outside the MRI. This not only indicates that GuISE performs better than RISE by accurately focusing on the MRI region, but also reveals that RISE introduces significant noise by focusing on irrelevant pixels. It must be noted here that before applying the XAI approaches, in all the above cases, we ensured that our fine-tuned classifier model correctly classified the respective images.

5 Conclusion

This paper introduces GuISE, a technique designed to interpret the workings of black-box models through the evaluation of input image regions, important for model predictions. Despite its straightforward and versatile nature, GuISE outperforms current explanation methodologies in causal metrics and the quantity of masks used. The guided mask method streamlines mask generation, optimizing mask quality and reducing computation by selecting the most informative masks. GuISE effectively balances broad and detailed perspectives, facilitating a smooth and consistent approach to mask creation. GuISE may occasionally produce noisy saliency maps, especially with objects of varied classification, due to sampling approximations. While we conducted evaluations of our method on medical images using quantitative metrics, there remains a need for assessing radiologist trust for qualitative metrics. This aspect can be explored in future through a combined approach, to gather insights from medical radiologists.

Acknowledgements. This research is supported by the Start-up Research Grant [SRG/2023/002658 to M.D.] from the SERB (DST), Govt. of India, New Delhi.

References

1. Bayer, J., Münch, D., Arens, M.: A comparison of deep saliency map generators on multispectral data in object detection. In: Counterterrorism, Crime Fighting, Forensics, and Surveillance Technologies V, vol. 11869, pp. 61–74. SPIE (2021)
2. Bayer, J., Münch, D., Arens, M.: Deep saliency map generators for multispectral video classification. In: 2022 26th International Conference on Pattern Recognition (ICPR), pp. 3757–3764. IEEE (2022)
3. Chattopadhay, A., Sarkar, A., Howlader, P., Balasubramanian, V.N.: Grad-CAM++: generalized gradient-based visual explanations for deep convolutional networks. In: 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 839–847. IEEE (2018)
4. Chen, T.C.T., Wu, H.C., Chiu, M.C.: A deep neural network with modified random forest incremental interpretation approach for diagnosing diabetes in smart healthcare. Appl. Soft Comput. **152**, 111183 (2024)
5. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. Int. J. Comput. Vision **88**, 303–338 (2010)
6. Hakkoum, H., Abnane, I., Idri, A.: Interpretability in the medical field: a systematic mapping and review study. Appl. Soft Comput. **117**, 108391 (2022)
7. Kaltsas, G.A., Nomikos, P., Kontogeorgos, G., Buchfelder, M., Grossman, A.B.: Diagnosis and management of pituitary carcinomas. J. Clin. Endocrinol. Metab. **90**(5), 3089–3099 (2005)
8. Nguyen, T.T.H., Truong, V.B., Nguyen, V.T.K., Cao, Q.H., Nguyen, Q.K.: Towards trust of explainable AI in thyroid nodule diagnosis. arXiv preprint arXiv:2303.04731 (2023)
9. Nickparvar, M.: Brain tumor MRI dataset. Data set. Kaggle (2021). https://doi.org/10.34740/KAGGLE/DSV/2645886. Accessed 3 Mar 2021
10. Petsiuk, V., Das, A., Saenko, K.: RISE: randomized input sampling for explanation of black-box models. arXiv preprint arXiv:1806.07421 (2018)
11. Petsiuk, V., et al.: Black-box explanation of object detectors via saliency maps. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11443–11452 (2021)
12. Ribeiro, M.T., Singh, S., Guestrin, C.: " Why should I trust you?" explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1135–1144 (2016)
13. Roshan, K., Zafar, A.: Utilizing XAI technique to improve autoencoder based model for computer network anomaly detection with Shapley additive explanation (SHAP). arXiv preprint arXiv:2112.08442 (2021)
14. Russakovsky, O., et al.: ImageNet large scale visual recognition challenge. Int. J. Comput. Vision **115**, 211–252 (2015)
15. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-CAM: visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 618–626 (2017)
16. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: visualising image classification models and saliency maps. arXiv preprint arXiv:1312.6034 (2013)
17. Soomro, S., Niaz, A., Choi, K.N.: Grad++ ScoreCAM: enhancing visual explanations of deep convolutional networks using incremented gradient and score-weighted methods. IEEE Access (2024)

18. Szczepankiewicz, K., et al.: Ground truth based comparison of saliency maps algorithms. Sci. Rep. **13**(1), 16887 (2023)
19. Truong, V.B., Nguyen, T.T.H., Nguyen, V.T.K., Nguyen, Q.K., Cao, Q.H.: Towards better explanations for object detection. Preprint arXiv:2306.02744 (2023)
20. Wang, H., et al.: Score-CAM: score-weighted visual explanations for convolutional neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp. 24–25 (2020)
21. Yang, Q., Zhu, X., Fwu, J.K., Ye, Y., You, G., Zhu, Y.: MFPP: morphological fragmental perturbation pyramid for black-box model explanations. In: 2020 25th International Conference on Pattern Recognition (ICPR), pp. 1376–1383. IEEE (2021)
22. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8689, pp. 818–833. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10590-1_53
23. Zhang, J., Bargal, S.A., Lin, Z., Brandt, J., Shen, X., Sclaroff, S.: Top-down neural attention by excitation backprop. Int. J. Comput. Vision **126**(10), 1084–1102 (2018)
24. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2921–2929 (2016)

Multi-target Attention Dispersion Adversarial Attack Against Aerial Object Detector

Panpan Wang[1], Shujuan Wang[1], Zhichao Lian[1](✉), and Shuohao Li[2]

[1] Nanjing University of Science and Technology, Nanjing 210094, China
{wpp_css,wsj-css,newlzcts}@njust.edu.cn
[2] National University of Defense Technology, Hunan 410015, China
lishuohao@nudt.edu.cn

Abstract. In recent years, object detection has become very common and has wide applications in various fields. As research deepens, CNN-based object detectors are vulnerable to adversarial examples, and this weakness can also plague aerial object detectors. However, existing attacks can only train one type of patch at one time, and these adversarial patches are usually universal patches. In scenarios where we only want to apply patch to attacks certain categories of targets but do not attack other categories, these patches are not applicable. Meanwhile, due to the training characteristics of universal patches (attacking all categories), they often have an impact on the detection of objects of other categories around them without patches. In this work, we focus on multi-target category adversarial attacks against aerial detection and propose a novel attack method, Multi target distraction attack (MTDA). Specifically, we aim to design different patches for different categories of targets, and generate multiple adversarial examples corresponding to each category through the patch-apply module in order to restrict the attack area to the target area. Furthermore, to achieve higher attack performance on aerial detector, we utilize a novel attention loss to reduce the attention of objects in the target area while improving the attention of non-target areas. The experimental results indicate that compared to traditional attack methods, our method can achieve a higher attack success rate (ASR) on the Yolov3, with the highest ASR of multi-target united attack reaching 99.28%.

Keywords: Multi-target Adversarial Attack · Attention Dispersion · Aerial Detection

1 Introduction

With the increasing popularity of drones in monitor and reconnaissance [1], more and more remote sensing image datasets have been published, and deep neural networks (DNN) have become the key to processing a large number of aerial images collected by earth observation satellite platforms. Aerial detection methods based on DNN are widely used in intelligent traffic monitoring, environmental monitoring, disaster response and rescue, border security and defense, and other fields. It exhibits excellent performance in both accuracy and efficiency.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 163–179, 2024.
https://doi.org/10.1007/978-3-031-78128-5_11

However, recent studies have validated the vulnerability of DNN to adversarial examples, which can also affect the object detector of aerial images. Only a small amount of malicious disturbance added to clean images can mislead deep neural networks to make wrong predictions. Therefore, studying new adversarial attack methods is essential to explain the vulnerability of DNN to adversarial examples.

At present, common adversarial attacks against object detection can be divided into two categories based on the attack area, they are pixel-wise adversarial attacks [2–9] and patch-wise adversarial attacks [10–14]. Pixel-wise adversarial attack mainly fine tunes the original image through iterative gradient descent to make the added noise distributed throughout the input image. This attack is often hardly noticeable. Patch-wise adversarial attack refers to adding patches to the fixed region in the clean image. These patches can be specific geometric shapes, textures, or colors, and are carefully designed to confuse the model. Due to the ease of extending patches from the digital domain to the physical domain, patch-wise adversarial attacks are still the mainstream attack method against object detection by printing patches in the physical world.

However, previous work mainly focused on single category of targets and can only generate one type of patch at a time. Meanwhile, the adversarial patches generated by their training mode are often universal patches, which lack target specificity. Due to the characteristics of global dispersion and local aggregation in small target remote sensing images, one-on-one patching can lead to mutual influence between different target classes, resulting in detection results deviating from expectations, as shown in Fig. 1. In addition, most existing methods focused on downstream tasks of object detectors. As the detection accuracy of detection models continues to improve, the generated adversarial examples often suffer from insufficient attack intensity.

Based on the above issues, our work focuses on multi-target adversarial attacks against aerial detection. Different patches are designed for different categories of targets, and multiple adversarial examples corresponding to each category are generated through the patch apply module. The attack area is limited to the target area. In order to improve the ASR of attack examples, a new loss function is designed to reduce the attention of objects in the target area while increasing the attention of non-target area.

In summary, the contributions of this paper are threefold.

- To the best of our knowledge, we are the first to design a multi-target categories patch-generation model for aerial detection, which can generate multiple patches at once. Each patch only attacks its corresponding category. Multi-patch training can reduce the detection bias for target categories without patches;
- We designed a new loss function that combined category loss and model attention to reduce attention regions while improving non-attention regions, and the united attack can achieve a higher attack success rate;
- In order to achieve the evaluation of multi-target adversarial patches, we have designed new metrics to measure their attack effectiveness and impact on the detection of other categories of targets.

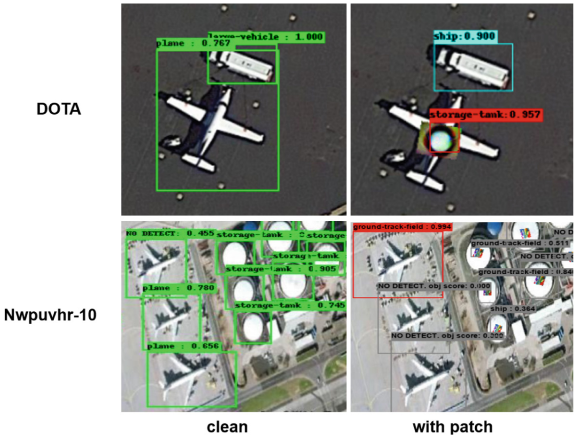


Fig. 1. Impact of universal patch on detection of surrounding objects. In DOTA, the plane with patch is incorrectly detected as a storage-tank, but the large vehicle without patch is also incorrectly detected as a ship. So the same in Nwpuvhr-10 dataset.

2 Related Work

2.1 Object Detection

Object detection is a prominent research focus in various computer vision tasks. The objective is to extract regions of interest from a given image and annotate the corresponding categories and positions. Currently, object detection based on DNN [15, 16] has superseded traditional methods that require manual feature extraction and classification with superior detection performance. Based on the existence of candidate boxes, CNN based object detection is divided into two-stage and single-stage detection. The two-stage detection network, such as Faster R-CNN [17], divides the detection of objects into two stages. Firstly, the position of the object is detected, and a region proposal network (RPN) is used to extract candidate boxes from images, followed by classification. A single-stage network does not require RPN, but directly completes classification and localization in one go. YOLO [18] is a typical representative of single stage, which is a grid-based object detection algorithm.

Due to the widespread adoption of Transformer [19] in the domain of natural language processing (NLP), it has also garnered significant attention and investigation in computer vision. In 2020, Carion et al. introduced a comprehensive object detection DETR [20] by integrating CNN with Transformer. Then the Deformable DETR [21] that leveraged adaptable CNN architectures was proposed in the subsequent year.

2.2 Adversarial Attack Against Object Detection

DNN-based object detection inherits the merits of neural networks, but it is susceptible to adversarial attacks, thereby engendering certain security risks in practical scenarios. Adversarial examples on object detectors involve adding specific perturbations to the clean image x to obtain the perturbed image x', with the aim of deceiving the object detector into generating random or erroneous results with targets. Based on the extent of alterations deployed to the image pixels, adversarial attacks can be categorized into global perturbation attacks and local perturbation attacks.

Global Perturbation Attacks. Global perturbation attack entails crafting adversarial examples by manipulating the pixels of the entire image. The added perturbations is not fixed to a single area, but is dispersed across the image. Lu et al. [22] first proposed adding noise to the "stop" sign to mislead detectors. The iterative method TOG [23] can attack both two-stage and single-stage object detectors simultaneously by reversing the training process, fixing network parameters, modifying the input image during each backpropagation, and generating adversarial examples through iteration.

Local Perturbation Attacks. Unlike global perturbation attacks requiring modification of all pixels, local perturbation attacks only add perturbations to a local specific area of the original image, thereby enabling the perturbations to impact the detection of entire image. The BPatch method [24] targets the local perturbation of two-stage detectors by adding perturbation blocks on the background outside the image target to attack the target detector. A visible adversarial example generation method based on YOLO network was designed by Thys et al. [12]. Their adversarial examples can make YOLO based detector unable to detect the presence of people.

Adversarial Attack Against Aerial Detection. Currently, adversarial attacks against aerial detection have attracted increasing attention and research. Adhikari et al. [25] conducted adversarial attacks on aircraft and discussed the impact of patches varying in size. Du et al. [26] studied adversarial attacks against remote sensing images in physical domain, optimized adversarial patches learning from Thys et al. [12], and developed a new patch method to generate three slender patches placed around the objects. Lian et al. [27] proposed a physical attack framework based on adaptive patches (AP-PA) and established a comprehensive and rigorous benchmark to evaluate the attack efficiency of adversarial patches.

Most of the current adversarial attack methods generate universal patches [14, 28–31] to attack all categories of targets, but we find that these methods do not have high attack success rate on every category. Therefore, patches with target categories specificity are needed to be effective only for the category they attack.

2.3 Attention Patterns in Computer Vision Tasks

In order to explain how deep neural networks predict the categories and positions of objects in images, and to make people intuitively feel where the network's focus is, David et al. first studied network visualization and proposed ZFnet [32] by improving AlexNet. Although different detection models have different backbones, when inferencing, the model usually focuses more on the target object regions rather than other irrelevant regions. Zhou et al. proposed CAM [33] to make neural networks visible, but it mandates the retraining of the model to obtain the weights obtained after global average pooling. To address the issues, Selvaraju et al. proposed Grad CAM [34], which used gradients in feature maps to calculate weights without retraining.

3 Methodology

In this paper, we proposed a novel multi-target distraction attack, which aims to fool the detector by generating different adversarial patches for different categories of targets at once. In this section, we will formulate the problem first, then introduce the MTDA framework. Finally, the design of multi-target patch application module, attention dispersion module and loss function will be introduced in detail.

3.1 Overview of Multi-target Problem

The purpose of multi-target adversarial attack is to induce targets with adversarial patch ignored or incorrectly detected, while targets without patches correctly detected by remote sensing detection model. Technically, we first initialize n random patches $p_i^0 \in \{p_1^0, p_2^0, \ldots, p_n^0\}$, and p_1^0 is only for targets such as aircraft, p_2^0 is only for targets of ships, etc. The number of patches n depends on the specific number of categories of the dataset. Given a clean aerial image x, we minimize the loss function $L(x_i^*, x^*)$ to optimize adversarial patches p_i^*, which can be represented as:

$$\underset{p_i^*}{\arg\min}\, L(x_i^*, x^*) \tag{1}$$

where adversarial examples x_i^* and x^* will be specified in Sect. 3.3. Compared with traditional mainstream attack methods, our method deeply studies the multi-target adversarial attack based on model attention according to the target characteristics of remote sensing images. In Sects. 3.2, 3.3 and 3.4, we will explain in detail how to train multi-target adversarial patches with strong adversarial attack capabilities.

3.2 Framework of MTDA

In order to generate highly robust adversarial patches with category specificity, we proposed a multi-target attention dispersion adversarial attack framework, as shown in Fig. 2. The specific attack steps are as follows:

Step1: We first randomly initialize multiple patches, and generate multiple masks following a sequence of angle and scale adaptive transformations. Then, we apply the adversarial patches to the clean image x, and use the multi-patch application module (Sect. 3.3) to correspond each patch to each category in the clean image one by one, forming two types of patches: multiple single target category-patch 1v1 adversarial examples and a multi-patch united adversarial example.

Step2: Next, adversarial examples will be sent to aerial detectors. Then, we extract category prediction probability from the detector as part of the class loss $\mathrm{L_{cls}}$ (Sect. 3.3), and minimize the $\mathrm{L_{cls}}$ to optimize adversarial patches.

Step3: In order to further enhance the attack intensity, we jointly attack the attention module of the model to extract the attention map and calculate the loss function L_{CAM} (Sect. 3.4) to distract the attention of the object detector by reducing the attention of the target area and increasing the attention of non-target areas.

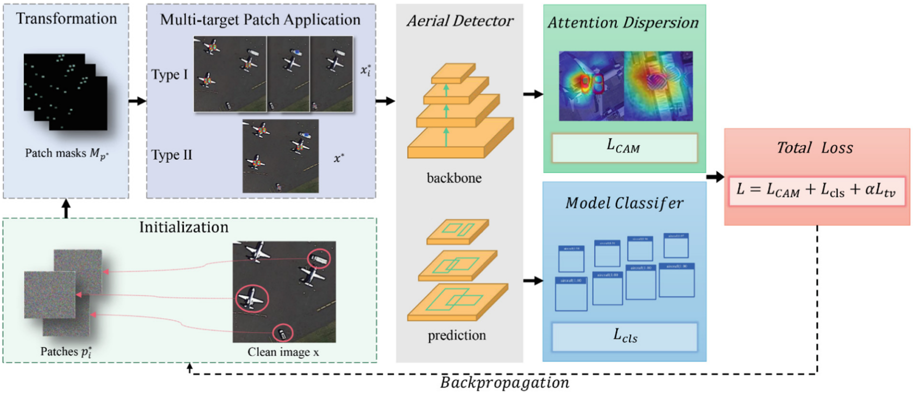


Fig. 2. Framework diagram of adversarial attacks with dispersed attention from multiple targets

3.3 Multi-target Patch Application

In order to make our adversarial patches more robust, we first perform various transformations PT$(\cdot)$ on each patch before optimizing the adversarial patch, including random rotation and scaling, so that each transformed patch can adapt to the target object in the image. Then, through an affine transformation matrix $\theta(\cdot)$, we generate the corresponding mask M_{p^*} and position category sequence number S_{id} for each patch.

$$M_{p_i^*}, S_{id} = PT\left(p_i^*, y\right) = \theta\left(p_i^*\right) \odot \theta(y) \tag{2}$$

where y is the label of each clean image, and n patches form a set of n masks $M = \{M_{p_i^*} | i \in \mathbb{N}\}$.

Next, in order to ensure that the masks of each patch correspond one-to-one with the target objects of each category, we input the transformed patch, mask, and position category sequence numbers along with the clean image into the patch application module PA$(\cdot)$. The adversarial examples obtained are as follows:

$$x^* = PA\left(x, p_i^*, M, S_{id}\right) = \left(1 - M_{p_i^*}\right) \odot x + M_{p_i^*} \odot PT\left(p_i^*\right) \tag{3}$$

We divide the generated adversarial examples into two types based on their location category sequence numbers S_{id}:

Type I applies corresponding patches only to one category of target objects in clean images, resulting in a set of adversarial examples:

$$x_i^* = PA\left(x, p_i^*, M_{p_i^*}, S_{id}\right) \tag{4}$$

Type II applies different patches to target objects of all categories in a clean image to obtain one adversarial example:

$$x^* = PA\left(x, p_i^*, M, S_{id}\right) \tag{5}$$

In order to ensure both the aggressiveness and category specificity of the adversarial patch, we feed both types of adversarial examples into the detection model. We take the category probability of the model's output result as the multi-target class loss, which is represented as follows:

$$L_{cls_i} = \frac{1}{s}\sum_{j=1}^{s} C_{i,j}\left(x_i^*, y_i\right) \tag{6}$$

$$L_{cls} = \sum_{i=1}^{n} L_{cls_i} + \frac{1}{S}\sum_{j=1}^{S} C_j\left(x^*\right) \tag{7}$$

where $\mathrm{C_{i,j}}(\cdot)$ is the category confidence of class i target j, $\mathrm{y_i}$ is the label corresponding to the patch, s is the number of target objects corresponding to each category, S is the number of all targets in an image, s ∈ S. For Type I adversarial examples, we only reduce the probability of the corresponding category of the patch, without paying attention to other categories to ensure category specificity of the adversarial patch. We take the average probability of the same category to obtain $\mathrm{L_{cls_i}}$. For Type II adversarial examples, we reduce the probability of all categories involved in the image, sum the two, and obtain the multi-objective category loss $\mathrm{L_{cls}}$ to ensure the patch's robustness.

3.4 Attention Module

With the popularity of interpretable models, more and more researchers are turning their attention to attacking models. However, in reality, the contextual information of remote sensing images is also an important factor affecting the classification of target objects by detection models. Therefore, we consider attacking the attention of the target area while dispersing the model's attention outside the target area.

In this article, we use Grad-CAM to extract the attention of the model, and input the adversarial example x^* generated in Sect. 3.3 into the attention map generator $G_{CAM}(x^*)$ of the target network to extract the model attention map AM:

$$AM = G_{CAM}\left(x^*\right) \tag{8}$$

Based on the attention attack algorithm in DAS [35], we calculate the attention loss function L_{CAM} to reduce the model's attention to the center of the target object, after obtaining the attention map. The attention loss function is as follows:

$$L_{CAM} = \frac{ReLU(AM)}{S_{bbox} - S_{msk}} \tag{9}$$

where, $ReLU(AM)$ is the total pixel value of the extracted model attention map that is greater than 0 after passing through the ReLU activation function, S_{bbox} is the area size of the detection box, and S_{msk} is the area size of the patch area. Since our patches are all applied to the center area of the target, the denominator in the formula is the target area excluding the patch. By minimizing L_{CAM}, the adversarial patches are optimized to minimize the total pixel value of the AM salient region, while increasing the attention of non-patched regions to distract the aerial detection model from the target center.

3.5 Total Loss Function

Based on the aforementioned modules, we will elucidate our total objective function.

In order to make the patterns and color shifts of adversarial patches p_i^* smoother, we use the common loss L_{tv} (total variational loss [36]):

$$L_{tv} = \sum_{i,j} \sqrt{\left(px_{i,j} - px_{i+1,j}\right)^2 + \left(px_{i,j} - px_{i,j+1}\right)^2} \tag{10}$$

where, $px_{i,j}$ is the pixel value at position (i, j) of each p_i^*.

In summary, we define the total loss as:

$$L = L_{cls} + L_{CAM} + \alpha L_{tv} \tag{11}$$

where, α is the hyperparameter. We optimize p_i^* by Adam [37] optimization algorithm. Our goal is to update the pixels of p_i^* through gradient backpropagation, so that all the weights of the aerial detector will be frozen during the patch optimization.

The specific adversarial example generation algorithm is shown in Algorithm 1.

Algorithm 1 MTDA

```
Input: Clean images x, labels y, detector confidence extraction $C(\cdot)$, original patch $p_i^0$,
Grad-CAM generator $G_{CAM}(\cdot)$, hyperparameters $\alpha$.
Output: Adversarial patches $p_i^*$ .
Initialize $p_i^0$ randomly in [0, 255], $p_i^* = p_i^0$;
for e in epochs do
    $M_{p_i^*}, S_{id} = PT(p_i^*, y)$ # Patch transformation
    Type I: $x_i^* = PA(x, p_i^*, M_{p_i^*}, S_{id})$ # Generate two types of patches
    Type II: $x^* = PA(x, p_i^*, M, S_{id})$
    $L_{cls} = C(x^*) + C_i(x_i^*)$ # Extract Classification confidence
    $L_{CAM} \leftarrow AM = G_{CAM}(x^*)$ # Extract model attention loss
    $L = L_{cls} + L_{CAM} + \alpha L_{tv}$ # Total loss
    Total loss back propagation, update patches
end for
return $p_i^*$
```

4 Experiments

In this section, we conducted comprehensive experiments to test the attack effectiveness of the proposed MTDA algorithm. We first described the experimental setup in detail in Sect. 4.1, and then conducted experiments on multi-patch joint attention attacks and ablation in Sects. 4.2 and 4.3, respectively.

4.1 Settings

Dataset. We use DOTA [38] as the dataset for our experiment. Due to limitations in experimental equipment performance, we selected 600 images for training and 200 images for testing. Moreover, with the intent of verifying the universality of the experiment on multiple datasets, the test set of this experiment also includes the NWPUVHR-10 [39] besides DOTA, which is also a commonly used dataset in the field of aerial detection. The detailed information of the two datasets is shown in Table 1.

Table 1. Detailed Description of DOTA and NWPUVHR-10 Datasets

Datasets	Categories	Images	Instances	Image width
DOTA	15	2806	118282	800 ~ 4000
NWPUVHR-10	10	800	3651	~ 1000

Target Models. Our experiment selected the first stage object detection model (YOLOv2 [40], YOLOv3 [41]), the second stage object detection model (Faster R-CNN [42]), and the transformer-based detector (Swin-t [43]) as the target models. These detection models were all aerial detection models trained on the DOTA dataset. We conducted multi-patch training on YOLOv3 and tested them on multiple detectors to validate the universality of the proposed attack methods.

Metrics. The effectiveness of adversarial patches is evaluated using two metrics: attack success rate (ASR) and other categories detection accuracy (OCAP). The formula for calculating the success rate of attacks on target objects with patches is as follows:

$$ASR = \sum_{i \in I} \frac{N_{MT}^{i} + N_{WT}^{i}}{N_{AT}^{i}}$$

where, I is the artificially selected categories in all categories C of DOTA, $I \in C$. N_{MT}^{i} represents the number of missed targets of category i, that is, how many targets were overlooked by the detector. N_{WT}^{i} represents the number of wrong-detected targets of category i, that is, how many targets were misclassified by the detector. N_{AT}^{i} represents the number of all targets of category i.

For target objects without patches, the calculation formula for OCAP (detection accuracy for other categories) is as follows:

$$\text{OCAP} = \sum_{j \in J} \frac{N_{CT}^{j}}{N_{AT}^{j}}$$

where, J represents several categories in category C of the DOTA dataset that have not been selected for applying patch, i.e. $J \in \{C | I \cap J = \emptyset\}$. N_{CT}^{j} is the number of correctly detected objects in category j, and N_{AT}^{j} is the number of all targets in category j.

In this experiment, the larger the ASR, the better the attack effect. And in the case of a larger ASR, the higher the OCAP, indicating that our patch has a smaller impact on the detection of other categories. (OCAP is not necessarily the bigger the better.)

Experimental Environments. We refer to the settings in [12] and set α in formula (11) to 2.5. In this article, the experiment was conducted using the NVIDIA GeForce RTX3090 graphics processing unit on PyTorch.

4.2 Multi-patch Distraction Experiment

For the convenience of demonstrating the experimental results, we select small vehicle, large vehicle, and aircraft in the DOTA dataset as multi-category attack targets, and generated three adversarial patches simultaneously for each of these three categories as shown in the Fig. 3. Then we test each patch separately on YOLOv3, and the detection results obtained are shown in Table 2.

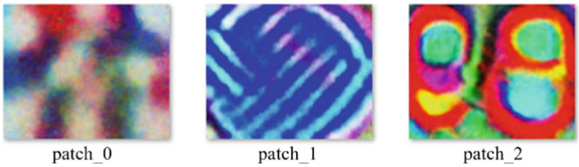

patch_0 patch_1 patch_2

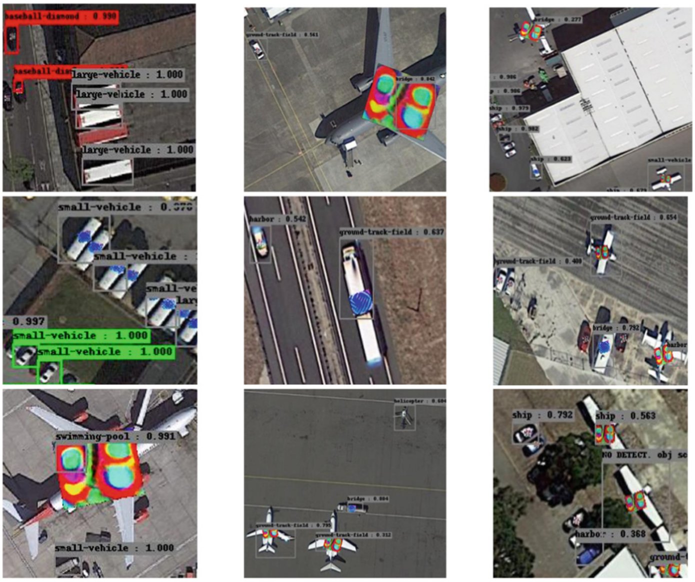
Fig. 3. Multi-target training patches

Table 2. ASR of Each Patch on the DATA Dataset

Patch	ASR(↑)	OCAP
patch_0	0.3120	0.7815
patch_1	0.9139	0.3369
patch_2	0.9655	0.3571
patch_all	0.9928	0.3730

From Table 2, except for the first patch(patch_0), the ASR of each patch reached over 90%, and the ASR of three patches simultaneously(patch_all) even reached 99.28%. Since we only experiment on aircrafts, small vehicles and large vehicles, and we hope not affecting the detection of other categories, our patch needs to ensure the high ASR, while ensuring that the targets of other categories can be accurately recognized. Therefore, the OCAP is not necessarily the higher the better or the lower the better.

The reason for the low ASR of the first patch may be the size of small vehicles is smaller than that of large vehicles and aircrafts, causing patch_0 to lose more high-frequency information in pixels when shrinking. Since the final loss function is a fusion of multiple objective functions, the Adam optimizer will have a bias towards an objective function that is easy to detect, so the pixel optimization for the first patch is not as good as the last two. But the OCAP of patch_0 is the highest, which also reflects the high detection accuracy of the yolov3 model we initially trained. However, in the case where the ASR of the latter two patches is extremely good, there is a certain decrease in the AP of other categories. It indicates that the patch we designed may still have a certain impact on the detection of other categories. However, comparing patch_1 and patch_2, patch_2 has a higher ASR than patch_1, and its OCAP is also higher than patch_1. This indicates that patch_2 has a reduced impact on the detection of targets without patches of other categories. Qualitative detection results are shown in Fig. 4.



one-patch apply two-patch apply three-patch apply

Fig. 4. Results of patch detection for each patch. The first column shows only one patch applied, the second column shows only two patches applied, and the third column shows the detection results of all three patches applied to the corresponding targets.

The targets with the patches we trained are not detected or misclassified by the detector, while the other targets without patches can be correctly detected. For example, in the last row of the two-patch column, when patching aircrafts and large vehicles, they are misclassified as playgrounds and bridges respectively, but the helicopter nearby can still be detected correctly.

We compared our method with randomly generated patches, the baseline method [26], and BA-AP [27], all of which were trained on YOLOv3 and tested on different detection models in Table 3 and Table 4.

From Table 3 and Table 4, our attack method outperforms other methods in terms of ASR and AP. Our patch trained on YOLOv3 has an over 96% ASR on CNN based detection models. Comparing the OCAP of Yolov2, Yolov3, and Faster RCNN, our method is effective in enhancing the detection precision of other categories without patches compared to the baseline method. This also indicates that our proposed method

Table 3. ASR and OCAP of different methods on different detection models

Method	Yolov2		Yolov3		Faster-RCNN		Swin-t	
	ASR(↑)	OCAP	ASR(↑)	OCAP	ASR(↑)	OCAP	ASR(↑)	OCAP
Random	0.2667	0.9419	0.1726	0.9682	0.1993	0.8142	0.0831	0.8905
Baseline [26]	0.9922	0.0753	0.7389	0.2924	0.5028	0.3023	0.3260	0.8322
Ours	**0.9967**	**0.1183**	**0.9928**	**0.3730**	**0.9671**	**0.3329**	**0.4008**	0.7161

Table 4. AP of different methods on different detection models

Method	Yolov2	Yolov3	Faster-RCNN	Swin-t
	AP(↓)	AP(↓)	AP(↓)	AP(↓)
Random	0.8950	0.8273	0.9033	0.8901
Baseline [26]	0.3247	0.5023	0.6550	0.8526
BA-AP [27]	0.0633	0.5924	0.3535	0.8278
Ours	**0.0032**	**0.3226**	**0.3367**	**0.8209**

can reduce the impact of patched category targets on the detection accuracy of other categories without patches to some extent.

Next, in order to validate the effectiveness of the multi-target patches we trained on other datasets, we conducted tests on the NWPUVHR-10. Since NWPUVHR-10 does not have large and small vehicles categories in DOTA, we selected aircraft, ships and storage tanks that both had in common as attack targets. Three patches were retrained. Meanwhile, patches for every 100 epochs (0, 100, 200, 300, 400, 500, 600) were selected from the optimization process for testing.

Table 5. Comparison of attack effects of multiple patches on different datasets

Epoch	DOTA		NWPUVHR-10	
	ASR(↑)	OCAP	ASR(↑)	OCAP
0	0	0.7315	0	0.4137
100	0.5561	0.6648	0.5846	0.4855
200	0.7772	0.6557	0.6183	0.4482
300	0.9054	0.3677	0.8623	0.4137
400	0.9079	0.3659	**0.8680**	0.4137
500	**0.9095**	0.3659	**0.8680**	0.4137
600	**0.9095**	0.3659	**0.8680**	0.4137

As shown in Table 5, our patch converges quickly during the optimization process, with a high ASR after 400–500 rounds of training. This is mainly because part of our class loss function is to use the average class probability of all detected objects, rather than just using the maximum objective score. This not only ensures our attack performance, but also improves the optimization efficiency of adversarial patches. At the same time, our attack method can still maintain a high ASR on other datasets.

On the basis of 94.40% mAP of yolov3 we used, OCAP usually decreases with higher attack performance. But on NWPUVHR-10 dataset, OCAP first increases and then decreases, and finally maintains a stable trend (Fig. 5). The initial low OCAP is due to the fact that the NWPUVHR-10 lacks data on multiple categories of targets in one image, resulting in insufficient data volume. Secondly, the most targets in NWPUVHR-10 are smaller and more blurry than that in DOTA, making detection difficult. But with the continuous optimization of the patch, OCAP rises, indicating that our training patch initially had a trend of improving the detection accuracy of other categories without patches. However, as patch optimization continues and attack performance continues to improve, the impact of patches on detecting other categories still exists. Nonetheless, since the final result is close to the initial value, it can be concluded that the impact of our patch on other categories is not very large.

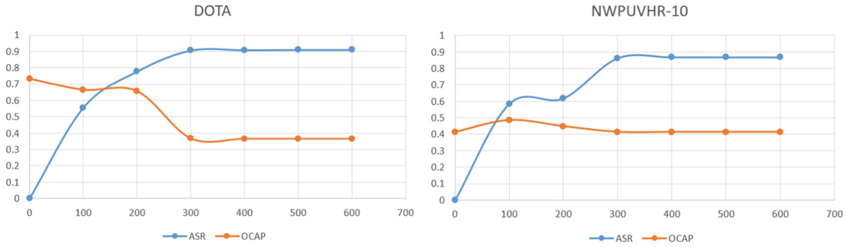


Fig. 5. ASR and OCAP of multiple patches on different datasets

4.3 Ablation Study

We also carried out ablation experiments on each loss term in the total loss function on the YOLOv3 detection model to illustrate the efficacy of MTDA. In the experiment, we used the class loss proposed in the baseline method [26], our class loss L_{cls}, our attention loss L_{CAM}, and the combination of them as the loss function. We trained one patch at a time and conducted comparative experiments.

As shown in the Table 6, our class loss function greatly improves the attack performance, compared to the class loss function in baseline methods. And the attention loss can also improve the ASR of patches. Compared to combining L_{CAM} with the L_{cls} of the baseline method, combining our class loss with L_{CAM} has a higher ASR.

Some attention map examples after patching of YOLOv3 are shown in Fig. 6. After applying patches, the detector's attention is dispersed, making it more focused on the surroundings of the object.

Table 6. Loss function ablation experiment

Loss Function	ASR(↑)
L_{cls} in Baseline	0.3389
L_{cls}	0.8992
L_{CAM}	0.4047
L_{cls}in Baseline + L_{CAM}	0.3714
$L_{cls} + L_{CAM}$ (*ours*)	0.9655

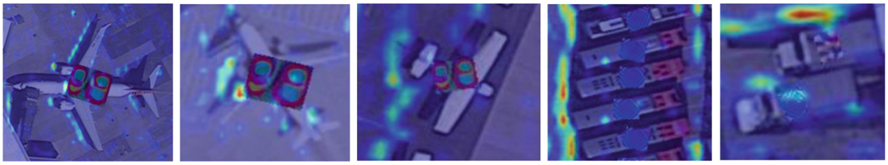


Fig. 6. Attention Map examples of YOLOv3

5 Conclusion

In this article, we proposed a Multi Target Distraction Attack (MTDA) method, which can generate multiple different patches simultaneously for different categories to address the issue of poor effectiveness of universal patches generated by traditional methods against certain category targets. By applying the patches to the corresponding category targets, we limit the attack area to the target area. Our method generates two types of adversarial patches to obtain the final class loss. And we incorporate the attention module and design an attention loss that disperses the attention of the detector from the center area of the target object to other irrelevant areas. Our experiments show that MTDA achieves an ASR of 99.28% on YOLOv3, and the multi-target design to some extent reduces the impact of patched objects on the detection of other unpatched objects. Our method solves the problem that the attack effect of the universal patch on some target objects is poor, and provides a more efficient method for targeted attack.

However, our method still has some shortcomings, and the target specificity of multi-target patches can be further strengthened. Our future research efforts involve verifying the generalizability of our method on more datasets and improving the concealment of adversarial patches and the attack efficiency against more powerful object detectors such as YOLOv5 and Swin Transformer.

Acknowledgment. This work is partially supported by Key R&D Program (2021YFF0602104-2), and partially supported by Qing Lan Project.

References

1. Zhu, X., et al.: Deep learning in remote sensing: a comprehensive review and list of resources. IEEE Geosci. Remote Sens. Mag. **5**(4), 8–36 (2017)
2. Szegedy, C., et al.: Intriguing properties of neural networks. CoRR; abs/1312.6199 (2014)
3. Lu, J., Sibai, H., Fabry, E.: Adversarial Examples that Fool Detectors (2017)
4. Xie, C., Wang, J., Zhang, Z., et al.: Adversarial examples for semantic segmentation and object detection, pp. 1378–1387. IEEE Computer Society (2017
5. Chen, S.T., Cornelius, C., Martin, J., et al.: ShapeShifter: robust physical adversarial attack on faster R-CNN object detector. In: European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (2019)
6. Li, Y., Bian, X., Lyu, S.: Attacking object detectors via imperceptible patches on background (2018)
7. Liao, Q., Wang, X., Kong, B., et al.: Category-wise attack: transferable adversarial examples for anchor free object detection (2020)
8. Zhang, H., Zhou, W., Li, H.: Contextual adversarial attacks for object detection. In: IEEE International Conference on Multimedia and Expo (ICME). IEEE (2020)
9. Wei, X., Liang, S., Chen, N., et al.: Transferable adversarial attacks for image and video object detection (2018)
10. Wu, X., Huang, L., Gao, C.: G-UAP: generic universal adversarial perturbation that fools RPN-based detectors. In: Asian Conference on Machine Learning. PMLR (2019)
11. Liu, X., Yang, H., Liu, Z., et al.: DPATCH: an adversarial patch attack on object detectors. In: National Conference on Artificial Intelligence (2019)
12. Thys, S., Ranst, W.V., Goedeme, T.: Fooling automated surveillance cameras: adversarial patches to attack person detection. In: Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 49–55, June 2019
13. Wang, Y., Lv, H., Kuang, X., et al.: Towards a physical-world adversarial patch for blinding object detection models. Inf. Sci. **556**(1), 459–471 (2020)
14. Lee, M., Kolter, Z.: On Physical Adversarial Patches for Object Detection (2019)
15. Liu, L., et al.: Deep learning for generic object detection: a survey. Int. J. Comput. Vision **128**(2), 261–318 (2020). https://doi.org/10.1007/s11263-019-01247-4
16. Ding, S., Zhao, K.: Research on daily objects detection based on deep neural network. IOP Conf. Ser. Mater. Sci. Eng. **322**(6), 062024 (2018)
17. Ren, S.Q., He, K.M., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Proceedings of the 28th International Conference on Neural Information Processing Systems, Montreal, Canada, pp. 91–99. MIT Press (2015)
18. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, USA, pp. 779–788. IEEE (2016). https://doi.org/10.1109/CVPR.2016.91
19. Vaswani, A., Shazeer, N., Parmar, N., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, Long Beach, pp. 5998–6008 (2017)
20. Carion, N., Massa, F., Synnaeve, G., et al.: End-to-End Object Detection with Transformers. Springer, Cham (2020)
21. Zhu, X., Su, W., Lu, L., et al.: Deformable DETR: deformable transformers for end-to-end object detection. In: International Conference on Learning Representations (2021)
22. Lu, J., Sibai, H., Fabry, E.: Adversarial examples that fool detectors (2017). https://doi.org/10.48550/arXiv.1712.02494
23. Chow, K.H., Liu, L., Gursoy, M.E., et al.: TOG: targeted adversarial objectness gradient attacks on real-time object detection systems (2020). https://doi.org/10.1109/TPS-ISA50397.2020.00042

24. Li, Y., Bian, X., Lyu, S.: Attacking object detectors via imperceptible patches on background (2018).https://doi.org/10.48550/arXiv.1809.05966
25. Adhikari, A., Hollander, R., Tolios, I., et al.: Adversarial patch camouflage against aerial detection. In: Artificial intelligence and Machine Learning in Defense Applications II, 20 September 2020, vol. 11543, pp. 77–86. SPIE (2020)
26. Du, A., Chen, B., Chin, T.J., et al.: Physical adversarial attacks on an aerial imagery object detector. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 1796–1806 (2022)
27. Lian, J., Mei, S., Zhang, S., et al.: Benchmarking adversarial patch against aerial detection. IEEE Trans. Geosci. Remote Sens. **60**, 1–16 (2022)
28. Song, D., et al.: Physical adversarial examples for object detectors. In: USENIX Workshop on Offensive Technologies. USENIX Association (2018)
29. Wu, Z., Lim, S., Davis, L.S., Goldstein, T.: Making an invisibility cloak: real world adversarial attacks on object detectors. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M. (eds.) Computer Vision – ECCV 2020. ECCV 2020. LNCS, vol. 12349. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58548-8_1
30. Pavlitskaya, S., Hendl, J., Kleim, S., et al.: Suppress with a patch: Revisiting universal adversarial patch attacks against object detection. In: 2022 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), pp. 1–6. IEEE (2022)
31. Xu, Y., Wang, J., Li, Y., et al.: Universal physical adversarial attack via background image. In: Zhou, J., et al. (eds.) International Conference on Applied Cryptography and Network Security. Springer, Cham (2022).https://doi.org/10.1007/978-3-031-16815-4_1
32. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: Part, I. (ed.) Computer Vision-ECCV 2014–13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, pp. 818–833. Springer (2014)
33. Zhou, B., Khosla, A., Lapedriza, À., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, 27–30 June 2016, pp. 2921–2929. IEEE Computer Society (2016)
34. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad–CAM: visual explanations from deep networks via gradient-based localization. In: IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, 22–29 October 2017, pp. 618–626. IEEE Computer Society (2017)
35. Wang, J., Liu, A., Yin, Z., Liu, S., Tang, S., Liu, X.: Dual attention suppression attack: generate adversarial camouflage in physical world. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, Virtual, 19–25 June 2021, pp. 8565–8574. Computer Vision Foundation/IEEE (2021)
36. Chambolle, A., Caselles, V., Cremers, D., Novaga, M., Pock, T.: An Introduction to Total Variation for Image Analysis. De Gruyter, pp. 263–340 (2010)
37. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: Proceedings of International Conference on Learning Representation (2015)
38. Xia, G.S., et al.: DOTA: a large-scale dataset for object detection in aerial images. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3974–3983 (2018)
39. Cheng, G., Zhou, P., Han, J.: Learning rotation-invariant convolutional neural networks for object detection in VHR optical remote sensing images. IEEE Trans. Geosci. Remote Sens. **54**, 7405–7415 (2016). Cheng, G., Zhou, P., Han, J.: RIFD-C 2016b
40. Redmon, J., Farhadi, A.: YOLO9000: better, faster, stronger. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017, pp. 7263–7271 (2017)

41. Redmon, J., Farhadi, A.: YOLOv3: an incremental improvement. In: Computer Vision and Pattern Recognition, vol. 1804. Springer, Heidelberg (2018)
42. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real time object detection with region proposal networks. In: Proceedings of Advanced Neural Information Processing System, vol. 28 (2015)
43. Liu, Z., et al.: Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of IEEE/CVF International Conference on Computer Vision (ICCV), October 2021, pp. 10012–10022 (2021)

Mask-TS Net: Mask Temperature Scaling Uncertainty Calibration for Polyp Segmentation

Yudian Zhang, Chenhao Xu, Kaiye Xu, and Haijiang Zhu(✉)

CIST, Beijing University of Chemical Technology, Beijing 100029, China
`{zyd,xuch,xuky}@buct.edu.cn`, `zhuhj@mail.buct.edu.cn`

Abstract. Lots of popular calibration methods in medical images focus on classification, but there are few comparable studies on semantic segmentation. In polyp segmentation of medical images, we find the most diseased area only occupies a small portion of the entire image, resulting in previous models not well-calibrated for lesion regions but well-calibrated for background, despite their overall seemingly better Expected Calibration Error (ECE) scores. Therefore, we proposed the four-branches calibration network with Mask-Loss and Mask-TS strategies to focus more on the scaling of logits within potential lesion regions, which serves to mitigate the influence of background interference. In the experiments, we compare the existing calibration methods with the proposed Mask Temperature Scaling (Mask-TS). The results indicate that the proposed calibration network outperforms other methods both qualitatively and quantitatively.

Keywords: Uncertainty estimation · Probability calibration · Binary segmentation

1 Introduction

With the rapid development of deep learning technology, neural networks are increasingly widely used in the field of medical image processing. Despite the promise of deep learning in medical imaging, concerns over reliability and trust impede their implementation in real-world clinical contexts, where any misjudgment will bring great risks to the doctor's diagnostic process and the patient's treatment. Ideal semantic segmentation networks should not merely excel in precision, but they should also possess the capability to indicate when and where their predictions may be unreliable or prone to error. If the segmentation network shows high uncertainty of its predictions, a medical expert is needed to double-check such doubtful regions. In order to achieve this goal, uncertainty estimation is essential.

Y. Zhang and C. Xu—Contributed equally to this work and should be considered co-first authors.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 180–194, 2025.
https://doi.org/10.1007/978-3-031-78128-5_12

However, the semantic segmentation model often outputs overconfident probabilities because of overfitting [1,8,10], which may mislead the final decision of doctors. To solve this, lots of calibration techniques [7,30] are proposed including regularization methods like data augmentation [19,25] and uncertainty estimation approaches like ensemble of neural networks [12,14]. But, these approaches can potentially compromise the segmentation model's accuracy. So, our focus lies in post-hoc calibration methodologies, which build an independent post-processing model separated from the prediction model. This approach can improve probabilistic calibration while keeping the model's original prediction unchanged.

What is more, the majority of existing post-hoc probability calibration methods are originally designed for image classification and typically yield a solitary class probability per entire image [8,20,28] while for segmentation pixel-wised probabilities of each pixel are called for. So these methods cannot be directly applied to medical image segmentation. Although there are some methods for semantic segmentation, the difference between background and target in the calibration process is ignored.

Our goal is to develop a post-hoc calibration method for two-label medical image semantic segmentation, which pays attention to the distinction between background and target in calibration process and retains the model's segmentation accuracy. Then, an intuitive uncertainty map with highly practical application value for clinical contexts is given.

Here are our specific contributions:

- The prediction-based mask for temperature scaling is proposed to finely select the region of interest, which enables precise calibration of the probabilities associated with the lesion area.
- For segmentation task, pixel-wised probabilities of each pixel are produced and spatial relations of pixels are concerned via our calibration networks.
- The probability calibration of two-label semantic segmentation problem is realized without compromising the original prediction accuracy.

2 Related Work

In recent years, many calibration methods have been proposed, especially in post-hoc calibration (Table 1). The most significant feature of the post-calibration method is that it makes the output probability closer to the true probability distribution without changing the prediction results of the original network, which enhances the effectiveness of the model.

Among them, Platt scaling (PS) [20] uses logistic regression for probability calibration and combines Support Vector Machine with Sigmoid to get a better regularized maximum likelihood estimation. However, because it contains a series of post-correction, it may fail when processing some independent and identically distributed (i.i.d) data for data-generating distributions [9].

As its extension, one of the most basic but effective methods in the field of classification is temperature scaling (TS) [8]. It divides the logits by a constant

T so that the output of the model is calibrated. However, it only uses one parameter T, which still has great limitations and it is not flexible enough for spatial distributions of probability.

Another extension of TS is ensemble temperature scaling (ETS) [28]. It improves calibration effect by considering different calibration methods into account. It ensembles uncalibrated vector, a maximum entropy logits vector and the temperature-scaled vector. As a consequence, it inevitably results in an increase in the cost of calculations.

Although PS, TS and ETS perform well in classification tasks, for semantic segmentation tasks, the positional relationship between pixels needs to be taken into account and probability calibration for the each pixels is needed.

For the segmentation problem, local temperature scaling (LTS) [5] takes the spatial information into account and considers that the parameter T of each pixel varies based on the relative positions. But it ignores the difference between background and target. As a result, it is not well-calibrated for lesion regions but well-calibrated for background, despite their seemingly better ECE scores.

Table 1. Comparison of Different Calibration Methods.

Methods	Mainly Application	Features	Disadvantages
PS	classification	combines Support Vector Machine and Sigmoid	fails to process i.i.d data for data-generating distributions and scaling parameter is globally fixed
TS	classification	divides the logits by a constant T, simplicity and effective	globally fixed parameter and not flexible for different distributions
ETS	classification	ensemble uncalibrated, maximum entropy and scaled vectors	globally fixed parameter and increases the cost of calculation
LTS	segmentation	takes the spatial information into account	not well-calibrated for small target area
Improved TS	segmentation	combines Shape prior module and Aleatoric-by-augmentation module	adapted high calculation consuming module

Improved TS [18], is proposed to reduce the calibration error when confronted with out-of-domain images. The model combines Shape prior module and Aleatoric-by-augmentation module. It leverages the pixel susceptibility against perturbations at the local level and the shape prior information at the global level. The shape prior and shape residual highlight potentially unreliable predictions. But the Shape prior module increases calculation costs.

Factually, by leveraging our Mask-Loss and Mask-TS strategies, these methods can effectively enhance their focus on regions of interest and refine the calibration of associated probabilities, thereby maximizing their overall effectiveness.

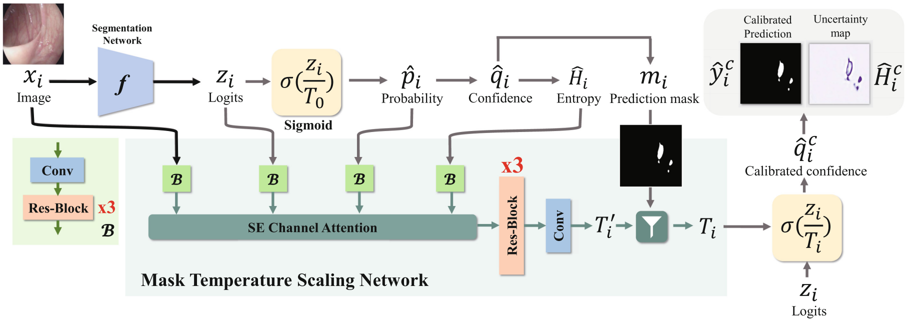


Fig. 1. Structure of Mask-TS Network.

3 Calibration Network

The existing probabilistic calibration methods, such as Platt scaling, TS, and ETS primarily focus on classification tasks rather than semantic segmentation tasks. Considering the spatial relations of pixels in an image, image segmentation should not be simply treated as classification. We therefore design a Mask-TS calibration network that performs different pre-temperature scaling on each pixel via convolutional networks. And then, a mask based on prediction is applied to the pre-temperature scaled parameters in order to pay more attention to the potential diseased area. Here are the details:

3.1 Four Branches Calibration Network

The predictive confidence of the two labels $\hat{q}_i$ is calculated by Eq. (1) because our task is two-label segmentation. The original probability $\hat{p}_i$ is the original Sigmoid output. The mask m_i based on prediction, namely the original prediction $\hat{y}_i$, is similarly obtained like Eq. (2) where $\hat{q}_i^c$ is the calibrated confidence and $\hat{y}_i^c$ is the calibrated prediction.

$$\hat{q}_i = \begin{cases} \hat{p}_i & \text{, if } \hat{p}_i \geq 0.5 \\ 1 - \hat{p}_i & \text{, otherwise} \end{cases} \tag{1}$$

$$\hat{y}_i^c = \begin{cases} 1 & \text{, if } \hat{q}_i^c \geq 0.5 \\ 0 & \text{, otherwise} \end{cases} \tag{2}$$

As shown in Fig. 1, the original image x_i, model output logits z_i, probability map $\hat{p}_i$ and uncertainty map $\hat{H}_i$ are separately input into network block $\mathcal{B}$

composed of convolution and residual neural networks, which means taking the spatial relationship between pixels into consideration. Then information from the four branches are adaptively weighted and combined under the action of a channel attention layer.

Our work is inspired by temperature scaling (TS) [8] for classification probability calibration, and improved TS [18]. However, the key component referred to as the shape prior network in improved TS, which requires higher training costs for a denoising autoencoder, is not utilized in our work. To avoid this, we creatively input $\hat{H}_i$ and $\hat{p}_i$ into proposed calibration network. We empirically found that our method yields comparable results to the improved TS. These two branches respectively play a role in enhancing the edge shape and treating the predicted truth value more carefully while x_i contains rich original information and z_i possesses segmentation-related features because it originates from a segmentation network.

We train the post-hoc calibration network using the binary cross entropy loss Eq. (3) for two-label image segmentation. Uncertainty map is used to quantify the uncertainty of predictive segmentation using the aleatoric uncertainty [11,13] for each pixel. For two-label segmentation task, this is measured by the entropy of the confidence Eq. (4).

$$\mathcal{L}_c = \frac{-1}{H \cdot W} \sum_{u,v=1,1}^{H,W} \left\{ y_i(u,v) \cdot \log[\sigma(\frac{z_i(u,v)}{T_i(u,v)})] + [1 - y_i(u,v)] \cdot \log[1 - \sigma(\frac{z_i(u,v)}{T_i(u,v)})] \right\} \tag{3}$$

$$\hat{H}_i = \mathcal{H}(\hat{q}_i) = -[\hat{q}_i \log_2(\hat{q}_i) + (1 - \hat{q}_i)\log_2(1 - \hat{q}_i)] \tag{4}$$

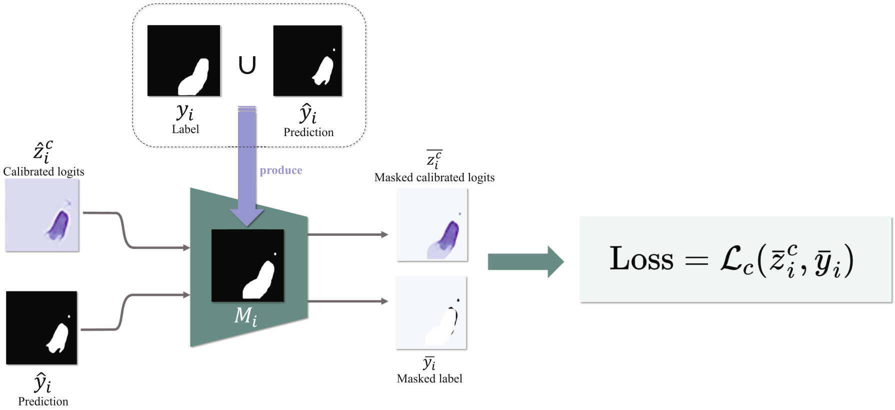


Fig. 2. Workflow of the proposed Mask-Loss method.

3.2 Mask-Loss

In practice, doctors focus more on positive predictions that may indicate potentially diseased area. Nevertheless, we find that most diseased area occupies only a small portion of the entire image, resulting the in less important predictions of background dominating the calculation of ECE. It also leads to previous models not well-calibrated for the actual lesion regions, despite their better ECE scores. This is detailed in Sect. 4.2.

So Mask-Loss strategy is proposed during the training stage. Specifically, as shown in Eqs. (5) (6) and (7), when computing BCE-Loss, we first perform a logical OR operation on the positive labels and the positive predictions to obtain the mask M_i. Then the masked and calibrated logits $\bar{z}_i^c$ and the masked label $\bar{y}_i$, both processed through M_i, are used to calculate BCE-Loss $\mathcal{L}_c$ using Eq. (3). Figure 2 also shows the workflow of Mask-Loss and it should be noted that the light blue area, seen as background, in $\bar{z}_i^c$ and $\bar{y}_i$ are excluded from the loss calculation. Thus, the background interference is weakened while positive predictions are more focused and concerned.

$$M_i = (y_i == 1)\text{or}(\hat{y}_i == 1) \tag{5}$$

$$\bar{z}_i^c = z_i^c \odot M_i = (\frac{z_i}{T_i}) \odot M_i \tag{6}$$

$$\bar{y}_i = y_i \odot M_i \tag{7}$$

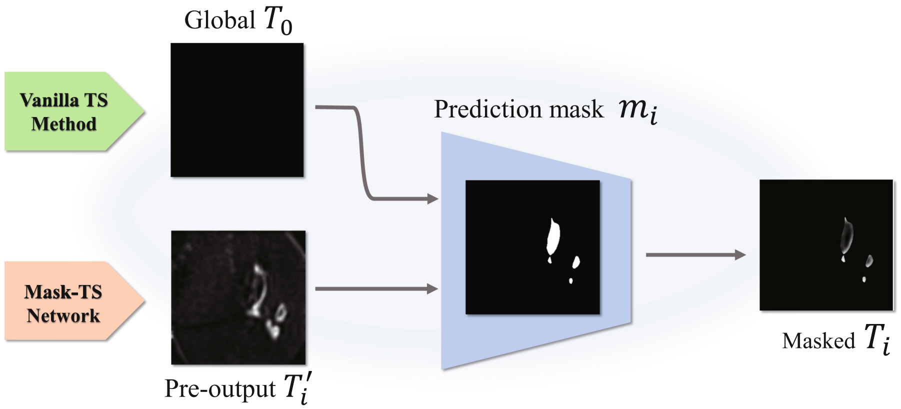


Fig. 3. Architecture for Mask-TS.

3.3 Mask-TS

To avoid the lack of constraints on the background pixels from causing an increase of the temperature parameters and pay attention to the calibration of potential lesion areas, we proposed the prediction-based Mask-TS method.

Through Mask-TS, for predictions classified as background, the corresponding temperature parameter T_i is set as the global temperature parameter T_0 produced by vanilla TS, while for predictions classified as lesion areas, T_i remains as the pre-output temperature parameter T_i' of the calibration network. Equation (8) shows how to obtain a masked temperature map T_i with the mathematical expression. Figure 3 shows the architecture for Mask-TS. It illustrates that T_i' may miscalibrate the background areas due to the lack of constraints caused by Mask-Loss and the masked T_i focus more on potential lesion areas while eliminating the background miscalibration.

Through practice, it is found that the final prediction of the model remains unchanged even z_i is scaled in different forms. The reason is that when we use Eqs. (1) and (2), the area where $\hat{p}_i$ is greater than 0.5 still maintain its original confidence value above 0.5 even after different scaling operations on z_i. Thus, the prediction results are unchanged, creating the conditions for confidence calibration without changing the original prediction results.

$$T_i = \begin{cases} T_i' & \text{, if } \hat{y}_i = 1 \\ T_0 & \text{, otherwise} \end{cases} \tag{8}$$

Table 2. Dataset allocation.

Dataset	Image Num	Segmentation Net	Calibration Net
CVC-ClinicDB [2]	550	Train	–
Kvasir [21]	900	Train	–
CVC-ColonDB [24]	380	–	Train
CVC-300 [26]	60	–	Validate
ETIS-LaribPolypDB [22]	196	–	Test
Piccolo [23]	333	–	Test

4 Experiment

We show the calibration performance of proposed method cooperating with four different segmentation models on two different datasets and also show the uncertainty maps that have highly practical application value. We compare our Mask-TS with baseline methods such as TS, ETS and LTS and also conduct the ablation studies based on four branches of our calibration network. As shown in

Table 2, for calibration part, CVC-ColonDB [24] is the training dataset, CVC-300 [26] is the validation dataset and ETIS-LaribPolypDB [22] along with Piccolo [23] are the test datasets. These datasets are public except for Piccolo that was provided by the Basque Biobank www.biobancovasco.org.

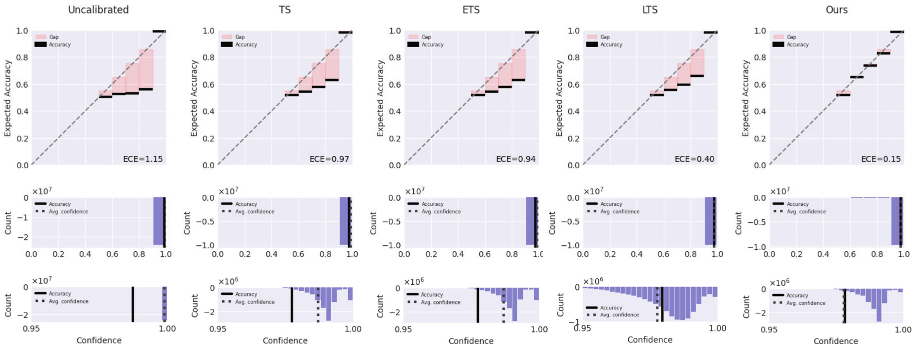


Fig. 4. Reliability diagrams and confidence histograms of comparative experiment on ETIS-LaribPolypDB. (Color figure online)

4.1 Evaluation Metrics

To measure the performance of probability calibration, we use the following metrics. Qualitatively, uncertainty maps compared with error maps are used to visually evaluate the calibration results. Quantitatively, the commonly used statistic metrics are chosen. They are reliability diagram [4,16], expected calibration error [15] (ECE), maximum calibration error [15] (MCE), static calibration error [17] (SCE) and adaptive calibration error [17] (ACE) are used. These metrics, originally devised for classification, are adjusted for semantic segmentation by treating each pixel's confidence as a separate sample. Computations are performed across 10 evenly-spaced bins.

When calculating quantitative metrics, local evaluation regions are used for testing. In practical applications, this helps to exclude the influence of less important backgrounds and focus more on the calibration performance of potential lesion areas. In practice, for each prediction map to be tested, the center point is randomly selected in the height and width range of (70, 352-70), and then 10 small patches (72×72) are taken based on the location of the center point. We select ten small patches from each image in each batch and then obtain these mentioned metrics for the whole batch. Then, the mean and standard deviation of all batches of indicators are obtained, so that these indicators can statistically reflect the calibration effects.

4.2 Evaluation and Results

As shown in Fig. 4, the top row shows our calibration method achieves the minimal misalignment (red gap) between confidence and accuracy. The middle and bottom rows of the confidence histogram illustrate that the proposed method better corrects overconfident predictions compared with the other results.

Note that there is a common problem of overconfident miscalibration in image segmentation [1,10]. In our task, each pixel is given a probability by Eq. (1). Because of this either-or mechanism where the confidences of positive and negative samples sum to one, overconfident miscalibration is greatly amplified. This leads to a very high confidence (close to 0.99 in our task) for non-boundary pixels with a large proportion, which has a large impact on calibration effect. This can also explain why there are many samples between the confidence interval [0.9,1) in the histograms.

Table 3. Quantitative metrics results for 4 different calibration methods cooperating with 4 segmentation models on ETIS-LaribPolypDB. Results are reported in mean (std) format

DATASETS		ETIS-LaribPolypDB			
Segmentation model		LNPNet [27]	PraNet [6]	MSNet [29]	PolypPVT [3]
ECE%↓	UC	13.98(21.59)	13.45(21.52)	14.30(21.81)	14.32(21.83)
	TS	13.98(20.16)	14.00(19.76)	13.86(19.33)	14.43(20.45)
	ETS	13.85(21.05)	13.5(20.70)	13.90(20.54)	14.21(21.23)
	LTS	13.59(20.20)	13.47(19.99)	13.30(18.82)	14.09(20.73)
	OURS	**2.38(5.10)**	**3.15(7.07)**	**3.69(7.37)**	**2.27(5.36)**
MCE%↓	UC	20.68(24.73)	17.88(23.47)	20.41(24.77)	20.58(24.67)
	TS	19.16(21.71)	19.8(22.80)	21.43(24.09)	20.33(22.80)
	ETS	19.55(23.10)	18.18(23.03)	20.62(24.38)	20.12(23.53)
	LTS	19.98(22.69)	19.88(23.38)	22.76(25.39)	21.26(23.91)
	OURS	**12.54(18.80)**	**10.84(15.54)**	**12.94(17.09)**	**11.35(16.04)**
SCE%↓	UC	14.12(21.71)	13.83(21.83)	14.94(22.38)	14.63(22.08)
	TS	15.18(20.68)	15.10(20.55)	15.60(20.71)	15.66(21.16)
	ETS	14.44(21.35)	14.27(21.30)	15.16(21.57)	14.97(21.74)
	LTS	14.49(20.67)	14.46(20.76)	14.89(20.16)	15.09(21.39)
	OURS	**2.81(5.54)**	**3.75(7.86)**	**4.46(8.94)**	**2.94(6.62)**
ACE%↓	UC	14.11(21.70)	13.83(21.82)	14.96(22.37)	14.64(22.07)
	TS	15.35(20.62)	15.36(20.44)	15.96(20.65)	16.04(21.05)
	ETS	14.45(21.33)	14.39(21.24)	15.37(21.54)	15.11(21.70)
	LTS	14.65(20.62)	14.64(20.69)	15.11(20.15)	15.28(21.34)
	OURS	**2.86(5.55)**	**3.76(7.88)**	**4.45(8.95)**	**2.96(6.63)**

Table 4. Quantitative metrics results for 4 different calibration methods cooperating with 4 segmentation models on Piccolo. Results are reported in mean (std) format

DATASETS		Piccolo			
Segmentation model		LNPNet	PraNet	MSNet	PolypPVT
ECE%↓	UC	20.82(28.80)	26.38(32.37)	21.09(27.97)	28.22(33.12)
	TS	20.50(27.28)	26.36(29.98)	19.99(24.33)	27.75(31.67)
	ETS	20.05(28.21)	26.10(31.14)	20.31(25.98)	27.80(32.40)
	LTS	19.98(27.14)	25.86(30.03)	19.33(23.67)	27.37(31.75)
	OURS	**8.78(20.00)**	**5.57(9.50)**	**12.64(22.96)**	**5.72(14.21)**
MCE%↓	UC	28.18(29.80)	33.3(32.29)	29.36(29.40)	35.36(32.79)
	TS	26.92(27.86)	35.29(30.70)	30.26(27.51)	35.11(32.13)
	ETS	22.74(28.68)	33.97(31.57)	29.82(28.35)	34.76(32.47)
	LTS	27.3(27.85)	35.48(30.24)	32.72(28.89)	36.00(32.06)
	OURS	**20.00(24.14)**	**18.8(17.82)**	**25.45(26.61)**	**18.21(20.41)**
SCE%↓	UC	21.09(28.95)	27.53(32.69)	22.16(28.47)	29.00(33.33)
	TS	22.35(27.65)	28.92(30.53)	22.63(25.38)	30.17(31.97)
	ETS	21.54(28.50)	28.10(31.63)	22.28(26.81)	29.44(32.74)
	LTS	21.44(27.48)	28.13(30.54)	21.62(24.71)	29.42(32.07)
	OURS	**9.53(20.35)**	**7.06(11.52)**	**14.28(23.72)**	**7.10(15.51)**
ACE%↓	UC	21.08(28.94)	27.53(32.68)	22.17(28.46)	29.00(33.11)
	TS	22.53(27.56)	29.13(30.37)	22.90(25.30)	30.46(31.79)
	ETS	21.59(28.46)	28.20(31.55)	22.41(26.77)	29.55(32.67)
	LTS	21.65(27.39)	28.29(30.42)	21.79(24.66)	29.60(31.95)
	OURS	**9.60(20.33)**	**7.06(11.56)**	**14.27(23.75)**	**7.10(15.52)**

Therefore, we find that in the task of two-label image segmentation calibration, although the previous models such as TS, ETS, and LTS reduce the value of ECE, they actually only reduce the ECE of the background area and do not really correct the confidence of the lesion area, nor can they give a calibration uncertainty map with reference value. This is why Mask-Loss and Mask-TS are needed so as to pay more attention to the lesion area.

Table 3 and Table 4 show the statistic metrics results for four different calibration methods cooperating with four segmentation models on two never-seen test datasets, ETIS-LaribPolypDB and Piccolo. It can be seen from the table that the calibration effect is affected by different segmentation methods. But compared with other calibration methods, the methods proposed by us all has better evaluation metrics. It means that our method have better calibration effect when the segmentation model is given.

As shown in Fig. 5 and Fig. 6, the results with some mispredictions are specifically selected to test whether the uncertainty map is well calibrated. The comparison between the error and the uncertainty map shows that when the prediction

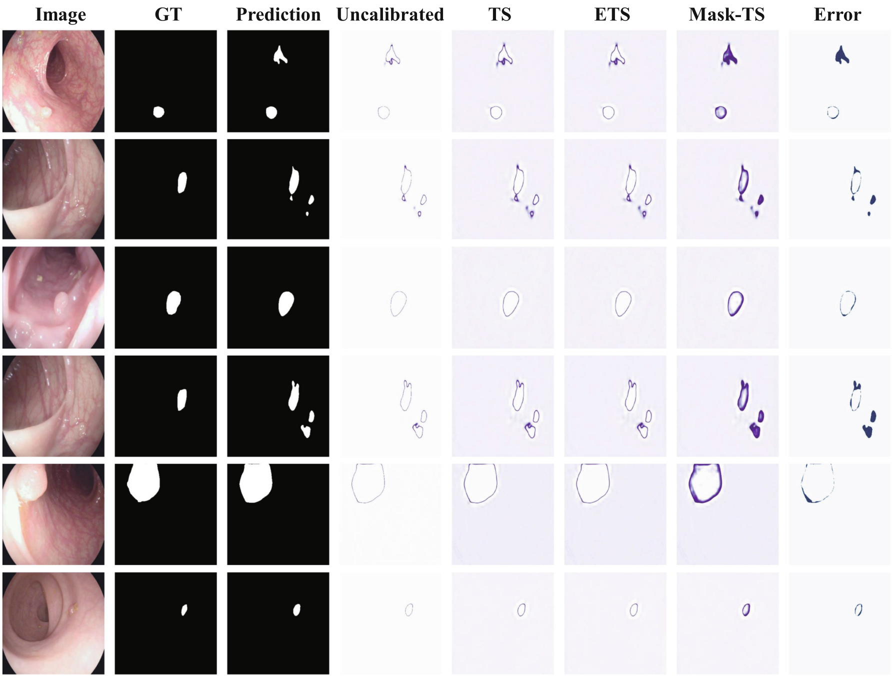


Fig. 5. Uncertainty maps of comparative experiment on ETIS-LaribPolypDB.

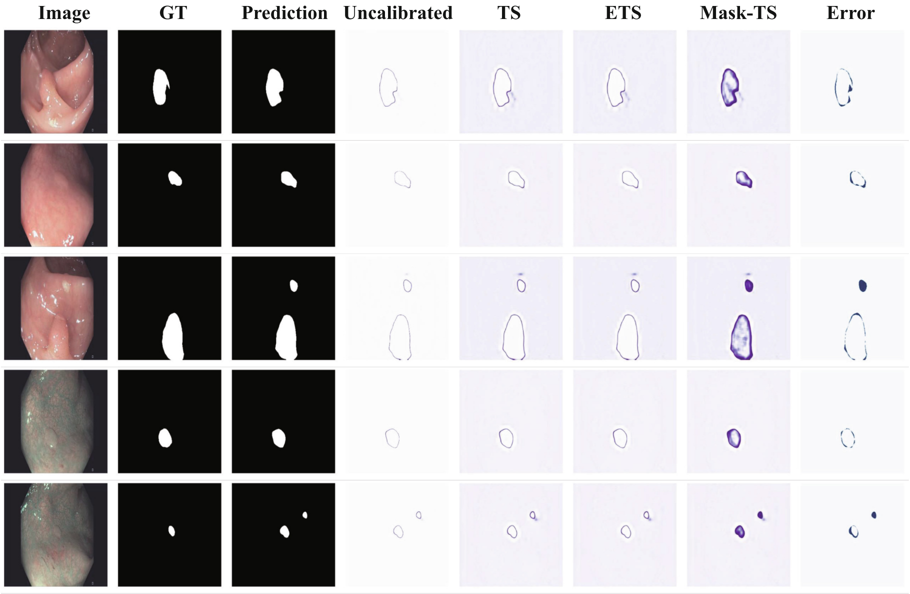


Fig. 6. Uncertainty maps of comparative experiment on Piccolo.

results exhibit high uncertainty, the likelihood of the prediction being incorrect increases. In other words, the uncertainty estimate of the calibration model correlates well with the actual segmentation error, denoting that the uncertainty is well calibrated.

4.3 Ablation Studies

As mentioned in Sect. 3.1, our Calibration Network primarily consists of four branches: uncertainty maps from the vanilla TS, probabilities from vanilla TS, output logits and input image, which are concatenated through an attention mechanism followed by process of Mask-Loss and Mask-TS. In the ablation experiments, we ablate some key components: Ablation01 removes Mask-TS, Ablation02 eliminates Mask-Loss & Mask-TS, and Ablation03 abolishes Mask-Loss & Mask-TS & probability from the vanilla TS. Subsequently, the calibration

Table 5. Quantitative metrics results for ablation experiment.

Dataset	Metrics	OURS	AB.01	AB.02	AB.03
ETIS-LaribPolypDB	ECE↓	**0.15**	3.25	0.71	0.66
	MCE↓	**3.30**	3.84	12.33	11.98
	SCE↓	**2.17**	7.06	5.45	5.70
	ACE↓	**2.35**	7.06	5.44	5.73
Piccolo	ECE↓	**8.16**	8.36	8.43	8.78
	MCE↓	**16.80**	18.96	19.1	19.99
	SCE↓	**9.02**	9.08	9.22	9.53
	ACE↓	**9.05**	9.15	9.31	9.60

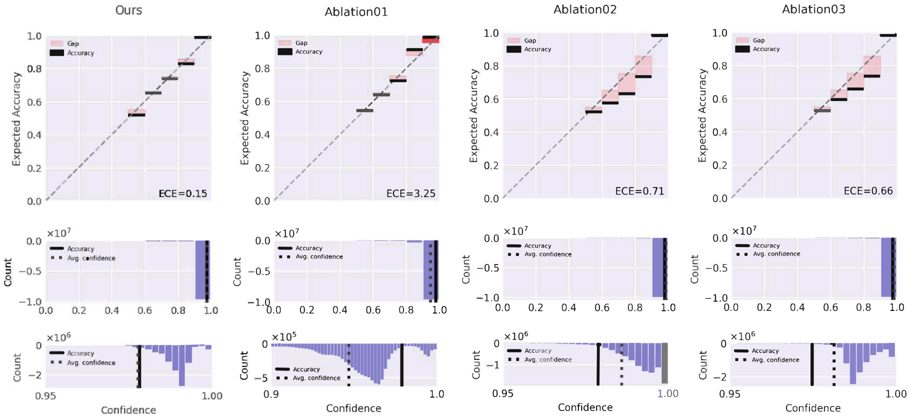


Fig. 7. Ablation results of reliability diagram & confidence histogram on ETIS-LaribPolypDB.

results are tested based on quantitative and qualitative metrics, as depicted in Table 5 and Figs. 7, 8 and 9.

In detail, the comparison of Ablation01 with Ablation02 and Ablation03 shows the calibration effect of Mask-Loss on the lesion area, which typically consists of a smaller proportion of pixels and exhibits lower confidence levels ranging from 0.5 to 0.9, manifested as a reduction in the red gap. But it causes misalignment in background where confidence levels are higher, between 0.9 and 1. Our method compared to Ablation01 illustrates the well calibrated effect in background area via introducing Mask-TS on overall network calibration. In a word, the calibration network achieves optimal performance when the four branches work together and the Mask-Loss & Mask-TS strategies are adopted.

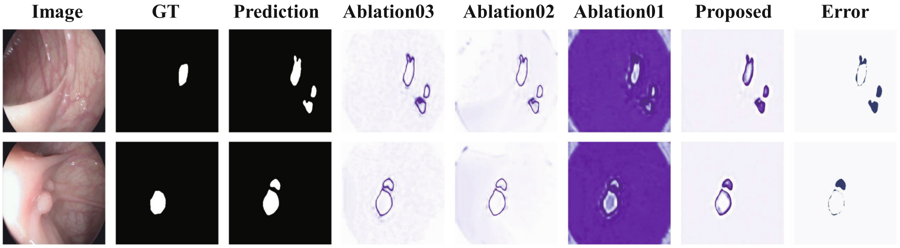


Fig. 8. Ablation results of uncertainty map on ETIS-LaribPolypDB.

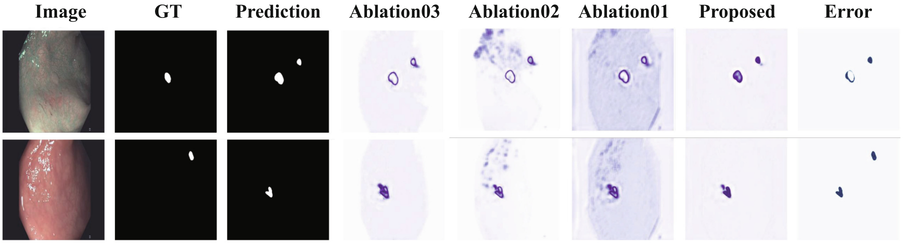


Fig. 9. Ablation results of uncertainty map on Piccolo.

5 Conclusion

In this work, we propose a better calibration method for binary-label polyp image segmentation. Further work can be done to improve the accuracy by improving the segmentation network structure preferably with smaller original miscalibration, improving the calibration network structure to reduce miscalibration or using more detailed masks to reduce misalignment.

Acknowledgements. Research supported by the Training Program of Innovation and Entrepreneurship for Undergraduates in Beijing University of Chemical Technology (X202410010085).

References

1. Bengio, Y., Goodfellow, I., Courville, A.: Deep Learning, vol. 1. MIT Press, Cambridge (2017)
2. Bernal, J., Sánchez, F.J., Fernández-Esparrach, G., Gil, D., Rodríguez, C., Vilariño, F.: WM-DOVA maps for accurate polyp highlighting in colonoscopy: validation vs. saliency maps from physicians. Comput. Med. Imaging Graph. **43**, 99–111 (2015)
3. Bo, D., Wenhai, W., Deng-Ping, F., Jinpeng, L., Huazhu, F., Ling, S.: Polyp-PVT: polyp segmentation with pyramid vision transformers. In: CAAI AIR (2023)
4. DeGroot, M.H., Fienberg, S.E.: Assessing probability assessors: calibration and refinement. Stat. Decis. Theory Relat. Topics **III**(1), 291–314 (1982)
5. Ding, Z., Han, X., Liu, P., Niethammer, M.: Local temperature scaling for probability calibration. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6889–6899 (2021)
6. Fan, D.-P., et al.: PraNet: parallel reverse attention network for polyp segmentation. In: Martel, A.L., et al. (eds.) MICCAI 2020. LNCS, vol. 12266, pp. 263–273. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59725-2_26
7. Gawlikowski, J., et al.: A survey of uncertainty in deep neural networks. Artif. Intell. Rev. **56**(Suppl 1), 1513–1589 (2023)
8. Guo, C., Pleiss, G., Sun, Y., Weinberger, K.Q.: On calibration of modern neural networks. In: International Conference on Machine Learning, pp. 1321–1330. PMLR (2017)
9. Gupta, C., Ramdas, A.: Online Platt scaling with calibeating. In: International Conference on Machine Learning, pp. 12182–12204. PMLR (2023)
10. Hastie, T., Friedman, J., Tibshirani, R., Hastie, T., Friedman, J., Tibshirani, R.: Kernel methods. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, pp. 165–192 (2001)
11. Kendall, A., Gal, Y.: What uncertainties do we need in Bayesian deep learning for computer vision? In: Advances in Neural Information Processing Systems, vol. 30 (2017)
12. Lakshminarayanan, B., Pritzel, A., Blundell, C.: Simple and scalable predictive uncertainty estimation using deep ensembles. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
13. McClure, P., et al.: Knowing what you know in brain segmentation using Bayesian deep neural networks. Front. Neuroinform. **13**, 67 (2019)
14. Mehrtash, A., Wells, W.M., Tempany, C.M., Abolmaesumi, P., Kapur, T.: Confidence calibration and predictive uncertainty estimation for deep medical image segmentation. IEEE Trans. Med. Imaging **39**(12), 3868–3878 (2020)
15. Naeini, M.P., Cooper, G., Hauskrecht, M.: Obtaining well calibrated probabilities using Bayesian binning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 29 (2015)
16. Niculescu-Mizil, A., Caruana, R.: Predicting good probabilities with supervised learning. In: Proceedings of the 22nd International Conference on Machine Learning, pp. 625–632 (2005)

17. Nixon, J., Dusenberry, M.W., Zhang, L., Jerfel, G., Tran, D.: Measuring calibration in deep learning. In: CVPR Workshops, vol. 2 (2019)
18. Ouyang, C., et al.: Improved post-hoc probability calibration for out-of-domain MRI segmentation. In: Sudre, C.H., et al. (eds.) Uncertainty for Safe Utilization of Machine Learning in Medical Imaging, UNSURE 2022. LNCS, vol. 13563, pp. 59–69. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-16749-2_6
19. Patel, K., Beluch, W., Zhang, D., Pfeiffer, M., Yang, B.: On-manifold adversarial data augmentation improves uncertainty calibration. In: 2020 25th International Conference on Pattern Recognition (ICPR), pp. 8029–8036. IEEE (2021)
20. Platt, J., et al.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. Adv. Large Margin Classif. **10**(3), 61–74 (1999)
21. Pogorelov, K., et al.: KVASIR: a multi-class image dataset for computer aided gastrointestinal disease detection. In: Proceedings of the 8th ACM on Multimedia Systems Conference, pp. 164–169 (2017)
22. Silva, J., Histace, A., Romain, O., Dray, X., Granado, B.: Toward embedded detection of polyps in WCE images for early diagnosis of colorectal cancer. Int. J. Comput. Assist. Radiol. Surg. **9**, 283–293 (2014)
23. Sánchez-Peralta, L.F., et al.: PICCOLO white-light and narrow-band imaging colonoscopic dataset: a performance comparative of models and datasets. Appl. Sci. **10**(23), 8501 (2020)
24. Tajbakhsh, N., Gurudu, S.R., Liang, J.: Automated polyp detection in colonoscopy videos using shape and context information. IEEE Trans. Med. Imaging **35**(2), 630–644 (2015)
25. Thulasidasan, S., Chennupati, G., Bilmes, J.A., Bhattacharya, T., Michalak, S.: On mixup training: improved calibration and predictive uncertainty for deep neural networks. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
26. Vázquez, D., et al.: A benchmark for endoluminal scene segmentation of colonoscopy images. J. Healthc. Eng. **2017**, 4037190 (2017)
27. Xu, C.: Lnpnet: Ladder net for polyp segmentation with multi-level feature fusion (2024). https://github.com/XUCH1008/lnpnet
28. Zhang, J., Kailkhura, B., Han, T.Y.J.: Mix-n-Match: ensemble and compositional methods for uncertainty calibration in deep learning. In: International Conference on Machine Learning, pp. 11117–11128. PMLR (2020)
29. Zhao, X., Zhang, L., Lu, H.: Automatic polyp segmentation via multi-scale subtraction network. In: de Bruijne, M., et al. (eds.) MICCAI 2021. LNCS, vol. 12901, pp. 120–130. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-87193-2_12
30. Zou, K., Chen, Z., Yuan, X., Shen, X., Wang, M., Fu, H.: A review of uncertainty estimation and its application in medical imaging. Meta-Radiol. **1**, 100003 (2023)

Label-Expanded Feature Debiasing for Single Domain Generalization

Jie Yang[1], Liwei Jing[1], Yuanzhuo Xu[1], Shaowu Wu[1], Steve Drew[2],
and Xiaoguang Niu[1](✉)

[1] School of Computer Science, Wuhan University, Wuhan 430072, China
{csyangjie,jingliwei,xyzxyz,wshaow,xgniu}@whu.edu.cn
[2] University of Calgary, Calgary T2N 1N4, Canada
steve.drew@ucalgary.ca

Abstract. In single domain generalization, the model is trained on a single source domain and requires being generalized to multiple unseen target domains. However, the presence of domain discrepancies poses a significant threat to this goal. A straightforward solution is to extract class-specific features that are robust to domain discrepancies. Existing methods typically learn class-specific features with the assistance of domain-specific features by enlarging the gap between them. Nevertheless, the absence of domain supervision limits the extraction of domain-specific features, leading to potential misidentification of class-specific features. To address this issue, we propose Label-expanded Feature Debiasing (LeFD), a novel method that learns class-specific features in a more robust manner. Technically, LeFD introduces domain supervision and explicitly extracts integrated domain and class features through label expansion. Subsequently, a rationale alignment module is employed to eliminate domain information from the integrated domain and class features, thereby obtaining class-specific features. Extensive experimental results on multiple benchmark datasets indicate the superiority of the proposed LeFD compared to other state-of-the-art methods.

Keywords: Domain supervision · Label expansion · Feature debiasing

1 Introduction

Deep neural networks (DNNs) have shown remarkable performance in a variety of tasks under the assumption that the training (i.e., source) and test (i.e., target) domains are drawn from similar or identical distributions [1]. Unfortunately, this assumption often fails to hold in the real world. When transferred from the source domain to unseen target domains, the performance of the model will typically degrade. The main reason originates that domain discrepancies

---

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78128-5_13.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 195–210, 2025.
https://doi.org/10.1007/978-3-031-78128-5_13

undermine the extraction of features [38]. To address this challenge, extensive studies on domain adaptation (DA) [30,39] and domain generalization (DG) [13,17,23] have emerged. Despite considerable success, they are constrained by the requirement for data from target domains or multiple source domains with different distributions. This paper considers an extreme case of generalization, where only a single domain is available for training, yet the trained model must generalize to multiple unseen target domains, which is termed as single domain generalization (SDG).

Due to the scarcity of diverse source domains in the training stage, most existing SDG methods [6,7,33] focus on data augmentation. They aim to extend the data distribution to encompass that of the target domain as closely as possible. However, the primary challenge encountered by these methods is the uncertainty regarding whether the augmented data can effectively mitigate the discrepancy between the source domain and the unseen target domains. By contrast, another line of methods [29,32,34] strives to disentangle class-specific (semantic) features that are robust to the domain discrepancy from domain-specific (superficial) features. Some of these methods consider class-specific and domain-specific features unrelated and thus achieve disentanglement by enlarging the gap between the two features. Other methods assume that the multi-factor, domain-specific features can infer the corresponding class labels, so they first use a multi-head classifier to extract the domain-specific features and then force the orthogonality between the weights of the multi-head classifier and the weights of a vanilla classifier to extract class-specific features. The key problem is that the features extracted through these methods are not guaranteed to be accurate. The absence of domain supervision limits the extraction of domain-specific features and thus hinders identifying the class-specific features. To this end, it is crucial to disentangle class-specific features from domain-specific features while ensuring their plausibility and interpretability.

In this paper, we present a straightforward yet effective method, named Label-expanded Feature Debiasing (LeFD), to facilitate reliable class-specific feature disentanglement. The proposed LeFD explicitly extracts integrated domain and class features under a multi-task paradigm with respect to learning the joint distribution of class labels and domain labels, and then disentangles these integrated features to obtain class-specific features. Operating based on the multi-task learning paradigm, the common constraints of class labels and domain labels enable the feature extractor to naturally extract abundant features that contain both domain and class information [22]. Technically, this is achieved through the construction of domain labels and the expansion of the original class labels. In particular, LeFD augments the source domain data with simple perturbations to obtain a different auxiliary domain. Subsequently, LeFD assigns domain labels (e.g., domain-0, domain-1) to the source data and augmented data, respectively, and then constructs joint labels encompassing domains and classes. Under the supervision of joint labels, a feature extractor is leveraged to extract integrated domain and class features. Finally, LeFD employs a rationale alignment module to align the decision-making process of the classifier on data from different domains. Concretely, this module is designed to disentangle class-specific features, which are then fed into the backbone classifier to predict class labels. In this process, both the class-specific features and the classifier predictions of the data belonging to the same class are restricted to be consistent despite the varying domains. Compared with existing feature disentanglement methods, the incorporation of domain labels in LeFD ensures the reliability of the extracted domain-specific features. Moreover, label expansion and feature debiasing enable LeFD to extract more fine-grained and plausible class-specific features.

We summarize our contributions as follows:

- To alleviate the ambiguity of the extracted domain-specific and class-specific features, we present a new feature debiasing method called LeFD, which explicitly extracts integrated domain and class features under a multi-task learning paradigm.
- We propose constructing domain labels for the source and augmented data to aid in extracting domain-specific features from the data. On this basis, we perform label expansion to obtain joint domain and class labels, which serve as supervisions for extracting integrated domain and class features.
- We introduce a rationale alignment module to align the decision-making process of the backbone classifier on data from different domains to further decouple class-specific features.
- Experimental results on multiple benchmark datasets verify the effectiveness of LeFD in tackling SDG problems. In addition, the outstanding performance in the DG scenario further demonstrates the favorable generalization ability of LeFD.

2 Related Work

2.1 Domain Generalization

The goal of domain generalization (DG) is to train the model on multiple source domains and generalize it to unseen target domains. Recent methods [4,5,13] primarily focus on domain alignment, operating under the assumption that data from different domains share a common representation in the feature space. For example, HGP [13] theoretically proved that aligning the Hessian matrix and gradient of the classifier across domains can increase the generalization ability. [5] aligned a joint distribution and a production distribution via the Relative Chi-Square (RCS) divergence. Apart from domain alignment, there are some other methods that tackle DG through data augmentation [17] and meta-learning [23]. Although these methods have achieved significant success, they cannot be applied directly to the single domain generalization due to their reliance on multiple source domains.

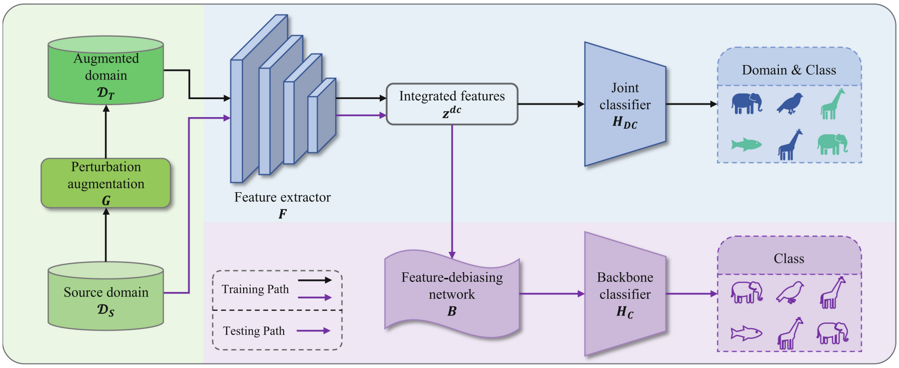


Fig. 1. The overall framework of LeFD. The augmented domain is generated first via perturbation augmentation G. Then we manually construct domain labels for both source and augmented data. With the domain and class labels, we expand them to obtain joint labels that indicate both the domain and the class to which the data belong. Subsequently, we use joint labels as supervisions to train the feature extractor F and the joint classifier H_{DC}, aiming to learn the integrated domain and class features. Based on the integrated features, we disentangle class-specific features through the feature-debiasing network B and the backbone classifier H_C.

2.2 Single Domain Generalization

As an extreme case of DG, single domain generalization (SDG) presents a more realistic and challenging scenario. In SDG, there is only one source domain available during training, yet the model must generalize to several unseen target domains during test. To tackle this challenge, some methods [8,36] proposed learning-free data augmentation schemes. For instance, RandConv [36] employed random convolutions for data augmentation, distorting local textures while preserving shape. Pro-RandConv [8] further advanced this approach by introducing random initialized offsets and recursively stacked convolutions to enhance diversity. Additionally, adversarial augmentation techniques have been proposed to generate safe and effective samples. Among them, PDEN [25] and M-ADA [28] trained the generator in a min-max manner. To simulate larger domain discrepancies, CADA [6] proposed a center-aware augmentation technique to push generated samples away from class centers through an angular center loss. Based on adversarial augmentation, ABA [7] introduced Bayesian neural networks and leveraged weight uncertainty to explore a wider augmentation space. Despite these efforts to introduce rich diversity, these methods still struggle to generate samples that adequately cover all unseen domains.

Different from data augmentation methods, other methods [29,32,34] focused on learning class-specific features that are robust to domain discrepancies, resulting in domain-specific features. Out of them, MetaCNN [32] decomposed the features into meta-features to construct a feature bag and proposed a compositional operation to eliminate domain-specific features through an addressing process. However, the proposed feature decomposition and composition incurred significant computational overhead. Heuristically, some methods considered class-specific and domain-specific features to be unrelated. Building on this assumption, [34] presented cyclic disentanglement to enlarge the gap between the features extracted by two branches, aiming to disentangle class-specific features. Another enlightening method, named MAD [29], enforced the orthogonality between the weights of the domain classifier and the vanilla class classifier. Unfortunately, the features extracted by these methods are not guaranteed to be truly domain-specific and class-specific due to the absence of domain supervision in SDG, which limits the extraction of domain-specific features and consequently hinders the identification of the class-specific features. Our work is inspired by this limitation and aims at learning class-specific features in a more robust manner.

3 Methodology

3.1 Preliminary

In the SDG problem, we aims to train the model with only a single source domain $\mathcal{D}_S$ and generalize it to several unseen target domains: $\{\mathcal{D}_T^j\}_{j=1}^J$, where $\mathcal{D}_S$ and $\mathcal{D}_T^j$ are sampled from different distributions with the same class label space. For the K-class classification, we define $\mathcal{D}_S = \{(x_i^s, y_i^c)\}_{i=1}^N$, where x_i^s represents the i-th source sample and $y_i^c \in \mathcal{Y}_C \subset \mathbb{R}^K$ represents the corresponding class label. Additionally, the set of variables will be in bold, e.g., $\boldsymbol{x^s} = \{x_i^s\}_{i=1}^N$.

To learn robust class-specific features, we propose a novel method called LeFD. LeFD manually constructs joint domain and class labels to explicitly extract integrated domain and class features based on a multi-task learning paradigm. Subsequently, LeFD aligns class-specific features and classifier predictions from different domains to disentangle features, ultimately yielding more reliable class-specific features. Particularly, LeFD consists of four components: a domain expansion module to generate auxiliary domains, a label-expanded multi-task module to extract integrated domain and class features, a rationale alignment module to identify class-specific features, and a backbone classifier for class label prediction, as illustrated in Fig. 1.

3.2 Perturbation-Based Domain Expansion

In order to collect more domain-specific features, we desire to expand the source domain to obtain a different auxiliary domain. So, we apply data augmentation. Instead of leveraging adversarial augmentation in most existing methods [6,28,33], we design a lightweight perturbation augmentation module G. G consists of a random 1×1 convolution layer L_{conv} followed by an AdaIN layer [18] to introduce more randomness. Concretely, the AdaIN used contains an element-wise normalization layer L_{in} and two trainable linear layers L_{fc1} and L_{fc2}:

$$AdaIN(\boldsymbol{x^s}, n) = L_{fc1}(n) \cdot L_{in}(L_{conv}(\boldsymbol{x^s})) + L_{fc2}(n), \tag{1}$$

where n is drawn from a normalized distribution. For a given source sample x_i^s, a corresponding augmented sample x_i^a is generated by

$$x_i^a = G(x_i^s) = \mathrm{Tanh}(AdaIN(x_i^s, n)), \tag{2}$$

where $\mathrm{Tanh}(\cdot)$ is a Hyperbolic Tangent activation function. Consequently, we denote the augmented auxiliary domain as $\mathcal{D}_A = \{(x_i^a, y_i^c)\}_{i=1}^N$. It should be noted that only the parameters of L_{fc1} and L_{fc2} are learnable, so the computational overhead required for domain expansion is particularly small.

3.3 Label-Expanded Multi-task Learning

Based on the source and augmented domains, we want to disentangle class-specific features from domain-specific features. Considering that class-specific and domain-specific features are irrelevant, existing methods design two branches and extract these two types of features by expanding the gap between the outputs of the two branches or orthogonalizing the weights of the two branches by using only class labels. A subsequent problem is that it is very fragile and untenable to ensure that they are really domain-specific and class-specific because of the lack of domain labels as supervisions. Accordingly, we present a domain-expanded multi-task learning schema. Our motivation is simple: 1) the lack of domain labels makes the extracted domain-specific features unreliable, and thus hindering the identification of class-specific features; 2) extracting class-specific features alone will be interfered by domain discrepancies, as dose domain-specific features. Therefore, we manually construct domain labels and simultaneously use class and domain labels to explicitly extract the integrated features containing both class and domain information.

For each sample x_i from source and augmented domains, we manually assign a domain label $y_i^d \in \mathcal{Y}_D \subset \mathbb{R}^2$ to it. In particular, $y_i^d = 0$ for sample from source domain and $y_i^d = 1$ for sample from augmented domain. In order to extract integrated features containing both domain and class information under a multi-task learning paradigm, we expand the domain and class label space to obtain joint label space via $\mathcal{Y}_{DC} = \mathcal{Y}_D \otimes \mathcal{Y}_C \subset \mathbb{R}^{2K}$, where $\otimes$ denotes Cartesian product operation. In this way, the corresponding joint label y_i^{dc} of x_i indicate both the domain and the class to which x_i belongs. For example, for an image of a dog from the cartoon domain, its joint label is considered to be $\{Cartoon, Dog\}$. In our implementation, a certain y_i^{dc} is constructed as follows for simplicity:

$$y_i^{dc} = y_i^c + y_i^d \cdot K. \tag{3}$$

For feature extraction, we introduce a feature extractor F and a joint classifier H_{DC}. During training, both source and augmented data are leveraged, and the following cross-entropy loss $\mathcal{L}_{dc}$ is minimized to optimize F and H_{DC}:

$$\mathcal{L}_{dc} = -\frac{1}{2N_b}\sum_{i=1}^{2N_b}\sum_{j=1}^{2K} \mathbb{1}_{j=y^{dc}} \log \frac{\exp(p_j^{dc}(x_i))}{\sum_{k=1}^{2K}\exp(p_k^{dc}(x_i))}, \tag{4}$$

where N_b represents the batch size and $p_k^{dc}(x_i) = H_{DC}(F(x_i))[k]$ denotes the k-th dimension of the logits output by the joint classifier for input sample $x_i \in \boldsymbol{x} = \boldsymbol{x^s} \cup \boldsymbol{x^a}$. Naturally, supervised by joint domain and class labels for multi-task learning, the features extracted by F, that is, $\boldsymbol{z^{dc}} = F(\boldsymbol{x})$, will simultaneously contain both domain and class information in that they are then fed into H_{DC} and constrained to correctly predict the domain and class to which data belong.

3.4 Feature Debiasing via Rationale Alignment

Based on the integrated domain and class features, we try to disentangle the class-specific features. For this purpose, we introduce a rationale alignment module. In detail, this module contains a feature-debiasing network B, which takes $\boldsymbol{z^{dc}}$ as input and outputs $\boldsymbol{z^c}$. Subsequently, $\boldsymbol{z^c}$ is fed into the backbone classifier H_C to predict class labels. To ensure that $\boldsymbol{z^c}$ is class-specific, we introduce another cross-entropy loss $\mathcal{L}_c$ to preserve class information.

$$\mathcal{L}_c = -\frac{1}{2N_b}\sum_{i=1}^{2N_b}\sum_{j=1}^{K}\mathbb{1}_{j=y^c}\log\frac{\exp(p_j^c(x_i))}{\sum_{k=1}^{2K}\exp(p_k^c(x_i))}, \tag{5}$$

where $p_k^c(x_i) = H_C(B(F(x_i)))[k]$ denotes the k-th dimension of the logits output by the backbone classifier for input sample x_i.

Furthermore, we introduce rationale alignment so that $\boldsymbol{z^c}$ contain no domain information, i.e., domain-invariant. The concept of rationale is first introduced in [4], which indicates the decision-making process of networks. Correspondingly, a rationale matrix $\mathcal{R}_i$ represents the rationale of classifying a sample x_i and it is formally described as follows:

$$\mathcal{R}_i = \begin{bmatrix} W_{1,1}\cdot z_{i,1}^c, & W_{1,2}\cdot z_{i,1}^c, & \ldots, & W_{1,K}\cdot z_{i,1}^c \\ W_{2,1}\cdot z_{i,2}^c, & W_{2,2}\cdot z_{i,2}^c, & \ldots, & W_{2,K}\cdot z_{i,2}^c \\ \vdots & \vdots & \ddots & \vdots \\ W_{M,1}\cdot z_{i,M}^c, & W_{M,2}\cdot z_{i,M}^c, & \ldots, & W_{M,K}\cdot z_{i,M}^c \end{bmatrix}, \tag{6}$$

where M represents the number of dimensions of $\boldsymbol{z^c}$, $z_{i,j}^c$ represents the j-th dimension of z_i^c and $W_{j,k}$ represents the corresponding weight in the H_C. To remove domain information from $\boldsymbol{z^c}$, we try to ensure that the decision to classify a sample is based on the same reasoning regardless of the variation in domains. We achieve this goal by aligning the rationale matrices of augmented data from the same class with those of the corresponding source data, and the alignment loss L_{align} is formulated as follows:

$$\mathcal{L}_{align} = \frac{1}{N_b}\sum_{k=1}^{K}\sum_{i=1,y_i^c=k}^{N_b}\|\mathcal{R}_i^A - \bar{\mathcal{R}}_k^S\|, \tag{7}$$

where $\|\cdot\|$ is the $l2$ norm, $\mathcal{R}_i^A$ is the rationale matrix of the i-th augmented sample and $\bar{\mathcal{R}}_k^S$ is the mean rationale matrix of source samples corresponding to the k-th class. In implementation, $\bar{\mathcal{R}}_k^S$ is updated in an exponential moving average manner:

$$\bar{\mathcal{R}}_{k,t}^S = (1-\lambda) \times \bar{\mathcal{R}}_{k,t-1}^S + \lambda \times \frac{1}{|y_i^c = k|} \sum_{i=1, y_i^c=k}^{N} \mathcal{R}_i^S, \tag{8}$$

where t is the iteration index and λ is the momentum value. Specially, $\bar{\mathcal{R}}_{k,0}^S$ is initialized with the rationale matrix of the first iteration.

3.5 Overall Framework

During training, we adopt a two-stage learning schema. In the first stage, we only minimize $\mathcal{L}_{dc}$ to encourage the feature extractor capable of learning the integrated domain and class features. In the second stage, we then introduce $\mathcal{L}_c$ and $\mathcal{L}_{align}$ to disentangle the class-specific features. Therefore, the overall loss $\mathcal{L}$ is defined as follows:

$$\mathcal{L} = \mathcal{L}_{dc} + \mathbb{1}_{t \geq T}(\mathcal{L}_c + \mathcal{L}_{align}), \tag{9}$$

where T denotes the first epoch of the second training stage. Consequently, the network framework is illustrated in Fig. 1 and the corresponding algorithm is included in supplementary materials. During testing, we can predict the class labels without the assistance of G and H_{DC}.

4 Experiments

4.1 Experiment Setup

Datasets. Following recent work [3,6,7], we validate the proposed method LeFD on three popular benchmark datasets: Digits, CIFAR-C and PACS. **Digits** consists of five sub-datasets: MNIST [21], MNIST-M [11], SVHN [27], SYN [11] and USPS [16], with 10 classes. Each dataset is considered a domain. We choose MNIST as the source domain and the rest of datasets as the target domains. The first 10,000 images in the training set of MNIST are used for training. **CIFAR-C** [14] is corrupted from the test set of CIFAR-10 [20] by 4 broad types (including 19 specific types) with 5 severity levels (1–5). Higher numbers indicate greater severity. All images in CIFAR-10 and CIFAR-C are categorized into 10 classes. We use the training set of CIFAR-10 as the source domain and CIFAR-C as the target domains. **PACS** [24] contains four domains: Art, Cartoon, Photo and Sketch, with seven classes. We choose one domain as the source domain and the other three domains as the target domains at a time. Therefore, there are four sets of experiments on this dataset.

Implementation Details. There are some common settings for different datasets. First, the joint classifier H_{DC} and the backbone classifier H_C are both constructed by a single linear layer. Second, the momentum to update $\bar{\mathcal{R}}_k^S$ is set to 1e−3. The details of different implementations for each dataset are as follows.

Digits. We leverage the ConvNet (*conv-relu-pool-conv-relu-pool-fc-relu-fc-relu*) adopted in [6,8,25] to construct the feature extractor F. The feature-debiasing network B is constructed by a single linear layer with ReLU following. During training, the Adam optimizer is utilized with an initial learning rate of 1e−4. We train the whole model for 600 epochs and start the second stage at the 30th epoch. The batch size is set to 32.

CIFAR-C. We use the backbone of WRN-16-4 [37] as the feature extractor F. The feature-debiasing network B is also constructed by a single linear layer followed by ReLU. During training, we use the Adam optimizer. The learning rate is initialized to 1e−3 and updated with a cosine annealing scheduler. We also train the whole model for 600 epochs and start the second stage at the 30th epoch. The batch size is set to 128.

PACS. We use the backbone of ResNet-18 [12] as the feature extractor F. The feature-debiasing network B is constructed by two linear layers with ReLU following each linear layer. During training, the Adam optimizer is used with an initial learning rate of 1e−5. We train the whole model for 200 epochs and start the second stage at the 20th epoch. The batch size is set to 64.

4.2 Experimental Results on SDG

For experiments targeting SDG, we compare LeFD with several state-of-the-art methods, which are as follows: (i) the methods of data augmentation: GUD [31], AA [9], RA [10], RSC [15], M-ADA [28], L2D [33], PDEN [25], CADA [6], ABA [7]; (ii) the methods of learning class-specific features: CCSA [26], d-SNE [35], JiGen [2], MetaCNN [32], MAD [29], SAR [3]. Besides, the results of ERM [19] are also included as a reference.

In addition, we also conduct experiments in the DG scenario. More details and results are included in supplementary materials.

Comparison on Digits. Table 1 illustrates the experimental results on Digits. We can observe that the proposed LeFD achieves the highest classification accuracy on average. For data augmentation methods, we have improved by 3.73% over the best method CADA, and for class-specific feature learning methods, we have improved by 5.47% over the best method SAR. For each target domain, LeFD improves on both SVHN and SYN compared to existing methods. For MNIST-M and USPS, LeFD is slightly weaker. This may be because L2D and CADA leverage more complex adversarial augmentation and MetaCNN weakens color sensitivity through meta-features.

Comparison on CIFAR-C. For CIFAR-C, we consider four types of corruption as different target domains and select the data with a corruption severity level of 5 for test. Detailed comparison results are illustrated in Table 2. As we

Table 1. Comparison on Digits. The model is trained on MNIST using ConvNet as the backbone and tested on the other four datasets.

	MNIST-M	SVHN	SYN	USPS	Avg.
ERM	52.72	27.83	39.65	76.94	49.29
GUD	60.41	35.51	45.32	77.26	54.62
AA	60.53	45.23	64.52	80.62	62.72
RA	74.05	54.77	59.60	77.33	66.44
M-ADA	67.94	42.55	48.95	78.53	59.49
L2D	87.30	62.86	63.72	83.97	74.46
PDEN	82.20	62.21	69.39	85.26	74.77
CADA	78.66	67.27	79.34	**96.96**	80.56
ABA	80.05	56.87	73.40	96.55	76.72
CCSA	49.29	25.89	37.31	83.72	49.05
d-SNE	50.98	26.22	37.83	93.16	52.05
JiGen	57.80	33.80	43.79	77.15	53.14
MetaCNN	**88.27**	66.50	70.66	89.64	78.76
SAR	78.34	69.94	78.47	88.54	78.82
LeFD	84.75	**74.85**	**82.29**	95.27	**84.29**

can see from Table 2, LeFD achieves optimal results in the Blur, Digital and Weather domains. Particularly in the Blur domain, LeFD improves by 2.40% compared to baselines, and in the Weather domain, LeFD improves by 7.19%. In general, LeFD outperforms baselines in average accuracy and this clearly val-

Table 2. Comparison on CIFAR-C. The model is trained on CIFAR-10 using WRN-16-4 as the backbone and tested on CIFAR-C. Data from the same broad corruption type (i.e., Blur, Digital, Noise, and Weather) form a target domain. Only data with a corruption severity level of 5 are used for the test.

	Blur	Digital	Noise	Weather	Avg.
ERM	56.73	62.30	30.02	67.28	54.08
GUD	67.04	66.62	39.97	72.67	61.58
M-ADA	63.76	65.10	54.21	75.54	64.65
L2D	69.16	72.02	73.29	75.98	72.61
PDEN	76.41	77.93	77.11	78.65	77.53
CADA	83.77	84.84	**84.84**	79.93	83.57
CCSA	57.81	61.96	28.73	67.66	54.04
d-SNE	56.59	61.83	33.97	67.90	55.07
MetaCNN	76.80	81.26	78.23	77.44	78.45
LeFD	**86.17**	**85.48**	81.54	**87.12**	**85.08**

Table 3. Comparison on PACS. The model is trained on one domain (name in column) using ResNet-18 as the backbone and tested on the other three domains.

	Art	Cartoon	Photo	Sketch	Avg.
ERM	70.90	76.50	42.20	53.10	60.70
RSC	73.40	75.90	41.60	56.20	61.80
L2D	76.91	77.88	52.29	53.66	65.18
CADA	76.33	79.08	56.65	61.59	68.41
ABA	75.34	77.49	58.86	53.76	66.36
Pro-RandConv	76.98	78.54	62.89	57.11	68.88
JiGen	67.70	72.23	41.70	36.83	54.60
MAD	75.51	77.25	52.95	57.75	65.87
SAR	77.13	80.14	59.60	62.55	69.86
LeFD	**81.04**	**82.84**	**68.42**	**65.92**	**74.56**

idates the effectiveness of LeFD. In addition, we show the experimental results across different levels of corruption severity in supplementary materials, in which we also achieve the optimal.

Comparison on PACS. For PACS, one domain is selected as the source domain and the other three domains are selected as the target domains. The results are illustrated in Table 3. Compared with baseline methods, LeFD performs favorably better than baseline methods in all domains. In particular, LeFD improves the optimal results of baselines by 5.53% and 3.37% in the Photo domain and Sketch domain, respectively. Taken together, the average accuracy of LeFD on the PACS dataset is 4.70% higher than the baselines. This further demonstrates the superiority of the proposed LeFD in dealing with more challenging situations.

4.3 Additional Analysis

Ablation Study. To assess the contribution of each component in our method, we conduct an ablation study on the PACS dataset. In these experiments, we either retain all components, remove one or two components, or eliminate all components to train the model. Specifically, the first and last settings correspond to the ERM method and the proposed LeFD respectively. The results are summarized in Table 4. As we can see all three components contribute significantly. Domain expansion can greatly enhance the ERM method, but it is still not optimal. On this basis, the introduction of label expansion and rationale alignment can further improve classification accuracy.

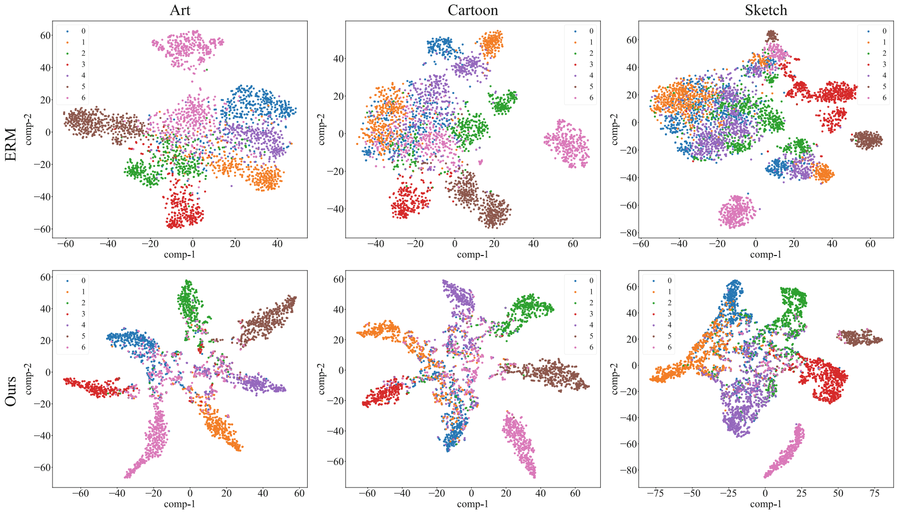


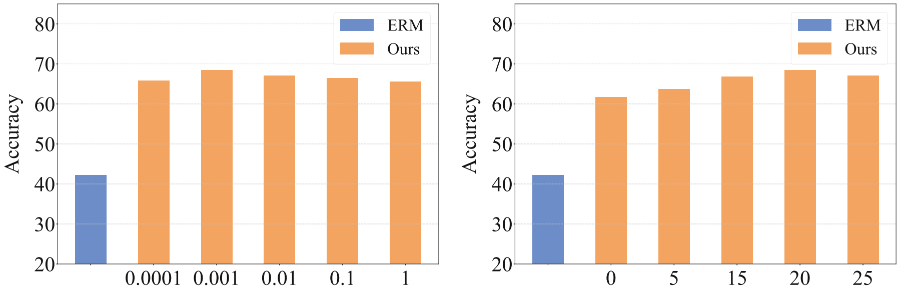
Fig. 2. t-SNE visualization of extracted class-specific features from the PACS. The Photo domain serves as the source domain, while the other three domains (Art, Cartoon, Sketch) are used as the target domains. Each column represents the visualization results under a specific target domain (indicated in the title). Features extracted from the test dataset of both the source and target domains are plotted in the same subgraph to facilitate the evaluation of the impact of domain discrepancies on class-specific feature extraction.

Hyper-parameter Sensitivity. We further conduct several experiments to explore the sensitivity of LeFD to hyperparameters λ and T in the SDG scenarios. λ is the momentum value used to update the mean rationale matrix of source samples and T denotes the epoch index at which the second training stage begins. In the experiments, we value λ on multiple scales from 1e$-$4 to 1 and value T from 0 to 25. From the results in Fig. 3, we know that when the value of each hyperparameter changes, the classification accuracy remains relatively stable.

Visualization. We present the t-SNE visualizations of the extracted class-specific features in Fig. 2. For the ERM method, the features of different classes are mixed, especially in the Cartoon and Sketch domain. In contrast, LeFD blends features of the same class but from different domains, while effectively separating features of different classes. This observation provides intuitive evidence that LeFD can effectively extract class-specific features that are robust to domain discrepancies.

Table 4. Ablation study on PACS. The model is trained on one domain (name in column) and tested on the other three domains. "DE", "LE" and "RA" denote Domain Expansion, Label Expansion and Rationale Alignment respectively.

DE	LE	RA	Art	Cartoon	Photo	Sketch	Avg.
			70.90	76.50	42.20	53.10	60.70
✓			71.93	73.77	58.27	56.26	65.06
	✓		76.58	77.18	60.91	56.42	67.77
✓		✓	75.68	76.94	63.81	63.05	69.87
✓	✓		78.83	80.38	63.50	62.38	71.27
✓	✓	✓	**81.04**	**82.84**	**68.42**	**65.92**	**74.56**



(a) Sensitivity to λ (T=20) (b) Sensitivity to T (λ=0.001)

Fig. 3. Sensitivity analysis on λ and T. Experiments are conducted on PACS using the Photo domain as the source domain and rest of three domains as the target domains.

5 Conclusion

In this paper, we present LeFD, a straightforward but efficient method to learn class-specific features in tackling SDG problems. Unlike existing class-specific feature learning methods, which directly enlarge the gap between the features learned by two separate branches in the absence of domain labels, LeFD manually constructs domain labels and explicitly extracts integrated domain and class features through label expansion. Subsequently, LeFD performs feature debiasing via rationale alignment to disentangle class-specific features from the integrated features. Extensive experiments in SDG and DG scenarios prove the superiority of the proposed LeFD. Nonetheless, there are still several worthy research directions for future work: 1) Our research is currently only based on the classification task, and more research on segmentation, detection, generation and other tasks is worth looking forward to. 2) We can try to explore more on different modalities (such as point cloud and text) and extend LeFD into a modality-agnostic method in solving SDG problems.

Acknowledgements. This study was supported by the NSFC (grant no. 42201460), the National Key Research and Development Program of China (grant no. 2023YFB3906600).

References

1. Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., Vaughan, J.W.: A theory of learning from different domains. Mach. Learn. **79**, 151–175 (2010)
2. Carlucci, F.M., D'Innocente, A., Bucci, S., Caputo, B., Tommasi, T.: Domain generalization by solving jigsaw puzzles. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2229–2238 (2019)
3. Chen, J., Gao, Z., Wu, X., Luo, J.: Meta-causal learning for single domain generalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7683–7692 (2023)
4. Chen, L., Zhang, Y., Song, Y., Van Den Hengel, A., Liu, L.: Domain generalization via rationale invariance. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 1751–1760 (2023)
5. Chen, S., Wang, L., Hong, Z., Yang, X.: Domain generalization by joint-product distribution alignment. Pattern Recogn. **134**, 109086 (2023)
6. Chen, T., Baktashmotlagh, M., Wang, Z., Salzmann, M.: Center-aware adversarial augmentation for single domain generalization. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 4157–4165 (2023)
7. Cheng, S., Gokhale, T., Yang, Y.: Adversarial Bayesian augmentation for single-source domain generalization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 11400–11410 (2023)
8. Choi, S., Das, D., Choi, S., Yang, S., Park, H., Yun, S.: Progressive random convolutions for single domain generalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10312–10322 (2023)
9. Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: AutoAugment: learning augmentation strategies from data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 113–123 (2019)
10. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V.: RandAugment: practical automated data augmentation with a reduced search space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp. 702–703 (2020)
11. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. In: International Conference on Machine Learning, pp. 1180–1189. PMLR (2015)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
13. Hemati, S., Zhang, G., Estiri, A., Chen, X.: Understanding hessian alignment for domain generalization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 19004–19014, October 2023
14. Hendrycks, D., Dietterich, T.G.: Benchmarking neural network robustness to common corruptions and perturbations. In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, 6–9 May 2019. OpenReview.net (2019). https://openreview.net/forum?id=HJz6tiCqYm

15. Huang, Z., Wang, H., Xing, E.P., Huang, D.: Self-challenging improves cross-domain generalization. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12347, pp. 124–140. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58536-5_8
16. Hull, J.J.: A database for handwritten text recognition research. IEEE Trans. Pattern Anal. Mach. Intell. **16**(5), 550–554 (1994)
17. Kang, J., Lee, S., Kim, N., Kwak, S.: Style neophile: constantly seeking novel styles for domain generalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7130–7140, June 2022
18. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4401–4410 (2019)
19. Koltchinskii, V.: Oracle inequalities in empirical risk minimization and sparse recovery problems: École D'Été de Probabilités de Saint-Flour XXXVIII-2008, vol. 2033. Springer, Cham (2011)
20. Krizhevsky, A., et al.: Learning multiple layers of features from tiny images (2009)
21. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998). https://doi.org/10.1109/5.726791
22. Lee, H., Hwang, S.J., Shin, J.: Self-supervised label augmentation via input transformations. In: International Conference on Machine Learning, pp. 5714–5724. PMLR (2020)
23. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.: Learning to generalize: meta-learning for domain generalization. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
24. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.M.: Deeper, broader and artier domain generalization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 5542–5550 (2017)
25. Li, L., et al.: Progressive domain expansion network for single domain generalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 224–233 (2021)
26. Motiian, S., Piccirilli, M., Adjeroh, D.A., Doretto, G.: Unified deep supervised domain adaptation and generalization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 5715–5725 (2017)
27. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning (2011)
28. Qiao, F., Zhao, L., Peng, X.: Learning to learn single domain generalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12556–12565 (2020)
29. Qu, S., Pan, Y., Chen, G., Yao, T., Jiang, C., Mei, T.: Modality-agnostic debiasing for single domain generalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 24142–24151 (2023)
30. Qu, S., et al.: Upcycling models under domain and category shift. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 20019–20028 (2023)
31. Volpi, R., Namkoong, H., Sener, O., Duchi, J.C., Murino, V., Savarese, S.: Generalizing to unseen domains via adversarial data augmentation. In: Advances in Neural Information Processing Systems, vol. 31 (2018)
32. Wan, C., et al.: Meta convolutional neural networks for single domain generalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4682–4691 (2022)

33. Wang, Z., Luo, Y., Qiu, R., Huang, Z., Baktashmotlagh, M.: Learning to diversify for single domain generalization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 834–843 (2021)
34. Wu, A., Deng, C.: Single-domain generalized object detection in urban scene via cyclic-disentangled self-distillation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 847–856, June 2022
35. Xu, X., Zhou, X., Venkatesan, R., Swaminathan, G., Majumder, O.: d-SNE: domain adaptation using stochastic neighborhood embedding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2497–2506 (2019)
36. Xu, Z., Liu, D., Yang, J., Raffel, C., Niethammer, M.: Robust and generalizable visual representation learning via random convolutions. In: International Conference on Learning Representations (2021). https://openreview.net/forum?id=BVSM0x3EDK6
37. Zagoruyko, S., Komodakis, N.: Wide residual networks. arXiv preprint arXiv:1605.07146 (2016)
38. Zhang, X., Zhou, L., Xu, R., Cui, P., Shen, Z., Liu, H.: Towards unsupervised domain generalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4910–4920, June 2022
39. Zhang, Z., et al.: Divide and contrast: source-free domain adaptation via adaptive contrastive learning. Adv. Neural. Inf. Process. Syst. **35**, 5137–5149 (2022)

Infrared and Visible Image Fusion Based on CNN and Transformer Cross-Interaction with Semantic Modulations

Yusu Zhang, Xiao-Jun Wu(✉), and Tianyang Xu

School of Artificial Intelligence and Computer Science, Jiangnan University, Wuxi, China
6213114006@stu.jiangnan.edu.cn, {wu_xiaojun,tianyang.xu}@jiangnan.edu.cn

Abstract. Convolutional Neural Networks (CNNs) have achieved success in the fusion of infrared and visible images, but they fall short in modeling long-range dependencies. In contrast, Transformers, with their global receptive field, demonstrate greater advantages in visual tasks. To jointly utilize local-global features and effectively leverage semantic information, we propose an infrared and visible image fusion method based on CNN and Transformer cross-interaction with semantic modulations (SMCFusion). Our approach integrates the strengths of CNNs and Transformers, enhancing the quality and interpretability of the fused images by introducing semantic features. We design a single-modality cross interaction module that facilitates the interaction between local and global features, and a cross-modality complementary mask fusion strategy for effective multi-modal feature fusion. The proposed semantic-oriented attention modulation improves semantic consistency during the fusion process. Experiments on multiple public datasets demonstrate that our SMCFusion outperforms other state-of-the-art methods in terms of visual quality and information retention.

Keywords: Infrared and Visible Image fusion · CNN and Transformer · Semantic Modulations

1 Introduction

Owing to the unique imaging properties of infrared and visible images, they exhibit complementarity in the information they provide. Fusing infrared and visible images can produce a single image that offers both clear background details and highlighted targets [1]. This fusion facilitates subsequent visual tasks, such as military applications, depth prediction, and facial recognition [2–4].

Infrared and visible image fusion is a crucial area of study, and numerous techniques have been developed to tackle this challenge. Early traditional image fusion methods mainly relied on mathematical transformations in the spatial

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 211–226, 2025.
https://doi.org/10.1007/978-3-031-78128-5_14

or transform domain to design fusion rules by manually analysing the activity level [5]. With the progress of deep learning, the domain of image fusion has investigated a range of deep learning-based approaches, leading to notable advancements. Among them, methods based on AE perform feature extraction and reconstruction using autoencoders, and achieve feature fusion through specific fusion strategies. Methods based on CNN utilize the strong feature extraction capabilities of CNN to make significant breakthroughs in image fusion. Additionally, methods based on GAN introduce adversarial mechanisms between generated and source images, achieving more realistic and credible fusion results through the competition between generators and discriminators.

Current deep learning-based image fusion methods are capable of producing high-quality fused images, but challenges remain. A significant limitation is that many of these algorithms rely on CNN architectures, which restrict the receptive field size. While methods based on Transformers improve the ability to capture global information, but there is significant redundancy between shallow and deep layers. Some research attempts to combine CNN and Transformer, but there is a lack of effective interaction mechanisms between them. Meanwhile, current approaches primarily emphasize the visual quality of the images while often overlooking the semantic content of the fused results, leading to weak semantic expression ability of features and insufficient understanding of image content by the network, which reduces the adaptability and robustness of the algorithm under different environmental conditions. To tackle the challenges in fusing infrared and visible images, we propose an innovative approach: infrared and visible image fusion based on CNN and Transformer cross-interaction with semantic modulations, which named SMCFusion. The key contributions of our study are summarized as follows:

- To explore the integration potential of CNN and Transformer, we designed them as parallel frameworks to extract fine-grained details of visible and target intensity of infrared. We proposed a single-modality cross interaction module to combine fine-grained and holistic features, promoting interaction between feature sub-networks, improving single-modality feature maps, and laying the foundation for subsequent high-quality multi-modal fusion.
- For the fusion of multi-modal features, we developed cross-modality complementary mask fusion to reduce redundant information and integrate effective information fully. Moreover, we introduced semantic features into the fused features using a powerful semantic knowledge base, forming semantic attention through interaction with the fused features, enhancing the expression of source image scene semantics by the fused features, and thereby improving the fusion performance.
- Comprehensive loss functions are formulated to take full advantage of the rich information in the input images, ensuring that key features and complex textures are preserved in the fused output.
- By conducting comprehensive tests across three widely used datasets, SMCFusion surpassed nine notable approaches in both quantitative and qualitative assessments.

2 Related Work

Deep learning-based image fusion techniques commonly employ network architectures such as autoencoders (AE), convolutional neural networks (CNN), and generative adversarial networks (GAN) [6]. DenseFuse [7] is a typical method based on AE, which applies dense blocks to the network structure to enhance feature representation and retain a large amount of useful intermediate information. Ma et al.'s PMGI [8] method considers the loss function perspective, adjusting the proportion of intensity and texture information preservation to achieve good fusion effects. FusionGAN [9] is the first model to apply generative adversarial networks to the field of infrared and visible image fusion, establishing an adversarial game between the generator and the discriminator, where the generator aims to generate the fused image, and the discriminator aims to force the fused image to have more details from the visible image [9]. There are also some fusion methods based on transformers, such as SwinFusion [10], a universal image fusion framework based on cross-domain remote learning and Swin Transformer proposed by some researchers, which achieves full integration of complementary information and global interaction. TgFuse [11] combines Transformer and generative adversarial network technologies, with a Transformer-based fusion module generating the fused image, which is then fed into the discriminator to calculate the loss function to improve the fusion effect. Recently, some methods have also started to focus on whether the fused image can promote downstream tasks, with these methods cascading fusion and downstream tasks. Liu et al. [12] proposed a dual-layer optimization form for joint fusion and detection problems, and developed it into a target-aware dual adversarial learning (TarDAL) network for fusion.

3 Methodology

3.1 Overall Framework

Our general framework of SMCFusion is shown in Fig. 1. For the pair of aligned infrared image $I_{ir} \in \mathbb{R}^{H \times W \times 1}$ and visible image $I_{vi} \in \mathbb{R}^{H \times W \times 3}$, we use two parallel feature extraction sub-networks to capture the detailed features and global features of both modalities. In order to aggregate the comprehensive features of a single modality extracted from the two sub-networks, we propose the single-modality cross interaction module (SCIM), which enhances the communication of the feature maps acquired by the different sub-networks at the same scale, respectively, and integrates the feature maps at different scales, so as to obtain the single modality rich and comprehensive feature information of a single modality. Secondly, in order to effectively fuse multi-modal features, we propose cross-modality complementary mask fusion (CCMF) strategy. In addition, we design a semantic-oriented attention modulation module (SAM), which provides more semantic attention to the fused features to ensure that the generated fusion results are more consistent with the semantic content of the image and reduce the presence of artefacts. Finally, we use feature reconstruction to obtain the fusion result $I_f \in \mathbb{R}^{H \times W \times 3}$.

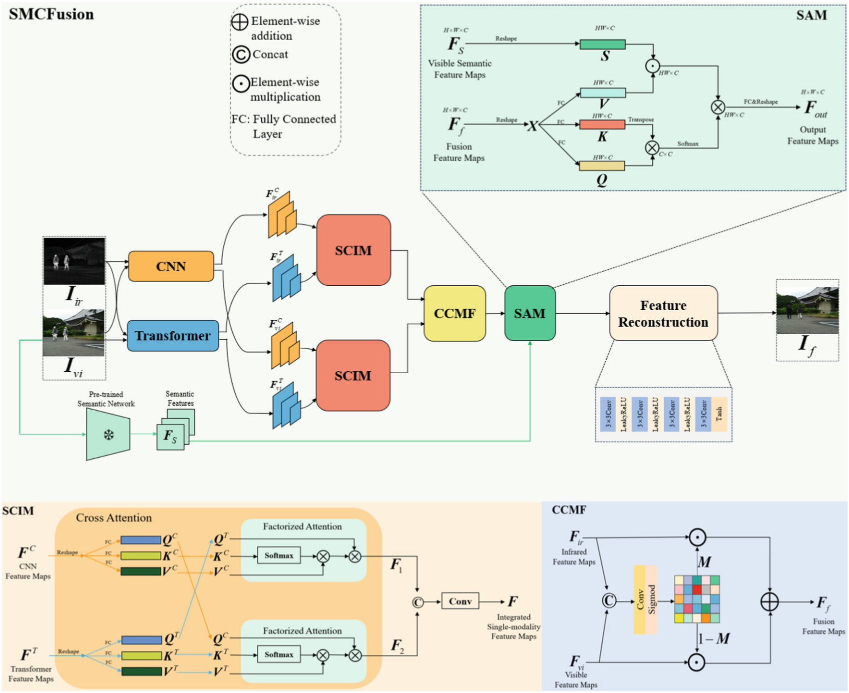


Fig. 1. Framework of the proposed SMCFusion. For the parallel extraction of single-modality features, a dual-branch structure is utilized to capture localized and overall features, respectively. Concurrently, a single-modality cross interaction module (SCIM) is constructed to effectively aggregate features. For multi-modal feature fusion, a cross-modality complementary mask fusion (CCMF) and semantic-oriented attention modulation module (SAM) are proposed to enhance the representation of fused features. Finally, feature reconstruction is applied to generate the final fused image.

3.2 Local and Global Feature Extraction

Inspired by the ability of convolutional neural networks to extract local information and the ability of Transformers to acquire long range dependencies and model global images, we propose to use two sub-networks to acquire local and global feature maps of the source image pair in parallel. Taking the input infrared image I_{ir} as an example, the local feature F_{ir}^{C} is obtained through the CNN branch, and the global feature F_{ir}^{T} is generated through the Transformer branch. In the CNN branch, we introduce the widely used DenseNet [13], which efficiently captures features in local regions of an image through its dense connectivity, thus enhancing the perception of image details. In the Transformer branch, we introduce the lightweight Lite Transformer [14], which utilises its long and short-range attention mechanism to efficiently process different ranges of information. By combining deep convolution and point-by-point convolution, Lite Transformer effectively reduces the parameter count while preserving performance, enabling efficient capture of global information from the source image. By combining CNN and Transformer, the ability to capture the full range of source image information is significantly improved.

3.3 Single-Modality Cross Interaction Module

How to make the feature maps extracted from different sub-networks communicate effectively within the equivalent scale is a key issue. To address this issue, we propose a SCIM that captures complementary information of the same image from two unique and different perspectives. This module introduces a cross-modal attention mechanism, allowing one branch to maintain its original features while insightfully integrating the representation of the other branch, thereby enhancing the overall feature expression capability. As shown in Fig. 1, we first reshape the feature maps extracted from the same image by CNN and Transformer, and then project them into query Q, key K, and value V vectors by three learned linear transformations, respectively. Then the computation of attention is executed. Note that the computational process of a single self-attention has significantly high complexity in both spatial and temporal aspects, leading to a significant increase in resource consumption. So we here draw on Factorized Attention as mentioned in the work [15]. Specifically, we approximate the attention map using a specific function and synchronously perform the computation of the second matrix multiplication (keys and values) [16]. For the cross-interaction of local and global features of the same image, we apply one branch of Q with another branch of K, V to implement the interaction computation. Taking infrared image as an example, the above calculation process can be expressed as:

$$\mathrm{CrossAtt}\left(Q_{ir}^{C}, K_{ir}^{T}, V_{ir}^{T}\right) = \frac{Q_{ir}^{C}}{\sqrt{\mathrm{d}}}\left(\mathrm{softmax}\left(K_{ir}^{T}\right)^{\mathrm{T}} V_{ir}^{T}\right) \tag{1}$$

$$\mathrm{CrossAtt}\left(Q_{ir}^{T}, K_{ir}^{C}, V_{ir}^{C}\right) = \frac{Q_{ir}^{T}}{\sqrt{\mathrm{d}}}\left(\mathrm{softmax}\left(K_{ir}^{C}\right)^{\mathrm{T}} V_{ir}^{C}\right) \tag{2}$$

where T denotes the transpose operation, $Q_{ir}, K_{ir}, V_{ir} \in \mathbb{R}^{N \times d}$, $N = H \times W$, denote the sequence of input vectors of the infrared modality, and d is the number of dimensionality reduction, C denotes from CNN branch and T denotes from Transformer branch. Additionally, we merge the two outputs F_1 and F_2, after the cross-interaction by performing cascade and convolution operations to obtain the integrated single-modality image features $F_{ir}(F_{vi})$.

3.4 Cross-Modality Complementary Mask Fusion

After obtaining interactive infrared and visible features in the above steps, we perform multi-modal feature fusion, aiming for the fused features to retain both the texture details of the visible modality and the prominent thermal target from the infrared modality. Common fusion strategies, such as directly adding the features of two modalities, do not adequately measure the amount of effective information in each modal feature, resulting in the omission of some important information. In addition, the operation of direct addition may introduce redundant information between features, thereby reducing network performance. Therefore we propose a more refined and effective fusion strategy called cross-modality complementary mask fusion that is able to transform the information we are interested in from both modalities into the fused image, the structure of CCMF is shown in Fig. 1. We first concatenate the interactive features F_{ir} and F_{vi} from both modalities, followed by passing them through convolutional layers and a sigmoid activation to generate an attention map M. M is then used to weight the input features, which are subsequently summed to produce the fused feature F_f. This process is mathematically represented by the following equation:

$$F_f = F_{ir} \odot M + F_{vi} \odot (1 - M) \tag{3}$$

3.5 Semantic-Oriented Attention Modulation

Large semantic knowledge bases can improve the representation capability of the network as they contain rich semantic information that can help the network to better understand and process the input data. Here, motivated by the idea of enhancing important features from semantic contexts, We incorporate visible image semantic features into the fusion features to provide richer and more profound semantic details, thereby improving the network's capacity to understand image content. Inspired by [17], SAM saves computational cost by considering each pixel of an image as a marker instead of the traditional pixel-based self-attention computation. First, we reshape the input fusion feature $F_f \in \mathbb{R}^{H \times W \times C}$ into $Z \in \mathbb{R}^{HW \times C}$. Z is then split into m heads, represented as $Z = [Z_1, Z_2, ..., Z_m]$, where $Z_i \in \mathbb{R}^{HW \times d_m}$, $d_m = \frac{C}{m}$, with i ranging from 1 to m, which is illustrated in Fig. 1 for the case of $m = 1$. Each head undergoes a linear projection to form a query element $Q_i \in \mathbb{R}^{HW \times d_m}$, a key element $K_i \in \mathbb{R}^{HW \times d_m}$, and a value element $V_i \in \mathbb{R}^{HW \times d_m}$ through three separate fully connected layers without bias:

$$Q_i = Z_i {W^{Q_i}}^{\mathrm{T}}, K_i = Z_i {W^{K_i}}^{\mathrm{T}}, V_i = Z_i {W^{V_i}}^{\mathrm{T}} \tag{4}$$

where W^{Q_i}, W^{K_i} and $W^{V_i} \in \mathbb{R}^{d_m \times d_m}$ represent the learnable parameters of the fully connected layers, and T signifies the matrix transpose. We note that visible images provide abundant visual information that enhances the ability of the network to recognize and understand objects, scenes, and structures within the images. Therefore, we incorporate the pre-trained semantic network [18] with multi-scale semantic features F_S extracted from visible images, F_S encodes the contextual information and is fed into SAM to provide more semantic attention to the fused multi-modal features. To match the shape of Z, we reshape $F_S \in \mathbb{R}^{H \times W \times C}$ as $S \in \mathbb{R}^{HW \times C}$ and divide it into m heads: $S = [S_1, S_2, ..., S_m]$, where

$S_i \in \mathbb{R}^{HW \times d_m}$, $d_m = \frac{C}{m}$ and $i = 1, 2, ..., m$. The self-attention mechanism for each head i is then given by the following equation:

$$Att\left(Q_i, K_i, V_i, S_i\right) = \left(S_i \odot V_i\right) \operatorname{softmax}\left(\frac{{K_i}^{\mathrm{T}} Q_i}{c_i}\right) \tag{5}$$

where c_i is a learnable parameter used to adaptively scale the result of the matrix multiplication to fit different ranges of attention weights. The results from the m heads are then combined using a fully connected layer and reshaped to generate the final output feature, $F_{out} \in \mathbb{R}^{H \times W \times C}$. By introducing the semantic features of visible modality into the fusion features, the fusion features can be made to have stronger semantic expression capability, thus improving the performance of the network in the image fusion task.

3.6 Loss Function

For the final fusion result, we want it to retain the thermal target information similar to the infrared image and the rich detail information of the visible image. Therefore we use intensity loss L_{int} and texture loss L_{tex} as follows:

$$L_{int} = \lambda_1 \frac{1}{HW} \left\|I_f - I_{ir}\right\|_1 + \lambda_2 \frac{1}{HW} \left\|I_f - I_{vi}\right\|_1 \tag{6}$$

$$L_{tex} = \lambda_1 \frac{1}{HW} \left\|\bigtriangledown I_f - \bigtriangledown I_{ir})\right\|_2^2 + \lambda_2 \frac{1}{HW} \left\|\bigtriangledown I_f - \bigtriangledown I_{vi})\right\|_2^2 \tag{7}$$

Here, H and W represent the height and width of the image, respectively. $\bigtriangledown$ indicates a gradient operator to obtain gradient information of the image. λ_1 and λ_2 are adaptive perceptual weights that determine how closely the fused image resembles each of the source images. Higher weights indicate greater level of resemblance, which corresponds to increased information retention from the source images. L_{int} constrains the generated image to have a similar histogram distribution as the source image, and L_{tex} constrains the generated image to have a fine texture. Therefore, these two losses essentially constrain the proportion of texture and intensity of the source image that is retained in the generated image. To enable the network to dynamically assign the appropriate information metric for retaining intensity and gradient details from the source images, we use the loss term weights to select the most relevant and significant information to be preserved in the fusion outcome. Inspired by the perceptual loss [19], we use VGG-19 to extract the features of the source images in order to evaluate the degree of information contained in it, and finally obtain two weight values (λ_1, λ_2) by the softmax function to map $(\mathrm{In}_1, \mathrm{In}_2)$ to real numbers between 0 and 1.

$$\mathrm{In}_\gamma = \frac{1}{5} \sum_{i=1}^{5} \frac{1}{\mathrm{H}_i W_i C_i} \sum_{j=1}^{C_i} \left\|\nabla \mathrm{f}_j^k\left(I\right)\right\|_{\mathrm{F}}^2, \gamma \in (1, 2) \tag{8}$$

$$\left(\lambda_1, \lambda_2\right) = \operatorname{softmax}\left(\frac{\mathrm{In}_1}{\partial}, \frac{\mathrm{In}_2}{\partial}\right) \tag{9}$$

We calculate the degree of information In using the feature maps from the outputs of the five convolutional layers in the VGG-19 model [20]. In the above formula, $f_j\left(I\right)$ denotes the feature maps obtained from the convolutional layer preceding the jth max-pooling layer. ∂ is a predefined constant used to adjust the weight values for more effective weight allocation.

The color of an image is often linked to its fine details. By managing the color distribution in the fused image, we can preserve the realism and naturalness of the colors. We use the discrete cosine distance [21] to regulate the color distribution of the fused image, ensuring it closely matches the source image while incorporating some intensity from the visible image. The color consistency loss is defined by the following formula:

$$L_{color} = \frac{1}{HWC}\sum_{z\in\varepsilon} cos(I_f^z, I_{vi}^z), \varepsilon = \{R, G, B\} \tag{10}$$

where z is an element of R, G, B and $cos(,)$ denotes the pixel-level discrete cosine similarity calculation between the fused image and visible image in the colour channel, the above equation is calculated by summing and normalising the similarity of the three channels in order to arrive at the final colour loss value. By constraining through L_{color}, it is possible to reduce color distortion and ensure that the resulting fused image is more consistent with the enhanced visible image in terms of color distribution.

The total loss function L_{total} of SMCFusion is denoted as:

$$L_{total} = L_{int} + L_{tex} + \alpha L_{color} \tag{11}$$

where α is the hyperparameter that balances the three losses.

4 Experiments

4.1 Experimental Configurations

To comprehensively evaluate our SMCFusion, we conducted extensive qualitative and quantitative experiments on MSRS, LLVIP and RoadScene datasets. We compared our method with nine state-of-the-art methods including one conventional fusion method, two AE-based fusion methods, one GAN-based fusion method, two generic CNN-based fusion methods and three Transformer-based fusion methods, which are: CBF [22], DenseFuse [7], FusionGAN [9], NestFuse [23], U2Fusion [19], SwinFusion [10], YDTR [24], CDDFuse [25] and TgFuse [11]. The image fusion algorithms used for comparison are all publicly available and are configured according to the parameters in their original papers. We selected six objective evaluation metrics to quantitatively assess the fusion performance of the algorithm, including structural similarity (SSIM) [26], Fusion Quality (Qabf) [27], Correlation Coefficient (CC) [28], sum of correlation coefficients (SCD) [29], entropy (EN) [30], and Pixel Feature Mutual Information ($\text{FMI}_{\text{pixel}}$) [31]. SSIM and Qabf effectively assess the visual quality and preservation of image details.

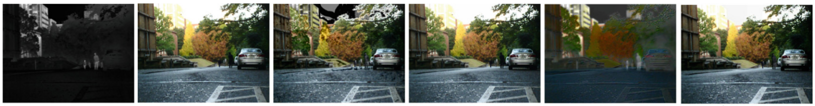
CC and $\text{FMI}_{\text{pixel}}$ quantify how well information is shared and retained between the fused and source images. SCD and EN reflect the relevance and informativeness of the image. Additionally, Higher values of SSIM, Qabf, CC, SCD, EN and $\text{FMI}_{\text{pixel}}$ signify superior fusion performance.

4.2 Implementation Details

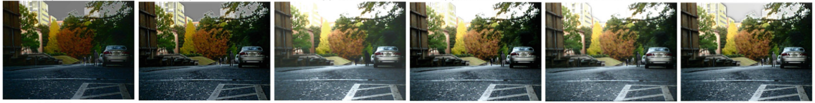
We trained SMCFusion using the MSRS dataset, normalizing all images to the range [0,1] before feeding them into the network. We expanded the training set by data enhancement means such as cropping. In our experiments, we used a batch size of 8 and trained the model for 200 epochs. The learning rate of the Adam optimizer was initialized to 0.001 and decayed exponentially. For the hyperparameters in Eq. (11), we set $\alpha = 1.5$. Notably, similar to most multimodal fusion approaches [10], all our fusion operations are performed on the Y-channel. Specifically, we convert the RGB visible image to the YCbCr color space. After performing the fusion on the Y channel, we combine it with the Cb and Cr channels of the original visible image to create the final color-fused output. The method we developed was implemented using PyTorch, and all experiments were executed on an NVIDIA RTX3090 GPU.

4.3 Comparative Experiment

To illustrate the performance of SMCFusion, we performed a comprehensive comparison of fusion performance with nine SOTA fusion methods on the MSRS dataset, and presented the qualitative experimental results in Fig. 2, and reported the quantitative experimental results in Table 1.



(a) Infrared (b) Visible (c) CBF (d) DenseFuse (e) FusionGAN (f) NestFuse



(g) U2Fusion (h) SwinFusion (i) YDTR (j) CDDFuse (k) TgFuse (l) SMCFusion

Fig. 2. Visual comparison of SMCFusion relative to nine advanced methods on #01501D image from MSRS dataset.

Visually, our method provides more comprehensive scene details compared to other methods, such as clearer textures of buildings and edges of branches, and pedestrians are also highlighted to some extent. The image exhibits strong contrast and excellent visual clarity. Quantitative analysis shows that all metrics reached optimal levels, suggesting that the images produced by SMCFusion are rich in information while preserving high structural similarity and strong correlation with the original images. The color and brightness information is also closer

Table 1. Quantitative results of the six metrics on MSRS dataset. The top-performing result is indicated in bold, and the second-best result is marked with an underline.

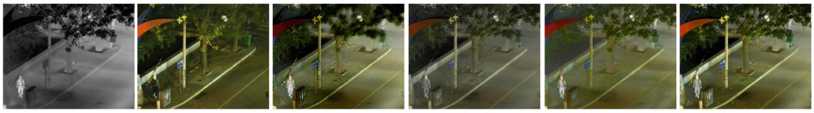
	CBF	DenseFuse	FusionGAN	NestFuse	U2Fusion	SwinFusion	YDTR	CDDFuse	TgFuse	SMCFusion
SSIM	0.7047	0.9391	0.5098	0.9440	0.9111	0.9488	0.9377	0.9398	0.9444	**0.9614**
Qabf	0.5571	0.6954	0.0703	0.6406	0.4872	0.6126	0.6739	0.6589	0.6505	**0.7040**
CC	0.4376	0.5088	0.5205	0.5326	0.4893	0.5054	0.5093	0.5346	0.5431	**0.6984**
SCD	0.8603	1.0007	0.6310	1.2881	1.4559	1.3625	0.9864	1.2633	1.3377	**1.7190**
EN	7.4370	7.4654	5.8167	7.5715	6.7451	7.0750	7.2611	7.5238	7.4689	**7.6335**
$\text{FMI}_{\text{pixel}}$	0.8554	0.8925	0.8320	0.8894	0.8715	0.8805	0.8790	0.8890	0.8903	**0.9516**

to the source images, ensuring high visual quality. This performance improvement is mainly due to the joint feature extraction by CNN and Transformer, which fully extracts the deep information of single modalities. The injection of semantic information into the fusion features enhances the model's perception of the scene, and the intensity loss and texture loss based on adaptive weight adjustment of the richness of source image information ensure that the fusion image has good visual perception.

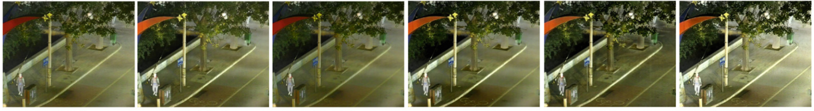
4.4 Generalization Experiment

Evaluation of generalization performance is crucial in deep learning methods. In order to validate the generalisation ability of SMCFusion in different scenarios, we performed experiments using the LLVIP and RoadScene datasets. It should be emphasised that SMCFusion was trained on the MSRS dataset and directly applied to the other two dataset for testing.

Figure 3 and Fig. 4 present visual comparison results of different algorithms on the LLVIP and RoadScene datasets. CBF and FusionGAN lose most of the texture information, resulting in blurred images with poor visual quality. DenseFuse, U2Fusion, and YDTR exhibit weaker saliency of targets such as pedestrians and cars, and the texture retention of background information, such as bike landmarks on nighttime roads, road signs, and the contours of trees across the road, is not clear. Compared to the aforementioned algorithms, SwinFusion, CDDFuse, and TgFuse can highlight infrared targets and show certain details, but the overall image of TgFuse is more biased towards visible images. The fusion results generated by our SMCFusion emphasize prominent pedestrians and vehicles while also effectively capturing the texture of road landmarks. Moreover, the trees across the road have clear edges, and the colors appear relatively natural. We attribute the superior fusion quality of SMCFusion to three primary factors. First, we extract rich source image features from two perspectives, with SCIM facilitating communication between feature sub-networks and CCMF achieving extensive multimodal feature integration. Second, SAM enhances the model's scene perception capability, improving the understanding and representation of scene features while preserving key information and details. Third, a comprehensive loss function constrains the contrast, texture details, and color naturalness of the fused image, significantly improving visual quality and detail retention.

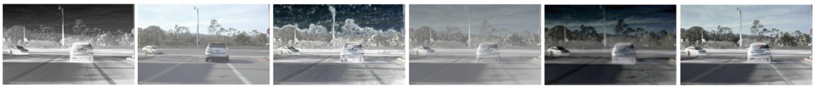


(a) Infrared (b) Visible (c) CBF (d) DenseFuse (e) FusionGAN (f) NestFuse

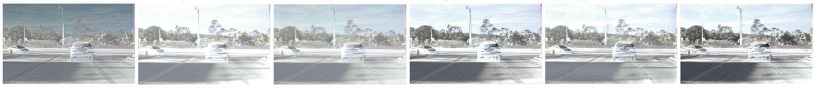


(g) U2Fusion (h) SwinFusion (i) YDTR (j) CDDFuse (k) TgFuse (l) SMCFusion

Fig. 3. Visual comparison of SMCFusion relative to nine advanced methods on #070220 image from LLVIP dataset.



(a) Infrared (b) Visible (c) CBF (d) DenseFuse (e) FusionGAN (f) NestFuse



(g) U2Fusion (h) SwinFusion (i) YDTR (j) CDDFuse (k) TgFuse (l) SMCFusion

Fig. 4. Visual comparison of SMCFusion relative to nine advanced methods on #00211 image from RoadScene dataset.

To quantitatively evaluate our method, we selected 25 pairs of images from each of the LLVIP and RoadScene datasets for testing. The corresponding results are shown in Table 2 and Table 3. These tables reveal that SMCFusion outperforms on both datasets. Considering both qualitative and quantitative analysis, SMCFusion demonstrates outstanding generalization capabilities and strong competitiveness in image fusion.

Table 2. Quantitative results of the six metrics on LLVIP dataset. The top-performing result is indicated in bold, and the second-best result is marked with an underline.

	CBF	DenseFuse	FusionGAN	NestFuse	U2Fusion	SwinFusion	YDTR	CDDFuse	TgFuse	SMCFusion
SSIM	0.8514	0.9441	0.7540	0.9353	0.9492	0.9087	0.8995	0.9533	0.9381	**0.9554**
Qabf	0.3206	0.3714	0.1075	0.4982	0.4551	0.5689	0.2231	0.5877	0.4376	**0.5914**
CC	0.5635	**0.7061**	0.5856	0.6515	0.6939	0.6466	0.6844	0.6602	0.6265	0.6982
SCD	0.6863	1.2448	0.7542	1.6123	1.3732	1.4260	1.0559	1.5996	1.2618	**1.6134**
EN	6.0748	6.3291	6.0773	6.7267	6.1951	6.7375	6.0346	**6.9460**	6.7244	6.8988
$\mathrm{FMI}_{\mathrm{pixel}}$	0.9181	0.9192	0.8992	0.9212	0.9146	0.9166	0.9174	0.9209	0.9274	**0.9381**

4.5 Ablation Studies and Discussion

Single-Modality Cross Interaction Module Analysis. To explore the impact of the SCIM on image feature aggregation and interaction communication, we designed a control group experiment. In the control group experi-

Table 3. Quantitative results of the six metrics on RoadScene dataset. The top-performing result is indicated in bold, and the second-best result is marked with an underline.

	CBF	DenseFuse	FusionGAN	NestFuse	U2Fusion	SwinFusion	YDTR	CDDFuse	TgFuse	SMCFusion
SSIM	0.7727	0.8661	0.5005	0.8549	0.8377	0.8742	0.9087	0.8686	**0.9318**	0.9226
Qabf	0.4492	0.3174	0.1187	0.5105	0.4376	0.4098	0.4611	0.4756	0.4629	**0.5525**
CC	0.1804	0.3670	0.2239	0.3015	0.3351	0.4466	0.4215	0.3128	0.4087	**0.5135**
SCD	1.4584	**1.8499**	1.6581	1.7259	1.7258	1.6903	1.6722	1.7626	1.7102	1.7686
EN	7.0526	6.2592	7.0784	7.2332	6.5712	6.8826	7.1102	**7.3140**	7.0367	7.2572
FMI_{pixel}	0.8224	0.8599	0.7964	0.8778	0.8535	0.8790	0.8878	0.8644	0.8900	**0.8928**

ment, after the two sub-networks extract the single-modality features, the module is not used for information interaction and aggregation. Instead, the two single-modality features are directly fused, and the other structures of the network remain unchanged. It can be seen in Fig. 5 that compared to our method, the images generated by the control group contain insufficient information, the pedestrians are not prominent, and the edge details of the bushes are also relatively blurry. This suggests that the module is capable of enhancing the representational ability of image features and the integration of information.

Table 4. Quantitative results of the ablation study on the MSRS dataset. The top-performing result is indicated in bold, and the second-best result is marked with an underline.

	SSIM	Qabf	CC	SCD	EN	FMI_{pixel}
w/o SCIM	0.8797	0.6153	0.5098	1.5982	7.0500	0.8316
w/o SAM	0.8770	0.5367	0.5703	1.5008	7.2179	0.8454
Addition	0.9110	0.6029	0.5423	1.5477	7.1988	0.8767
ours(SMCFusion)	**0.9614**	**0.7040**	**0.6984**	**1.7190**	**7.6335**	**0.9516**

Cross-Modality Complementary Mask Fusion Analysis. To validate the effectiveness of the cross-modality complementary mask fusion strategy, we designed addition fusion strategy for comparison with our SMCFuison, and we named as Addition in Fig. 5. Notably, the only modification made was to the fusion strategy within the network. By comparing the ablation experiment results in Table 4, it becomes clear that SMCFusion shows a substantial improvement across all evaluation metrics. This suggests that CCMF more effectively leverages the complementary information between visible and infrared images, thereby enhancing the fusion quality.

Semantic-Oriented Attention Modulation Analysis. SAM can embed the extracted semantic features into the fused features, thereby providing the model

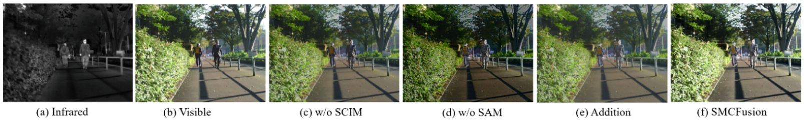


Fig. 5. Comparison of visual quality in the ablation study for key modules.

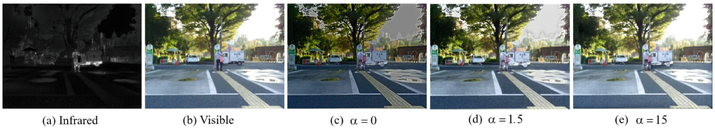


Fig. 6. Comparison of visual quality in the ablation study for important loss.

with richer semantic focus points, enhancing the expressiveness of the multi-modal fusion features at the semantic level, and thus improving the fusion effect of the model. To verify the effectiveness of this design, we conducted an ablation experiment comparing the performance of the network model without semantic features and the complete fusion network integrated with semantic features. The experimental results in Fig. 5 demonstrate that the fused images produced by the network incorporating semantic features from visible images offer a more comprehensive and detailed representation of the scene, with colors and brightness that closely resemble the source images, leading to impressive visual effects. This further confirms the role and value of the semantic-oriented attention modulation module.

Color Consistency Loss and Hyperparameter Analysis. For the color consistency loss, the parameter α is utilized to adjust the level of color correction and determine the amount of visible information introduced. To investigate how the value of α affects the fusion outcome, a series of experiments were conducted, with the results displayed in Fig. 6. When α is set to 0, which implies the absence of color consistency loss, the fusion result tends to favor the characteristics of the infrared image, leading to a degradation in color. As the value of α gradually increases, the fused image more comprehensively retains background information; however, the contrast of thermal targets diminishes. When α is set to 1.5, a balance between image clarity and contrast is achieved, resulting in a natural and authentic fusion effect.

Table 5. Segmentation results on the MSRS dataset. The best result is highlighted in bold.

	Background	Car	Person	Bike	Curve	Car stop	Guardrail	Color cone	Bump	**mIoU**
Infrared	98.085	89.632	69.319	69.650	48.338	61.532	57.340	54.635	65.998	68.325
Visible	98.243	90.110	64.366	69.767	53.182	68.974	74.486	60.495	69.781	72.156
Fuse	**98.394**	**90.183**	**73.565**	**70.686**	**55.646**	**69.424**	**76.123**	**60.960**	**73.050**	**74.226**

Segmentation Performance. To explore whether the fused images generated by SMCFusion can promote the performance improvement of subsequent visual tasks, we trained the SegFormer [32] model on both the source images and the fused images separately, and compared the semantic segmentation performance. Table 5 presents the quantitative results of segmentation. We utilize the mIoU evaluation metric, a standard in semantic segmentation tasks, representing the mean intersection and merger ratio. It is evident that SMCFusion effectively integrates valuable information from the source images, and the modulation of semantic attention contributes to enriching the semantic content of the fused images, leading to improved segmentation performance.

5 Conclusion

We propose an infrared and visible image fusion framework named SMCFusion. To fully integrate local features and global information, we adopt the design concept of parallel feature extraction by CNN and Transformer. The single-modality cross interaction module focuses on the complementarity between local details and global objectives, achieving fine interaction between features. The cross-modality complementary mask fusion strategy is dedicated to mining the complementary information between images to achieve more comprehensive and accurate fusion. Moreover, the semantic-oriented feature modulation injects semantic information into the fused features, enabling the network to generate fusion images with more semantic consistency and visual coherence. We conducted extensive experimental validation on publicly available datasets, and the results demonstrate that our proposed SMCFusion significantly enhances image fusion quality and performance when compared to published competitors.

Acknowledgements. This work is supported in part by the National Key Research and Development Program of China (2023YFF1105102, 2023YFF1105105), the National Natural Science Foundation of China (Grant NO. 62020106012, 62332008, 62106089, U1836218, 62336004), the 111 Project of Ministry of Education of China (Grant No. B12018).

References

1. Li, S., Kang, X., Fang, L., Hu, J., Yin, H.: Pixel-level image fusion: a survey of the state of the art. Inf. Fusion **33**, 100–112 (2017)
2. Tianyang, X., Zhu, X.-F., Xiao-Jun, W.: Learning spatio-temporal discriminative model for affine subspace based visual object tracking. Visual Intell. **1**(1), 4 (2023)
3. Xu, T., Wu, X.-J., Kittler, J.: Non-negative subspace representation learning scheme for correlation filter based tracking. In: 2018 24th International Conference on Pattern Recognition (ICPR), pp. 1888–1893. IEEE (2018)
4. Jia, Z., Sun, S., Liu, G., Liu, B.: MSSD: multi-scale self-distillation for object detection. Visual Intell. **2**(1), 8 (2024)
5. Dogra, A., Goyal, B., Agrawal, S.: From multi-scale decomposition to non-multi-scale decomposition methods: a comprehensive survey of image fusion techniques and its applications. IEEE Access **5**, 16040–16067 (2017)

6. Zhang, H., Han, X., Tian, X., Jiang, J., Ma, J.: Image fusion meets deep learning: a survey and perspective. Inf. Fusion **76**, 323–336 (2021)
7. Li, H., Xiao-Jun, W.: DenseFuse: a fusion approach to infrared and visible images. IEEE Trans. Image Process. **28**(5), 2614–2623 (2018)
8. Zhang, H., Han, X., Xiao, Y., Guo, X., Ma, J.: Rethinking the image fusion: a fast unified image fusion network based on proportional maintenance of gradient and intensity. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 12797–12804 (2020)
9. Ma, J., Wei, Yu., Liang, P., Li, C., Jiang, J.: FusionGAN: a generative adversarial network for infrared and visible image fusion. Inf. fusion **48**, 11–26 (2019)
10. Ma, J., Tang, L., Fan, F., Huang, J., Mei, X., Ma, Y.: SwinFusion: cross-domain long-range learning for general image fusion via swin transformer. IEEE/CAA J. Automatica Sinica **9**(7), 1200–1217 (2022)
11. Rao, D., Xu, T., Wu, X.-J.: TGFuse: an infrared and visible image fusion approach based on transformer and generative adversarial network. IEEE Trans. Image Process. (2023)
12. Liu, J., et al.: Target-aware dual adversarial learning and a multi-scenario multi-modality benchmark to fuse infrared and visible for object detection, pp. 5802–5811 (2022)
13. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708 (2017)
14. Wu, Z., Liu, Z., Lin, J., Lin, Y., Han, S.: Lite transformer with long-short range attention. arXiv preprint arXiv:2004.11886 (2020)
15. Xu, W., Xu, Y., Chang, T., Tu, Z.: Co-scale conv-attentional image transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9981–9990 (2021)
16. Bello, I.: LambdaNetworks: modeling long-range interactions without attention. arXiv preprint arXiv:2102.08602 (2021)
17. Cai, Y., Bian, H., Lin, J., Wang, H., Timofte, R., Zhang, Y.: Retinexformer: one-stage retinex-based transformer for low-light image enhancement. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 12504–12513 (2023)
18. Chen, L.-C., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587 (2017)
19. Han, X., Ma, J., Jiang, J., Guo, X., Ling, H.: U2Fusion: a unified unsupervised image fusion network. IEEE Trans. Pattern Anal. Mach. Intell. **44**(1), 502–518 (2020)
20. Sengupta, A., Ye, Y., Wang, R., Roy, K.: Going deeper in spiking neural networks: VGG and residual architectures. Front. Neurosci. **13**, 425055 (2019)
21. Strang, G.: The discrete cosine transform. SIAM Rev. **41**(1), 135–147 (1999)
22. Shreyamsha Kumar, B.K.: Image fusion based on pixel significance using cross bilateral filter. SIViP **9**, 1193–1204 (2015)
23. Li, H., Xiao-Jun, W., Durrani, T.: NestFuse: an infrared and visible image fusion architecture based on nest connection and spatial/channel attention models. IEEE Trans. Instrum. Meas. **69**(12), 9645–9656 (2020)
24. Tang, W., He, F., Liu, Y.: YDTR: infrared and visible image fusion via Y-shape dynamic transformer. IEEE Trans. Multimedia (2022)
25. Zhao, Z., et al.: CDDFuse: correlation-driven dual-branch feature decomposition for multi-modality image fusion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5906–5916 (2023)

26. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)
27. Piella, G., Heijmans, H.: A new quality metric for image fusion. In: Proceedings 2003 International Conference on Image Processing (Cat. No. 03CH37429), vol. 3, pp. III–173. IEEE (2003)
28. Larson, E.C., Chandler, D.M.: Most apparent distortion: full-reference image quality assessment and the role of strategy. J. Electron. Imaging **19**(1), 011006–011006 (2010)
29. Aslantas, V., Bendes, E.: A new image quality metric for image fusion: the sum of the correlations of differences. AEU-Int. J. Electron. Commun. **69**(12), 1890–1896 (2015)
30. Roberts, J.W., Van Aardt, J.A., Ahmed, E.B.: Assessment of image fusion procedures using entropy, image quality, and multispectral classification. J. Appl. Remote Sens. **2**(1), 023522 (2008)
31. Haghighat, M., Razian, M.A.: Fast-FMI: non-reference image fusion metric. In: 2014 IEEE 8th International Conference on Application of Information and Communication Technologies (AICT), pp. 1–3. IEEE (2014)
32. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J.M., Luo, P.: SegFormer: simple and efficient design for semantic segmentation with transformers. Adv. Neural Inf. Process. Syst. **34**, 12077–12090 (2021)

Mining Long Short-Term Evolution Patterns for Temporal Knowledge Graph Reasoning

Fengyi Wang, Guanghui Zhu(✉), Haojun Hou, Chunfeng Yuan, and Yihua Huang

State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
{fengyiwang,haojunhou}@smail.nju.edu.cn, {zgh,cfyuan,yhuang}@nju.edu.cn

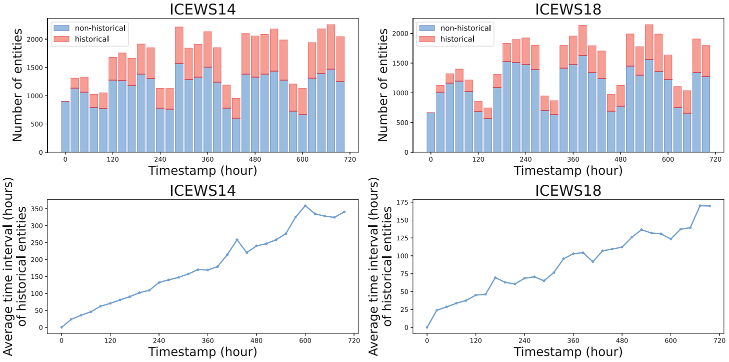
Abstract. Temporal knowledge graphs (TKGs), consisting of graph snapshots evolving over time, have attracted substantial research attention in various areas like recommender systems and relation networks. Although significant research efforts have been devoted to this field, the regularity and abruptness of graph evolution make it still challenging to model temporal dynamics in TKGs. Existing works only characterize the short-term evolution due to the limitations of modeling long sequences, or lack consideration of fine-grained differences between recent snapshots in time. To tackle the issues, we propose LSEN (**L**ong **S**hort-Term **E**volution Learning **N**etwork), an effective model that jointly captures short-term and long-term evolution patterns in TKGs. By introducing a short-term and a long-term evolution pattern mining module, we boost the memorization and generalization of LSEN for TKG reasoning. Specifically, the short-term module of LSEN simultaneously consider the topology structure at each moment and model the chronologically sequential effect across recent snapshots. In addition, we devise a long-term module leveraging the frequencies of constrained triple occurrences to explore long-term evolution patterns in the entire historical sequence. We conduct extensive experiments on five real-world datasets, and LSEN achieves state-of-the-art results, demonstrating the significant superiority of our method.

Keywords: Deep learning · Knowledge graph reasoning · Temporal knowledge graph

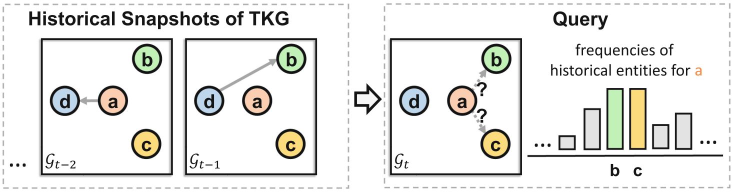
1 Introduction

Knowledge Graphs (KGs) [12], which provide a fundamental abstraction to complex network systems by viewing entities as nodes and their relations as edges, are applicable to numerous downstream tasks [38]. Recently, Temporal Knowledge Graphs (TKGs) [8] have been extensively researched and applied, which allow for a more elaborate investigation of the underlying dynamics in evolving graphs. As many real-world systems can be formulated as evolving KGs, TKG

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 227–242, 2025.
https://doi.org/10.1007/978-3-031-78128-5_15



(a) Recurrence of interactions in TKGs. Past events associated with the entities in current queries are called historical events, and otherwise non-historical events. Their corresponding entities are called historical and non-historical entities.



(b) An example of TKG reasoning. It is to be predicted that entity a will interact with either b or c at moment t. Entity b and entity c have historically been related to entity a with the same frequency.

Fig. 1. Importance of considering both long-term and short-term evolution patterns in TKGs.

reasoning [16] has achieved inspiring success in various fields like international relations [13], stock prediction [6], and recommendation systems [31].

Existing works on TKG reasoning typically obtain the structural information in each snapshot through GNNs (Graph Neural Networks) [14,30] and use RNNs (Recurrent Neural Networks) [3,11,22] to capture temporal dynamics in time-evolving graphs [18,19]. Some works consider encoding the recurrence and periodicity of interactions [33,37].

Despite the success of previous works, they are subject to significant problems because of their one-sidedness. Due to the shortcomings of RNNs in processing long sequences (e.g., gradient vanishing and error accumulation problems) [1,24], methods using RNNs are less likely to capture long-term evolution patterns such as the repetition and the regular occurrence of facts. Other sequential models such as Transformer [29] also only process recent temporal information due to the computational complexity and memory constraints, and thus cannot include the full history of sequences. As shown in Fig. 1a, we investigate the proportion of historical entities that are recurrent at several moments in TKG datasets. It can be observed that the repetition of facts is a common phenomenon and has a certain periodicity. Also, the mean time interval between the last appearance and the present of recurring entities increases over time, requiring that long-distance time dependencies based on frequencies should be considered.

On the other hand, models based on temporal feature encoding primarily focus on general evolution patterns in history and lack the consideration of subtle variations across moments. For example, in Fig. 1b, it is to be predicted that entity a will interact with either b or c at moment t. Entity b and entity c have historically been related to entity a with the same frequency. If the prediction is made based merely on the repetition frequencies of historical entities, it is hard to reach an accurate conclusion. But if the structural information of recent graph snapshots is considered, it is reasonable to perceive that both a and b have recently interacted with d, and therefore a should more likely be connected with b at t. This demonstrates that it is also undesirable to consider only long-term information and ignore recent patterns of evolution.

Recently, some work such as HGLS [36] considers both long- and short-term patterns of TKGs. However, HGLS ignores the significant importance of repetition frequencies for TKG reasoning. It only integrates long- and short-term representations through a simple gating mechanism without mining the complex coupling between them.

Based on the aforementioned insights, we make the first attempt to close the research gap by proposing a new model, LSEN[1] (short for ***L**ong* **S***hort-Term* **E***volution Learning* **N***etwork)*, for TKG reasoning. LSEN is capable of jointly modeling the long-term and short-term evolution patterns of TKGs. In the whole process, the output of the short-term evolution pattern mining module is used as the input of the long-term module. Specifically, after characterizing the short-term evolution patterns through GNNs and RNNs, LSEN further improves embeddings by innovatively integrating long-term evolution patterns. In the depiction of long-term evolution characteristics, LSEN takes full advantage of the frequencies of constrained triple occurrences, i.e., the frequencies with which entities have been associated with the query quadruple containing a missing entity. By recording and effectively processing such frequencies, our model is able to take into account the entire historical information in TKGs. Through the refinement of short-term characteristics via long-term frequencies, both long-term and short-term evolution patterns in TKGs are obtained by LSEN for better prediction. The contributions of this paper are summarized as follows:

- **New Findings.** We present the necessity of modeling both long-term and short-term evolution patterns for temporal knowledge graph reasoning. Moreover, it is essential to utilize the frequencies of interactions.
- **New Method.** In the proposed model LSEN, we devise specific modules for modeling short-term and long-term evolution patterns, respectively, and

---

[1] The code of LSEN is available at https://github.com/PasaLab/LSEN.git.

integrate them into a unified framework for end-to-end training and learning. A novel long-term evolution pattern mining module based on constrained triple occurrence frequencies is designed to improve embeddings. Our model is able to take into account the entire historical information for TKGs.
- **SOTA Performance.** We conduct extensive experiments on five real-world datasets and LSEN achieves the best results among all the comparable baselines. The experimental results demonstrate that the proposed method can improve the prediction performance of TKG reasoning.

2 Related Works

2.1 Static KG Reasoning Methods

To infer missing facts in static KGs, existing works focus on learning representations of entities and relations through GNNs, tensor factorization [15], and so on [26]. GNNs are widely applied for reasoning over KGs since they can effectively capture graph structural information and learn relation-aware representations. Such approaches include R-GCN [25], VR-GCN [35], CompGCN [28], etc.

2.2 Temporal KG Reasoning Methods

According to how temporal dynamics are modeled in TKGs, recent works on TKG reasoning can be broadly classified into two categories: models using RNNs to capture temporal dynamics in time-evolving graphs and models based on encoding the recurrence or periodicity of events.

Among RNN-based TKG methods, RE-GCN [19] leverages a relation-aware GCN (Graph Convolutional Network) [14] to model the structural dependencies of entities and relations within each moment and uses GRU (Gate Recurrent Unit) [3] to characterize the sequential patterns of evolving graphs. HisMatch [18] is also RNN-based, which proposes to model entities as queries or candidates and apply different encoders from the two perspectives.

As a method based on encoding the recurrence or periodicity of events, CyGNet [37] divides the entities to be predicted into two categories: those that have been historically related and those that have not. Processing the two types of entities differently actually distinguishes different entities in the time dimension by means of statistics. Similar to CyGNet, CENET [33] also distinctively considers seen or unseen entities as well as relations, and it proposes a contrastive learning framework to make predictions on TKGs. HIP [10] passes information from temporal, structural and repetitive perspectives, which are used to model the temporal evolution of events, the interactions of events at the same time step, and the known events respectively. EvoKG [23] captures the ever-changing structural and temporal dynamics in TKGs via recurrent event modeling, and models the interactions between entities based on the temporal neighborhood aggregation framework.

3 Problem Formulation

3.1 Static Knowledge Graph

A static knowledge graph $\mathcal{SKG}$ can be viewed as a collection of entities $\mathcal{E}$, relations $\mathcal{R}$, and facts $\mathcal{F}$. That is, $\mathcal{SKG} = \{\mathcal{E}, \mathcal{R}, \mathcal{F}\}$. Each fact in can be represented by a triple (e_s, r, e_0), where e_s and e_o are entities, and r is the relation type.

3.2 Temporal Knowledge Graph

A temporal knowledge graph can be regarded as a sequence of static KGs at different time, i,e., $\mathcal{TKG} = \{\mathcal{SKG}_1, \mathcal{SKG}_2, \cdots, \mathcal{SKG}_t\}$. The knowledge graph at t is defined as $\mathcal{SKG}_t = \{\mathcal{E}, \mathcal{R}, \mathcal{F}_t\}$, where $\mathcal{F}_t$ represents the set of all facts at t and each fact is extended to a quadruple (e_s, r, e_0, t).

3.3 Temporal Knowledge Graph Reasoning

Reasoning over temporal knowledge graphs is to predict whether future entities and relations will co-occur given historical snapshots of knowledge graphs, i.e., given $\{\mathcal{SKG}_1, \mathcal{SKG}_2, \cdots, \mathcal{SKG}_{t-1}\}$ to predict $\mathcal{SKG}_t$. In practical modeling, the temporal knowledge graph reasoning is to typically predict a missing element except t in each query quadruple, i.e., $(?, r, e_o, t)$, $(e_s, r, ?, t)$ or $(e_s, ?, e_o.t)$, where ? is the object to be predicted. Entities and relations are treated equally as elements in each quadruple, and the model generates representations for them. Therefore, relations can be seen as entities to simplify the problem, and all the prediction tasks can be considered given a query $(e_s, r, ?, t)$ to predict ?. We use $|\mathcal{E}| = n$ to denote the number of entities and $|\mathcal{R}| = m$ to denote the number of relations. Input embeddings of entities are denoted as $\mathbf{E} \in \mathbb{R}^{n \times d}$ and input embeddings of relations are denoted as $\mathbf{R} \in \mathbb{R}^{m \times d}$.

4 Method

4.1 Overall Process of LSEN

The overall process of the proposed method is shown in Fig. 2. Specifically, all facts at the same time t in the TKG will be grouped together and constructed into a graph $\mathcal{G}_t$ based on their corresponding entities and relations. Entities are modeled as nodes, and relations are modeled as edges in the graph. If there exists a triple like (e_s, r, e_0) at t, two nodes corresponding to e_s and e_o are connected by an edge of type r and the direction is from e_s to e_o. After the above processing, the input of the algorithm becomes a sequence of graphs. Accordingly, the objective of the algorithm becomes to predict $\mathcal{G}_t$ given $\{\mathcal{G}_1, \mathcal{G}_2, \cdots, \mathcal{G}_{t-1}\}$.

The input graphs are sequentially processed by two modules of LSEN: (a) the Short-term Evolution Pattern Mining Module based on GNN [28] and RNN [3], and (b) the Long-term Evolution Pattern Mining Module based on the frequencies of constrained triple occurrences. The short-term evolution pattern mining

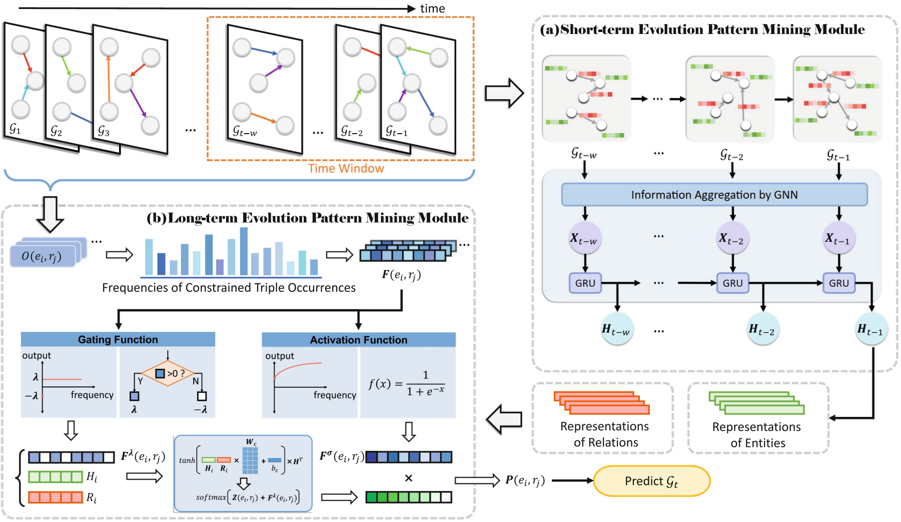

time
Time Window
(a)Short-term Evolution Pattern Mining Module
Information Aggregation by GNN
GRU
Representations of Relations
Representations of Entities
Predict $\mathcal{G}_t$
(b)Long-term Evolution Pattern Mining Module
Frequencies of Constrained Triple Occurrences
Gating Function
Activation Function
output
frequency
>0 ?
Y
N
$f(x) = \frac{1}{1+e^{-x}}$
$F(e_i, r_j)$
$F^\lambda(e_i, r_j)$
$F^\sigma(e_i, r_j)$
$P(e_i, r_j)$
$O(e_i, r_j)$
softmax

Fig. 2. Overall framework of proposed LSEN for TKG reasoning.

module encodes the recent changes. It selects the recent data within a small time window and uses a GCN [28] to learn the relevance between entities and relations. Then GRUs [3] are applied to model the changing representations over time. Given the entity and the relation in a query, the long-term module calculates the probability that all other entities may become the missing element of the quadruple. The long-term module takes the representations generated by the short-term module as input. In order to simultaneously take into account the long-term characteristics of the TKG, the long-term module utilizes the statistical frequencies of constrained triple occurrences. Finally, LSEN predicts the state of the TKG at the next moment based on the output probabilities.

4.2 Short-Term Evolution Pattern Mining Module

This module encodes the most recent facts within a small time window, and the window size is denoted as w. In a typical TKG, the number of facts that actually appear in a snapshot at each moment is limited, so the corresponding constructed graphs may contain many nodes with degree 0. This means that features of the 0-degree nodes remain unchanged at that moment, which should be considered. Therefore, we add a self-loop to each node and add an additional learnable vector to the embedding table of relations to represent the self-loop.

Graph Neural Network. After constructing the graph, LSEN applies a graph convolutional neural network to theconstructed graph to learn the structural correlations, outputting representations of the entities at each moment. The computation procedure of the graph convolutional network [28] is as follows:

$$\mathbf{x}_{i,t}^{l+1} = \frac{1}{\sqrt{\deg(i)}} \sum_{j \in \mathcal{N}_{i,t}} \mathbf{W}_1^l (\mathbf{x}_{j,t}^l - \mathbf{r}) + \mathbf{W}_2^l \mathbf{x}_{i,t}^l, \tag{1}$$

where $\mathbf{x}_{i,t}^{l+1}$ denotes the representation of node v_i at the $(l+1)$-th layer of the graph convolutional neural network at moment t, and $\mathbf{x}_{i,0}^0 = \mathbf{e}_i \in \mathbf{E}$. $\deg(i)$ represents the entry degree of node v_i. $\mathcal{N}_{i,t}$ represents the set of neighbors of node v_i in the graph at t. $\mathbf{r} \in \mathbf{R}$ represents the features of the relation between entities. $\mathbf{W}_1^l$ and $\mathbf{W}_2^l$ are the learnable parameters of the l-th graph convolutional layer. The entity representations generated by the graph convolutional network are denoted as: $\{\mathbf{X}_{t-w}, \mathbf{X}_{t-w+1}, \cdots, \mathbf{X}_{t-1}\}$, which constitute a time series.

Recurrent Neural Network. To characterize the sequential patterns, the short-term module introduces GRU [3] to encode the sequences. The specific computational procedure of GRU is as follows:

$$\mathbf{r}_t = \sigma(\mathbf{W}_r \mathbf{x}_t + \mathbf{b}_{wr} + \mathbf{U}_r \mathbf{h}_{t-1} + \mathbf{b}_{ur}) \tag{2}$$

$$\mathbf{z}_t = \sigma\left(\mathbf{W}_z \mathbf{x}_t + \mathbf{b}_{wz} + \mathbf{U}_z \mathbf{h}_{t-1} + \mathbf{b}_{uz}\right) \tag{3}$$

$$\mathbf{n}_t = \tanh(\mathbf{W}_n \mathbf{x}_t + \mathbf{b}_{wn} + \mathbf{r}_t \odot (\mathbf{U}_n \mathbf{h}_{t-1} + \mathbf{b}_{un})) \tag{4}$$

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{n}_t + \mathbf{z}_t \odot \mathbf{h}_{t-1}, \tag{5}$$

where $\mathbf{h}_t$ is the implicit variable at moment t, $\mathbf{x}_t$ is the input at moment t, and $\mathbf{h}_{t-1}$ is the hidden state at moment $t-1$ or at the initial moment. $\mathbf{r}_t$ and $\mathbf{z}_t$ are the reset gate and update gate, respectively. $\odot$ is the hadamard product, and σ is the sigmoid activation function. $\mathbf{W}_*$, $\mathbf{U}_*$ and $\mathbf{b}_*$ are learnable parameters. After the computation is completed, the output of the gated recurrent unit corresponding to the last moment $t-1$ will be used as the output of the module.

4.3 Long-Term Evolution Pattern Mining Module

As shown in Fig. 2, the long-term evolution pattern mining module based on the query's corresponding frequencies of constrained triple occurrences takes the entire historical data (from $\mathcal{G}_1$ to $\mathcal{G}_{t-1}$) as input. Compared with the short-term evolution pattern mining module, the long-term module has a longer horizon and is better able to focus on long-term historical information.

Processing Frequencies of Constrained Triple Occurrences. Given the entity e_i and the relation r_j of a query, the long-term evolution pattern mining module obtains the set $O(e_i, r_j)$ containing entities that have had a relation r_j with e_i, i.e., $O(e_i, r_j) = \{e_k | (e_i, r_j, e_k) \in \mathcal{F}_{t'}, t' \in [1, t-1]\}$. Also, this module records the frequency of each entity occurrence in $O(e_i, r_j)$, and the frequencies of all entities will form an n-dimensional vector $\mathbf{F}(e_i, r_j)$. Specifically, $\mathbf{F}_k(e_i, r_j)$ (the k-th dimension of $\mathbf{F}(e_i, r_j)$) records the frequency of occurrence of the triple (e_i, r_j, e_k). The frequency corresponding to the entity that is not in $O(e_i, r_j)$ is 0. Based on $\mathbf{F}(e_i, r_j)$, the long-term evolution pattern mining module generates two vectors $\mathbf{F}^{\lambda}(e_i, r_j)$ and $\mathbf{F}^{\sigma}(e_i, r_j)$ for decoding. $\mathbf{F}^{\lambda}(e_i, r_j)$ is computed as follows:

$$\mathbf{F}_k^{\lambda}(e_i, r_j) = f^{gating}(\mathbf{F}_k^{\lambda}(e_i, r_j), \lambda), \tag{6}$$

$$f^{gating}(x, \lambda) = \lambda \cdot (\phi_{x>0} - \phi_{x=0}), \tag{7}$$

where the ϕ_ρ function returns 1 when ρ is true and 0 when ρ is false. λ is a hyper-parameter used to control the threshold. $\mathbf{F}^{\lambda}(e_i, r_j)$ is formed through a designed gating function f^{gating}. The purpose of $\mathbf{F}^{\lambda}(e_i, r_j)$ is to set the elements of $\mathbf{F}(e_i, r_j)$ greater than 0 to λ and elements equal to 0 to $-\lambda$. This further emphasizes the impact of historical entities on the prediction and reduces the impact of non-historical entities.

In addition, candidate entities corresponding to different frequencies should be distinguished. Entities with a frequency of 1 and 10 are significantly different in their possibilities of establishing a connection with the query. Thus, $\mathbf{F}^{\sigma}(e_i, r_j)$ is generated to distinguish the effects of historical entities with different frequencies (σ is the sigmoid activation function):

$$\mathbf{F}_k^{\sigma}(e_i, r_j) = \sigma(\mathbf{F}_k(e_i, r_j)). \tag{8}$$

Calculating Representation Distances. Given the query entity e_i and the query relation r_j, the long-term module fuses the corresponding representations of the two to obtain the query vector and then takes the dot products of the vector and all candidate entity representations as distances, calculated as:

$$\mathbf{Z}(e_i, r_j) = \tanh(\mathbf{W}_c[\mathbf{H}_i; \mathbf{R}_j] + \mathbf{b}_c)\mathbf{H}^T, \tag{9}$$

where $\mathbf{H}$ represents the evolving representations of all entities output by the short-term module at each moment. $\mathbf{H}_i$ and $\mathbf{R}_j$ denote the representations of the entity e_i and the relation r_j, respectively. $[\cdot;\cdot]$ is the concatenation operation of variables. $\mathbf{W}_c \in \mathbb{R}^{d \times 2d}$ and $\mathbf{b}_c \in \mathbb{R}^d$ are learnable parameters, where d is the embedding dimension. In order to better capture the recurrence as well as the periodicity of interactions in TKGs and to better grasp the evolution characteristics over a long historical period, the module corrects the final prediction results based on the frequencies of constrained triple occurrences as follows:

$$\mathbf{P}(e_i, r_j) = \mathbf{F}^{\sigma}(e_i, r_j) \cdot \mathrm{softmax}(\mathbf{Z}(e_i, r_j) + \mathbf{F}^{\lambda}(e_i, r_j)), \tag{10}$$

where $\mathbf{P}(e_i, r_j)$ represents the probabilities that each candidate entity may have a relation r_j with e_i.

4.4 Model Training and Predicting

The task of temporal knowledge graph reasoning can be seen as a multi-class classification task, where all the candidate entities of a query $q(e_i, r_j)$ can be seen as categories, and entities that actually generate the relations are real labels. Therefore, we can use the cross-entropy loss function, which is calculated as:

$$\mathcal{L} = \frac{1}{B} \sum_{i=1}^{B} \sum_{c=1}^{n} y_{ic} \log\left(p_{ic}\right), \tag{11}$$

where B represents the number of queries (usually the size of a batch during training). y_{ic} is a sign function (outputs 0 or 1) and if the true result of a query $q(e_i, r_j)$ is e_c, then y_{ic} takes 1, otherwise takes 0. p_{ic} represents the probability that entity e_c is the result of the query $q(e_i, r_j)$. All the probability values are formed into $\mathbf{P}(e_i, r_j)$. When making the prediction, LSEN chooses the entity corresponding to the largest value of $\mathbf{P}(e_i, r_j)$ as the final prediction e_{pred}, i.e.:

$$e_{pred} = \mathrm{argmax}(\mathbf{P}(e_i, r_j)). \tag{12}$$

5 Experiments

In this section, we conduct extensive experiments to evaluate the performance of the proposed LSEN and other baselines. The environment of experiments, information about datasets, and the setup of experiments are described in detail. In addition, we perform ablation studies and hyper-parameter investigations to comprehensively analyze our method.

5.1 Experimental Setup

Datasets. We use five real-world TKG datasets that are widely used in previous works to conduct evaluation experiments, including ICEWS18 [13], ICEWS14 [7], GDELT [17], WIKI [16], and YAGO [21]. ICEWS18 and ICEWS14 take political events that have taken place internationally for different periods. GDELT is a collection of human social behaviors extracted from a news media. WIKI and YAGO are subsets of Wikipedia history and YAGO3. In order to have a fair comparison with baselines, we follow [33,37] to use an 8:1:1 (train-val-test) temporal

Table 1. Statistics of datasets.

Dataset	Entities	Relations	Train Edges	Valid Edges	Test Edges	Granularity	Time Granules
ICEWS18	23,033	256	373,018	45,995	49,545	24 h	304
ICEWS14	12,498	260	323,895	–	341,409	24 h	365
GDELT	7,691	240	1,734,399	238,765	305,241	15 min	2,751
WIKI	12,544	24	539,286	67,538	63,110	1 year	232
YAGO	10,623	10	161,540	19,523	20,026	1 year	189

split for datasets except ICEWS14. ICEWS14 is only divided into a training set and a testing set since the original ICEWS14 is not provided with a validation set. All these settings are the same as in previous works. The statistical information of each dataset after delineation can be seen in Table 1.

Baselines. To evaluate the effectiveness of our method, we compare LSEN against 16 prominent KG reasoning models, including static and temporal approaches. Static KG reasoning methods include TransE [2], DistMult [34], ComplEx [27], R-GCN [25], and ConvE [4]. Temp [32], RE-NET [13], TLogic [20], RE-GCN [19], TANGO-TuckER [9], TANGO-Distmult [9], CyGNet [37], EvoKG [23], HIP [10], TSHGE [5], CENET [33], and HGLS [36] are selected as TKG models.

Evaluation Metrics. We use MRR and Hits@N as evaluation metrics in our experiments, as they are standard metrics for evaluating TKG models, and this is consistent with baselines for comparison fair. MRR is short for Mean Reciprocal Ranks, which is the average of the reciprocal of the rank of ground truth. Hit@N is the proportion of correct predictions ranked within the top N. The higher the rank of the probability corresponding to the correct entity, the higher will be the values of MRR and Hits@N.

Implementation Details. The results are reported under a widely adopted filtering setup [10,13,33,37]. We run all experiments using PyTorch and on a Tesla V100 GPU. As for default configurations of LSEN, we use the Adam optimizer with a learning rate of 0.001, a weight decay ratio of 0.00001, a batch size of 1024, and an embedding dimension of 200. The dropout rate of GNN layers is set to 0.2, and the dropout rate of the long-term module is set to 0.4. For the settings of baselines, we use their recommended configurations.

5.2 Quantitative Results

Table 2 shows the link (event) prediction experiment results of our LSEN and 15 up-to-date baselines. Here the results are under the traditional offline setting (i.e., models are trained only using the training set). Almost all temporal knowledge graph methods outperform static knowledge graph methods, so it is inappropriate to directly apply existing static knowledge graph algorithms to TKG prediction and the temporal information must be taken into account. Our proposed LSEN achieves the best performance compared to other competitive methods and the performance improvement on WIKI and YAGO datasets is remarkable. The experimental results prove the effectiveness of LSEN, and show that comprehensive consideration of the short-term and long-term evolution patterns of TKGs is necessary to improve prediction performance.

Table 2. Performance (in percentage) for the temporal link prediction task. The best and second-best results in each column are highlighted in **bold** font and underlined. LSEN outperforms other baselines on all datasets.

Method	ICEWS18			ICEWS14			GDELT			WIKI			YAGO		
	MRR	Hits@1	Hits@3	MRR	Hits@1	Hits@3	MRR	Hits@1	Hits@3	MRR	Hits@1	Hits@3	MRR	Hits@1	Hits@3
TransE	17.56	2.48	26.95	18.65	1.12	31.34	16.05	0.00	26.10	46.68	36.19	49.71	48.97	46.23	62.45
DistMult	22.16	12.13	26.00	19.06	10.09	22.00	18.71	11.59	20.05	46.12	37.24	49.81	59.47	52.97	60.91
ComplEx	30.09	21.88	34.15	24.47	16.13	27.49	22.77	15.77	24.05	47.84	38.15	50.08	61.29	54.88	62.28
R-GCN	23.19	16.36	25.34	26.31	18.23	30.43	23.31	17.24	24.96	37.57	28.15	38.66	41.30	32.56	44.44
ConvE	36.67	28.51	39.80	40.73	33.20	43.92	35.99	27.05	39.32	47.57	38.76	50.10	62.32	56.19	63.97
TeMP	40.48	33.97	42.63	43.13	35.67	35.67	37.56	29.82	40.15	49.61	46.96	50.24	62.25	55.39	64.63
RE-NET	42.93	36.19	45.47	45.71	38.42	49.06	40.12	32.43	43.40	51.97	48.01	52.07	65.16	63.29	65.63
TLogic	37.52	30.09	40.87	38.19	32.23	41.05	22.73	17.65	24.66	57.73	57.43	57.88	1.29	0.49	0.85
RE-GCN	32.78	24.99	35.54	32.37	24.43	35.05	29.46	21.74	32.01	44.86	39.82	46.75	65.69	59.98	68.70
TANGO-TuckER	44.56	37.87	47.46	46.42	38.94	50.25	38.00	28.02	43.91	53.28	52.21	53.61	67.21	65.56	67.59
TANGO-Distmult	44.00	38.64	45.78	46.68	41.20	48.64	41.16	35.11	43.02	54.05	51.52	53.84	68.34	67.05	68.39
CyGNet	46.69	40.58	49.82	48.63	41.77	52.50	50.29	44.53	54.69	45.50	50.48	50.79	63.47	64.26	65.71
EvoKG	29.67	12.92	33.08	18.30	6.30	19.43	11.29	2.93	10.84	50.66	12.21	63.84	55.11	54.37	81.38
HIP	48.37	43.51	51.32	50.57	45.73	54.28	52.76	46.35	55.31	54.71	53.82	54.73	67.55	66.32	68.49
TSHGE	–	–	–	36.22	26.26	40.73	–	–	–	39.59	–	44.24	57.33	–	64.89
CENET	51.06	47.10	51.92	53.35	49.61	54.07	58.48	55.99	58.63	68.39	68.33	68.36	84.13	84.03	84.23
HGLS	32.72	21.04	37.35	46.99	34.74	53.05	23.56	18.03	30.16	49.63	39.62	55.17	59.02	48.17	65.73
LSEN	**52.12**	**48.37**	**52.95**	**54.82**	**51.15**	**55.53**	**59.47**	**57.44**	**59.38**	**76.13**	**74.01**	**76.82**	**88.07**	**86.70**	**88.61**
Improv.	+1.06	+1.27	+1.03	+1.47	+1.54	+1.25	+0.99	+1.45	+0.75	+7.74	+5.68	+8.46	+3.94	+2.67	+4.38

5.3 Ablation Study

We further conduct ablation studies on ICEWS18, ICEWS14, and YAGO with offline setting to validate the effectiveness of critical modules of LSEN. The results are shown in Table 3. Details of the ablation study are as follows:

- **w/o long**: We remove the long-term evolution pattern mining module of proposed LSEN and denote this variant as **w/o long**.
- **w/o short**: We remove the short-term evolution pattern mining module of proposed LSEN and denote this variant as **w/o short**.
- **w/o freq**: We set all the co-occurrence frequencies unchanged when evaluating LSEN on the testing set and denote this variant as **w/o freq**.
- **w/o loop**: We remove all the self-loops from graph snapshots and denote this variant as **w/o loop**.
- **w/o GRU**: We remove the GRUs int the short-term evolution pattern mining module of LSEN and denote this variant as **w/o GRU**.
- **w/o gating**: We don't use the f^{gating} function in the long-term evolution pattern mining module of LSEN and $\mathbf{F}^{\lambda}(e_i, r_j)$ is the same as $\mathbf{F}(e_i, r_j)$.
- **w/o $\mathbf{F}^{\sigma}$**: We remove $\mathbf{F}^{\sigma}$ in the long-term evolution pattern mining module of LSEN. In this case, Eq. 10 will not include $\mathbf{F}^{\sigma}(e_i, r_j)$.

It can be seen that all components of the model are beneficial to the prediction performance and indispensable. As the two main components of LSEN, the short-term and long-term evolution pattern mining modules have the greatest impact on the prediction performance (as indicated by the results of **w/o long** and

Table 3. Ablation study of LSEN. Statistics are presented in percentages.

Ablation	ICEWS18			ICEWS14			YAGO		
	MRR	Hits@1	Hits@3	MRR	Hits@1	Hits@3	MRR	Hits@1	Hits@3
w/o long	36.94	28.03	40.92	51.63	47.68	52.47	66.66	58.2	71.88
w/o short	50.42	47.25	50.75	53.28	50.11	53.56	71.03	69.47	71.40
w/o loop	42.82	39.72	43.22	51.85	48.43	52.44	82.06	81.47	81.92
w/o GRU	51.91	48.16	52.61	54.55	50.99	55.26	85.90	84.78	85.89
w/o freq	51.87	47.97	52.67	54.38	50.61	55.22	87.58	86.03	88.31
w/o gating	34.55	29.76	35.77	30.74	26.89	31.53	81.88	80.64	82.15
w/o F^{σ}	51.63	47.68	52.47	54.60	50.78	55.42	87.13	85.96	87.40
LSEN	**52.12**	**48.37**	**52.95**	**54.82**	**51.15**	**55.53**	**88.07**	**86.70**	**88.61**

w/o short), which reflects the significance of considering both short-term and long-term evolution patterns for TKGs.

In the design of the short-term evolution pattern mining module, the self-loop needs to be included in the composition, as demonstrated by the results of **w/o loop**. A 0-degree node indicates that the entity is not involved in any event, which is also a necessary characteristic. GRUs can more accurately model the temporal patterns, which can be indicated by the results of **w/o GRU**.

As for the long-term evolution pattern mining module, the results of **w/o freq** indicate that the frequencies of constrained triple occurrences in the long-term module are beneficial for portraying long-term features. The variant **w/o gating** results in significant performance degradation of LSEN on ICEWS18 and ICEWS14 datasets, which highlights the importance of distinguishing between historical and non-historical entities. Also, $\mathbf{F}^{\sigma}$ is indispensable as it can distinguish the influence of candidate entities with different frequencies. All of the above observations can be verified in the ablation experiments.

5.4 Hyper-parameter Investigation

Impact of the Time Window Size. We conduct experiments to analyze the effect of the time window size selected by the short-term module, which is shown in Fig. 3. For ICEWS18, ICEWS14, and GDELT, the model achieves best performance on these datasets with time window size between 2 and 5. While for WIKI and YAGO, the model performs best when the time window is chosen as 1. The granularities of ICEWS18, ICEWS14, and GDELT are at the minute or hour level, so the information of adjacent moments is closely relevant, and the larger time window can aggregate more short-term information and better capture the evolutionary characteristics. The interval time of data in both WIKI and YAGO is in years, and a time window of length 2 provides an excessive duration. A large time window may introduce too many irrelevant historical facts, so a window size of 1 results in a good performance.

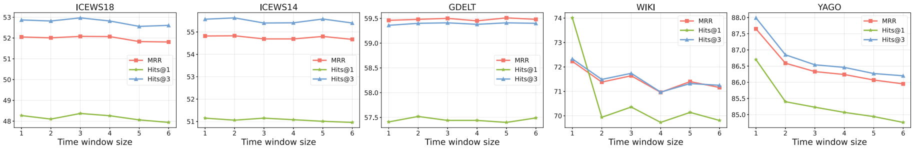


Fig. 3. Performance (%) of LSEN with different time window sizes.

Impact of the Number of GNN Layers. Experimental results on the effect of the number of GNN layers are presented in Fig. 4. The performance on ICEWS18, ICEWS14, and GDELT datasets show a similar trend with increasing GNN depth. The performance is optimized at about 2 layers, after which it decreases with more layers. On the YAGO and WIKI datasets, the model performance improves as the number of GNN layers increases, reaching an optimum at around 6–7 layers. The numbers of entities in the five datasets are close, but the numbers of relations in WIKI and YAGO are much smaller than the other three datasets. So the WIKI and YAGO datasets are more like entity-dominated datasets. Deeper GNNs generate higher-order representations and lead to better performance in entity-dominated datasets, which may be the reason why deeper GNNs are needed in WIKI and YAGO.

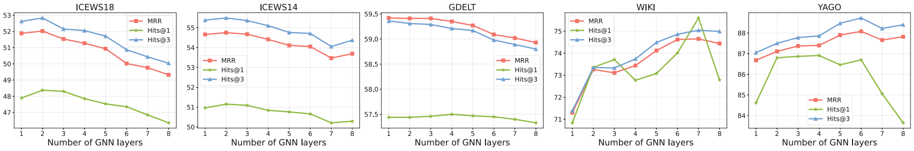


Fig. 4. Performance (%) of LSEN with different numbers of GNN layers.

Impact of λ in the Long-Term Evolution Pattern Mining Module. We investigate the effect of the hyper-parameter λ, as demonstrated in Fig. 5. λ controls how much importance the model places on long-term historical information, and either too large or too small λ will degrade the model's performance. For datasets with small temporal granularity like ICEWS18, ICEWS14, and GDELT, larger λ are more appropriate because the data with a certain window size has a smaller time span and cannot capture sufficient historical features, thus requiring more emphasis on long-term historical information. As for WIKI and YAGO with large temporal granularity, the data with a certain time window size has a large time span and contains more long-term historical characteristics. Too large λ values may introduce historical noise instead, so the weight needs to be reduced. Therefore, for ICEWS14, ICEWS18, and GDELT, λ is set to 2. For WIKI and YAGO, λ is set to 1.

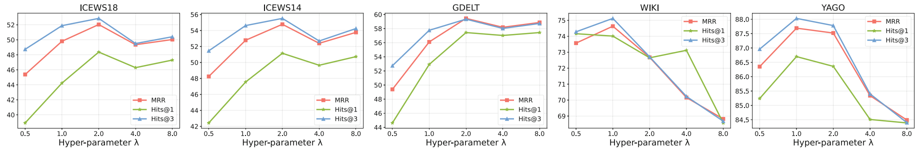


Fig. 5. Performance (%) of LSEN with different λ values.

6 Conclusion

In this paper, we proposed LSEN, a temporal knowledge graph reasoning method that can simultaneously capture long-term and short-term evolution patterns of TKGs. To express the recent and fine-grained evolution patterns, the short-term evolution pattern mining module of LSEN applies GCN to capture structural information in graphs and uses GRU to model the sequential patterns across snapshots. The long-term evolution pattern mining module of LSEN takes into account the frequencies of constrained triple occurrences. Extensive experiments show that LSEN can achieve the best performance of TKG reasoning.

Acknowledgment. This work was supported by the National Natural Science Foundation of China (#62102177), the Natural Science Foundation of Jiangsu Province (#BK20210181), and the Collaborative Innovation Center of Novel Software Technology and Industrialization, Jiangsu, China.

References

1. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Trans. Neural Networks **5**(2), 157–166 (1994)
2. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Advances in Neural Information Processing Systems, vol. 26 (2013)
3. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling (2014)
4. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2D knowledge graph embeddings. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
5. Duan, H., Jin, H., Chen, K., Du, S., Fang, T., Huo, H.: Jointly learning time-structure-history graph embedding for temporal knowledge graph reasoning. In: 2022 IEEE Smartworld, Ubiquitous Intelligence & Computing, Scalable Computing & Communications, Digital Twin, Privacy Computing, Metaverse, Autonomous & Trusted Vehicles (SmartWorld/UIC/ScalCom/DigitalTwin/PriComp/Meta), pp. 1387–1393. IEEE (2022)
6. Feng, F., He, X., Wang, X., Luo, C., Liu, Y., Chua, T.S.: Temporal relational ranking for stock prediction. ACM Trans. Inf. Syst. (TOIS) **37**(2), 1–30 (2019)
7. García-Durán, A., Dumančić, S., Niepert, M.: Learning sequence encoders for temporal knowledge graph completion. arXiv preprint arXiv:1809.03202 (2018)

8. Gottschalk, S., Demidova, E.: EventKG-the hub of event knowledge on the web-and biographical timeline generation. Semant. Web **10**(6), 1039–1070 (2019)
9. Han, Z., Ding, Z., Ma, Y., Gu, Y., Tresp, V.: Learning neural ordinary equations for forecasting future links on temporal knowledge graphs. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (2021)
10. He, Y., Zhang, P., Liu, L., Liang, Q., Zhang, W., Zhang, C.: Hip network: historical information passing network for extrapolation reasoning on temporal knowledge graph. In: IJCAI, pp. 1915–1921 (2021)
11. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
12. Ji, S., Pan, S., Cambria, E., Marttinen, P., Philip, S.Y.: A survey on knowledge graphs: representation, acquisition, and applications. IEEE Trans. Neural Netw. Learn. Syst. **33**(2), 494–514 (2021)
13. Jin, W., Qu, M., Jin, X., Ren, X.: Recurrent event network: autoregressive structure inference over temporal knowledge graphs (2019)
14. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
15. Kolda, T.G., Bader, B.W.: Tensor decompositions and applications. SIAM Rev. **51**(3), 455–500 (2009)
16. Leblay, J., Chekol, M.W.: Deriving validity time in knowledge graph. In: Companion Proceedings of the Web Conference 2018, pp. 1771–1776 (2018)
17. Leetaru, K., Schrodt, P.A.: GDELT: global data on events, location, and tone, 1979–2012. In: ISA annual Convention, vol. 2, pp. 1–49. Citeseer (2013)
18. Li, Z., et al.: HiSMatch: historical structure matching based temporal knowledge graph reasoning. arXiv preprint arXiv:2210.09708 (2022)
19. Li, Z., et al.: Temporal knowledge graph reasoning based on evolutional representation learning. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 408–417 (2021)
20. Liu, Y., Ma, Y., Hildebrandt, M., Joblin, M., Tresp, V.: TLogic: temporal logical rules for explainable link forecasting on temporal knowledge graphs. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, pp. 4120–4127 (2022)
21. Mahdisoltani, F., Biega, J., Suchanek, F.: YAGO3: a knowledge base from multilingual Wikipedias. In: 7th Biennial Conference on Innovative Data Systems Research. CIDR Conference (2014)
22. Medsker, L., Jain, L.C.: Recurrent Neural Networks: Design and Applications. CRC Press, London (1999)
23. Park, N., Liu, F., Mehta, P., Cristofor, D., Faloutsos, C., Dong, Y.: EvoKG: jointly modeling event time and network structure for reasoning over temporal knowledge graphs. In: Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, pp. 794–803 (2022)
24. Ribeiro, A.H., Tiels, K., Aguirre, L.A., Schön, T.: Beyond exploding and vanishing gradients: analysing RNN training using attractors and smoothness. In: International Conference on Artificial Intelligence and Statistics, pp. 2370–2380. PMLR (2020)
25. Schlichtkrull, M., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: Gangemi, A., et al. (eds.) ESWC 2018. LNCS, vol. 10843, pp. 593–607. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93417-4_38
26. Sun, Z., Deng, Z.H., Nie, J.Y., Tang, J.: RotatE: knowledge graph embedding by relational rotation in complex space. arXiv preprint arXiv:1902.10197 (2019)

27. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: International Conference on Machine Learning, pp. 2071–2080. PMLR (2016)
28. Vashishth, S., Sanyal, S., Nitin, V., Talukdar, P.: Composition-based multi-relational graph convolutional networks. arXiv preprint arXiv:1911.03082 (2019)
29. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
30. Velickovic, P., et al.: Graph attention networks. stat **1050**(20), 10–48550 (2017)
31. Wang, X., He, X., Cao, Y., Liu, M., Chua, T.S.: KGAT: knowledge graph attention network for recommendation. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 950–958 (2019)
32. Wu, J., Cao, M., Cheung, J.C.K., Hamilton, W.L.: TeMP: temporal message passing for temporal knowledge graph completion (2020)
33. Xu, Y., Ou, J., Xu, H., Fu, L.: Temporal knowledge graph reasoning with historical contrastive learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, pp. 4765–4773 (2023)
34. Yang, B., Yih, W.T., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. arXiv preprint arXiv:1412.6575 (2014)
35. Ye, R., Li, X., Fang, Y., Zang, H., Wang, M.: A vectorized relational graph convolutional network for multi-relational network alignment. In: IJCAI (2019)
36. Zhang, M., Xia, Y., Liu, Q., Wu, S., Wang, L.: Learning long- and short-term representations for temporal knowledge graph reasoning. In: Proceedings of the ACM Web Conference 2023, WWW 2023 (2023)
37. Zhu, C., Chen, M., Fan, C., Cheng, G., Zhang, Y.: Learning from history: modeling temporal knowledge graphs with sequential copy-generation networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 4732–4740 (2021)
38. Zou, X.: A survey on application of knowledge graph. J. Phys. Conf. Ser. **1487**, 012016 (2020)

Rethinking Attention Gated with Hybrid Dual Pyramid Transformer-CNN for Generalized Segmentation in Medical Imaging

Fares Bougourzi[1(✉)], Fadi Dornaika[2,3], Abdelmalik Taleb-Ahmed[4], and Vinh Truong Hoang[3]

[1] Junia, UMR 8520, CNRS, Centrale Lille, Univerity of Polytechnique, 59000 Lille, Hauts-de-France, France
`faresbougourzi@gmail.com`
[2] University of the Basque Country UPV/EHU, San Sebastian, Spain
`fadi.dornaika@ehu.eus`
[3] Ho Chi Minh City Open University, Ho Chi Minh City, Vietnam
`vinh.th@ou.edu.vn`
[4] Université Polytechnique Hauts-de-France, Université de Lille, CNRS, Valenciennes, 59313 Hauts-de-France, France
`Abdelmalik.Taleb-Ahmed@uphf.fr`

Abstract. Inspired by the success of Transformers in Computer vision, Transformers have been widely investigated for medical imaging segmentation. However, most of Transformer architecture are using the recent transformer architectures as encoder or as parallel encoder with the CNN encoder. In this paper, we introduce a novel hybrid CNN-Transformer segmentation architecture (PAG-PVTUnet) designed for efficiently building a strong CNN-Transformer encoder. Our approach exploits attention gates within a Dual Pyramid hybrid encoder. The contributions of this methodology can be summarized into three key aspects: (i) the utilization of Pyramid input for highlighting the prominent features at different scales, (ii) the incorporation of a PVT transformer to capture long-range dependencies across various resolutions, and (iii) the implementation of a Dual-Attention Gate mechanism for effectively fusing prominent features from both CNN and Transformer branches. Through comprehensive evaluation across different segmentation tasks including: abdominal multi-organs segmentation, infection segmentation (Covid-19 and Bone Metastasis), microscopic tissues segmentation (Gland and Nucleus). The proposed approach demonstrates state-of-the-art performance and exhibits remarkable generalization capabilities. This research represents a significant advancement towards addressing the pressing need for efficient and adaptable segmentation solutions in medical imaging applications.

Keywords: Transformer · Convolutional Neural Network · Deep Learning · Segmentation · Unet · Synapse · (Gland and Nucleus) · Covid-19

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 243–258, 2025.
https://doi.org/10.1007/978-3-031-78128-5_16

1 Introduction

Medical imaging segmentation plays a crucial role in diagnosing, assessing severity, and monitoring progress in various medical conditions [23]. Despite significant advancements in utilizing machine learning for medical imaging segmentation, several challenges persist in developing efficient segmentation approaches. These challenges include limited labeled data availability, which is a laborious and error-prone task [5,23]. The ultimate goal remains to devise a generalized approach for different medical segmentation tasks. However, achieving efficiency across various medical imaging segmentation tasks remains challenging due to the high variability among diseases, ranging from single classes to multi-classes, and from disease to organ segmentation. Consequently, many approaches are tailored to specific tasks, limiting their applicability to other tasks.

In the last decade, Convolutional Neural Networks (CNNs) have emerged as the primary approach for medical imaging segmentation [4,5,7,11,23]. However, CNNs are predominantly adept at extracting local features, thereby overlooking long-range dependencies, which are crucial for modeling global contextual features. Transformers have demonstrated high capability in encoding long-range dependencies, leading to their integration into segmentation architectures either as pure architectures or hybrid ones combined with CNNs [7,11,23,28,31]. However, existing architectures often utilize transformers as single or parallel encoders alongside CNN encoders [7–9,29,31,32,37], indicating limitations in efficiently combining transformer and CNN features.

To address this, we propose revisiting attention gates to build a stronger encoder, introducing our Dual-Attention Gate. Unlike conventional attention gates originally designed to select prominent features from the encoder during decoding [16], our Dual-Attention Gate selects prominent features between CNN features via an input pyramid and from the transformer branch via the main CNN feature path. This results in a more compact main path.

The paper introduces a novel approach called PAG-PVTUnet, which combines Transformer and CNN architectures using Dual-Attention Gates. These gates aim to extract significant feature regions and merge features from both CNN and Transformer models. The encoder structure of PAG-PVTUnet consists of three branches. The first branch undergoes contraction through four pyramid levels using convolutional blocks, producing features that act as a gating signal for highlighting prominent features in the second branch. The second branch, termed the main branch, focuses on extracting features from the input data. Simultaneously, the features from the main branch are used to highlight important features in the third branch, which utilizes Transformer architecture. The attention features from both branches are concatenated to form the new main branch features for the subsequent level. Overall, the proposed approach aims to capture both local and global features through attention mechanisms, resulting in a comprehensive representation of the input data.

In summary, the main contributions of this work are:

- Introduction of a novel hybrid architecture for medical imaging segmentation, which seamlessly integrates CNN, Transformers, and a fusion branch encoder.

- Enhancement of the Att-Unet attention gate through our proposed Dual-Attention Gate. This refinement involves redesigning its structure, repositioning it within the encoder, and optimizing its functionality within the fusion objective.
- Demonstration of the remarkable capability of our approach to achieve state-of-the-art performance across a diverse range of medical imaging segmentation tasks, including organ scans segmentation, infection detection, and microscopic tissue segmentation (Fig. 1 shows examples of the considred segmentation tasks).

This paper is organised as follows: Sect. 2 highlights the related works. In Sect. 3 the proposed approach is described. Section 4 illustrates the evaluated datasets and tasks. Section 5 depicts and analyzes the obtained results and ablation study. In Sect. 6 models size and inference time are discussed. Finally, Sect. 7 concludes this paper.

2 Related Works

In recent years, Convolutional Neural Networks (CNNs) have achieved state-of-the-art performance in medical image segmentation, particularly following the proposition of the U-Net architecture by Ronneberger et al. in 2015 [21]. Since then, numerous variants such as Attention U-Net (Att-U-Net) [16], U-Net++ [36], and ResU-Net [34] have emerged, each aiming to enhance segmentation performance. The U-Net architecture, characterized by an encoder-decoder structure with skip connections, has proven effective in preserving fine-grained details through feature concatenation. On the other hand, attention mechanisms have been widely investigated for medical imaging segmentation. One of the most famous attention mechanisms is the Attention Gate (AG), proposed by Oktay et al. in 2018 [16], which integrates attention into U-Net after the skip connection, producing a variant known as Att-U-Net. The main objective of Att-U-Net is to highlight salient regions in encoder features using decoder features. However, the efficacy of attention gates can vary, prompting the introduction of our approach: the Dual-Attention Gate, integrated into the encoding phase, leveraging Pyramid features, CNN features, and Transformer features to enhance feature extraction and emphasize prominent regions.

Despite the great success of CNNs in medical imaging segmentation, their main shortcoming lies in their weakness in capturing long-range dependencies, as CNNs are primarily focused on extracting local features [8,23,37]. On the other hand, Transformers, renowned for their ability to capture long-range dependencies in sequences, have shown promising performance in medical imaging tasks, including classification, detection, and segmentation [9,23,29,31,32]. In segmentation, both 2D and 3D transformer-based approaches, such as Fat-Net and U-Transformer, have showed promising performance by fusing CNN and Transformer components to enhance segmentation accuracy [8,19,32].

The integration of CNN and Transformer blocks into single architectures has been a focal point, particularly in the encoding phase [8,9,29,31,32,37].

Various encoder configurations have been proposed, including solely Transformer-based encoders [8,37], parallel CNN and Transformer encoders with subsequent fusion [9,32], and CNN encoders followed by Transformer blocks [29,31]. However, many existing approaches lack robust connectivity between Transformer and CNN features, indicating a gap in feature integration. To address this, our approach introduces a novel encoder architecture incorporating Pyramid features, CNN features, Transformer features, and Dual-Attention Gates, aiming to significantly enhance feature fusion and improve segmentation performance.

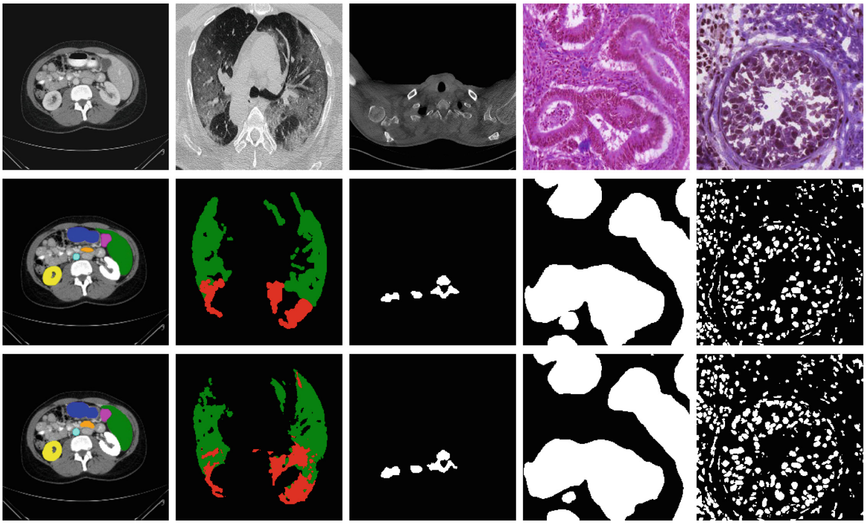


Fig. 1. Examples of Medical Imaging Segmentation, the first, second and third rows represent the input image, ground truth and the prediction of our approach, respectively. First, second, third, fourth and fifth columns depict abdominal multi-organ segmentation, Covid-19, Bone Metastasis, Gland, and Nucleus, respectively.

3 Proposed Approach

Our proposed Pyramid Dual-Attention Gate PVT Unet (PAG-PVTUnet) has three encoder branches and a Unet-like decoder as shown in Fig. 2. The detailed architecture is illustrated in Fig. 3.

As shown in Fig. 2, the encoder of our proposed architecture consists of four main components: (i) a Pyramid Vision Transformer (PVT-v2), (ii) a pyramid representing the input image with four levels, each level followed by convolution blocks (P_1ConvB, P_2ConvB, P_3ConvB and P_4ConvB), (iii) a main encoder path that merges the PVT features and the main encoder features using Dual-Attention Gates, and (iv) a classic Transformer (Base ViT) serving as the final stage of encoding.

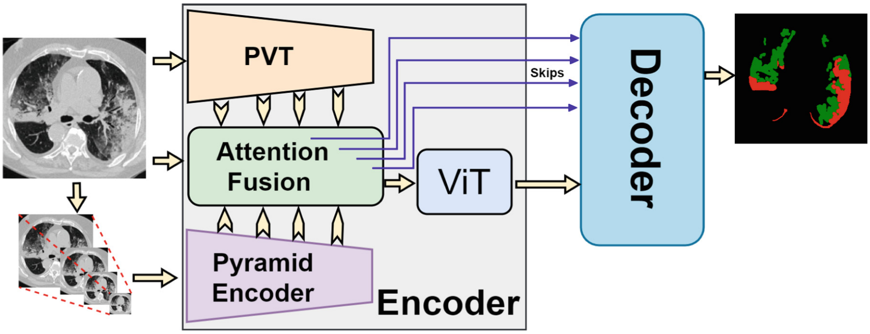


Fig. 2. Our proposed PAG-PVTUnet architecture.

3.1 Pyramid Encoder

The pyramid encoder branch aims to provide convolutional features at four levels of the input image pyramid, which are subsequently utilized in the spatial gate attention mechanism. The image undergoes transformation into a pyramid with four levels, each level being resized separately. There are four pyramid levels, each with a pyramid input (P_1, P_2, P_3, and P_4) derived from the input image (I). These pyramid levels generate pyramid feature maps (P_{f_1}, P_{f_2}, P_{f_3}, and P_{f_4}) using pyramid convolutional blocks (PConvB), which consist of double convolutional blocks (DConvB). Notably, the first pyramid level contains one DConvB, whereas the fourth level incorporates a cascade of four DConvBs, as shown in Fig. 4. Additionally, as depicted in Fig. 4.a, the DConvB comprises two 3×3 convolutional blocks and a residual skip connection that uses a 1×1 convolutional kernel to match the input number of channels C_{in} to C_{out}. The output of the two 3 by 3 kernels is summed with the features of the skip connection.

These pyramid feature maps play a crucial role in maintaining spatial attention awareness across all main encoder layers. They serve as gating signals for the main encoder path, facilitating the integration of spatial attention information throughout the encoding process.

3.2 Main Encoder: Attention Fusion

As shown in Fig. 3, the input image is fed to both the Transformer and the main encoder branch. For the Transformer branch, we utilize PVT-v2-Li [30], which was designed for a progressive shrinking pyramid and a spatial-reduction attention. This makes the PVT flexible for learning multi-scale and high-level features, similar to the CNN encoder design. For the main branch, we start with a double convolution module as depicted in Fig. 4.a. From this point, it hierarchically merges the current features of the main branch and the Transformer features using a dual-gate attention mechanism (explained in the next section). The first level of the main branch is fused with the Transformer first stage features through the proposed Dual-Attention Gate. This attention fusion process

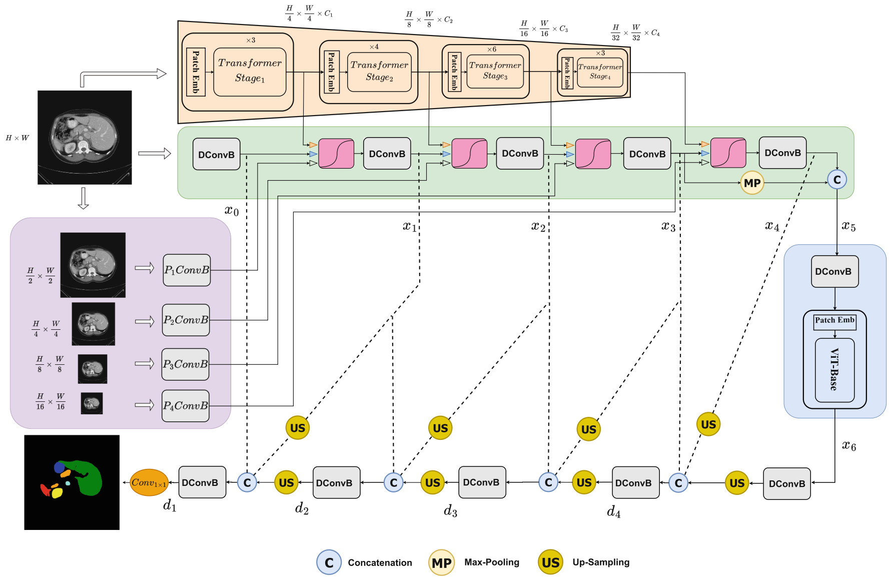

$\frac{H}{4} \times \frac{W}{4} \times C_1$
$\frac{H}{8} \times \frac{W}{8} \times C_2$
$\frac{H}{16} \times \frac{W}{16} \times C_3$
$\frac{H}{32} \times \frac{W}{32} \times C_4$
×3
×4
×6
×3
Patch Emb
Transformer Stage$_1$
Transformer Stage$_2$
Transformer Stage$_3$
Transformer Stage$_4$
$H \times W$
DConvB
MP
C
x_0
x_1
x_2
x_3
x_4
x_5
x_6
$\frac{H}{2} \times \frac{W}{2}$
$\frac{H}{4} \times \frac{W}{4}$
$\frac{H}{8} \times \frac{W}{8}$
$\frac{H}{16} \times \frac{W}{16}$
P_1ConvB
P_2ConvB
P_3ConvB
P_4ConvB
ViT-Base
US
$Conv_{1\times1}$
d_1
d_2
d_3
d_4
C Concatenation
MP Max-Pooling
US Up-Sampling

Fig. 3. The detailed description of our proposed PAG-PVTUnet approach.

is performed in the main branch for four levels. At each level the corresponding features of Transformer stage, pyramid levels, and previous main branch features are combined.

Upon the completion of the fourth fusion, the resulting features (x_4) are concatenated with the features from the previous level (x_3) in the main encoder. These are then fed into a classic ViT (ViT Base) with a spatial resolution of 14×14, corresponding to 196 tokens. The output features from the ViT (x_6) are subsequently passed through a dual convolution module. The resulting features (x_7) are then forwarded to the decoder. Finally, the decoder of the proposed PAG-PVTUnet consists of four stages and follows a conventional architecture, with skips provided by two levels of features from the main encoder.

3.3 Dual-Attention Gate

The Dual-Attention gates play a crucial role in providing an effective fusion mechanism for the Transformer features at four different stages and the features of the main encoder branch with attention provided by the pyramid level convolutional features. As depicted in the Fig. 5, this module has three inputs: Transformer features, the previous main branch level features, and the features associated with the corresponding pyramid level. The module consists of two classical Attention Gates (AG). The first AG considers the signal and the current features as input, while the second AG considers the current features as the signal and the lower pyramid feature as the gate. The outputs of both AGs are then concatenated to form the signal in the main Encoder, which is used in the

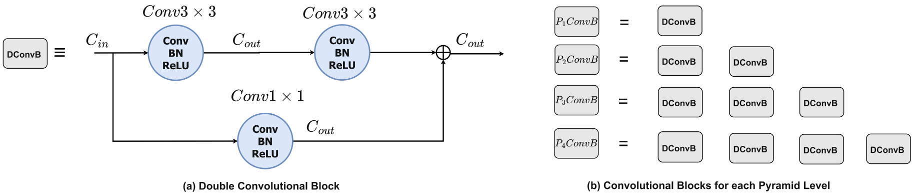

DConvB ≡
C_{in}
$Conv3 \times 3$
Conv BN ReLU
C_{out}
$Conv3 \times 3$
Conv BN ReLU
C_{out}
$Conv1 \times 1$
Conv BN ReLU
C_{out}
(a) Double Convolutional Block
P_1ConvB = DConvB
P_2ConvB = DConvB DConvB
P_3ConvB = DConvB DConvB DConvB
P_4ConvB = DConvB DConvB DConvB DConvB
(b) Convolutional Blocks for each Pyramid Level

Fig. 4. Detailed representation of the convolutional blocks used in our proposed PAG-PVTUnet architecture.

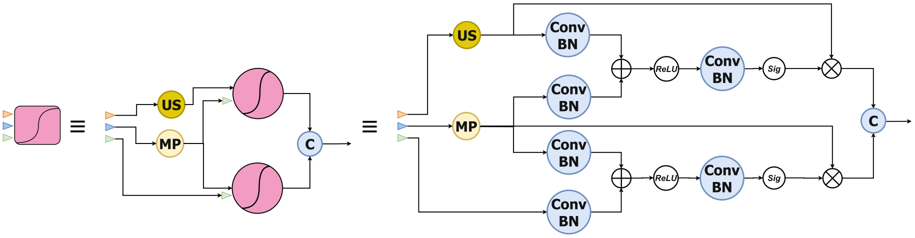


Fig. 5. The proposed Dual-Attention Gate, left the general structure and the right is the detailed one.

skips to decoder levels. Each module incorporates Max pooling and Up sampling to match the spatial resolution of all three input features.

In summary, the main encoder branch receives the output of the convolution block at the input image of resolution $H \times W$, and then four Dual-Attention gates are utilized to obtain the encoded features, which is passed through DConvB to extract the next level features of the main branch. Both Dual-Attention Gate and DConvB are used to fuse then extract higher features, respectively, constructing a strong encoder for medical imaging segmentation.

4 Datasets and Tasks

For abdominal organ segmentation, we utilized the Synapse multi-organ segmentation dataset, which has emerged as a benchmark dataset for evaluating the performance of medical imaging segmentation approaches in recent years. Following the precedent set by many state-of-the-art works, we adopted the training and validation splits introduced in the TransUnet paper [7]. In summary, the Synapse dataset consists of 30 abdominal CT scans introduced first hand in the MICCAI 2015 Multi-Atlas Abdomen Labeling Challenge and it has the pixel level annotation of 8 abdominal organs (aorta, gallbladder, spleen, left kidney, right kidney, liver, pancreas, spleen, and stomach).

For infection segmentation tasks, we focused on the multi-class segmentation of Covid-19, specifically Ground Glass Opacity (GGO) and Consolidation, along with Bone Metastasis (BM) segmentation. These tasks present significant challenges due to the variability in infection shape, position, intensity, and type. For

Covid-19 segmentation, we followed the methodology outlined in [3,5] and utilized two datasets from [20]. In total, we used 879 slices for training and 50 slices for testing. Among the training slices, 345 and 272 slices contained GGO and Consolidation infection types, respectively. The remaining slices without infection were included to enable the models to learn more features about healthy tissues. For BM segmentation, we utilized the BM-Seg dataset [1], employing the same data splits as in [1]. The BM-Seg dataset comprises 23 CT-scans, each covering one of multiple organs depending on the spread and primary cancer (e.g., lung, breast). In total, the dataset contains 1517 slices, and we employed a five-fold cross-validation strategy to evaluate the performance of the segmentation models.

For Gland and Nuclear segmentation tasks, we utilized two distinct datasets: the Gland segmentation dataset (GlaS) [24] and the MoNuSeg dataset [12], respectively. The GlaS dataset comprises 165 images specifically designed for gland segmentation tasks. On the other hand, the MoNuSeg dataset consists of 44 images tailored for nuclear segmentation tasks. Following the evaluation scheme proposed in [28], we conducted three iterations of five-fold cross-validation for each task. This approach ensures robust evaluation by splitting the dataset into five subsets, using each subset as a validation set once while training on the remaining four subsets. The results are corresponding to the mean and standard deviation of the three runs, where each run result corresponds the five folds cross-validation results.

Table 1. Comparison on Abdominal Multi-Organs Segmentation. DSC and HD95 are the average dice score and 95% Hausdorff distance of the 8 classes, respectively. The fourth column to the last show the Dice-score (DSC) for each class.

Architecture	Average		Aorta	Gallbladder	Kidney (L)	Kidney (R)	Liver	Pancreas	Spleen	Stomach
	DSC↑	HD95↓								
U-Net [21]	74.68	36.87	84.18	62.84	79.19	71.29	93.35	48.23	84.41	73.92
Att-Unet [16]	75.57	36.97	55.92	63.91	79.20	72.71	93.56	49.37	87.19	74.95
V-Net [15]	68.81	–	75.34	51.87	77.10	80.75	87.84	40.05	80.56	56.98
TransUnet [7]	77.48	31.69	87.23	63.13	81.87	77.02	94.08	55.86	85.08	75.62
MTUnet [29]	78.59	26.59	87.92	64.99	81.47	77.29	93.06	59.46	87.75	76.81
UCTransNet [28]	78.23	26.75	–	–	–	–	–	–	–	–
TransClaw U-Net [28]	78.09	26.38	85.87	61.38	84.83	79.36	94.28	57.65	87.74	73.55
ST-Unet [33]	78.86	20.37	85.68	69.05	85.81	73.04	95.13	60.23	89.15	72.78
Swin-Unet [14]	77.58	27.32	81.76	65.95	82.32	79.22	93.73	53.81	88.04	75.79
VM-UNet [22]	81.08	19.21	86.40	69.41	86.16	82.76	94.17	58.80	89.51	81.40
TransCeption [2]	82.24	20.89	87.60	**71.82**	86.23	80.29	95.01	65.27	91.68	80.02
Ours	**83.43**	**15.82**	**89.67**	68.89	**86.74**	**84.88**	**95.87**	**68.75**	**92.01**	**80.66**

5 Experiments and Results

5.1 Experimental Setup

To produce our experiments, we mainly used PyTorch [18] library for deep learning. Each architecture is trained for 100 epochs with an initial learning rate of 0.1 and Adam optimizer. The batch size is set to 16 images. The used machine has NVIDIA RTX A5000 GPU with 24 GB of memory, 11th Gen Intel(R) Core(TM) i9-11900KF (3.50GHz) CPU and 64 of RAM. Three types of active data augmentation are used; random rotate with an angle between $-35°$ and $35°$ with a probability of 10% and random Horizontal and vertical Flipping with probability of 20% for each.

5.2 Results

Tables 1, 2, 3, and 4 summarizes the comparison results with the state-of-the-art architectures in **Synapse, BM-Seg, Covid-19, and GlaS** and **MoNuSeg** datasets, respectively. These results show the superiority of our approach over the state-of-the-art architectures.

For the **Synapse dataset** results (Table 1), we selected comparison approaches that followed the same evaluation splits as [7]. Comparing with the TransUnet architecture, considered as the baseline for the Synapse dataset, our architecture demonstrated superior performance with improvements of 5.95% and 15.87 for Dice-Score and HD95, respectively. This indicates the efficacy of our approach in leveraging both Transformer and CNN features through the proposed Dual-Attention Gate.

Furthermore, our architecture surpassed state-of-the-art methods in terms of both average Dice-Score and HD95 metrics, demonstrating superior performance across all segmented classes. However, it is noteworthy that our approach did not achieve the highest segmentation performance specifically for the Gallbladder class.

In the **BM** segmentation comparison, we present the results obtained by comparing our method with the competing approaches outlined in the dataset paper [1] and four recent transformer-based architectures: SwinUnet [6], MTUnet [29], MISSFormer [10], and UCTransNet [28]. Our proposed approach showcased superiority over these architectures (see Table 2).

Moreover, the lower performance of Transformer-based architectures, such as SwinUnet and MTUnet, raises concerns about their ability to generalize across different tasks, especially for infection segmentation tasks. Infection segmentation tasks involve high variability in shape, type, position, and intensity of infections, which may cover only a small portion compared to the background.

In contrast, our approach exhibits a high ability to segment infection regions due to the rich features extracted and combined during the encoding phase. Additionally, the proposed Dual-Attention Gate effectively highlights prominent parts through multi-scale feature maps, making it well-suited for detecting infection regions.

Table 2. Comparison on Bone Metastasis Segmentation. F1-S, DSC and IoU are F1-score, Dice-Score and Intersection over Union, respectively.

Model	F1-S↑	DSC↑	IoU↑
U-Net [21]	79.46	72.26	65.93
AttUnet [16]	79.41	71.76	65.86
Unet++ [36]	79.74	71.99	66.31
AttUnet++ [13]	80.28	72.36	67.06
SwinUnet [6]	61.09	39.17	44.01
MTUnet [29]	58.59	44.30	41.45
MISSFormer [10]	81.44	70.42	68.73
UCTransNet [28]	83.62	73.88	71.85
Hybrid-AttUnet++ [1]	82.27	75.70	69.89
EDAUnet++ [1]	83.67	77.05	71.92
Ours	**85.01**	**79.70**	**73.92**

Table 3 provides a comprehensive summary for **Multi-classes Covid-19** segmentation results. This Table includes the results of our proposed approach and its comparison with three baseline CNN architectures (U-Net, Att-Unet, and Unet++), four state-of-the-art approaches for Covid-19 segmentation (CopleNet [27], AnamNet [17], SCOATNET [35], and EMB-TrAttUnet [5]), and four recent Transformer-based medical imaging segmentation approaches (SwinUnet [6], MTUnet [29], MISSFormer [10], and UCTransNet [28]).

Our analysis revealed that Transformer-based approaches exhibit limited generalization ability, achieving performance close to that of baseline CNN architectures. Additionally, a significant performance gap was observed between the segmentation of the two classes, primarily due to the minor presence of Consolidation compared to GGO, both in appearance and distribution within the lung. Remarkably, our proposed approach achieved the best performance, effectively reducing the gap in segmenting both classes compared to the comparison approaches. This highlights our method's exceptional capability to accurately highlight infection regions throughout all encoding blocks, leveraging the proposed Dual-Attention Gates.

Following the evaluation protocol and comparing the performance with the results obtained in [28], Table 4 presents a comprehensive comparison of our approach with the state-of-the-art methods for microscopic segmentation tasks, specifically **Gland** and **Nucleus** segmentation. From these results, it is evident that our proposed architecture outperforms the state-of-the-art methods, achieving the best performance in both Gland and Nucleus segmentation tasks. This further confirms the efficiency and versatility of our approach in various medical imaging segmentation tasks.

Table 3. Comparison on Multi-classes Covid-19 Segmentation. F1-S, DSC and HD95 are F1-score, Dice-Score and 95% Hausdorff distance, respectively. GGO and Con are the two types of Covid-19 infection known as Ground-Glass Opacity and Consolidation.

Architecture	Average			F1-S		DSC	
	F1-S↑	DSC↑	HD95↓	GGO	Con	GGO	Con
U-Net [21]	48.58	32.79	35.69	65.81±1.26	31.35±12.96	50.13±1.31	15.45±5.66
Att-Unet [16]	51.92	34.85	35.84	64.81±1.89	39.04±6.81	50.44±1.35	19.26±3.55
Unet++ [36]	48.51	41.48	44.06	65.69±1.29	31.31±6.67	51.65±4.12	31.31±6.67
CopleNet [27]	54.64	31.355	39.04	60.44±1.54	29.70±10.29	46.25±3.13	16.46±4.76
AnamNet [17]	48.53	34.875	34.78	65.10± 3.56	31.97±6.12	51.69±4.8	18.06±4.61
SCOATNET [35]	45.07	37.06	30.99	65.77±3.28	43.52±1.67	50.80±4.63	23.32±2.07
SwinUnet [6]	47.47	31.11	39.42	62.74±2.63	32.2±6.68	42.46±2.61	19.77±3.87
MTUnet [29]	42.30	30.60	37.50	57.83±2.57	26.78±7.39	42.97±2.78	18.24± 4.56
MISSFormer [10]	56.70	39.79	42.08	65.66 ±3.06	47.75±4.77	51.57±4.01	28.02±2.72
UCTransNet [28]	58.33	41.41	34.67	67.46±2.97	49.21±4.27	53.42±4.24	29.41±3.48
EMB-TrAttUnet [5]	65.16	48.18	27.47	70.06±0.03	60.26±0.92	59.14±0.87	37.23±0.97
Ours	**68.71**	**51.03**	**24.22**	**73.12±0.37**	**64.30±0.90**	**60.38±0.94**	**41.68±0.98**

Table 4. Comparison on Glas and MoNuSeg Segmentation datasets.

Ex	Architecture	GlaS		MoNuSeg	
		DSC	IoU	DSC	IoU
1	U-Net [21]	85.45±1.3	74.78±1.7	76.45±2.6	62.86±3.0
2	Unet++ [36]	87.56±1.2	79.13±1.7	77.01±2.1	63.04±2.5
3	AttUNet [16]	88.80±1.1	80.69±1.7	76.67±1.1	63.47±1.2
4	MRUNet [26]	88.73±1.2	80.89±1.7	78.22±2.5	64.83±2.9
5	TransUNet [7]	88.40±0.7	80.40±1.0	78.53±1.1	65.05±1.3
6	MedT [25]	85.92±2.9	75.47±3.5	77.46±2.4	63.37±3.1
7	Swin-Unet [14]	89.58±0.6	82.06± 0.7	77.69±0.9	63.77± 1.2
8	UCTransNet [28]	90.18±0.7	82.96±1.1	79.08±0.7	65.50±0.9
9	**Ours**	**94.20±0.55**	**89.29±0.91**	**79.62±0.7**	**66.31±0.6**

Table 5. Ablation study on Synapse Dataset and Covid-19. The importance of the following elements is studied: CNN Pyramid path (Pyr), PVT path (PVT) and the Vit Transformer (ViT). Mean Dice-Score (DSC) and 95% Hausdorff distance metrics are used for both tasks plus F1-Score (F1-S) for for Covid-19 task.

Architecture	Ablation			Synapse		Covid-19		
	Pyr	PVT	ViT	DSC↑	HD95↓	F1-S↑	DSC↑	HD95↓
(1) No Pyramid Path	✗	✓	✓	82.32	21.45	67.84	51.07	23.23
(2) No PVT	✓	✗	✓	79.44	22.92	65.98	50.25	24.05
(3) No ViT	✓	✓	✗	82.39	17.67	**68.92**	**51.69**	**21.53**
(4) **PAG-PVTUnet**	✓	✓	✓	**83.43**	**15.82**	68.71	51.03	24.22

5.3 Ablation Study

The aim of this section is to investigate the significance of the proposed encoding elements within our approach. We examine the importance of the following components: CNN Pyramid path (Pyr), PVT path (PVT), and the ViT Transformer (ViT), considering multi-organ abdominal segmentation (Synapse) and infection segmentation (Covid-19). The results are summarized in Table 5. In the first ablation experiment, it is evident that the Pyramid path plays a crucial role in Synapse segmentation, as removing it leads to a decrease in performance by 1.11% and 5.63 for DSC and HD95, respectively. Conversely, the results for Covid-19 segmentation show stable performance despite removing the Pyramid path.

In the second ablation study, it becomes apparent that Transformer features are vital for both tasks. Removing the PVT path results in a significant decrease in performance on the Synapse dataset, with a reduction of 4% and 7.1 for Dice-score and HD95, respectively. Similarly, for Covid-19 segmentation, the performance decreases by 2.73% and 0.78% for F1-score and Dice-score, respectively. Regarding the ViT block, the experiments demonstrate its importance in Synapse segmentation, likely due to the complexity of Synapse having more classes compared to Covid-19 segmentation. Additionally, the relatively smaller size of the Covid-19 dataset makes it challenging to train the ViT (base varaint), leading to potential overfitting. However, the experiments show only a minor decrease in performance. Overall, these findings underscore the significance of each component in achieving high performance in both Synapse and Covid-19 segmentation tasks, with particular emphasis on the Transformer features in enhancing segmentation accuracy.

In Table 6, we investigate the importance of the ViT block in our proposed architecture for binary segmentation tasks (BM-Seg, GlaS, and MoNuSeg Datasets). The results show that the ViT block can be omitted while preserving similar performance to PAG-PVTUnet.

Table 6. Ablation study for ViT block importance for binary segmentation (BM-Seg, GlaS and MoNuSeg Datasets).

Architecture	Bone Metastasis			GlaS		MoNuSeg	
	F1-S↑	DSC↑	IoU↑	DSC↑	IoU↑	DSC↑	IoU↑
No ViT	84.99	79.28	73.91	94.20	89.26	79.31	65.98
PAG-PVTUnet	85.01	79.70	73.92	94.20	89.29	79.62	66.31

6 Discussion

In this section, we investigate the number of parameters, number of FLOPs, and inference times of our approach in comparison with others. Table 7 summarizes these comparisons. Notably, our architecture (PAG-PVTUnet) slightly increases the number of parameters and FLOPs compared to state-of-the-art architectures. Compared to baseline architectures such as U-Net and AttUnet, our proposed architecture has a higher parameter count due to the inclusion of the Hybrid Encoder, which has proven its efficiency in handling complex and varied medical imaging segmentation tasks. Despite the larger parameter count, our approach still delivers competitive inference times. In fact, the inference time for a batch size of 50 slices is less than half a second, making it suitable for real-time scenarios.

On the other hand, Table 7 shows that our proposed architecture without the ViT Base block reduces the number of parameters and FLOPs by half and has a faster inference time. As depicted in Sect. 5.3, the ViT Base can enhance performance with more complicated tasks, such as the Synapse dataset, which contains 9 classes. In contrast, for binary tasks, this block can be omitted while preserving similar performance. Based on task complexity, the trade-off between model size, testing time, and accuracy can be decided.

Table 7. Number of parameters and FLOPs of different architectures and Testing Time for a batch size of 50 images.

Architecture	Numb of FLOPs	Numb of Params	Inf Time
U-Net	10.73 GMac	7.85 M	89 ms
AttUnet	11.05 GMac	7.98 M	102 ms
Unet++	26.51 GMac	9.16 M	244 ms
CopleNet	12.58 GMac	10.52 M	95 ms
AnamNet	19.48 GMac	15.63 M	117 ms
SCOATNET	29.75 GMac	40.21 M	407 ms
MISSFormer	7.21 GMac	42.46 M	223 ms
SwinUnet	15.12 GMac	41.38 M	160 ms
MTUnet	44.73 GMac	79.07 M	629 ms
UCTransNet	32.94 GMac	66.43 M	423 ms
EMB-TrAttUnet	47.71 GMac	121.13 M	475 ms
TransUnet	24.66 GMac	105.28 M	267 ms
PAG-PVTUnet (No ViT)	17.06 GMac	59.56 M	301 ms
PAG-PVTUnet	33.65 GMac	144.22 M	470 ms

7 Conclusion

In this paper, we introduce a novel hybrid architecture, termed PAG-PVTUnet, designed for medical imaging segmentation. By seamlessly integrating Convolutional Neural Networks (CNNs), Transformers, and a fusion branch encoder, we aim to address the limitations of existing approaches and improve segmentation accuracy. Our key innovation lies in enhancing the Att-Unet attention gate with our proposed Dual-Attention Gate mechanism. This mechanism facilitates the extraction of prominent features from multiple encoder branches, thereby capturing both local and global contextual information more effectively.

Through comprehensive evaluation across various segmentation tasks, including abdominal multi-organs segmentation, infection detection (Covid-19 and Bone Metastasis), and microscopic tissue segmentation (Gland and Nucleus), our proposed approach demonstrates state-of-the-art performance and remarkable generalization capabilities. The utilization of the Dual-Attention Gate mechanism enables efficient fusion of features from different encoder branches, leading to enhanced segmentation accuracy and robustness across diverse medical imaging datasets.

The contributions of this work extend beyond the development of a novel segmentation architecture. We present a significant advancement towards addressing the pressing need for efficient and adaptable segmentation solutions in medical imaging applications. By seamlessly integrating CNNs and Transformers, our approach provides a versatile framework capable of handling the high variability among diseases and segmentation tasks. Furthermore, our methodology lays the foundation for future research endeavors aimed at advancing medical imaging segmentation techniques and facilitating clinical decision-making processes.

References

1. Afnouch, M., et al.: Bm-seg: a new bone metastases segmentation dataset and ensemble of CNN-based segmentation approach. Expert Syst. Appl. **228**, 120376 (2023)
2. Azad, R., Jia, Y., Aghdam, E.K., Cohen-Adad, J., Merhof, D.: Enhancing medical image segmentation with transception: a multi-scale feature fusion approach. arXiv preprint arXiv:2301.10847 (2023)
3. Bougourzi, F., Distante, C., Dornaika, F., Taleb-Ahmed, A.: D-trattunet: dual-decoder transformer-based attention unet architecture for binary and multi-classes covid-19 infection segmentation. arXiv preprint arXiv:2303.15576 (2023)
4. Bougourzi, F., Distante, C., Dornaika, F., Taleb-Ahmed, A.: Pdatt-unet: pyramid dual-decoder attention unet for covid-19 infection segmentation from ct-scans. Med. Image Anal. **86**, 102797 (2023)
5. Bougourzi, F., Dornaika, F., Nakib, A., Taleb-Ahmed, A.: Emb-trattunet: a novel edge loss function and transformer-CNN architecture for multi-classes pneumonia infection segmentation in low annotation regimes. Artif. Intell. Rev. **57**(4), 90 (2024)
6. Cao, H., et al.: Swin-unet: Unet-like pure transformer for medical image segmentation. In: European Conference on Computer Vision, pp. 205–218. Springer (2022). https://doi.org/10.1007/978-3-031-25066-8_9
7. Chen, J., et al.: Transunet: Transformers make strong encoders for medical image segmentation. arXiv preprint arXiv:2102.04306 (2021)
8. Hatamizadeh, A., Tang, Y., Nath, V.: UNETR: Transformers for 3D Medical Image Segmentation, pp. 574–584 (2022)

9. He, X., Qi, G., Zhu, Z., Li, Y., Cong, B., Bai, L.: Medical image segmentation method based on multi-feature interaction and fusion over cloud computing. Simul. Model. Pract. Theory **126**, 102769 (2023)
10. Huang, X., Deng, Z., Li, D., Yuan, X., Fu, Y.: Missformer: an effective transformer for 2d medical image segmentation. IEEE Trans. Med Imaging (2022)
11. Khan, S., Naseer, M., Hayat, M., Zamir, S.W., Khan, F.S., Shah, M.: Transformers in vision: a survey. ACM Comput. Surv. (CSUR) (2021)
12. Kumar, N., et al.: A multi-organ nucleus segmentation challenge. IEEE Trans. Med. Imaging **39**(5), 1380–1391 (2019)
13. Li, C., Tan, Y.: Attention unet++: a nested attention-aware u-net for liver CT image segmentation. In: 2020 IEEE International Conference on Imaging Processing (ICIP), pp. 345–349. IEEE (2020)
14. Liu, Z., et al.: Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 10012–10022 (2021)
15. Milletari, F., Navab, N., Ahmadi, S.A.: V-net: fully convolutional neural networks for volumetric medical image segmentation. In: 2016 Fourth International Conference on 3D Vision (3DV), pp. 565–571. Ieee (2016)
16. Oktay, O., Schlemper, J., Folgoc, L.L.: Attention U-Net: learning Where to Look for the Pancreas. arXiv:1804.03999 [cs] (May 2018), arXiv: 1804.03999
17. Paluru, N., Dayal, A., Jenssen, H.B.: Anam-Net: anamorphic depth embedding-based lightweight CNN for segmentation of anomalies in COVID-19 Chest CT Images. IEEE Trans. Neural Netw. Learn. Syst. **32**(3), 932–946 (2021)
18. Paszke, A., Gross, S., Massa, F.: Pytorch: an imperative style, high-performance deep learning library. Adv. Neural Inform. Process. Syst. 8026–8037 (2019)
19. Petit, O., Thome, N., Rambour, C., Themyr, L., Collins, T., Soler, L.: U-Net transformer: self and cross attention for medical image segmentation. In: Lian, C., Cao, X., Rekik, I., Xu, X., Yan, P. (eds.) MLMI 2021. LNCS, vol. 12966, pp. 267–276. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-87589-3_28
20. RADIOLOGISTS: COVID-19 CT-scans segmentation datasets (2019). http://medicalsegmentation.com/covid19/, last visited: 18-08-2021
21. 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
22. Ruan, J., Xiang, S.: Vm-unet: Vision mamba unet for medical image segmentation. arXiv preprint arXiv:2402.02491 (2024)
23. Shamshad, F., Khan, S., Zamir, S.W.: Transformers in Medical Imaging: a Survey, arXiv:2201.09873 (Jan 2022) [cs, eess]
24. Sirinukunwattana, K., et al.: Gland segmentation in colon histology images the glas challenge contest. Med. Image Anal. **35**, 489–502 (2017)
25. Valanarasu, J.M.J., Oza, P., Hacihaliloglu, I., Patel, V.M.: Medical transformer: gated axial-attention for medical image segmentation. In: de Bruijne, M., et al. (eds.) MICCAI 2021. LNCS, vol. 12901, pp. 36–46. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-87193-2_4
26. Wang, F.K., Huang, Y.Q.: Mrunet: a two-stage segmentation model for small insect targets in complex environments. J. Integrative Agricult. **22**(4), 1117–1130 (2023)
27. Wang, G., Liu, X., Li, C., Xu, Z.: A noise-robust framework for automatic segmentation of COVID-19 pneumonia lesions from CT images. IEEE Trans. Med. Imaging **39**(8), 2653–2663 (2020). https://doi.org/10.1109/TMI.2020.3000314

28. Wang, H., Cao, P., Wang, J., Zaiane, O.R.: Uctransnet: rethinking the skip connections in u-net from a channel-wise perspective with transformer. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, pp. 2441–2449 (2022)
29. Wang, H., et al.: Mixed transformer u-net for medical image segmentation. In: ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2390–2394. IEEE (2022)
30. Wang, W., Xie, E.e.a.: Pvt v2: improved baselines with pyramid vision transformer. Comput. Vis. Media **8**(3), 415–424 (2022)
31. Wang, W., Chen, C., Ding, M., Yu, H., Zha, S., Li, J.: TransBTS: multimodal brain tumor segmentation using transformer. In: de Bruijne, M., et al. (eds.) MICCAI 2021. LNCS, vol. 12901, pp. 109–119. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-87193-2_11
32. Wu, H., Chen, S., Chen, G.e.a.: FAT-Net: feature adaptive transformers for automated skin lesion segmentation. Med. Image Analysis **76**, 102327 (2022). https://doi.org/10.1016/j.media.2021.102327
33. Zhang, J., Qin, Q., Ye, Q., Ruan, T.: St-unet: swin transformer boosted u-net with cross-layer feature enhancement for medical image segmentation. Comput. Biol. Med. **153**, 106516 (2023)
34. Zhang, Z., Liu, Q., Wang, Y.: Road extraction by deep residual U-Net. IEEE Geosci. Remote Sensing Lett. **15**(5), 749–753 (2018). https://doi.org/10.1109/LGRS.2018.2802944, conference Name: IEEE Geoscience and Remote Sensing Letters
35. Zhao, S., Li, Z., Chen, Y.e.a.: Scoat-net: a novel network for segmenting covid-19 lung opacification from ct images. Pattern Recogn. 108109 (2021)
36. Zhou, Z., Rahman Siddiquee, M.M., Tajbakhsh, N., Liang, J.: UNet++: a nested U-Net architecture for medical image segmentation. In: Stoyanov, D., et al. (eds.) DLMIA/ML-CDS -2018. LNCS, vol. 11045, pp. 3–11. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00889-5_1
37. Zhu, Z., He, X., Qi, G., Li, Y., Cong, B., Liu, Y.: Brain tumor segmentation based on the fusion of deep semantics and edge information in multimodal mri. Inform. Fus. **91**, 376–387 (2023)

A Weighted Discrete Wavelet Transform-Based Capsule Network for Malware Classification

Tonghua Qiao[1,2], Chunjie Cao[1,2], Binghui Zou[1,2], Fangjian Tao[1,2], Yinan Cheng[1,2],
Qi Zhang[3], and Jingzhang Sun[1,2](✉)

[1] Lab of Heterogeneous Network Security and Governance, Haikou, China
{tonghuaqiao,caochunjie,bhzou,tfj,Innacheng,
jingzhangsun}@hainanu.edu.cn
[2] School of Cyberspace Security, Hainan University, Haikou, China
[3] Faculty of Data Science City University of Macau, Macau SAR, China
qizhang@cityu.mo

Abstract. The rise of sophisticated malware poses a grave threat to computer security, challenging traditional detection methods. Traditional malware detection techniques, which primarily rely on feature engineering and defining rules to identify malware, are gradually failing to comprehensively and accurately detect emerging malware variants. To address this problem, we proposed an efficient and accurate novel malware classification model based on capsule network with Weighted Discrete Wavelet Transform (WDWTCN). In the field of malware classification, datasets are predominantly composed of grayscale images, which present samples in a single-channel format. To more adeptly capture the key features within malware samples, WDWTCN incorporates the Weighted Discrete Wavelet Transform (WDWT) as a preprocessing step for input data. By converting the single-channel input of grayscale images into a four-channel input, our model becomes significantly more efficient in extracting feature information from malware samples. We used ResNet50 as the feature extraction layer of the capsule network. By integrating global and local features, the model can comprehensively understand and classify malware, thereby enhancing the detection performance of malware. Finally, to validate the performance of the WDWTCN, we conducted an extensive experimental evaluation on three large datasets, i.e., BIG2015, MalImg and BODMAS. The experimental results showed that WDWTCN achieves significant improvements in malware detection accuracy, reaching 98.93%, 99.21% and 96.14% accuracy, respectively.

Keywords: Malware Classification · Image Processing · Data Preprocessing · Wavelet Transform · Capsule Network

1 Introduction

Malware describes software programs designed maliciously, posing significant threats to computer systems, network security, and user privacy. According to the World Economic Forum Global Risks Report 2023 [1], malware manifests in various forms, such as

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 259–274, 2025.
https://doi.org/10.1007/978-3-031-78128-5_17

information leakage, financial fraud, denial-of-service attacks, malicious propagation, exploitation of system vulnerabilities, cryptocurrency mining, and Internet of Things (IoT) threats. Cybersecurity Ventures predicted that global cybercrime expenses will reach $10.5 trillion by 2025 because of data theft from malware [2]. Moreover, KELA Research Q1 2023 report highlights the phenomenal growth of the ransomware market [3]. Supply chain attacks and ransomware-as-a-service have emerged as prominent trends in the industry. Building a reliable and precise malware detection method is vital in these situations.

Typically, the malware detection process includes malware analysis, feature extraction, and classification. In malware analysis, dynamic [4] and static analysis [5] are commonly employed, followed by reversing engineering to represent malware in various forms [6]. The emergence of deep learning techniques has led to the use of Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Generative Adversarial Networks (GAN), etc., as the basic methods for malware detection. These techniques have significantly advanced malware detection in various fields [7–10]. Researchers have focused on extracting malware features and converting them into grayscale or color graphics. For instance, CNN-based classification methods may overlook the spatial location associations of features, resulting in the loss of location information. RNN encounters challenges such as long-term dependency, gradient vanishing, or having problems dealing with longer sequences. GAN can suffer from pattern collapse, producing samples that are heavily concentrated on a few patterns and lack diversity.

To address these issues, Capsule Network (CN) has been proposed for malware classification [11]. Unlike traditional CNN, CN utilize dynamic routing to establish connections between lower-level and higher-level capsules. This mechanism ensures the preservation of spatial hierarchical relationships and the part-to-whole relationships. By analyzing these connections, we can trace the contribution of specific features to the final classification, thereby clearly understanding the decision-making process. The use of capsules encapsulates the positional information of objects (such as location, orientation, and scale), adding a layer of interpretability. For example, if a type of malware is characterized by a specific structural pattern, the capsules will capture these patterns, making it easier to understand why the model classifies a sample into a particular category. Traditional CN uses only one convolutional layer to extract features. To better capture the features of the original samples, portray the positional relationship between each feature, and improve the performance of malware classification, this study proposed an efficient and accurate novel malware classification model based on capsule network with Weighted Discrete Wavelet Transform (WDWTCN). The main contributions of our work can be summarized as follows:

- We introduce a method that uses Weighted Discrete Wavelet Transform (WDWT) to preprocess grayscale images into four-channel inputs. This method improves feature representation and detail capturing by weighting different frequency bands, highlighting important features. By utilizing the multi-channel capability of modern deep learning models, this preprocessing significantly boosts classification performance.
- Moving away from traditional CN setups, we remove the single convolutional layer and use ResNet50, purposely leaving out the average pooling and fully connected layers to make the network simpler. This updated method greatly boosts the network's

skill in identifying and showing complex patterns in the data, thanks to a multilayer convolutional setup supported by residual connections.

- In the encoder segment, we applied transposed convolution as a replacement for the traditional CN's fully connected layers. This approach enables a more effective feature extraction and better preservation of spatial information, which contributes to enhanced model performance.
- We conducted experiments on three large malware datasets, i.e., BIG2015, MalImg and BODMAS to demonstrate the performance of WDWTCN in image-based malware classification tasks.

2 Related Works

2.1 Static Analysis and Dynamic Analysis Methods

Static analysis methods focus on analyzing static features and code of malware samples. The extracted features were further used to classify the malware into different families. Patricia and Ganorkar [12] proposed a static analysis approach for malware detection using various machine learning models, such as Artificial Neural Networks (ANN) and Support Vector Machines (SVM). Their study found that ANN achieved the highest classification accuracy of 93.44%, making it the most effective model under memory constraints. Lee et al. [13] proposed a method for automatically detecting malware mutants and grouping them for classification. The method utilizes n-gram analysis and clustering coefficients to improve accuracy. Its objective is to efficiently identify malware mutations and categorize them into groups based on code sequence patterns. However, static analysis techniques rely primarily on manual analysis and do not execute code [14].

Dynamic analysis detects malware from its operations, analyzing how it behaves and interacts in real-time [15]. This involves capturing live Windows API function calls [16] [17]. Advancing these methods, AI-based techniques have emerged, Molina-Coronado et al. [18] analyze ten influential research works on Android malware detection using a unified evaluation framework, and Huang et al. [19] innovating with a multi-task deep learning framework for classifying software. Additionally, Pan et al. [20] integrated hardware tracking with interpretable ML for a balance between precision and computational efficiency. Despite these advances, dynamic analysis still wrestles with environment specificity, detection balance, computational demands, and evasion tactics [21]. Often, it's augmented by combining static and dynamic analysis, among other security measures, which, albeit thorough, can be time-consuming [22].

2.2 Image-Based Methods

To minimize the expense of feature engineering and reduce dependence on domain expert knowledge, Nataraj et al. [23] pioneered the research by transforming malicious binary files into grayscale images. Jian et al. [24] proposed a visual malware detection framework with deep neural networks converting executable files into RGB images for improved detection performance and highlighting the benefits of data augmentation.

Xiao et al. [25] also proposed a novel visualization method, called Coloured Labeled Boxes (CoLab), for labeling sections of PE files and highlighting information about the distribution of sections in the transformed malware images. Yajamanam et al. [26] found that GIST descriptors performed quite well in image-based malware classification. Deep learning has a potential advantage in that GIST features do not need to be extracted during the training process.

2.3 CapsuleNet-Based Methods

CN is a deep learning model originally developed by Hinton et al. [27] for its greater expressiveness in dealing with spatial structure and pose changes. Although CN has achieved some impressive results in other areas, its use in malware detection is still at a relatively early stage. Cayir et al. [28] proposed a random CN based on bootstrap aggregation methods for the classification of unbalanced malware, which can achieve 99.56% accuracy on the dataset, but there are some limitations in this paper. Firstly, the estimation of the number of capsules in the RCNF model is limited, and its implementation can only contain a maximum of 10 capsules per RCNF model. Secondly, the training time is long; for the MMCC dataset, it takes 5 h to train the RCNF with 10 capsules. Pei et al. [29] proposed a version of CN that successfully detected DGA botnets. They achieved this by using raw pixel values extracted from domain names and extracting visual features from grayscale images, which helped to improve classification accuracy. Zou et al. [30] proposed FACILE, a CN designed for malware image classification. The goal of FACILE is to achieve effective classification with fewer capsules while retaining rich hierarchical information. The network architecture prioritizes the capture of basic hierarchical features in malware images, which helps to improve the classification performance.

3 Methodology

3.1 Model Overview

The WDWTCN architecture is shown in Fig. 1, mainly composed of three parts: WDWT, decoder, and encoder. WDWT and the differences from traditional CN will be explained in detail in the following sections.

Weighted Discrete Wavelet Transform (WDWT). In our proposed model, we introduced the WDWT in the pre-processing stage. This method involved a two-dimensional discrete wavelet transform to the sample image, utilizing weighting coefficients. As a result, the number of channels in the image increased from its original value to four. During the wavelet transform, the images were decomposed into sub-bands representing distinct frequency components. Specifically, each wavelet transform generated a low-frequency sub-band (LL: low frequency on row and column) and three high-frequency sub-bands (LH: low frequency row, high frequency column; HL: high frequency row, low frequency column; HH: high frequency on row and column). The wavelet transform depended on the sub-band LL, obtained from the previous level. This iterative process continued until the desired level was reached, completing the wavelet transform of the image. The value ranged from 1 to the specified level. Weighting coefficients were assigned to the sub-band LL to enhance the capture of local structures, features, and overall image description.

Figure 2 shows the distributions of wavelet coefficients in the first-level and second-level wavelet transforms, with each sub-band containing its respective coefficients. It can be observed that each wavelet transform can be seen as a sampling of the image in the horizontal and vertical directions at specific intervals. With each transformation, the spatial resolution decreases by half, resulting in a sub-band spatial resolution of compared to the original image after the i-th level wavelet transform.

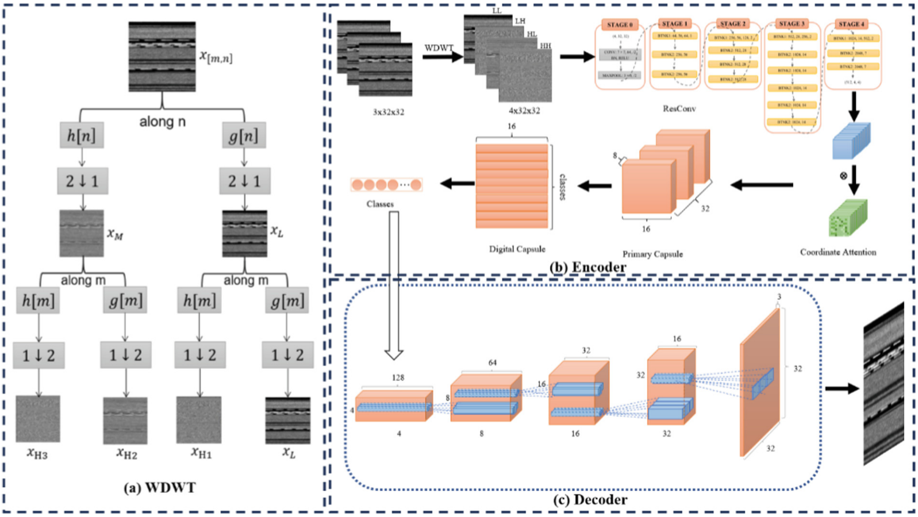


Fig. 1. The architecture of WDWTCN. (a) the WDWT process, the first step is to perform a high pass, a low pass, and a down-sampling in the n-direction. Then, the high and low passes, along with down-sampling, are performed in the m-direction, generate images for four channels. (b) the encoder, comprising the Resnet50 feature extraction module, the primary capsule module, and the digital capsule module. (c) the decoder of WDWTCN.

In WDWT, we use Haar as the wavelet basis for image classification tasks. The Haar wavelet basis has a powerful ability to capture local features while also being able to capture detailed features of an image at different scales. We first define signals and filters that would be utilized.

- $x[n]$: Discrete input signal of length n.
- $g[n]$: Low-frequency filter.
- $h[n]$: High-frequency filter.
- $\downarrow Q$: Down-sampling filter, if $x[n]$ is used as input, the output is $y[n] = x\left[Q_n\right]$.

The input signal becomes $x[m, n]$ as shown in Fig. 1 (a). The n-direction is first processed with high and low pass frequency reduction. The process is as follows:

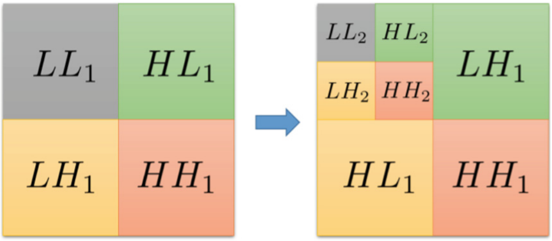
$$v_{1,L}[m, n] = \sum_{k=0}^{K-1} x[m, 2n - k]g[k] \tag{1}$$

$$v_{1,H}[m,n] = \sum_{k=0}^{K-1} x[m, 2n-k]g[k] \tag{2}$$

Then $v_{1,L}[m,n]$ and $v_{1,H}[m,n]$ are high-passed, low-passed and downgraded frequency in the m direction, and for the LL band multiplied by our weighting factor w as followed:

$$x_{1,LL}[m,n] = w * \sum_{k=0} v_{1,L}[2m-k, n]g[k] \tag{3}$$

The WDWT stage enriches feature extraction by utilizing a 4-channel image, allowing more nuanced information capture at multiple scales and orientations. Each channel reflects distinct image traits, including edges, textures, and corners, through different wavelet coefficients. The wavelet transform decomposes the image into multiple bands, securing detail preservation while offering a layered representation, which is critical for robust image classification.



(a) First-level wavelet transform distribution for i=1

(b) Second-level wavelet transform distribution for i=2

Fig. 2. The wavelet transform of an image generates a low-frequency sub-band (LL) and three high-frequency sub-bands (LH, HL, and HH). These sub-bands are obtained by sampling the horizontal and vertical directions of the image. From the previous LL level, the process can be repeated at any image level.

Encoder. To preserve spatial information, we utilized Resnet50 without the average pooling and fully connected layers of the traditional CN. By using Resnet50 as a feature extractor, the model can more efficiently capture valuable image features, leading to improved classification performance.We introduced the CoordAttLayer, which was proposed by Hou et al. [31] between the feature extractor and the primary capsule in the encoder. The fundamental concept is to use location information to improve the model's attention to different location features and can help the network capture the importance of different locations in an image and consider the influence of location information when calculating attention weights to perceive fine-grained features and distinguish between different classes of images.

Decoder. To enable the encoder part to better extract features, preserve spatial information, and diminish the number of parameters and computational complexity, we followed the DCGAN [32] and used five transposed convolutions, while adding a BN layer after each transposed convolutional layer and using the ReLU function to elevate the model performance and enhance the quality of the generated images.

3.2 Capsule Network

Capsule Network establish connections between lower-level and higher-level capsules through dynamic routing, enabling them to capture spatial hierarchical relationships and part-whole relationships in data. Lower-level capsules extract basic features, while higher-level capsules integrate these features to form more complex abstract concepts, such as an object's pose information (position, orientation, scale, etc.). This mechanism makes the model's decision path more transparent; even without visualization, the basis for classification decisions can still be explained by analyzing feature contributions. This is particularly suitable for security application scenarios where understanding and trust are essential. Below, we will introduce the main components of Capsule Network and how they operate.

Primary Capsule. The Primary Capsule is a set of capsules located at the lower level of the mesh. It is used to extract primary features from the input feature map into capsules. In our model, the feature maps are extracted by Resnet50. Each primary capsule is connected to a different location in the input feature map and generates a vector representing the presence and properties of the features detected at the corresponding location. The capsule's output is a vector and then passed to the next layer of capsules for dynamic routing through the routing algorithm for further learning of the higher-level feature representation.

Digital Capsule. The Digital Capsule is similar to the Fully Connected Layer in that it is used to encode spatial information and make the final classification.

Dynamic Routings. To overcome the problems of spatial invariance and pose variation that exist in traditional CNN, the dynamic routing algorithm is not just a weighting and updating technique, but a way of communicating and coordinating the flow of information between capsules. It establishes the correlation between capsules by dynamically assigning importance weights between capsules, allowing information to flow along the most important paths. Similar to a dynamic routing system in traffic management, the algorithm intelligently adapts and optimizes the communication between capsules based on the different attributes of the input data. The capsules can interact adaptively, learning and extracting high-level features.

The input for dynamic routing is a lower-level capsule (Primary Capsule), denoted as $u_i \in R^{k\times 1}$. k represents the quantity of neurons in each capsule. Next, a transformation matrix $W_{ij} \in R^{p\times k}$ is applied to encode the positional relationships between lower-level and higher-level features. The output capsule has p neurons, and the input $u_i \in R^{k\times 1}$ is transformed into a prediction vector:

$$\hat{u}_{j|i} = W_{ij} \cdot u_i \tag{4}$$

The prediction vectors are then weighted and summed:

$$s_j \leftarrow \sum_i c_{ij} \cdot \hat{u}_{j|i} \tag{5}$$

where s_j is the total input to the high-level capsule, c_{ij} is the weighting coefficient, i.e., the coupling coefficient. c_{ij} Denotes the probability distribution that capsule i will activate capsule j, which is learned by the dynamic routing algorithm, a workflow we describe in detail later. c_{ij} Are calculated as follows:

$$c_{ij} = \frac{\exp(b_{ij})}{\sum_i \exp(b_{ij})} \tag{6}$$

where b_{ij} represents the predicted logarithmic prior probability of coupling $u_{j|i}$ with the senior capsule s_j. b_{ij} is calculated as:

$$b_{ij} = b_{ij+} + v_j \cdot \hat{u}_{j|i} \tag{7}$$

At the beginning of the first dynamic routing, the value of b_{ij} is initialized to zero. The number of routes is also set to 3 times for best results with reference to the experience. Next, the value of the vector c_{ij} is calculated, which is the ownership weight of the lower capsule i. The use of Softmax here ensures that all ownership weights c_{ij} are non-negative and that their sum is equal to 1. Since the values of all b_{ij} are initialized to zero, the values of all coefficients c_{ij} will be equal after the first iteration. We then compute a linear combination of the input vectors s_j. The vectors from the previous step are passed through the non-linear squash function to generate the output vector v_j, which is passed to all the higher-level capsules, whose values are updated with each iteration. At the end of the routing iteration, a high-level capsule v_j is output.

Squash. Capsules are specialized groups of neurons that collectively act as information carriers. Unlike individual neurons, the extent of a capsule in a CN signifies the likelihood or probability that a particular feature is present. To preserve this representation, squashing is used in the CN instead of the traditional activation function ReLU, which ensures that short vectors can be compressed to a length close to 0 and long vectors to a length close to 1. The transformation process can be characterized as:

$$v_j \leftarrow \frac{\|s_j\|^2}{1+\|s_j\|^2} \frac{s_j}{\|s_j\|} \tag{8}$$

Loss Function. In capsule network, a commonly used loss function is Margin Loss, which is designed to ensure that correctly classified capsules output high values and misclassified capsules output low values. Since the wavelet transform is capable of extracting multi-scale features of an image, we have considered adding a reconstruction error to the Margin Loss to ensure that the wavelet-transformed image can be efficiently reconstructed and classified by the capsule network. The overall loss function is defined as follows:

$$L_{loss} = L_{margin} + \alpha L_{recon} \tag{9}$$

where

$$L_{\text{recon}} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_i - \hat{\mathbf{x}}_i^2 \tag{10}$$

The Margin Loss L_{margin} is given by:

$$L_{\text{margin}} = T_j \cdot max\big(0, m^+ - \|v_j\|\big)^2 + \lambda\big(1 - T_j\big) \cdot max\big(0, \|v_j\| - m^-\big)^2 \tag{11}$$

T_j is 1 if the capsule represents the correct class and 0 otherwise, m^+ and m^- are the margin thresholds typically set to 0.9 and 0.1 respectively, and λ is a down-weighting factor typically set to 0.5. The term L_{recon} represents the reconstruction error, ensuring that the capsule network not only classifies the image correctly but also reconstructs it accurately, enhancing the robustness and interpretability of the model. The inclusion of the reconstruction loss helps in capturing the intricate details of the wavelet-transformed image, thereby improving the overall performance of the capsule network in image classification tasks.

4 Experiments

4.1 Experimental Environment and Dataset

The three datasets we selected for the experiment are BIG2015, MalImg, and BODMAS, as shown in the Fig. 3. For three malware datasets:

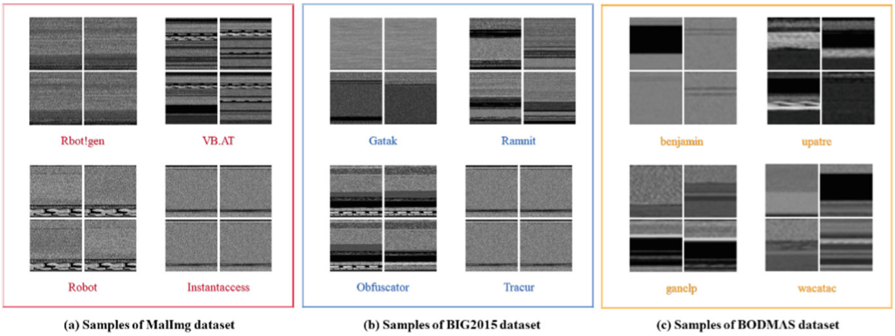
- Nataraj et al. [23] published the MalImg dataset, which comprises 9339 grayscale images encompassing 25 malware families. These families include Rbot!gen, VB.AT, Fakerean, Instantaccess, and several family variants such as C2Lop.p, C2Lop.gen!g and Swizzor.gen! etc.,
- Microsoft [33] provided the data science community with an unprecedented malware dataset: BIG2015, consisting of 10,868 samples covering 9 malware families.
- We also evaluated our model on the BODMAS [34] dataset, which contains 57,293 malware samples and 77,142 benign samples. From this dataset, we selected 14 families, totaling 34,368 samples (with each family containing more than 1,000 samples) for our experiments. Before training, we converted the binary files into 32x32 grayscale images.

Our experiments were all performed under Intel(R) Xeon(R) CPU E5-2630L v3, 32G RAM, and a GPU of GeForce RTX 3080 Ti-12G. We used PyTorch 1.13.0 and Python 3.8 to create the training environment, and we used AdamW as the model of the optimizer. To evaluate the generalization performance of WDWTCN, we used a 5-fold cross-validation approach. During each round of cross-validation, four of these subsets (80% of the total) were used to train the model, while the remaining subset (20% of the total) was used for testing. The parameter settings remained the same as when no cross-validation was performed, except that the data sets were divided differently. The training of the model involved 50 epoch, 32 batch size, and an initial learning rate of 0.00038. The learning rate is also decayed by a smaller percentage at the end of each epoch. The tuning strategy is as follows, we set α to 0.99.

$$lr = lr_0 \cdot \alpha \tag{12}$$

4.2 Baselines

WDWTCN was compared with the following state-of-the-art experimental methods (both CNN methods and CN methods): MalCaps [35], 1D CNN [36], DenseNet + DEAM [37], MCFT-CNN [38], MSAAM [39], RCNF [28], ImgConvAttn-Greyscale[40], Mobilenet V2 [41] and Inception V3[41].



(a) Samples of MalImg dataset (b) Samples of BIG2015 dataset (c) Samples of BODMAS dataset

Fig. 3. Samples of malware images sourced from three distinct datasets: MalImg, BIG2015 and BODMAS.

4.3 Evaluation Metrics

We adopt the standard metrics of Accuracy, Precision, Recall and F1-Score to evaluate the performance of different malware classification methods as follows:

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \tag{13}$$

$$Precision = \frac{TP}{TP+FP} \tag{14}$$

$$Recall = \frac{TP}{TP+FN} \tag{15}$$

$$F1\text{ score} = \frac{2*Precision*Recall}{Precision+Recall} \tag{16}$$

TP, TN, FP, FN stand for true positive, true negative, false positive and false negative.

4.4 Results and Analysis

Evaluation of the Weighting Coefficients for WDWT. We set the proportion of training and test sets to 4:1, and performed five cross-validations for each weighting coefficient. During the experiments, we only weighted the LL sub-bands. The classical CN and DWT achieved accuracies of 97.01% and 96.14% on the MalImg and BIG2015 datasets, respectively. With the addition of WDWT, the accuracy of both datasets is improved as shown in Fig. 4. On the MalImg dataset, the optimal LL coefficient is 0.312, resulting in a 1.1% increase in accuracy. On the BIG2015 dataset, the optimal LL coefficient is 0.25, resulting in a 0.913% increase in accuracy.

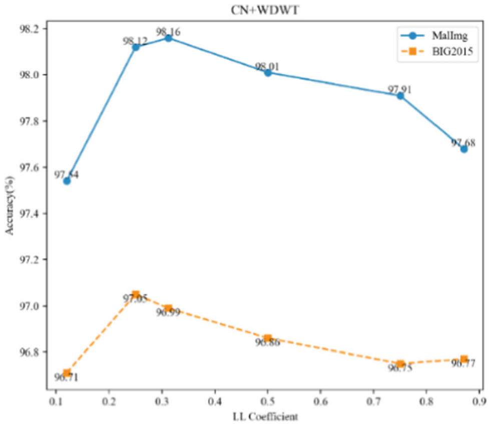


Fig. 4. The evaluation of various LL weighting coefficients was conducted on the MalImg dataset and BIG2015 dataset.

Evaluation of WDWTCN as Compared to the Baselines. The WDWTCN was evaluated against other baseline models on the MalImg, BIG2015 and BODMAS datasets. Figure 5 depicts the confusion matrix of WDWTCN on three datasets. Figure 6 illustrates the precision, recall, and F1 scores for each malware family on the three datasets. WDWTCN is excellent at detecting the two families Kelihos_ver3 and Simda. As can be seen from Fig. 6 (a), the overall performance is very good for the different family classifications. However, there are some errors for the C2LOP family and the Swizzor family, which could be attributed to the relatively small sample size. Tables 1, 2 and 3 show the performance comparison of WDWTCN with other major malware classification methods.

Experimental results on the BIG2015 dataset show that the WDWTCN model exhibits superior performance in four metrics. In particular, our feature extraction layer is richer compared to the MalCaps model. WDWTCN improves accuracy by 1.15%, precision by 6.265%, recall by 6.849%, and F1 score by 7.134%. WDWTCN outperforms 1D CNN, DenseNet + DEAM, MCFT-CNN, and MSAAM on all four metrics.

On the other hand, experimental results on the MalImg dataset confirm the robust performance of the WDWTCN model. WDWTCN achieves an accuracy of 99.213% on this dataset, outperforming other baseline models. It also shows significant advantages in precision, recall and F1 score, reaching 97.176%, 97.648%, and 97.412%, respectively. These results further demonstrate the superior performance of WDWTCN in malware detection and classification.

On the BODMAS dataset, our model achieved an accuracy of 96.144 and a recall of 96.129. The model performed well across various families, but its performance on the sillyp2p family was average, possibly due to the small image size leading to incomplete feature extraction.

In summary, the WDWTCN model demonstrates exceptional performance metrics such as accuracy, precision, recall, and F1 score in the malware classification task. By using WDWT to introduce a multi-scale adaptive attention mechanism, we effectively

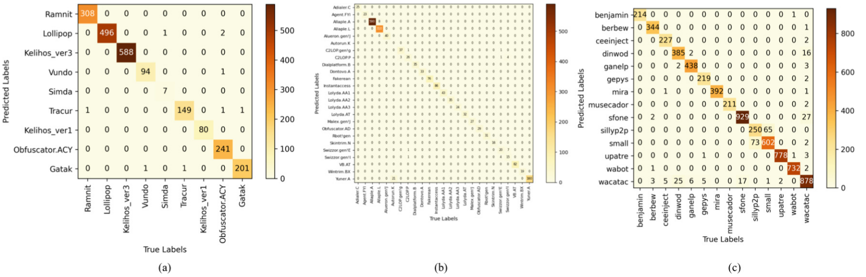


Fig. 5. Confusion matrix of WDWTCN on (a) BIG2015 dataset, (b) MalImg dataset and (c) BODMAS dataset.

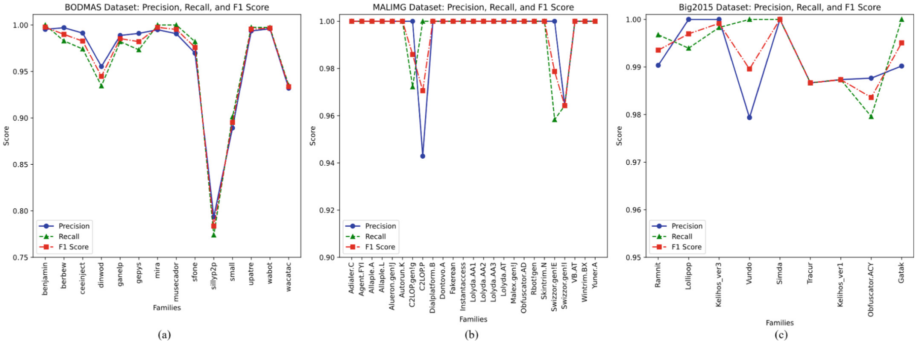


Fig. 6. Precision, Recall, and F1 Score of WDWTCN on (a) BODMAS dataset, (b) MalImg dataset and (c) BIG2015 dataset.

extracted feature information, increased the granularity of the capsules, and improved the results of malware detection and classification.

4.5 Ablation Experiments

To investigate the effect of the WDWT, ResConv as the extraction layer of the CN on WDWTCN, we conducted four ablation experiments. Table 4 shows the accuracy and F1 values on both datasets, where we refer to the wavelet transform as DWT, the weighted wavelet transforms as WDWT and ResNet50 as the CN extraction layer as ResConv. From Table 4, we meticulously analyzed each component of the model to determine their individual contributions to its overall efficacy. Notably, the DWT component achieved an accuracy of 97.017% and an F1 score of 94.258% on the MalImg dataset, alongside an accuracy of 96.14% and an F1 score of 94.423% on the BIG2015 dataset, establishing a foundational performance benchmark for our experiments. The enhanced WDWT variant, building upon DWT, registered a 1% improvement in accuracy for both datasets by incorporating additional features, indicating a notable enhancement in model performance attributable to the WDWT element.

Furthermore, the ResConv model recorded an accuracy of 98.72% and an F1 score of 96.892% on the MalImg dataset, and an accuracy of 98.263% and an F1 score of

Table 1. Performance comparison between WDWTCN and other baselines on the BIG2015 dataset.

	Models	Accuracy	Precision	Recall	F1 Score
	MalCaps [34]	0.97782	0.92359	0.91463	0.91334
	1D CNN [35]	0.96290	–	–	–
	Densenet + DEAM [36]	0.97300	0.95300	0.95400	0.95400
	MCFT-CNN [37]	0.98055	0.95114	0.95793	0.95401
	MSAAM [38]	0.98200	0.96600	0.96300	0.96400
Ours	**WDWTCN**	**0.98932**	**0.98624**	**0.98312**	**0.98468**

Table 2. Performance comparison between WDWTCN and other baselines on the MALIMG dataset.

	Models	Accuracy	Precision	Recall	F1 Score
	MalCaps [34]	0.98190	0.94213	0.95348	0.95679
	1D CNN [35]	0.98910	–	–	–
	Densenet + DEAM [36]	0.98500	0.96900	0.96600	0.96700
	RCNF [28]	0.98720	–	–	0.96610
	MCFT-CNN [37]	0.99100	0.97360	0.97300	0.97280
	MSAAM [38]	0.99200	0.98000	0.97900	0.97900
Ours	**WDWTCN**	**0.99213**	**0.97176**	**0.97648**	**0.97412**

Table 3. Performance comparison between WDWTCN and other baselines on the BODMAS dataset.

	Models	Accuracy	Precision	Recall	F1 Score
	MalCaps [34]	0.94800	0.94700	0.95000	0.94800
	SeqConvAttn [40]	0.93330	–	–	0.92900
	ImgConvAttn-Grey [40]	0.95020	–	–	0.96700
	Mobilenet V2 [41]	0.92290	0.98020	0.93340	0.97280
	Inception V3 [41]	0.91870	0.**98780**	0.92140	0.95350
Ours	**WDWTCN**	**0.96144**	**0.95914**	**0.96129**	**0.96021**

96.253% on the BIG2015 dataset, evidencing the advantageous effect of the ResConv component's integration. Ultimately, the WDWTCN model outperformed with an accuracy of 99.213% and an F1 score of 97.412% on the MalImg dataset, and an accuracy of 98.932% and an F1 score of 98.468% on the BIG2015 dataset. These results distinctly highlight the substantial impact of each component on the model's performance, underscoring the WDWTCN model's exceptional capability in processing the MalImg and BIG2015 datasets.

Table 4. Accuracy and F1 Score of WDWTCN and its variants on the MalImg dataset and BIG2015 dataset.

Model Variations	MalImg		BIG2015	
	Accuracy	F1 Score	F1 Score	F1 Score
DWT	0.97017	0.94258	0.96140	0.94423
WDWT	0.98160	0.96482	0.97050	0.95856
ResConv	0.98720	0.96892	0.98263	0.96253
WDWTCN (WDWT + ResConv)	**0.99213**	**0.97412**	**0.98932**	**0.98468**

5 Conclusion

We have developed a capsule-based malware classification model named WDWTCN, which incorporates a WDWT module to accentuate salient frequency information, thereby enhancing classification accuracy. By integrating ResNet50 in place of the conventional feature extraction layer used in traditional convolutional networks, our model achieves a more robust and comprehensive feature representation. The WDWTCN model outperforms the baseline across the BIG2015, MalImg and BODMAS datasets. For future research, we aim to employ automated methods for navigating the optimal parameter configuration and use generative models to create samples. This will address the imbalance within software family distributions in the datasets. As for interpretability of capsule network, we intend to devote a future paper to this issue.

Acknowledgment. This work was supported by the Science and Technology Talents Innovation Project of Hainan Province (KJRC2023B13 and KJRC2023D30).

References

1. Wef global risks report 2023. https://www.weforum.org/publications/global-risks-report-2023/, Accessed 29 Mar 2024
2. 2023 cybersecurity almanac. https://cybersecurityventures.com/cybersecurity-almanac-2023/, Accessed 29 Mar 2024
3. Ransomware and Network Access Sales. https://www.kelacyber.com/ransomware-and-network-access-sales-report-q2-2023/, Accessed 29 Mar 2024
4. Li, C., et al.: DMalNet: dynamic malware analysis based on API feature engineering and graph learning. Comput. Sec., 102872 (2022)

5. Taheri, R., et al.: Similarity-based Android malware detection using Hamming distance of static binary features. Futur. Gener. Comput. Syst. **105**, 230–247 (2020)
6. Abusitta, A., et al.: Malware classification and composition analysis: a survey of recent developments. J. Inform. Sec. Appli, **59**, 102828 (2021)
7. Li, Q., et al.: CNN-based malware variants detection method for the internet of things. IEEE Internet of Things J. **8**(23) (2021)
8. Kumar, S., et al.: SDIF-CNN: stacking deep image features using fine-tuned convolution neural network models for real-world malware detection and classification. Appl. Soft Comput. **146**, 110676 (2023)
9. Liu, Z., Cao, C., Tao, F., Sun, J.: Revisiting graph contrastive learning for anomaly detection. Accepted by 2023 European Conference on Artificial Intelligence (ECAI)
10. Zhu, E., Zhang, J., Yan, J., Chen, K., Gao, C.: N-gram MalGAN: evading machine learning detection via feature n-gram. Digital Communications and Networks **8**(4), 485–491 (2022)
11. Wang, S.W., Zhou, G., Lu, J.C., Zhang, F.J.: A novel malware detection and classification method based on capsule network. In Artificial Intelligence and Security: 5th International Conference, ICAIS 2019, New York, NY, USA, 26–28 July 2019, Proceedings, Part I 5, pp. 573–584. Springer International Publishing (2019).
12. Baker del Aguila, R., et al.: Static malware analysis using low-parameter machine learning models. Computers **13**(3), 59 (2024)
13. Lee, T., et al.: Automatic malware mutant detection and group classification based on the n-gram and clustering coefficient. J. Supercomput. **74**, 3489–3503 (2018)
14. Or-Meir, O., Nissim, N., Elovici, Y., Rokach, L.: Dynamic malware analysis in the modern era—A state of the art survey. ACM Computing Surveys (CSUR) **52**(5), 1–48 (2019)
15. Darabian, H., et al.: Detecting cryptomining malware: a deep learning approach for static and dynamic analysis. J. Grid Comput. **18**, 293–303 (2020)
16. Bayer, U., et al.: Scalable, behavior-based malware clustering. NDSS **9**, 8–11 (2009)
17. Santos, I., Devesa, J., Brezo, F., Nieves, J., Bringas, P. G.: Opem: a static-dynamic approach for machine-learning-based malware detection. In: International joint conference CISIS 2012-ICEUTE 2012-SOCO 2012 special sessions, pp. 271–280. Springer, Berlin (2013)
18. Molina-Coronado, B., et al.: Towards a fair comparison and realistic evaluation framework of android malware detectors based on static analysis and machine learning. Comput. Sec. **124**, 102996 (2023)
19. Santos, I., Devesa, J., Brezo, F., Nieves, J., Bringas, P.G.: OPEM: A Static-Dynamic Approach for Machine-Learning-Based Malware Detection. In: Herrero, Á., et al. (ed.) International Joint Conference CISIS 2012-ICEUTE 2012-SOCO 2012 Special Sessions. AISC, vol 189. Springer, Berlin (2013). https://doi.org/10.1007/978-3-642-33018-6_28
20. Pan, Z., Sheldon, J., Mishra, P.: Hardware-assisted malware detection and localization using explainable machine learning. IEEE Trans. Comput. **71**(12), 3308–3321 (2022)
21. Sihwail, R., Khairuddin O., Zainol Ariffin, K.A.: A survey on malware analysis techniques: static, dynamic, hybrid and memory analysis. Int. J. Adv. Sci. Eng. Inf. Technol **8**(4–2), 1662–1671 (2018)
22. Ucci, D., et al.: Survey of machine learning techniques for malware analysis. Comput. Sec. **81**, 123–147 (2019)
23. Nataraj, L., et al.: Malware images: visualization and automatic classification. In: Proceedings of the 8th International Symposium on Visualization for Cyber Security, pp. 1–7 (July 2011)
24. Jian, Y., et al.: A novel framework for image-based malware detection with a deep neural network. Comput. Sec. **109**, 102400 (2021)
25. Xiao, M., et al.: Image-based malware classification using section distribution information. Comput. Sec. **110**, 102420 (2021)
26. Yajamanam, S., et al.: Deep learning versus gist descriptors for image-based malware classification. In: Icissp, pp. 553–561 (2018)

27. Sabour, S., et al.: Dynamic routing between capsules. In: Advances in Neural Information Processing Systems 30 (2017)
28. Çayır, A., et al.: Random CapsNet forest model for imbalanced malware type classification task. Comput. Sec., 102133 (2021)
29. Pei, X., et al.: A two-stream network based on capsule networks and sliced recurrent neural networks for DGA botnet detection. J. Netw. Syst. Manage. **28**, 1694–1721 (2020)
30. Zou, B., et al.: FACILE: A capsule network with fewer capsules and richer hierarchical information for malware image classification. Comput. Secur. **137**, 103606 (2024)
31. Hou, Q., Zhou, D., Feng, J.: Coordinate attention for efficient mobile network design. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. (2021)
32. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)
33. Ronen, R., et al.: Microsoft malware classification challenge. arXiv preprint arXiv:1802.10135 (2018)
34. Yang, L., et al.: BODMAS: an open dataset for learning based temporal analysis of PE malware. In: 2021 IEEE Security and Privacy Workshops (SPW). IEEE (2021)
35. Zhang, X., Wu, K., Chen, Z., Zhang, C.: MalCaps: a capsule network based model for the malware classification. Processes **9**(6), 929 (2021)
36. Lin, W.-C., Yeh, Y.-R.: Efficient malware classification by binary sequences with one-dimensional convolutional neural networks. Mathematics **10**(4), 608 (2022)
37. Wang, C., Zhao, Z., Wang, F., Li, Q.: A novel malware detection and family classification scheme for IoT based on DEAM and DenseNet. Security and Communication Networks **2021**, 1–16 (2021)
38. Kumar, S.: Malware classification with fine-tune convolution neural networks using traditional and transfer learning in Internet of Things. Futur. Gener. Comput. Syst. **125**, 334–351 (2021)
39. Wang, C., Zhao, Z., Wang, F., Li, Q.: MSAAM: A multiscale adaptive attention module for IoT malware detection and family classification. Sec. Commun. Netw. (2022)
40. Lu, Q., et al.: Self-attentive models for real-time malware classification. IEEE Access **10**, 95970–95985 (2022)
41. Hai, T.H., et al.: A proposed new endpoint detection and response with image-based malware detection system. IEEE Access **11**, 122859–122875 (2023)

Data-Driven Spatiotemporal Aware Graph Hybrid-hop Transformer Network for Traffic Flow Forecasting

Lin Hang(✉), Qianqian Ren, and Xingfeng Lv

Department of Computer Science and Technology, Heilongjiang University,
Harbin 150080, China
`linhang@s.hlju.edu.cn`, `{renqianqian,lvxingfeng}@hlju.edu.cn`

Abstract. In recent years, researchers have shown significant interest in various fields such as transportation, social networks, and recommendation algorithms regarding graph neural networks. However, achieving high-accuracy traffic flow prediction in transportation proves highly challenging due to its complex spatiotemporal dependencies and nonlinear traffic patterns. This paper proposes a method that utilizes a Hybrid-hop attention mechanism to establish spatiotemporal correlations through the combination of geographical and semantic information between different time steps. To reduce time complexity, we set the receptive field of attention to neighboring spatial nodes and introduce a dynamic spatiotemporal perception graph driven by data to capture hidden spatiotemporal dependencies. Additionally, we design a multi-scale gated convolution mechanism to extract dynamic temporal dependencies from multi-receptive field features of various scales. Experimental results on public transportation network datasets (METR-LA and PEMS-BAY) demonstrate the model's excellent performance.

Keywords: Traffic prediction · Transformer · Data-driven Graph

1 Introduction

With the continuous upgrade of relevant hardware and the popularity of ride-hailing applications, the scale of traffic data has reached an unprecedented level. Against this backdrop, data-driven traffic flow prediction has attracted widespread attention. Accurate and timely traffic prediction not only provides a reference for urban traffic road planning and emergency traffic management but also offers travel advice to people, helping them devise better route plans to improve their quality of life.

In recent years, the rapid development of deep learning has led to its widespread application in processing spatiotemporal data. For example, compared to traditional methods, the use of Graph Convolutional Networks (GCN),

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 275–289, 2025.
https://doi.org/10.1007/978-3-031-78128-5_18

can better capture spatiotemporal correlations in non-Euclidean spatial structures of road networks [8]. However, existing GCN methods mostly rely on predefined graphs constructed from distance metrics or other geographical connections, resulting in static adjacency matrices. This limitation hinders their ability to reflect the complex, dynamic spatiotemporal correlations within the road network.

Furthermore, the impact of traffic flow between nodes in time series data is significant. This is specifically evident in a spatial graph, where one node can influence another across multiple time intervals, sometimes extending several jumps away. Previous research has struggled to effectively model these direct temporal and spatial effects by using different modules and mechanisms for time and space separately and then integrating them. Clearly, this approach may not perform well in the situation mentioned above. Therefore, it is necessary to consider directly integrating these spatiotemporal influences into modeling.

At different time steps, the time series exhibits dynamic pattern similarity and the interweaving of random irregularities. In terms of similarity, homogeneous wide dynamic congestion between morning and evening peaks and stable traffic pattern deviations between workdays and holidays are evident. Meanwhile, the irregularity is influenced by various factors such as personal driving habits, adaptive traffic control strategies, and unexpected road events. This will cause variations in traffic conditions at any given time step, making it impractical to use the same node update mechanism at different time steps. Therefore, simultaneous consideration of the time correlation at different time steps is crucial for capturing the dynamic temporal dependence in the road network. Given the outstanding performance of Transformer architecture in language modeling [5]and computer vision [14], we posit that employing self-attention mechanisms to model the dynamic spatiotemporal relationships in the graph will be an effective approach.

To address the above-mentioned issue, we propose a new neural network framework called Data-driven Spatiotemporal Aware Graph Hybrid-hop Transformer Network (D2HTN) for Traffic Flow Forecasting. Our specific contributions are as follows:

- The paper introduces a novel Transformer design to simulate different types of causality in spacetime. By employing the M-GSA Block for joint modeling of spacetime, the model is able to better capture the hidden semantics in the spacetime graph compared to the conventional segregated modeling approaches.
- Furthermore, we introduce EMAG (Earth Mover Augmented Graph) as a data-driven graph to explore spatial correlations between nodes in the past, replacing the static predefined graph.
- The introduction of Hybrid-hop is aimed at reducing model complexity, where temporal attention is focused on the node's data at different time steps, while the spatial attention is constrained within one hop, with additional relevant points selected based on EMAG to maintain its scalable complexity. Historical

data of the attended points in spacetime also become part of the attention scope.
- Additionally, the use of multi-scale gating technology enhances the perception of temporal dependency across the entire road network by extracting time features through convolution at multiple scales.

2 Related Work

2.1 Traffic Forecasting

In recent years, with the breakthrough and development of deep learning techniques in various fields such as speech recognition(SR) and image processing, an increasing number of researchers have also gradually started to apply deep learning techniques to traffic prediction research work. Deep neural network models have gained a lot of attention due to their ability to better capture the dynamic characteristics of traffic data and thus get better results for the target task [16]. The models can be divided into two categories according to whether spatial dependence is being considered or not. Some methods consider only temporal dependences, such as feed-forward neural networks to achieve the traffic flow prediction task. Since recurrent neural networks (RNNs) and their variants long short-term memory networks (LSTMs) [18], gated recurrent units (GRU) [4] make effective use of the self-loop mechanism, they learn temporal dependencies well and obtain better prediction results. However, these models consider temporal characteristics but ignore spatial dependencies, making the changes in traffic data unconstrained by the urban road network and unable to accurately predict the traffic state on the road. Making full use of spatiotemporal correlation is the key to solving the traffic prediction problem, and many studies have improved on this basis to further capture the spatiotemporal features in the traffic data to achieve more accurate traffic flow prediction. Some research [17]employs CNNs to capture spatial correlations of adjacency in traffic data, which is inspired by the rapid development of computer vision (CV) research [10]. Subsequent work has also combined CNN and RNN to model spatial-temporal correlations simultaneously, obtaining good traffic prediction results, A graph convolutional recurrent attention network is proposed for traffic flow prediction [3].

In order to better characterize the spatiotemporal features of traffic data, many studies have improved on this basis to further capture the spatiotemporal features in traffic data [21] to achieve more accurate traffic flow prediction [11], etc. A spatiotemporal sequence-to-sequence network is proposed in [2], which adopts an enhanced diffusion convolutional network and a temporal convolutional network to capture the dynamic spatial correlations and sequential and periodic temporal correlations. A learnable positional attention mechanism and a sequential component are designed in [20] to effectively aggregate information from adjacent roads and model the traffic flow dynamics.

2.2 Transformer Models

The Transformer was first proposed by Google's research team in 2017, primarily for natural language processing tasks. Its design marks a departure from traditional sequence models that rely on recurrent neural networks (RNN) or long short-term memory networks (LSTM), as it employs a self-attention mechanism to globally model sequence information. In recent times, numerous variants of the Transformer, such as Informer [23], Pdformer [7], and Pyraformer [13], have been introduced, each tailored to enhance its efficiency in specific domains. In the field of computer vision and video processing, similar to traffic flow prediction, modeling attention in both spatial and temporal domains is essential. Vivit [1] extracts temporal and spatial tokens from videos, while Video Swin transformer [15] employs a sliding window approach to compute local spatiotemporal attention within the window. Inspired by these respective studies, we incorporated some of these methods into our own model. In the realm of traffic prediction, researchers have begun integrating graph datasets into their models to better model spatiotemporal information. This has led to the introduction of attention-based spatiotemporal graph convolutional networks, such as GMAN [22] and DSTAGNN [9]. They embed spatiotemporal information into each input token, capturing dynamic correlations using attention mechanisms and learning spatiotemporal features using graph convolution. In response to the limitations of these methods in directly simulating spatiotemporal effects, our model performs attention in both spatial and temporal dimensions.

3 Preliminary

3.1 Problem Formulation

Definition 1 *(Road Network). We define the road network as* $G = (V, E, A)$*, where* $V = \{v_i\}_{i=1,2,\cdots,N}$ *represents the set of* N *nodes in the road network,* $E = \{e_{ij}\}$ *represents the set of edges indicating connectivity between the nodes, and* $A \in R^{N \times N}$ *describes the connectivity of the network. For* $v_i, v_j \in V$*, if* $(v_i, v_j) \in E$*, then* $A_{ij} = 1$ *indicating that corresponding road segments are connected, 0 otherwise.*

Definition 2 *(Traffic Feature Matrix). On the road network* G*, at any time step* t*, we can use* $X_t \in R^{N \times C \times d}$ *to represent its traffic state which will be dynamically changing over time, where* C *represents the types of traffic parameters (such as traffic volume, speed, etc.).In this paper, we consider the speed as the feature (hence* $C = 1$*)*

Problem Statement(Traffic Foretasting): Given a graph G and feature matrix of historical T' time steps, the goal of traffic prediction is to learn a function $\mathcal{F}$ to predict the traffic volume on the road network G for the future T time steps $X^{(t+1):(t+T)} \in R^{N \times C \times T}$, The mapping relationship is shown as follows:

$$X^{(t+1):(t+T)} = \mathcal{F}[X^{(t-T'+1):(t)}; G] \tag{1}$$

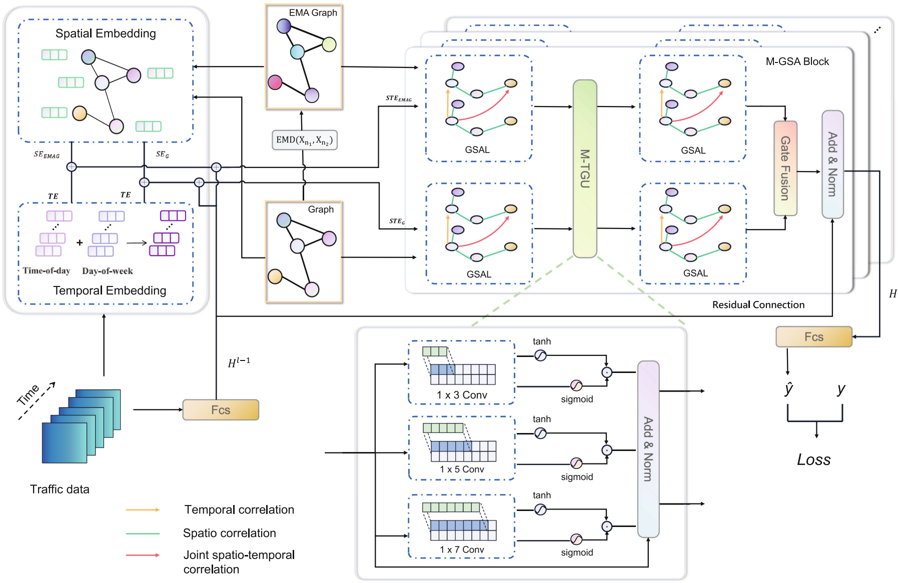


Fig. 1. The overall network architecture of the D2HTN and the specific structure of the M-GSA Block and M-STGU, especially, when $l = 1$, H^{l-1} is the output of X passing through the FCS layer.

3.2 Earth Mover's Distance

The Earth Mover's Distance (EMD) is a method for measuring the difference between two probability distributions, representing the minimum cost to transform one distribution into another. This distance is also known as the Wasserstein distance. Given the original distribution P_r and the target distribution P_θ, the Earth Mover's distance is defined as:

$$W[P_r, P_\theta] = \frac{inf}{\gamma \in \Pi[P_r, P_\theta]} \int_x \int_y \gamma(x, y) d(x, y) dx dy \tag{2}$$

where γ is a joint probability distribution of $\in \Pi[P_r, P_\theta]$ with marginal distributions that exactly match P_r and P_θ, that is $\int \gamma(x, y)dy = P_r(x)$ and $\int \gamma(x, y)dx = P_\theta(y)$. Here, d(x, y) represents the cost of moving a unit mass from x to y, often derived from the Minkowski distance. The inf denotes the minimum value among all possible schemes for transforming a probability distribution P_r into another distribution P_θ, the solution with the smallest cumulative moving distance, and the cost of this scheme is $W[P_r, P_\theta]$.

4 Methodolody

In this section, we introduced the proposed D2HTN and how it addresses the traffic flow prediction problem through direct spatiotemporal modeling.

D2HTN adopts the architecture shown in Fig. 1, which primarily consists of four parts: the Embedding Layer, Geographic-Semantic Attention Layer (GSA Layer), Multi-scale Spatial Temporal Gated Unit block (M-STGU), and Gate Fusion.

4.1 Spatial-Temporal Embedding

The Transformer model lacks the ability, possessed by recurrent neural networks (RNNs), to capture the relative positional information of elements within a sequence. Therefore, the introduction of Positional Encoding (PE) becomes necessary to provide the model with information concerning the positions of elements in the input sequence, enabling it to better understand the temporal and spatial relationships among different elements within the sequence. In this section, we will introduce how to obtain Spatial Embedding(SE_*) and Temporal Embedding(TE) in the model respectively. Adding the two can obtain the unique representation STE_* of different nodes at different time steps, and the mathematical expression for this process is as follows:

$$STE_* = Concat(SE_*, TE) \tag{3}$$

Spatial Embedding. In terms of spatial embedding, inspired by [6], we adopt Laplacian Positional Encodings, which utilizes graph structural information to precompute Laplacian eigenvectors and employ them as the position information for nodes. Subsequently, we integrate this approach into the Transformer, where it effectively characterizes the distances between nodes in a graph. The Laplacian positional encoding formula is given by:

$$\begin{aligned} SE_* &= f(\Delta, k) \\ \Delta &= I - D^{-1/2} A D^{-1/2} = U^T \Lambda U \end{aligned} \tag{4}$$

where A is a $n \times n$ dimensional adjacency matrix, D is the degree matrix, Λ and U correspond to the eigenvalues and eigenvectors, respectively. Specifically, we utilize k smallest non-trivial eigenvectors of a node as its positional encoding. To maintain the same dimensionality as the input X, we apply a linear transformation to it in the FC Layer and finally obtain the spatial embedding SE_*.

Temporal Embedding. In terms of temporal embedding, to incorporate real-world temporal features, we employ a straightforward time-step generation for temporal embedding. This involves encoding each token with a periodic time encoding from the day-of-week and time-of-day dimensions to create two-dimensional temporal embeddings. Subsequently, these embeddings are then also fed into the FC layer to be dimensionally adjusted to be summed with the positional embeddings, and finally obtain the temporal embedding TE.

4.2 Earth Mover Augmented Graph

In traffic prediction, accurately extracting the spatial correlations between nodes is a critical issue. It is widely recognized that the simple traffic flow propagation between nodes and their neighboring nodes is one source of spatial dependency. However, the spatial dependency between nodes with similar functionalities, despite being distant, is often overlooked. This is due to the existence of similar traffic flow patterns, leading to spatial dependency. Capturing such information is challenging through static graphs.

Inspired by Panaretos and Zemel's work, We calculate the similarity between nodes using Earth Mover (EM) distance and then derive EMAG from it. The traffic data collected for each node in the road network at different time steps can be seen as discrete data in a multi-dimensional space. We then transformed the discrete data into probability distributions for each node. The EM distance measures the minimal effort required to reshape one probability distribution into another, indicating that smaller EM distances imply a higher similarity between nodes. Specifically, let's denote the traffic flow data for N recording points over D days as $X \in R^{N \times D \times d_t}$, where d_t represents the time step of daily records (e.g., d_t=288 for records taken every 5 min). Each record point represents a day's traffic data in the form of a vector. Multiple days of traffic data can be represented as a sequence of vectors: $X = (r_{n1}, r_{n2}, ..., r_{nD})$,$r_{nd} \in R^{d_t}$, where$d \in [1, D]$. The probability distribution at a certain node n can be expressed as:

$$P_n(X_d = m_{nd}), m_{nd} = \frac{\|r_{nd}\|_2}{\sum_{d=1}^{D} \|r_{nd}\|_2} \tag{5}$$

where m_{nd} represents the proportion of traffic volume on a certain day over a period of time.

After obtaining the probability distribution, we use the Earth Mover's Distance (EMD) mentioned in Sect. 3.2 (Eq. 2)to calculate the difference between the two probability distributions. For example, in the model application, the EMD from the traffic flow at point n_1 on day i to the traffic flow at point n_2 on day j is calculated using the following formula:

$$EMD(X_{n_1}, X_{n_2}) = \frac{inf}{\gamma \in \mathit{\Pi}[P_{n_1}, P_{n_2}]} \int_x \int_y \gamma(x, y)(1 - \frac{r_{n_1,x}^{\top} \cdot r_{n_2,y}}{\|r_{n_1,x}\|_2 \times \|r_{n_2,y}\|_2})dxdy \tag{6}$$

the symbol 'inf' represents the infimum,γ is the joint probability distribution, with the requirement that the marginal distributions are u and v, $d(x, y)$ is the cost function, in the model, we use the cosine distance between traffic flow vectors:

$$cost(r_{n_1,i}, r_{n_2,j}) = 1 - \frac{r_{n_1,i}^{\top} \cdot r_{n_2,j}}{\|r_{n_1,i}\|_2 \times \|r_{n_2,j}\|_2} \tag{7}$$

the marginal distributions u and v as:

$$
\begin{aligned}
u &= \int \gamma(x,y)dy = \frac{\|r_{n_1,x}\|_2}{\sum_{x=1}^{D}\|r_{n_1,x}\|_2} \\
v &= \int \gamma(x,y)dx = \frac{\|r_{n_2,y}\|_2}{\sum_{y=1}^{D}\|r_{n_2,y}\|_2}
\end{aligned}
\tag{8}
$$

After calculating the EMD distances between all pairs of nodes, we obtain a matrix $A' \in R^{N\times N}$. To simplify subsequent computations, we define $A_E[i,j] = 1 - a'_{i,j}$. Higher values in A_E indicate stronger correlations between nodes i and j. We also introduce a learnable parameter W to adjust A_E, facilitating the dynamic generation of EMAG. Additionally, we set a hyperparameter θ such that $N_r = N \times \theta$. In each i node, we set N_r largest values to 1 and the remaining values to 0 to binaryize A_E and obtain $\hat{A} \in R^{N\times N}$ as the graph structure. This will be used in the subsequent Geographic-Semantic Attention Layer to capture the attention of N_r nodes most relevant to the given node.

4.3 Multi-scale Geographic-Semantic Attention

M-GSA Block consists of two components, geographic-semantic attention and M-STGU. These two modules are stacked together to update node embeddings in both spatial and temporal dimensions simultaneously. Additionally, EMAG generation and Multi-head attention will be introduced in this section.

Geographic-Semantic Attention. As mentioned in Sect. 4.2 we believe that the spatial dependency between nodes primarily arises from two sources: first, the simple traffic flow propagation between adjacent nodes, which we refer to as Geographic information, and second, from the similarity in functionality between urban nodes, which we refer to as Semantic information [7]. Thus, in terms of spatial dependency, we combine both to extract attention. Regarding temporal dependency, it is evident that the current traffic flow at a node is influenced by its own flow at the previous time step. Therefore, we extract attention from the node itself and its previous state. Finally, due to the need for joint modeling across space and time, we also capture the Geographic and Semantic information before time t, referring to this approach as Hybrid-hop.

The most direct way to perform joint spatiotemporal modeling, as shown in Fig. 2, is to calculate attention scores for all node pairs at different time steps (full attention). However, this approach not only involves high computational complexity but also introduces significant noise. To extract Geographic information, we only consider the neighboring nodes within one hop of the current node to compute their attention scores. This not only reduces the complexity of attention but also aligns more closely with real-life logic. The representation of node i at time step t is computed as follows:

$$h_{i,t} = \sum_{j\in\mathcal{N}(i)\cup i} \sum_{t'} \alpha(i,j,t,t')v_j^{t'} \tag{9}$$

where $\mathcal{N}(i)$ represents the set of all one-hop neighbors of node i. $\alpha(i, j, t, t')$ denotes the attention score between node (t, i) and (t', j), and $v_j^{t'}$ represents the projection of $h_{t',j}$. We can consider this as a form of masking with full attention. The node update computation utilizes the multi-head attention mechanism described in Sect. 4.2. The calculation formula is as follows:

$$GSA(X, A_{Geo}) = Softmax[(\frac{Q^{(l)}(K^{(l)})^T}{\sqrt{D_{QK}}})A_{Geo}]V^{(l)} \tag{10}$$

For the extraction of Semantic information, we used the EMAG calculated in 4.2 as a mask to filter nodes with the same semantics, that is, having similar city functions to extract attention. Like extracting Geographic information, we also use multi-head attention, and just replace A_{Geo} with A_{EMAG}. It is worth mentioning that the spatial correlation can be captured as the number of layers is stacked even if the distance between nodes is far away, but relatively speaking, the T is much smaller than the number of different locations N, so we use full attention in the time dimension.

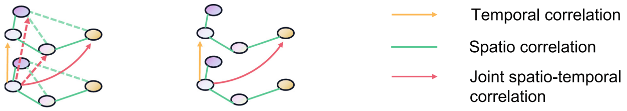


Fig. 2. A simple example of unified temporal-spatial modeling demonstrates the difference between the direct modeling approach (left) and the Hybrid-hop method. The dashed lines represent the additional attention scores required by the direct modeling method. Evidently, the Hybrid-hop method is more effective in improving efficiency and mitigating the influence of noise.

Multi-scale Spatial Temporal Gated Unit. As mentioned in Sect. 4.3 when using the Hybrid-hop method, there is less available temporal information. To address this issue, we integrate the logic of time cycles in real life and propose the incorporation of the M-STGU module, enabling the discovery of time patterns at different scales and capturing both short-term and long-term dynamics of traffic data. When considering convolutional networks, the selection of the kernel size is crucial. An excessively large kernel may fail to effectively capture short-term signals, while an overly small one may result in decreased capture of long-term signals. In image processing, the inception strategy is often employed, which involves the use of three kernel sizes (1×1, 3×3 and 5×5) with their outputs concatenated. However, this strategy may not be entirely suitable for time signals. In reality, time often exhibits periodicity. Daily life cycles are usually associated with fixed time slices such as 7, 12, 24 and 60. Thus, suggesting that the three kernels used in M-STGU are κ_1: 1×3, κ_2 : 1×5 (for covering weekdays), and κ_3 : 1×7. The specific structure is depicted in Fig. 1, where

M-STGU's input denoted as $Y^{(l)} \in R^{N\times M\times C^{(l)}}$ and its convolutional kernel as $K \in R^{1\times\kappa\times c^{(l)}\times 2c^{(l)}}$ result in $Y'^{(l)} = K * Y^{(l)} \in R^{N\times(M-(\kappa-1))\times 2C^{(l)}}$ with doubled channels after convolution. Then we perform gating operations on it, this can be expressed as:

$$K *_g Y^{(l)} = tanh(E) \odot sigmoid(F) \tag{11}$$

where $*_g$ is gated convolution operator,E and F respectively cut $Y^{(l)}$ in half based on the channel size. As the receptive field of the gated convolution increases, the ability to capture long-term temporal dynamics will also gradually improve. Based on Eq. 11, we propose that $Y_{out}^{(l)}$ can be represented as:

$$\begin{aligned} &Y_{out}^{(l)} = M\text{-}STGU(Y^{(l)}) \\ &= ReLU(Concat(Pooling(K_1 *_g Y^{(l)}), \\ &Pooling(K_2 *_g Y^{(l)}), Pooling(K_3 *_g Y^{(l)}) + Y^{(l)}) \end{aligned} \tag{12}$$

whereK_1, K_2, K_3 represent the use of convolutional kernel $\kappa_1, \kappa_2, \kappa_3$. The pooling layer with window size W reduces the feature dimensions to $(M - K - 1)/W$. The features are then concatenated to yield a feature dimension of $(3M - (\kappa_1 + \kappa_2 + \kappa_3 - 3))/W = M$. This results in an output dimension equal to the input dimension (where $\kappa_1, \kappa_2, \kappa_3, w$ are hyperparameters) enabling it to be connected with the input using residual connections. Finally, the output Y is obtained by applying the ReLU activation function. The use of gated and residual connections in M-STGU helps alleviate gradient vanishing, maintains non-linearity, and plays a role in extracting long-term and short-term features.

4.4 Gate Fusion and Output Layer

In the Gate Fusion block of our model, we employ the updating gate method of GRU. Firstly, we obtain the update gate control signal z through the static graph G and EMAG. Then, we use z to simultaneously forget and select memories from h_G and h_{EMAG}. The specific calculation is as follows:

$$\begin{aligned} z &= sigmoid(h_G + h_{EMAG}) \\ h_{out} &= (1 - z) \odot h_G + z \odot h_{EMAG} \end{aligned} \tag{13}$$

where the range of the gate control signal z is from 0 to 1. when z closer to 1 signifies that more data is retained as memory, while a signal closer to 0 signifies more data is forgotten. After connecting h_{out} and X through residual connection, we apply ReLU activation and LayerNorm, and finally map the output through the FC layer to obtain the predicted value in the future time dimension T'. In our experiments, we utilized the widely used root mean square error function as the loss function for model training. Assuming the predicted value is $\hat{X}_{t'}$ and the actual value is $X_{t'}$, the loss is as follows:

$$Loss(X_{t'}, \hat{X}_{t'}) = \frac{1}{T'} \sum_{t'=t+1}^{t+T'} |X_{t'} - \hat{X}_{t'}|^2 \tag{14}$$

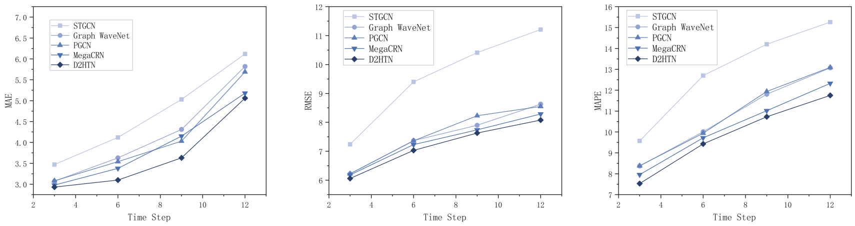
5 Experiment

5.1 Datasets

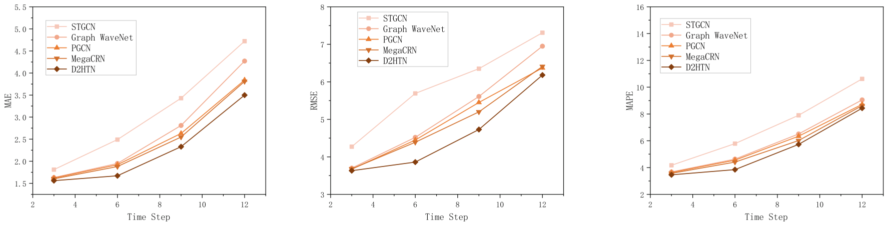
To evaluate the performance of our proposed model, we present experimental results of D2HTN and competing baselines on two spatiotemporal traffic datasets in this section. These datasets include METR-LA by [12]and PEMS-BAY which was published by [19]. The original traffic data is aggregated at 5-minute intervals, and spatial adjacency graphs for each dataset are constructed based on the actual road network (Table 1).

Table 1. Traffic Forecasting Result Comparison On Different Datasets.

Datasets	Methods	30 min			60 min			120 min		
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
METR-LA	ARIMA	5.15	10.45	12.70%	6.90	13.23	17.40%	9.32	15.67	20.64%
	LSVR	5.05	10.87	12.10%	6.72	13.76	16.70%	8.94	13.85	18.15%
	FC-LSTM	3.77	7.23	10.90%	4.37	8.69	13.20%	6.53	10.61	14.45%
	DCRNN	3.15	6.45	8.80%	3.60	7.60	10.50%	6.34	10.28	14.34%
	STGCN	3.47	7.24	9.57%	4.59	9.40	12.70%	6.82	11.21	15.26%
	ASTGCN	3.31	6.98	9.32%	4.52	9.24	12.62%	6.12	9.76	13.65%
	Graph WaveNet	3.07	6.22	8.37%	3.53	7.37	10.01%	5.82	8.63	13.07%
	GMAN	3.15	6.78	9.02%	4.03	8.11	11.73%	5.42	8.64	12.92%
	PGCN	3.08	6.22	8.38%	3.54	7.36	9.94%	5.69	8.55	13.09%
	PM-MemNet	3.03	6.29	8.42%	3.46	7.29	9.97%	5.38	8.28	12.83%
	MTGNN	3.05	6.17	8.19%	3.49	7.23	9.87%	6.25	9.83	13.67%
	MegaCRN	2.98	6.18	7.96%	3.38	7.23	9.72%	5.18	8.19	12.33%
	D2HTN	**2.93**	**6.06**	**7.53%**	**3.10**	**7.03**	**9.43%**	**5.06**	**8.08**	**11.75%**
	improvement	*0.05*	*0.11*	*0.43%*	*0.28*	*0.20*	*0.29%*	*0.12*	*0.11*	*0.58%*
PEMS-BAY	ARIMA	2.33	4.76	5.40%	3.38	6.50	8.30%	5.13	8.04	11.87%
	LSVR	2.48	5.18	5.50%	3.28	7.08	8.12%	4.97	7.76	11.45%
	FC-LSTM	2.20	4.55	5.20%	2.37	4.96	5.70%	4.63	7.12	10.38%
	DCRNN	1.74	3.97	3.90%	2.07	4.74	4.90%	4.52	7.15	10.08%
	STGCN	1.81	4.27	4.17%	2.49	5.69	5.79%	4.72	7.31	10.62%
	ASTGCN	1.75	3.98	3.95%	2.32	5.41	5.51%	4.42	7.02	9.76%
	Graph WaveNet	1.63	3.70	3.67%	1.95	4.52	4.63%	4.27	6.95	9.05%
	GMAN	1.62	3.72	3.63%	1.86	4.32	4.31%	3.89	6.61	8.89%
	PGCN	1.62	3.67	3.63%	1.92	4.45	4.55%	3.84	6.37	8.72%
	PM-MemNet	1.65	3.76	3.71%	1.95	4.47	4.54%	3.85	6.54	8.79%
	MTGNN	1.65	3.74	3.69%	1.94	4.49	4.53%	3.90	6.83	8.95%
	MegaCRN	1.60	3.68	3.57%	1.88	4.39	4.41%	3.81	6.41	8.62%
	D2HTN	**1.56**	**3.63**	**3.46%**	**1.67**	**3.86**	**3.85%**	**3.70**	**6. 18**	**8.43%**
	improvement	*0.04*	*0.05*	*0.11%*	*0.21*	*0.23*	*0.36%*	*0.11*	*0.19*	*0.19%*



(a) MAE on METR-LA (b) RMSE on METR-LA (c) MAPE on METR-LA



(d) MAE on PEMS-BAY (e) RMSE on PEMS-BAY (f) MAPE on PEMS-BAY

Fig. 3. Performance of different horizons on PEMS-BAY and METR-LA.

5.2 Forecasting Results and Visualization

The performance of D2HTN and ten baseline methods on the METR-LA and PEMS-BAY datasets at 30 min (6 timesteps), 60 min (12 timesteps), and 120 min (24 timesteps) are presented in Table 2. Our observations are as follows:

(1) Our model and other deep learning-based models that consider graphical structures generally outperform traditional time series methods such as ARIMA, LSVR, and FC-LSTM. This indicates the robust capabilities of deep learning models, as well as the effectiveness of considering spatiotemporal correlations in traffic prediction.
(2) The graph-based models WaveNet and our model outperform the graph models DCRNN and STGCN, highlighting that a more precise and realistic graph construction method can capture more genuine spatial dependencies, thereby enhancing model performance.
(3) GMAN and our model also outperform traditional graph-based deep learning models, indicating the significance of capturing dynamic spatiotemporal correlations.
(4) D2HTN achieves nearly the best results among the existing state-of-the-art models, demonstrating that direct spatiotemporal modeling can significantly enhance the accuracy of traffic prediction.

In Fig. 3, we display the performance of several models on the PEMS-BAY and METR-LA datasets at different horizons. The results indicate that the predictive performance of D2HTN is superior to that of the other models, with

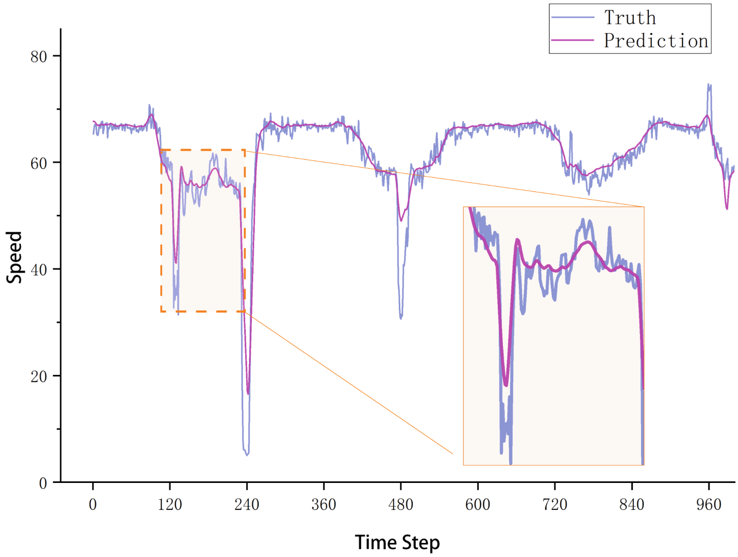


Fig. 4. Forecasting visualization of D2HTN during 1000 Time Step on PEMS-BAY.

all models experiencing a decline in predictive performance as the forecasting horizon increases. Although the performance of our proposed model also shows a decreasing, it remains superior to the other models. We visualized a portion of the predictions from the PEMS-BAY dataset at a forecasting interval of 12 steps. Figure 4 shows the predictions for a specific traffic node over 1000 intervals, with a focus on amplifying the complex and fluctuating data. Our model demonstrates strong predictive performance in periods of low volatility, and accurately captures the trends in traffic data even during challenging intervals, responding quickly and accurately to significant fluctuations.

5.3 Ablation Study

To examine the effectiveness of the components in the validation model, disintegration experiments were conducted on the PEMS-BAY dataset. The variants were named as follows:

- RemE: removal of spatiotemporal node embedding
- RemGSAL: removal of the GSAL module
- RemM-STGU: traditional convolution used to replace the M-STGU module.
- RemEMAG: removal of EMAG as the graph structure in the STA-block in the model

Table 2 presents the results of the disintegration experiments. We observed that D2HTN outperforms its variants in three criteria, thus affirming the effectiveness of each component in our model.

Table 2. Ablation analysis on the PEMS-BAY dataset (12 steps).

Method	MAE	RMSE	MAPE
RemE	1.86	4.35	4.32%
RemGSAL	1.82	4.27	4.21%
RemM-STGU	1.75	4.05	4.03%
RemEMAG	1.70	3.95	3.91%
D2HTN	1.65	3.86	3.85%

6 Conclusion

In this paper, we propose a novel deep learning framework, D2HTN, for traffic prediction, which differs from existing methods by modeling spatial and temporal dependencies jointly through cross-temporal-spatial modeling. Our approach incorporates the EM distance derived from historical traffic data to produce the EMAG, replacing the static adjacency matrix, thus reducing reliance on prior knowledge and enhancing the representation of dynamic inter-node relationships within the road network. Moreover, we integrate the Multi-scale Spatial Temporal Gating Unit (M-STGU) to further enhance the information mining of time series data. Comprehensive empirical studies on two traffic datasets demonstrate that our proposed method outperforms state-of-the-art baselines. Moving forward, we aim to incorporate external factors such as weather and unexpected traffic incidents into our research to improve prediction results. Additionally, we plan to integrate more efficient transformer variants into the model to enhance model efficiency.

References

1. Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: a video vision transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6836–6846 (2021)
2. Cao, S., Wu, L., Wu, J., Wu, D., Li, Q.: A spatio-temporal sequence-to-sequence network for traffic flow prediction. Inf. Sci. **610**, 185–203 (2022)
3. Cao, S., Wu, L., Zhang, R., Li, J., Wu, D.: Capturing local and global spatial-temporal correlations of spatial-temporal graph data for traffic flow prediction. In: International Joint Conference on Neural Networks, IJCNN 2022, Padua, Italy, 18-23 July 2022. pp. 1–8. IEEE (2022)
4. Cho, K., et al.: Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078 (2014)
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
6. Dwivedi, V.P., Bresson, X.: A generalization of transformer networks to graphs. arXiv preprint arXiv:2012.09699 (2020)

7. Jiang, J., Han, C., Zhao, W.X., Wang, J.: Pdformer: Propagation delay-aware dynamic long-range transformer for traffic flow prediction. arXiv preprint arXiv:2301.07945 (2023)
8. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
9. Lan, S., Ma, Y., Huang, W., Wang, W., Yang, H., Li, P.: Dstagnn: dynamic spatial-temporal aware graph neural network for traffic flow forecasting. In: International conference on machine learning. pp. 11906–11917. PMLR (2022)
10. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015)
11. Li, F., Feng, J., Yan, H., Jin, G., Jin, D., Li, Y.: Dynamic graph convolutional recurrent network for traffic prediction: Benchmark and solution. arXiv preprint arXiv:2104.14917 (2021)
12. Li, Y., Yu, R., Shahabi, C., Liu, Y.: Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. arXiv preprint arXiv:1707.01926 (2017)
13. Liu, S., et al.: Pyraformer: low-complexity pyramidal attention for long-range time series modeling and forecasting. In: International Conference on Learning Representations (2021)
14. Liu, Z., et al.: Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 10012–10022 (2021)
15. Liu, Z., et al.: Video swin transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision And Pattern Recognition, pp. 3202–3211 (2022)
16. Lv, Y., Duan, Y., Kang, W., Li, Z., Wang, F.Y.: Traffic flow prediction with big data: a deep learning approach. IEEE Trans. Intell. Transp. Syst. **16**(2), 865–873 (2014)
17. Ma, X., Dai, Z., He, Z., Ma, J., Wang, Y., Wang, Y.: Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction. Sensors **17**(4), 818 (2017)
18. Ma, X., Tao, Z., Wang, Y., Yu, H., Wang, Y.: Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Trans. Res. Part C: Emerging Technol. **54**, 187–197 (2015)
19. Song, C., Lin, Y., Guo, S., Wan, H.: Spatial-temporal synchronous graph convolutional networks: a new framework for spatial-temporal network data forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 914–921 (2020)
20. Wang, X., et al.: Traffic flow prediction via spatial temporal graph neural network. In: Huang, Y., King, I., Liu, T., van Steen, M. (eds.) WWW 2020: The Web Conference 2020, Taipei, Taiwan, 20-24 April 2020, pp. 1082–1092. ACM/IW3C2 (2020)
21. Zhang, Q., Chang, J., Meng, G., Xiang, S., Pan, C.: Spatio-temporal graph structure learning for traffic forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 1177–1185 (2020)
22. Zheng, C., Fan, X., Wang, C., Qi, J.: Gman: a graph multi-attention network for traffic prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 1234–1241 (2020)
23. Zhou, H., et al.: 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)

Automatic Diagnosis Model of Gastrointestinal Diseases Based on Tongue Images

Baochen Fu[1,2], Miao Duan[3], Zhen Li[3], Xiuli Zuo[3], and Xu Qiao[2,4](✉)

[1] School of Software, Shandong University, Jinan 250101, Shandong, China
[2] School of Control Science and Engineering, Shandong University,
Jinan 250061, Shandong, China
qiaoxu@sdu.edu.cn
[3] Department of Gastroenterology, Qilu Hospital, Shandong University,
Jinan 250012, Shandong, China
[4] Shandong National Center for Applied Mathematics, Jinan 250100, Shandong, China

Abstract. Gastrointestinal diseases have become a significant global health issue, with early diagnosis and timely intervention being crucial for improving human health. Although endoscopy is considered the primary examination tool, it is time-consuming and invasive, limiting its widespread implementation. According to Traditional Chinese Medicine (TCM) theories, the tongue reflects the body's physiological and pathological conditions. Therefore, tongue images can be used to develop a non-invasive and rapid diagnostic model for gastrointestinal diseases. We collected tongue images from 1123 patients using a smartphone and categorized them into normal and abnormal groups based on their diagnostic reports. Comparison experiments were conducted using four models: ResNet-50, RepVGG,Deit-B and CrossViT. To address dataset imbalances, we introduced a hybrid loss function to enhance the models' learning for underrepresented classes. The experimental results indicate that Vision Transformer demonstrates superior performance in early diagnosis of gastrointestinal diseases. Specifically, the dual-branch cross-attention model with a hybrid loss function achieved an AUC of 0.87 and an accuracy of 0.84.

Keywords: Gastrointestinal disease · Cross attention · Hybrid loss function

1 Introduction

Gastrointestinal diseases, such as gastritis, peptic ulcer disease, and gastroesophageal reflux disease, are highly prevalent worldwide. In the United States alone, the annual medical expenditure on gastrointestinal disorders amounts to 135.9 billion U.S. dollars, with over 3 million hospitalizations and numerous outpatient visits for individuals initially diagnosed as having gastrointestinal diseases [1]. These diseases not only contribute to increased healthcare costs globally but also have a significant impact on an individual's quality of life [2]. Currently, medical professionals rely on endoscopes to directly examine the lesion area by inserting them into the body through natural cavities. However, endoscopy does have limitations. Many patients with gastrointestinal

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 290–301, 2024.
https://doi.org/10.1007/978-3-031-78128-5_19

disorders may not exhibit noticeable symptoms in the early stages, and this invasive procedure carries the risk of mucosal damage, bleeding, and further infection, which can limit its effectiveness in the early diagnosis and screening of gastrointestinal diseases. Thus, there is an urgent demand for a rapid, non-invasive screening tool.

Tongue diagnosis is essential in traditional Chinese medicine (TCM) as a diagnostic technique. Traditional Chinese medicine believes that various organs in the human body are related to the tongue through the meridians [3], with the closest association between the tongue image and the spleen and stomach [4]. The human body's physiological processes and pathological changes are reflected in changes to the tongue's appearance, which can occasionally occur before other symptoms [5]. Although tongue diagnosis is valued for its non-invasive, contact-free, and non-traumatic characteristics, its clinical application is limited by its inherent subjectivity, relying heavily on practitioners' experience and expertise, making it difficult to be used as a large-scale screening tool [6].

With the advancement of big data and computing technology, artificial intelligence is becoming increasingly prevalent in the medical field [7]. Integrating tongue images with deep learning methods may be the most cost-effective, non-invasive, and socially acceptable approach to early screening and diagnosis. Gholami et al. [8] used a combination of deep neural networks, support vector machines, and deep Convolutional Neural Network (CNN) to improve the accuracy of gastric cancer diagnosis based on tongue surface and color features. Zhu et al. [9] applied the Deeplabv3 + network to tongue segmentation and extracted nine tongue features, used statistical methods and deep learning to explore the relationship between tongue features and gastric cancer, and finally designed a gastric cancer prediction framework. Experimental findings suggest that this framework significantly enhances the early and accurate diagnosis of gastric cancer patients. Li et al. [10] designed three different deep learning models APINet, TransFG, and DeepLabV3+ to reduce the conclusion deviation caused by model differences. The results showed that all three models proved the effectiveness of tongue image discrimination, indicating that tongue images can be as a stable tool for gastric cancer diagnosis. Wu et al. [11] used statistical methods to analyze the relationship between tongue characteristics and gastroesophageal reflux disease (GERD), and used the automatic tongue diagnosis system (ATDS) to extract and analyze the characteristics of the tongue, including tongue shape, color, tooth marks, Tongue coating thickness, etc. Research has demonstrated that tongue coating thickness and saliva volume in the spleen and stomach area can reflect gastroesophageal reflux disease, and TCM tongue diagnosis can provide a non-invasive preliminary diagnosis of GERD.

In this research, we collected tongue pictures of patients with gastrointestinal diseases and healthy people, divided them into normal and abnormal groups, and marked the tongue area for each original picture. We used two advanced deep learning architectures to build the gastrointestinal disease screening model: convolutional neural network (CNN) and visual transformer (ViT). In addition, we propose a hybrid loss function to ensure that the model performs well across categories. The results show that our model can effectively screen for gastrointestinal diseases, providing strong support for clinical practice and patient health.

2 Method

2.1 Data Collection

From September 2021 to June 2022, we collected data on patient information and their tongue images at the Department of Gastroenterology, Qilu Hospital, Shandong University. All participants had clinical and image information. A retrospective study was conducted in alignment with the Declaration of Helsinki (2013 revision). The Ethics Committee of Qilu Hospital, Jinan, Shandong Province, China (Approval No. 2022009) approved the study and waived the need for individual consent for this retrospective analysis.

In this study, we collected 3,311 tongue images from 1,123 patients. Participants were instructed by researchers to capture images by swallowing, opening their mouths as widely as possible, protruding their tongues, and keeping them relaxed and flat. All tongue images were captured using the same mobile phone with a resolution of 1000 × 1333 pixels and saved in PNG format. We conducted rigorous quality screening to exclude foggy, underexposed, overexposed, stained tongue coatings, and abnormal tongue shapes, resulting in 3,020 images meeting our criteria. The images of tongues displayed are shown in Fig. 1.

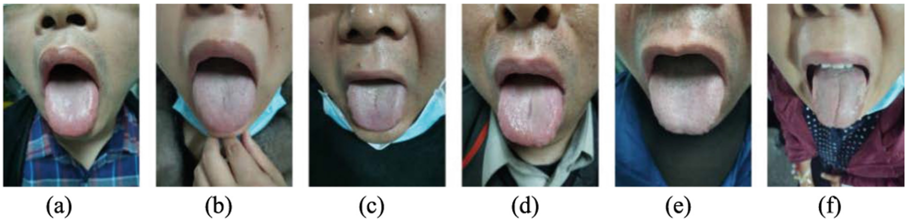


Fig. 1. Tongue image dataset display diagram: (a) Health (b) Helicobacter pylori infection (c) Bile reflux (d) Reflux esophagitis (e) Gastric erosion (f) Duodenal erosion

2.2 Data Preprocessing and Splitting

The original image contains a large amount of background that is unrelated to the tongue, making it necessary to segment the tongue area. In this study, the tongue is completely segmented by manual segmentation, and then the largest circumscribing rectangle of the tongue contour is calculated, the redundant black background is removed, and the size is uniformly scaled to 224 × 224, as shown in Fig. 2. All patients were divided into normal group and abnormal group, in which the abnormal group included Helicobacter pylori infection, bile reflux, reflux esophagitis, gastric erosion and duodenal erosion, and the images of the two groups were compared according to 7:3 Split, so that multiple images of the same patient will not enter the training set and test set at the same time. The Table1 shows how the datasets were divided.

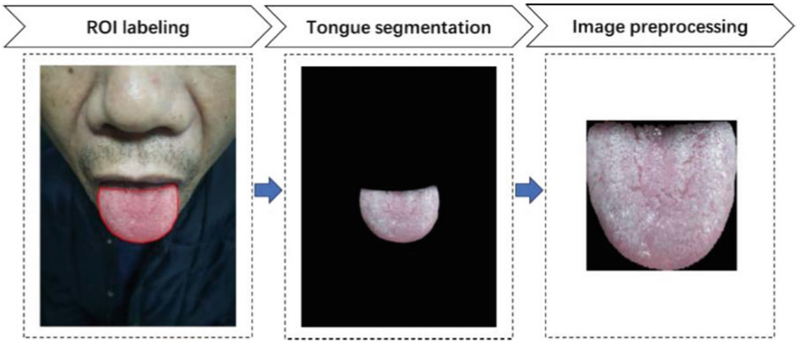


Fig. 2. Image preprocessing flow chart.

Table 1. The number of training and testing sets of samples.

Category	Training Set	Testing Set	Total
Normal	617	257	874
Abnormal	1497	649	2146

2.3 Neural Network Architecture

In the field of computer vision, Convolutional Neural Network and Vision Transformer have become the mainstream Backbone network. Due to the self-attention mechanism, ViT shows excellent generalization ability on large-scale data sets [10]. In this study, we chose the two-branch cross-attention ViT model (CrossViT) (Fig. 3) for the examination of gastrointestinal diseases and compared with ResNet 50 [12], RepVGG [13], Deit-B [14]. CrossViT [15] is a Transformer architecture designed for image classification tasks. Its key innovation lies in its two-branch structure and cross-attention mechanism, which integrates image features at multiple scales to capture details at different scales in the image, thereby improving the accuracy of classification. Each branch of CrossViT processes image patches of different sizes. Patch size is generally thought to affect the accuracy and complexity of ViT, and finer grained patch sizes lead to better ViT performance but higher FLOPs and memory consumption. The model introduces two branches, Small-Branch and Large-Branch, with patch sizes of 12 and 16 respectively. The two branches are fused together through the Cross-Attention module. The Class token of the last two branches are used for diagnosis. Figure 4 explains the attention fusion of Large-Branch. It first collects patch tokens from Small-Branch and connects its own Class token to it. $f(\cdot)$ and $g^l(\cdot)$ are size-aligned projection and back-projection functions respectively. The module executes Cross-Attention (CA) between $\mathbf{x}^l{}_{cls}$ and $\mathbf{x}'^l$. CA performs the following calculation:

$$q = x'^l{}_{cls} W_q,\ k = x'^l W_k,\ v = x'^l W_v \tag{1}$$

$$A = soft\max\left(qk^T / \sqrt{\frac{c}{h}}\right),\ CA\left(X'^l\right) = Av \tag{2}$$

where $W_q, W_k, W_v, \in R^{c\times(c/h)}$ W_q, W_k, $W_v \in R^{C\times(C/h)}$ are learnable parameters, C and h are the embedding dimension and the number of heads respectively, in addition, multiple heads are used and represented in CA For MCA, the final output $\mathbf{y}'^{l}$ is defined as follows:

$$y^{l}_{cls} = f^{1}\left(x^{l}_{cls}\right) + MCA\left(LN\left(\left[f^{1}\left(x^{l}_{cls}\right) \middle\| x^{s}_{patch}\right]\right)\right) \tag{3}$$

$$y^{1} = \left[g^{1}\left(y^{l}_{cls}\right) \middle\| x^{1}_{patch}\right] \tag{4}$$

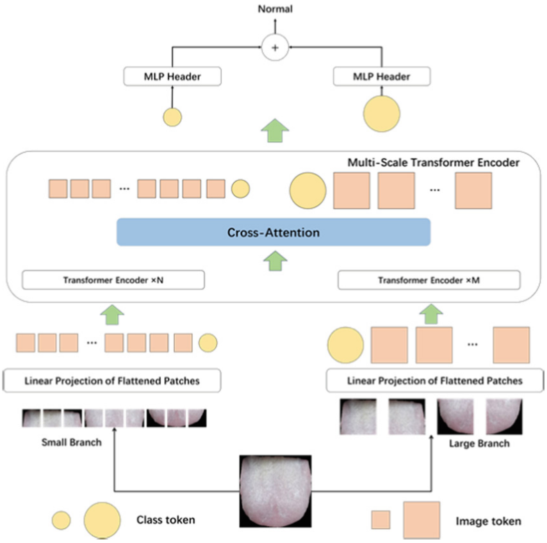


Fig. 3. Transformer architecture for learning multi-scale features with cross-attention.

2.4 Hybrid Loss Function

The tongue image dataset utilized in this study displays a notable category imbalance. This category imbalance will cause the model to be more biased towards the majority category, thereby leading to suboptimal performance for the minority category. To mitigate this concern, we employ the Large Margin Aware Focal (LMF) loss, defined by the following formula:

$$L_{LMF} = \alpha L_{FL} + \beta L_{LDAM} \tag{5}$$

$$L_{FL} = -(1-p_t)^{\gamma}\log(p_t) \tag{6}$$

$$l_{LDAM}((x,y);f) = -\log\frac{e^{zy-\Delta y}}{e^{zy-\Delta y}+\sum_{j\neq y} e^{z}j}, \ \Delta_j = \frac{c}{\frac{1}{n^{\frac{4}{j}}}} \tag{7}$$

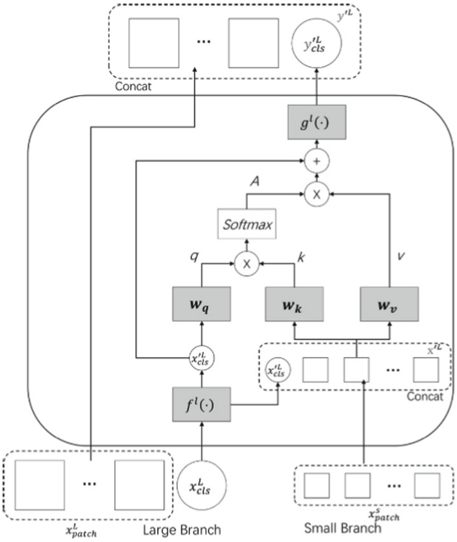


Fig. 4. Cross-attention module for Large branch.

The loss function consists of two parts, where L_{FL} represents Focal Loss [16], as shown in the formula (6), Focal Loss adds a tuning factor $(1 - p_t)^\gamma$ on the basis of cross-entropy loss, and p_t represents the predicted probability of the model, γ is a hyperparameter that can be adjusted. A higher value of γ will reduce the loss of easy samples and make the model focus on difficult samples. Here, we use $\gamma = 1.5$.

L_{LDAM} is (Label Distribution aware Marginal Loss) [17]. Here, j represents a specific class, n_j is the number of samples in this class, and C is a constant, set to 0.5.

α and β are hyperparameters, here set to 1. Treat x as a specific sample, y is the corresponding label of x, z_y represents the model output of the specific sample, and LDAM Loss determines the weight through the category distribution of the data set. A linear combination of the two loss functions yields efficient results.

2.5 Experimental Setting

The Adam optimizer is used in model training, beta1 is set to 0.9, and beta2 is set to 0.99. The input size of the image is 224 × 224. The initial learning rate is set to 5e-5 and is adjusted during training using a cosine annealing function, with a minimum value of 5e-8. All models use pre-trained parameters on the ImageNet dataset with a batch size of 32. Random rotation, horizontal and vertical flipping, and random changes in brightness, contrast, and hue are used as data augmentation techniques. The experiment utilizes the PyTorch framework on Ubuntu 20.04. Model training and testing are performed on an NVIDIA Tesla V100 32GB GPU.

2.6 Evaluation Metrics

The model's performance is evaluated across four distinct metrics.

$$Recall = TP/(TP + FN) \tag{8}$$

$$Precision = TP/(TP + FP) \tag{9}$$

$$F1 - Score = (2 * Precision * Recall)/(Precision + Recall) \tag{10}$$

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \tag{11}$$

TP, FP, TN, and FN represent true positives, false positives, true negatives, and false negatives. Recall was calculated for each category, and the macro average was used to evaluate overall classification performance. Classification models were assessed using the area under the curve (AUC) and receiver operating characteristic (ROC) curves, which plot the true positive rate against the false positive rate. A higher AUC indicates better model performance.

3 Results

3.1 Cross-Validation Evaluation

We selected four models for comparative analysis: ResNet-50, RepVGG, Deit-B, and CrossViT. Each model utilized two distinct loss functions: Cross-Entropy Loss (CE Loss) and Large Margin aware Focal Loss (LMF Loss). Table 2 presents the classification results of these models through five-fold cross-validation using 2,114 training set images. Compared to the CNN model, ViT demonstrates superior performance, especially Deit-B and CrossViT, both achieving an AUC exceeding 0.8. After introducing LMF Loss, the four models showed varying degrees of improvement: ResNet and.

RepVGG increased their recall by 0.06 and 0.07 respectively, while Deit-B and CrossViT improved by 0.04 and 0.02 in recall. Among them, CrossViT performed the best with an AUC of 0.84, an accuracy (ACC) of 0.82, and a recall of 0.76.

Table 2. Cross-validation results of different models.

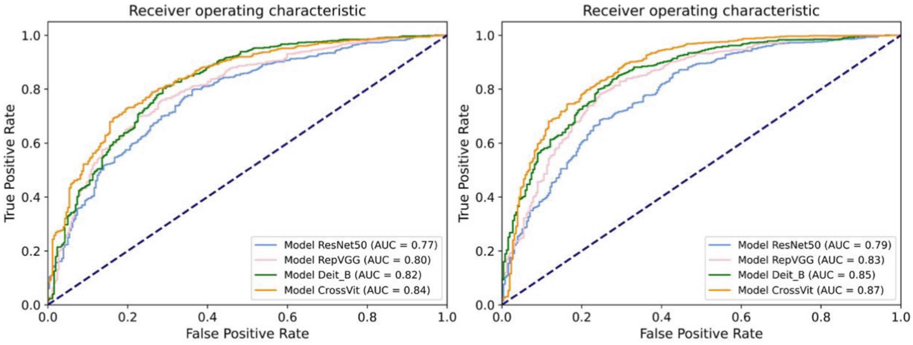
Model	Loss Function	AUC	ACC	Recall	Precision	F1-Score
ResNet 50	CE Loss	0.76	0.72	0.66	0.79	0.81
	LMF Loss	0.78	0.74	0.72	0.86	0.81
RepVGG	CE Loss	0.78	0.75	0.65	0.80	0.83
	LMF Loss	0.80	0.77	0.72	0.84	0.84
Deit_B	CE Loss	0.80	0.78	0.69	0.81	0.85
	LMF Loss	0.81	0.77	0.73	0.81	0.83
CrossViT	CE Loss	0.83	0.80	0.74	**0.86**	0.86
	LMF Loss	0.84	0.82	0.76	0.84	0,87

3.2 Test Set Evaluation

To further evaluate the model's performance and assess the impact of the loss function, we developed four deep learning models using the training set and evaluated them on the test set. The test set comprised 906 tongue images, with a training-to-test ratio of 7:3. Figure 5 illustrates the ROC curves of the various models, and CrossViT achieved a superior AUC compared to the other models. Table 3 presents a quantitative comparison of the different loss functions employed by each model. Results in Table 3 indicate that LMF-Loss effectively enhances model performance. Specifically, the recall for ResNet-50 increased from 0.68 to 0.70, while RepVGG and Deit-B saw increases of 0.07, and CrossViT improved by 0.08. Notably, CrossViT demonstrated the best overall performance with an AUC of 0.87, ACC of 0.84, Recall of 0.78, Precision of 0.86, and F1-Score of 0.89.

Table 3. The results of different models on the test set.

Model	Loss Function	AUC	ACC	Recall	Precision	F1-Score
ResNet 50	CE Loss LMF Loss	0.77 0.79	0.75 0.78	0.68 0.70	0.81 0.82	0.83 0.86
RepVGG	CE Loss LMF Loss	0.80 0.83	0.77 0.80	0.67 0.74	0.80 0.85	0.85 0.87
Deit_B	CE Loss LMF Loss	0.82 0.85	0.81 0.81	0.70 0.77	0.81 **0.87**	0.88 0.87
CrossViT	CE Loss **LMF Loss**	0.84 **0.87**	0.80 **0.84**	0.70 **0.78**	0.81 0.86	0.87 **0.89**



(a) (b)

Fig. 5. ROC curves using different loss functions for four models:(a) Cross entropy loss (b) Hybrid loss.

3.3 Ablation Study

We conducted ablation experiments using CrossViT on the tongue image dataset, including the selection of hyperparameters for the hybrid loss function and the model's patch size. According to Eq. 5, the LMF Loss comprises both Focal Loss and LDAM Loss. Table 4 shows the influence of the Focal Loss hyperparameter γ on the model's classification results. When $\gamma = 0$, the Focal Loss simplifies to Cross-Entropy Loss. By comparing the model's classification results under various parameters, we found that the highest classification accuracy is achieved when $\gamma = 1.5$. Consequently, we set γ to 1.5 in subsequent experiments.

Table 4. Focal loss ablation experiment.

γ	AUC	ACC	Recall	Precision	F1-Score
0	0.85	0.81	0.75	0.85	0,87
0.5	0.84	0.81	0.76	0.86	0.84
1	0.83	0.81	0.74	0.85	0.87
1.5	**0.87**	**0.84**	**0.78**	**0.86**	**0.89**
2	0.84	0.81	0.75	0.85	0.87

Table 5 presents the selection of hyperparameters α and β for LMF Loss. When = 0, only Focal Loss is used as the loss function, and when $\alpha = 0$, only LDAM Loss is employed. The comparative results in the table clearly indicate that a linear combination of these two loss functions enhances the accuracy of the classification model. Table 6 demonstrates the performance of CrossViT using only the large branch and small branch. The patch size significantly impacts the accuracy and complexity of ViT. Using a finer patch size can potentially enhance model performance but will result in higher FLOPs. Although using a finer patch size can potentially enhance model performance, it results in higher FLOPs. For instance, setting the patch size for both branches to 12 increases the FLOPs from 8.2 to 13.4 without improving accuracy. Conversely, setting the patch size to 16 reduces FLOPs from 8.2 to 7.9, with a decrease in accuracy by 0.03. Therefore, setting the patch sizes of the two branches to 12 and 16, respectively, facilitates the model in learning multi-scale image features while balancing accuracy and complexity.

Table 5. LMF loss ablation experiment.

α	β	AUC	ACC	Recall	Precision	F1-Score
1	0	0.83	0.81	0.74	0.84	0.87
1	**1**	**0.87**	**0.84**	**0.78**	**0.86**	**0.89**
0	1	0.82	0.82	0.75	0.85	0.88

Table 6. Ablation experiments with different patch size.

Patch size				
Small	Large	ACC	FLOPs(G)	Params(M)
12	12	0.84	13.4	42.9
12	16	**0.84**	**8.2**	**43.1**
16	16	0.81	7.9	43.2

4 Discussion

In this study, we divided patients with bile reflux, reflux esophagitis, gastric erosion, duodenal erosion, and H. pylori infection into disease groups. Bile reflux was identified by the presence of bile in the stomach during endoscopic examination. Reflux esophagitis was characterized by mucosal erosion of the distal esophagus due to inflammation. Gastric and duodenal erosions were defined as mucosal injuries or ulcerations in their respective regions, with all conditions being diagnosed via endoscopy.H. pylori infection is diagnosed by a rapid urease test, a positive 13C-urea breath test (UBT), or identification of the organism in a biopsy specimen. All patients have detailed test reports. After data cleaning, a total of 3020 tongue images were retained. The performance of four deep learning models was compared: ResNet 50, RepVGG, Deit-B, and CrossViT. We combined two loss functions, Focal Loss and LDAM Loss, to address the issue of data imbalance. Focal Loss assists the model in focusing to difficult-to-classify samples. LDAM Loss introduces stronger regularization to minority categories than majority categories. The combination of these two loss functions produced better results. Finally, CrossViT achieved the best result (AUC = 0.87) using the mixed loss function. Compared with the cross-entropy loss, the Recall of CrossViT increased by 0.08, indicating that the loss function we proposed can effectively balance the number of two categories. Furthermore, the unique dual-branch structure of CrossViT realizes the multi-scale feature fusion of images by dividing the tongue image into patches of different sizes, and then using the Cross-Attention module to perform vector calculations on the class token of the master branch and the patch token of the slave branch, resulting in notable performance advantages.

In recent years, numerous studies on tongue diagnosis have primarily utilized convolutional neural network (CNN) models. Ma et al. [19] developed a screening model for gastric cancer precancerous lesions based on tongue images using ResNet50. By integrating clinical features with tongue characteristics, they achieved an AUC of 0.76. Zhu et al. [9] used the EfficientNet network to classify tongue images of gastric cancer and non-gastric cancer patients, addressing the limitations of traditional image classification algorithms, including single feature scale, dimension scaling saturation, and poor classification performance. he experimental results demonstrated that this method accurately distinguished between gastric cancer patients and non-gastric cancer patients, achieving an accuracy of 93.6%. Wang et al. [20] collected a dataset of 1548 tongue images using specialized medical equipment, which professional Traditional Chinese.

Medicine physicians categorized into tooth-marked tongues and non-tooth-marked tongues. They manually segmented the tongue regions and constructed both the original tongue image dataset and the tongue region image dataset, comparing the performance of ResNet34 and VGG16 on these datasets. The experimental results indicated that, after segmenting the tongue regions, ResNet34 achieved higher performance with an accuracy of 91.47%. Although the above research has made some progress, several issues remain. Firstly, most studies have utilized specialized collection equipment, which is bulky and expensive, making long-term monitoring challenging. Our study, however, eliminates the need for specialized equipment by using smartphones to capture tongue images, thereby enhancing practical applicability. Secondly, previous studies typically focused on classifying a single disease, whereas our research is capable of accurately diagnosing five common gastrointestinal diseases, thereby aiding a larger number of patients in receiving timely treatment. Furthermore, we introduced a hybrid loss function specifically designed to address the issue of data imbalance-LMF Loss, which has not been explored in prior research. LMF Loss incorporates the principle of large margins and strategies that focus on difficult-to-classify samples within deep learning frameworks, making it suitable for scenarios that require handling class imbalance and high discriminative feature demands. Experimental results demonstrated the efficacy of this loss function in enhancing the model's ability to learn from minority class samples. Finally, we conducted comparative experiments using two CNN models and two ViT models. Given the intricate manifestation of gastrointestinal diseases on the tongue, it is essential to extract deeper and more comprehensive tongue features for diagnosis. CrossViT utilizes its self-attention mechanism to establish spatial relationships among different regions of the tongue, extracting image features at multiple scales to capture both fine-grained local features and macroscopic global information. This method has demonstrated superior classification accuracy, validating the efficacy of ViT models in tongue diagnosis.

Although the model achieved commendable classification results, our research still has some limitations. First, we collected less data from a single center and a single race, which may limit the model's generalizability. Additionally, manual segmentation of tongue images is employed, which cannot meet the requirements for automated and intelligent diagnostic systems. In the future, we will focus on collecting more diverse tongue image datasets and implementing automated tongue region segmentation.

5 Conclusion

Based on the principle of tongue diagnosis in traditional Chinese medicine, We developed a deep learning-based diagnostic model for gastrointestinal diseases using tongue images to achieve non-invasive patient detection. The model utilizes tongue images captured by mobile phones for diagnosis, obviating the need for costly and cumbersome specialized equipment, thereby reducing unnecessary costs. Moreover, our model can diagnose multiple common gastrointestinal disorders, not limited to a single disease. Even for class datasets with uneven sample distribution, our proposed method can consistently maintain high accuracy, thereby facilitating timely and accurate diagnosis for more patients, ultimately promoting prompt treatment and enhancing patient outcomes.

Our study demonstrates that when combined with an innovative loss function and multiscale vision model architecture, the accuracy and practicality of the tongue diagnosis model for gastrointestinal diseases can be significantly improved.

References

1. Messner, E.-M., et al.: Mobile apps for the management of gastrointestinal diseases: Systematic search and evaluation within app stores. J. Med. Internet Res. **24**(10) (2022): e37497
2. Sperber, A.D., et al.: Worldwide prevalence and burden of functional gastrointestinal disorders, results of Rome Foundation Global Study. Gastroenterology **160**(1), 99114 (2021)
3. Nestler, G.: Traditional Chinese medicine. Medical Clinics **86**(1), 63–73 (2002)
4. Sun, Z.-M., et al.: Metabolic markers and microecological characteristics of tongue coating in patients with chronic gastritis. BMC Complementary Alternative Med. **13**, 1–10 (2013)
5. Liu, Q., et al.: A survey of artificial intelligence in tongue image for disease diagnosis and syndrome differentiation. Digital Health **9**, 20552076231191044 (2023)
6. Dai, Y., Wang, G.: Analyzing tongue images using a conceptual alignment deep autoencoder. IEEE Access **6**, 5962–5972 (2018)
7. Yang, G., et al.: Predicting stroke-associated pneumonia from intracerebral hemorrhage based on deep learning methods. In: International KES Conference on Innovation in Medicine and Healthcare. Springer Nature Singapore, Singapore (2023)
8. Gholami, E., Tabbakh, S.R.K.: Increasing the accuracy in the diagnosis of stomach cancer based on color and lint features of tongue. Biomed. Signal Process. Control **69**, 102782 (2021)
9. Zhu, X., et al.: A framework to predict gastric cancer based on tongue features and deep learning. Micromachines **14**(1), 53 (2022)
10. Yuan, L., et al.:Development of a tongue image-based machine learning tool for the diagnosis of gastric cancer: a prospective multicentre clinical cohort study. EClinical Med. **57** (2023)
11. Wu, T.-C., et al.: Tongue diagnosis indices for gastroesophageal reflux disease: a crosssectional, case-controlled observational study. Medicine **99**(29), e20471 (2020)
12. He, K., et al: .Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016)
13. Ding, X., et al.: Repvgg: Making vgg-style convnets great again. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2021)
14. Touvron, H., et al.: Training data-efficient image transformers & distillation through attention. In: International Conference on Machine Learning. PMLR (2021)
15. Chen, C.-F.R., Fan, Q., Panda, R.: Crossvit: cross-attention multi-scale vision transformer for image classification. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2021)
16. Lin, T.-Y., et al.: Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision (2017)
17. Cao, K., et al.: Learning imbalanced datasets with label-distribution-aware margin loss. Adv. Neural Inform. Process. Syst. **32** (2019)
18. Fu, B., et al:. Deep learning system for automatic diagnosis of ocular diseases in infants. In: International KES Conference on Innovation in Medicine and Healthcare. Springer Nature Singapore, Singapore (2023)
19. Ma, C., et al.: Construction of tongue image-based machine learning model for screening patients with gastric precancerous lesions. J. Personalized Med. **13**(2), 271 (2023)
20. Wang, X., et al.: Artificial intelligence in tongue diagnosis: Using deep convolutional neural network for recognizing unhealthy tongue with tooth-mark. Comput. Struct. Biotechnol. J. **18**, 973–980 (2020)

TinyConv-PVT: A Deeper Fusion Model of CNN and Transformer for Tiny Dataset

Yi Zhang[1], Bowei Zhang[1], and Kai Yi[2](✉)

[1] Department of Computer Science, Sichuan University, Chengdu, China
yi.zhang@scu.edu.cn
[2] Sichuan Police College, Intelligent Policing Key Laboratory of Sichuan Province, Luzhou, China
yikai@scpolicec.edu.cn

Abstract. Transformer has taken computer vision field by storm in recent years and is becoming increasingly popular in both academia and industry. However, the remarkable success is largely fueled by training on massive samples. In real applications, it is not always possible to have sufficient annotated data. When only a small set of labeled data is available (called tiny dataset), Transformer performs far worse than convolution neural network (CNN). Moreover, it occupies large memory footprint during training. Under such circumstances, a deep fusion model has been created by integrating CNN with Transformer architecture. Specifically, Conv-stem is implanted in the first stage of Transformer to reduce memory footprint. Then, the second and the third stages of the Transformer encoder have been modified into parallel structure with integration of CNN. Finally, depth-wise separable convolution blocks are appended to the encoder to enhance feature representation. Extensive experiments prove the effectiveness of our proposed model, in which it outperforms other popular methods on tiny datasets by sage margins.

Keywords: Vision Transformer · Convolution Neural Networks (CNN) · Downstream Vision tasks

1 Introduction

Early in the field of computer vision, many classical algorithms were developed to recognize features. Those methods were indeed efficient but were subject to many limitations. For instance, the primitive algorithms can only recognize simple features (e.g. lines, circles), but cannot reason complex patterns. Later, with the rapid development of deep learning techniques, computer vision has come to a new era. The performances of some advanced models have even surpassed human (e.g. object tracking, face recognition, behavior analysis etc.), which have already replaced human labor in some areas (e.g. intelligent surveillance, autonomous driving).

Currently, convolutional neural network (CNN) dominates computer vision domain. Such success is largely due to the powerful inductive bias of spatial

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 302–318, 2025.
https://doi.org/10.1007/978-3-031-78128-5_20

equivariance encoded by convolutional layers, which has been the key to strong visual representations. However, it only aggregates local features instead of modelling global contextual information. Recent work has shown that Transformer achieves equal or even superior performance than CNN on large scale image classification tasks. It relies on self-attention (rather than convolution) to establish long range dependencies across locations, which has aroused great interests of scholars. However, Transformer has obvious shortcomings: firstly, the classical Transformer cannot be used as a backbone to implement feature extraction. Secondly, the spatial-temporal complexity of Transformer is proportional to the number of input tokens, which results in high computational cost and undoubtedly impedes its application in high-resolution scenarios. And thirdly, Transformer lacks inductive bias, it hence only outshines CNN on large scale datasets.

To better illustrate this problem, we train ResNet-50 and Pyramid Vision Transformer (PVT) on a tiny dataset (CIFAR-100 with only 50k training images) using simple data augmentation techniques (padding and random cropping). As shown in Fig. 1, PVT converges quickly on training set, but it only reaches less than 60% (of accuracy) on the testing set, suffering overfitting. In comparison, CNN converges slowly on training set but achieves higher than 70% (of accuracy) on the testing set. Usually, the best way to remedy overfitting is to train the model on large scale datasets. However, in reality, the ideal large-scale dataset is not always available (e.g. medical imaging). Moreover, those models that perform well on large scale datasets are expected to have the same performances on tiny dataset is quite demanding [1]. Besides, the training time and inference speed are also important factors, to train the existing popular ViT based models on large datasets are extremely time consuming, while lightweight models have clear advantage. At last, for some real-time applications, the models have to be implemented on mobile devices, while the heavy ViT models (e.g. ViT-L) could not.

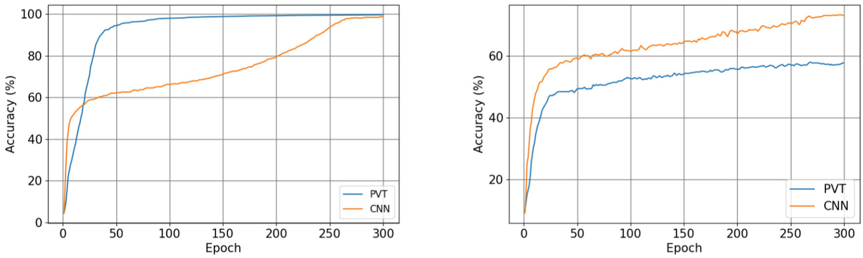


Fig. 1. Comparison of convergence speeds on training (left) and testing sets (right) on CIFAR-100 between PVT and CNN

To delve deeper into this issue, we replace the first layer of PVT with CNN and compare their attention distances after 100 and 200 epochs training (by following the instruction of [2]). The attention distance is the weighted sum of the distance between any 2 tokens and their corresponding attention weights. As shown in Fig. 2, the attention distances of the attention head in the first 2 layers are smaller than those in the deeper layers, which means more attention is paid to the local information in the shallower layers, while the attention is paid to global information in the deeper layers. After 100 epochs training, the average attention distances of block_0 and block_1 (blue and orange lines in (a) and (b)) do not have significant changes. While block_2 and block_3 (green and red lines in (a) and (b)) have slight decrease, and those of the deeper layers (brown and purple lines) have almost no changes. This phenomenon indicates that the model would adjust the attention weights during training in a continuous way. This adjustment is more evident in the shallower layers due to the introduction of CNN, and it gradually weakens in the middle layers, since Transformer lacks the inductive bias so that the model needs more training samples to converge.

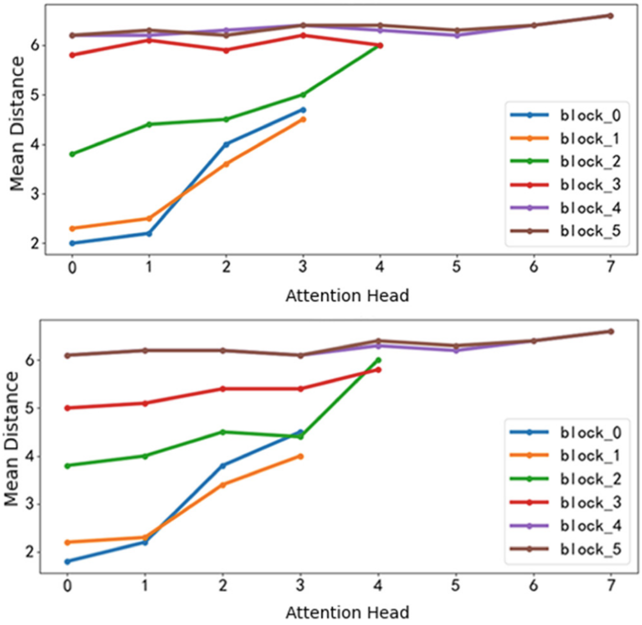


Fig. 2. The mean distances of Conv-PVT after 100 (upper) and 200 (lower) epochs training

In order to improve the general performance of Transformer and to accelerate the convergence speed on tiny datasets, we make the following 3 contributions in this paper:

1. Conv-stem is implanted in the first stage of Transformer to reduce the memory footprint and accelerate the training process.

2. The second and the third stages of the original Transformer encoder have been modified into a double branch parallel structure, while CNN is deployed on one branch to compensate the lack of inductive bias in Transformer, which also reduces the dependency of Transformer on massive training samples.
3. Depth-wise separable convolution blocks have been appended to the second and the third stages of Transformer to enhance the feature representation abilities.

2 Related Works

Transformer was initially proposed to solve machine translation, parallel data processing and other cross-modal tasks [[3,4]]. It has strong global context modelling ability due to its serialized input structure and self-attention mechanism. Meanwhile, Transformer also breaks the dominance of recurrent neural network (RNN) and CNN. Vision Transformer (ViT) [5] is a milestone, which splits input images into image patches to perform various computer vision tasks and paves the way for subsequent works. Although some improved models have achieved appealing performances on ImageNet-1K , they cannot compete with CNN based models on tiny datasets (e.g. CIFAR-100) due to the lack of inductive bias, and therefore they need large training samples to acquire preferable visual representation. Some subsequent works strengthen the local attention of the network through deploying CNN into neighboring embeddings. Among them, CCT [6] introduces compact Transformer and replaces class token with sequence pooling for small-scale learning. SL-ViT [7] presents Shifted Patch Tokenization and Locality Self-Attention which solves the lack of locality inductive bias for tiny datasets. EdgeViT [8] advised a highly cost effective information exchange scheme by integrating self-attention and convolutions. Recently, the improvement of Transformer has mainly focused on the following 3 aspects:

The Improvement of Token: The improvement of token in computer vision domain can be roughly divided into 2 directions. Wang et al. [9] proposes a token mixer free model architecture and explores the improved learning paradigm to break the limitation of simple token mixer free backbone. Ryoo et al. [10] learns to mine important tokens in visual data and models the pairwise attention between such tokens, which limits the number of tokens within 8 16 so as to reduce the computation cost. Swin-Transformer [11] is a hierarchical architecture with shifted windows, which brings greater efficiency by limiting the computation within non-overlapping local windows. Another direction centers around tokenization methods. Most of them are still based on ViT, which split input image into patches to yield tokens through linear projection. T2T-ViT [12] develops a layer-wise Tokens-to-Tokens (T2T) transformer to progressively structurize the image to tokens by recursively aggregating surrounding tokens into one.

Positional Encoding (PE): PE aims to add the positional information of each token into Transformer in NLP, which can be divided into fixed PE and learned PE. The former is based on the phase change of sine and cosine functions. The latter learns positional embeddings through a set of learnable parameters. However, it may fail when dealing with high-resolution input images. CVT [13] advises hierarchical structures containing a new convolutional token embedding and convolutional Transformer block leveraging a convolutional projection.

Introduction of CNN: CNN and ViT use different ways to enforce geometric structure priors. Inductive bias is defined as a set of hypothesis about data distribution and solution space, which is represented as locality and translation invariance in CNN. However, ViT utilizes the global self-attention to learn the relationships between global image features. A combination of the advantages of CNN (in processing local visual clues) and ViT (in processing global relationship) is potentially a better way for image modelling. DeiT [14] introduces a teacher-student strategy to Transformer, which relies on a distillation token to ensure that the student learns from the teacher through attention. ConViT [15] devises a form of positional self-attention soft convolutional inductive bias (called gated positional self-attention) to mimic the locality of CNN. BotNet [16] replaces the spatial convolutions with global attention in the last 2 stages of ResNet, which achieves surprisingly good results on various vision tasks. CMT [17] puts forth a new Transformer based hybrid network by taking advantages of both Transformers and CNN to capture long-range dependencies and model local features respectively. A Conv-MSA structure is proposed by FlexFormer [18] to extract local contextual information with linear computational complexity on input image size. SWFormer [19] is a scalable and accurate model for 3D object detection which is built upon sparsity of point clouds and converts 3D points into sparse voxels and windows. MaskGIT [20] uses a bidirectional Transformer decoder to predict randomly masked tokens by attending to tokens in all directions. Wang et al. [21] designs a matching operator, combining a multi-head attention with a modulation module across the inputs. MobileViTc [22] learns global representations via incorporating convolution blocks. CMT [17] captures both local and global structural information via local perception unit and inverted residual feed-forward network.

Vision Transformer on Tiny Datasets: Most existing works focus on middle sized datasets (e.g. ImageNet), while the performances on tiny datasets are still less satisfactory. One possible solution is to migrate the pre-trained models to tiny datasets. Liu et al. [29] introduces self-supervised training strategy and loss function to train ViT on small datasets. SL-ViT [30] is a cascaded model that utilizes hierarchical patch partition to recover the high resolution image in a progressive way. Normally, for the same model the difference in accuracy on CIFAR-100 is greater than that on ImageNet, the reason is that the fusion of convolution blocks in the model is more conducive to improving the results on tiny datasets (compared with updating the attention mechanism or adding new feature extraction blocks).

Although the above mentioned based methods narrow the gap between ViT and standard CNN based models on large scale datasets, they still cannot compete with CNN on tiny datasets due to their lower capability in extracting fine-grained local feature patterns. Considering this, we integrate CNN into the attention module of Transformer to enhance its ability in extracting local features so as to improve the performance of ViT on tiny datasets.

3 Proposed Method

In this work, our goal is to construct an efficient vision Transformer architecture by taking the advantages of both CNNs and attention mechanisms. In the original ViT, the straightforward tokenization of input images leads to ignorance of local structures (e.g. lines and edges), which therefore needs multiple times more training samples than CNN to reach comparable results. The fusion of CNN in the model effectively alleviates such deficiency.

3.1 Overall Architecture

We divide the model into 4 stages according to the design of CNN in order to generate feature maps of different scales. Between each stage, we down-sample the image by 2 through the Patch Embedding layer (that consists of convolution and normalization layers). To implement down-sampling operation, we introduce Conv-stem to extract fine-grained local features. Besides, in each stage we stack different number of Transformer blocks, each block has the same structure, which consists of attention blocks and DSCFFL layers. Unlike, ViT, we deploy parallel branches in stage 2 and 3 with CNN on one of the branch, which is proven to be effective for tiny datasets. At the end of the last stage, the feature map is further processed via a global average pooling layer and a classification layer with softmax to output different classification results. The overall architecture of our proposed network and our proposed Transformer encoder are drawn in Fig. 3 and Fig. 4 respectively.

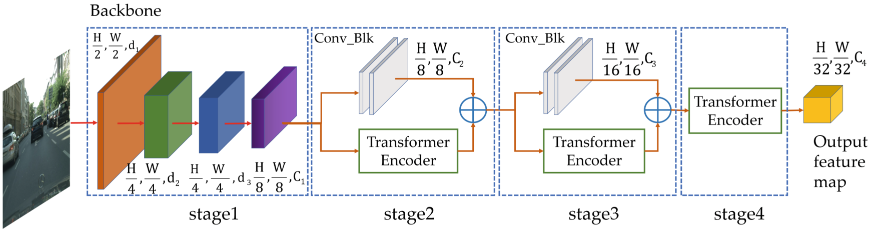

Backbone
$\frac{H}{2},\frac{W}{2},d_1$
$\frac{H}{4},\frac{W}{4},d_2$
$\frac{H}{4},\frac{W}{4},d_3$
$\frac{H}{8},\frac{W}{8},C_1$
Conv_Blk
$\frac{H}{8},\frac{W}{8},C_2$
Transformer Encoder
Conv_Blk
$\frac{H}{16},\frac{W}{16},C_3$
Transformer Encoder
Transformer Encoder
$\frac{H}{32},\frac{W}{32},C_4$
Output feature map
stage1
stage2
stage3
stage4

Fig. 3. The overall architecture of TinyConv-PVT

For a certain input image, we obtain 4 layered feature maps (i.e.multi-resolution representation) with different resolutions through consecutive down-sampling operation, which allows us to easily solve downstream tasks such as object detection and semantic segmentation.

In general, we have made the following two majors changes to the original Transformer architecture:

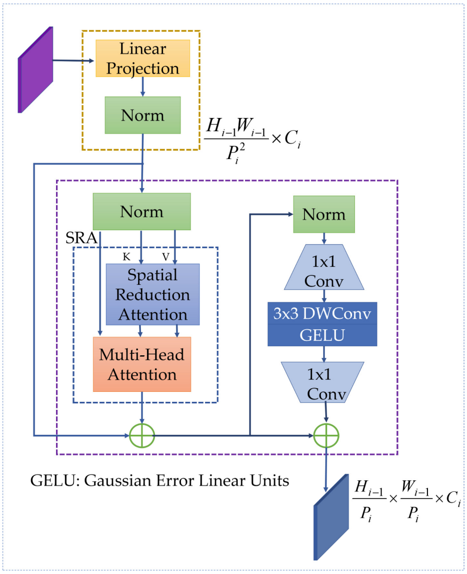

Linear Projection
Norm
$\frac{H_{i-1}W_{i-1}}{P_i^2} \times C_i$
Norm
SRA
K
V
Spatial Reduction Attention
Multi-Head Attention
Norm
1x1 Conv
3x3 DWConv
GELU
1x1 Conv
GELU: Gaussian Error Linear Units
$\frac{H_{i-1}}{P_i} \times \frac{W_{i-1}}{P_i} \times C_i$

Fig. 4. The structure of our proposed Transformer Encoder

1. Firstly, we replace the first stage of Transformer with 4 layers of CNN to downsample the input images by a factor of 8, which accelerates the convergence speed and also reduces the memory footprint.
2. Secondly, we fuse CNN into middle layers of Transformer and change the Transformer encoder into a parallel structure, which introduces the inductive bias of CNN into Transformer to improve the performance on tiny dataset.

3.2 Conv-Stem

We firstly implant 4 layers of CNN in the first stage of Transformer (called Conv-stem) to generate feature maps of different scales. A Conv-stem is created by stacking 3×3 and 7×7 convolutions. Each layer consists of 3 Batch Normalization (BN) + 3ReLu + 1Projection to restrict the data in a small range. In the first layer, the kernel size is set to (7, 7) with stride 2 and padding 3 to obtain a larger receptive field. The rest of the layers adopt kernel sizes (3, 3, 3) and strides (2, 2, 1). As a result, the inductive bias of CNN is introduced into transformer structure by stacking overlapped convolutions to extract more fine-grained local features. In the meantime, by replacing the attention blocks in the first layer of Transformer with convolution layers reduces the computation cost.

3.3 Middle Layer

The self-attention mechanism of ViT achieves better results by capturing content based global interaction. In comparison, CNN only focuses on the local feature patterns. In order to complement each other's strengths and weaknesses, we implant CNN into the second and the third stages of Transformer to form a parallel structure. The attention operation is computed as:

$$Attention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d_{head}}})V \tag{1}$$

here the inputs are Q, K, V, and the output is a refined feature. Then the spatial reduction operation (SR) is performed on K and V as:

$$SR(X) = Reshape(x, R_i) \tag{2}$$

here x is either K or V. The above reshape operation is realized via convolution operation, where R_i is the reduction ratio. The SR based attention mechanism is called Spatial Reduction Attention (SRA). Then, our modified Transformer Encoder processes the input feature map X as:

$$TransEncoder(X) = Attn(X) + DSC\_FFL(X), \tag{3}$$

where Attn denotes the attention operation with the residual structure, and DSC_FFL represents the proposed depth-wise separable feedforward layer. Next, for the second and the third stages of our model, we append convolution blocks to form parallel structures. Specifically, we use two 7 × 7 kernels, each of which has a BN layer and a ReLU layer. For each convolution kernel, the input feature map is processed as:

$$Conv\_Blk(X) = Relu(BN(Conv(X))) \tag{4}$$

Then, for each stage, the results of Eqn. (3) and (4) are added to yield the output as:

$$Xout = TransEncoder(X) + Conv\_Blk(X) \tag{5}$$

3.4 Positional Encoding (PE) and Cls Token

As claimed by Islam [23], positional information and semantic feature co-exist in CNN, and deeper layers have more positional information. Based on this, we introduce CNN into deeper layers of Transformer, in which the output feature maps by CNN could provide positional information for Transformer. In our model, we intentionally delete PE module, so that our network will no longer be restricted by the size of the input images, and we can acquire the positional information from CNN.

LeViT [24] has proved that replacing cls token with average pooling has almost no impact on the accuracy of the model, while it somehow improves the robustness of the model. In this light, we remove cls-token and insert an average pooling into the classification head to produce the classification output.

3.5 Feed-Forward Layer (FFL)

The purpose of Feed-Forward Layer (FFL) in Transformer is to implement nonlinear transformation so as to improve the representation ability. It consists of 2 linear transformations, namely a fully connected layer and an activation function. The former converts the input position vector to an intermediate variable, the latter performs nonlinear transformation on this variable to better capture the local and global image features. Depth-wise Separable Convolutional Feed-Forward Layer (DSCFFL) is a variant of FFL, in which a depth-wise separable convolution (DWS) layer is added after the self-attention blocks in Transformer. It has the following merits:

1. Less parameters: DSCFFL is based on the idea of local connection and weights sharing, it thereby needs less parameters than fully connected layers, resulting in lightweight model and faster training. Meanwhile, it is more suitable to process large-scale images than fully connected layers.
2. More focus on local features: Since DSCFFL utilizes depth-wise separable convolution operations, it can easily learn local features. In comparison, fully connected layers tends to capture global features.

Considering this, we replace the traditional FFL with DSCFFL.

4 Experiment

In this section, we will verify the effectiveness of our proposed scheme on improving the performance of ViT on tiny datasets. Comprehensive experiments include image classification (Sect. 4.1), object detection (Sect. 4.2) and semantic segmentation (Sect. 4.3).

4.1 Image Classification

We compare the image classification results on 3 tiny datasets: CIFAR-100[25], Flowers[26] and Chaoyang[27].

CIFAR-100: CIFAR-100 is a subset of the tiny Images dataset, which consists of 60k color images (32×32). The 100 classes in the CIFAR-100 are divided into 20 super classes. There are 600 images per class. Each image has a “fine” label (of its category) and a “coarse” label (of its superclass it belongs to). And there are 500 training images and 100 testing images per class.

Flowers: An image classification dataset including 102 flower categories. Each class has 40 to 258 images, which have large scale, pose and light variations. Besides, some categories have large intra-class variance.

Chaoyang: A professional dataset labeled by several pathologists. There are 1111 normal samples, 842 serrated, 1404 adenocarcinoma, 664 adenoma for training, and 705 normal, 321 serrated, 840 adenocarcinoma, 273 adenoma samples for testing.

In this experiment, we compare our results with small scale Transformer models. Firstly,we randomly crop the input image into 224 × 224, and adopt random horizontal flipping and mix-up for data augmentation. We also adopt Label-smoothing regularization during training. AdamW optimizer is used with a weight decay of 5×10^{-2} and a momentum of 0.9. The initial learning rate is set to 10^{-3} that decreases following the cosine schedule. All models are trained 300 epochs on 2 RTX 3090 GPUs. In the meantime, center-crop is employed on the validation set, wherein 224 × 224 pixels are cropped to test the recognition accuracy.

Table 1. Comparison of Classification results on 3 tiny datasets

Models	Params(M)	GFLOPs	Top-1 Acc. (%)		
			CIFAR-100	Flowers	Chaoyang
CNN:					
ResNet-18[28]	11.7	1.8	70.43	59.83	78.12
ResNet-50[28]	25.6	4.1	79	69.23	82.71
Pure Transformer:					
DeiT-Ti[14]	5.7	1.3	65.08	50.06	82
T2T-ViT-7[12]	4.2	1.1	69.37	65.2	80.74
Transformer with Hierarchy Structure:					
PiT-Ti[29]	4.9	0.7	73.58	68.4	82.7
PVT-Ti[30]	13.2	1.9	69.22	62.32	73.68
PVTv2-B0[31]	3.4	0.6	77.44	67.51	82.05
mobileViT-S[15]	5.6	2.0	70.97	65.26	80.98
CMT-XS[17]	15.2	1.5	80.98	53.68	82.37
EdgeViT-S[8]	11.1	1.9	82.03	62.09	80.18
Transformer with Convolutional Inductive Bias:					
ConViT-Ti[15]	6.0	1.0	75.32	57.51	82.47
Conv-PVT[32]	13.0	2.2	77.17	66.38	82.05
TinyConv-PVT (ours)	11.8	2.9	**82.37**	**70.64**	**83.22**

A comparative result on tiny dataset is listed in Table 1, pure Transformer models (including DeiT-Ti [14] and T2T-ViT-7 [12]) make poor results due to the lack of inductive bias. For some Transformer based models with partial fusion of CNN (including PVTv2 [31], ConViT-Ti [15] and Conv-PVT [32]), they have increased the classification accuracy to some extents over the previous two, which indicates that the fusion of CNN with Transformer indeed improves the recognition accuracy. Finally, our model achieves the highest results on all 3 datasets. It's worth noting that Chaoyang is a medical image dataset with fewer annotated samples, which illustrates the application prospects of our model in medical imaging field. As shown in Table 1, our model outperforms other popular methods on tiny dataset with large margins.

4.2 Object Detection

Object detection is implemented on MS COCO [33] dataset. MS COCO 2017 has 80 categories, which includes 118k training images, 5k validation images and 20k test dev images without labels. We use RetinaNet [34] and Mask R-CNN [35] as our detector. We use Xaiver [36] to initialize the newly added layers, and train our model on 2 RTX3090 using AdamW as the optimizer with an initial learning rate of 10–4. We adopt 1x training schedule (i.e. 12 epochs) to train both RetinaNet and Mask R-CNN. And we resize the training image to 800 and 1333 pixels for the shorter and longer sides respectively. And in the testing stage, the shorter size of the input image is fixed to 800 pixels.

Table 2. Comparison of Object Detection on MS COCO dataset

Models	Box AP	
	RetinaNet	Mask R-CNN
PVT-Tiny[30]	36.7	36.7
PVTv2-b0[31]	37.2	38.2
LightVit-T[37]	–	37.8
PoolFormer-S12[38]	36.2	37.3
Conv-PVT[32]	33.1	37.4
TinyConv-PVT (ours)	**38.9**	**39.6**

As shown in Table 2, by introducing CNN into the middle layer of the Transformer, we have achieved some performance improvement in object detection. In RetinaNet, compared to the original SRA models PVT and PVTv2, our model has improved by 2.2% and 1.7%, respectively. These improvements mainly come from the contribution of parallel convolution blocks. Similarly, we have achieved similar effects in Mask-RCNN. Compared to the original ViT backbone, we have improved by 2.9% and 1.4% in Mask-RCNN. This indicates that after incorporating convolutional fusion, our PVT can also effectively replace the original backbone for object detection.

4.3 Semantic Segmentation

The semantic segmentation task is implemented on ADE20k dataset [39]. ADE20k is a challenging scene parsing benchmark for semantic segmentation, containing 150 fine-grained semantic categories. It has 20210 images for training, 2000 images for validation and 3352 images for testing. We use TinyConv-PVT as the backbone (that is pre-trained on ImageNet-1k [40]) and use FPN as our detector. In the training stage, we use Xaiver [36] to initialize with AdamW as the optimizer with an initial learning rate of 10^{-4}. The batch size is set to 16. The training process is conducted on 2 RTX 3090 for 160K epochs. The learning rate decays according to the polynomial decay scheme. During training, the input images are randomly cropped into 512*512, and resize the short side to 512 during testing.

Table 3. Experimental Results of Semantic Segmentation on ADE20k

Models	mIoU(%)
PVT-Tiny [30]	36.6
TopFormer-T[41]	32.8
Xcit [42]	39.9
PoolFormer-S12 [38]	37.2
Conv-PVT [32]	37.2
TinyConv-PVT(ours)	**40.9**

As shown in Table 3, we again make notable improvement over the previous methods for 160K. This indicates that the deployment of CNN in the middle layer of Transformer is indeed beneficial to segmentation task.

4.4 Ablation Study

As mentioned earlier, we made the following changes to the original Transformer structure: firstly, we introduce CNN into the middle layer of Transformer to adjust the assignment of attention. Then, we remove PE to get rid of the restriction of the field input image size. Finally, we replace the traditional FFL with DSCFFL to reduce the computation cost. In this section, we testify the effectiveness of the above key designs through ablation experiments of image classification.

The Function of the Convolution Block. To further validate the function of the convolution blocks that we implanted into ViT, we have conducted 3 groups of comparative experiments on CIFAR-10 and CIFAR-100 among different versions of the original ViT (including PVT, PVTv2 and scalable ViT) and the integrated models with Conv. The experiment is conducted on a single 3090 card using the same training strategy as mentioned in Sect. 4.1. The experimental results are shown in Table 4, all our models (tiny conv) surpass the corresponding original tiny models (PVT-Tiny, PVTv2-Tiny and ScalableViT-Tiny) with safe margin, some even surpass the corresponding small versions (PVT-Small, ScalableViT-Small), which demonstrates the effectiveness of our proposed scheme.

Table 4. The results of convolution blocks.

Models	Params	CIFAR-10	CIFAR-100
PVT-Tiny	13.23	86.72	66.30
PVT-Small	24.49	88.68	68.60
PVT-Tiny+conv	24.41	**93.35**	**74.40**
PVTv2	13.54	93.37	77.03
PVTv2+conv	24.37	**95.91**	**77.86**
ScalableVit-Tiny	14.90	91.23	71.14
ScalableVit-Small	23.32	**93.15**	74.73
ScalableVit-Tiny+conv	22.93	92.93	**75.05**

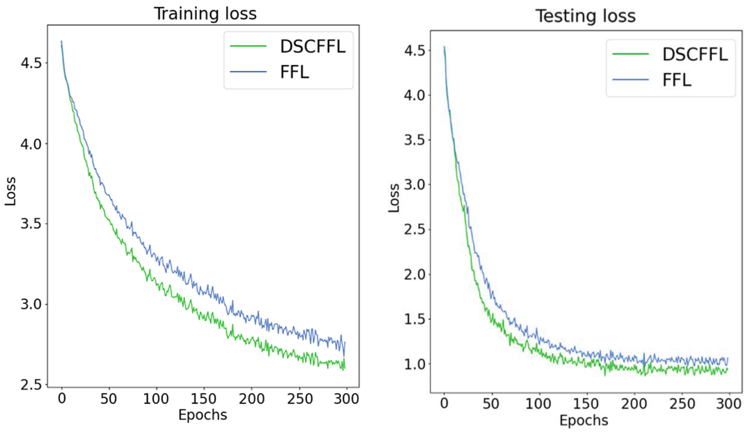
Removal of Positional Encoding (PE). In this section, we compare the accuracies by deploying PE at different positions. We train different models 100 epochs on 2 RTX 3090. As shown in Table 5, apparently, the deployment of PE at different positions make no major differences than without PE. This result reflects the fact that the integration of CNN and Transformer gets rid of the dependency on PE.

Table 5. Comparison of accuracies by deploying PE at different layers

Positions of PE	Pos1+Pos2+Pos3	Pos1+Pos3	Pos2+Pos3	Pos1+Pos2	None
Accuracy (%)	**69.33**	69.32	69.26	69.30	69.32

The Impact of Depth-Wise Separable Convolutional Feed-Forward Layer (DSCFFL). In Sect. 3.4, we propose to replace the traditional FFL with DSCFFL. Here we investigate the impact of DSCFFL on our network. We train the model (with FFL and DSCFFL) on 2 RTX 3090 cards 300 epochs on the training set and evaluate their performances on the testing set. Both the training and testing losses are shown in Fig. 5 below.Apparently, the loss curves of DSCFFL on both the training and testing sets drop faster than that of the traditional FFL, indicating a faster convergence speed.

The Optimal Size of Convolution Blocks. As shown in Fig. 3, we add 2 convolution blocks that are parallel to Transformer encoder. In this section, we will derive the optimal size of the convolutional blocks through comparative experiments. As shown in Table 6, we compare different sizes and train the network 300 epochs on CIFAR-100. Obviously, (7, 7) achieves the highest accuracy than other sizes, while more complex convolution blocks do not bring any improvements on the performance of Transformer.



(a) Training loss (b) Testing loss

Fig. 5. Comparison of loss curves for training (a) and testing sets (b) between FFL and DSCFFL

Table 6. Comparative results by using different sizes of convolution blocks on CIFAR-100

Convolution Blocks	(7,7,7,7)	(3,3,3,3)	(7,7,7)	(3,3,3)	(7,7)	(3,3)	(7)	(3)
Accuracy(%)	79.5	79.1	81.2	80.6	**82.4**	81	79.2	78.6

5 Conclusions

Aiming to improve the general performance of Transformer in implementing downstream vision tasks on tiny dataset and reducing its computation cost. We propose a deep fusion network that incorporates CNN with Transformer in many aspects. Specifically, we remove the positional encoding, replace the Transformer block in the first stage with CNN, replace FFL with DSCFFL and deploy 2 convolution blocks that are parallel to Transformer in the second and the third stages. Extensive experiment have been conducted on several mainstream tiny datasets to validate the effectiveness of our proposed network, where it manifests superior performance by outshining other state-of-the-art methods in accuracy.

Acknowledgment. This work is supported by the Intelligent Policing Key Laboratory of Sichuan Province, No. ZNJW2024FKMS004.

References

1. Liu, Y., Sangineto, E., Bi, W., Sebe, N., Lepri, B., Nadai, M.: Efficient training of visual transformers with small datasets. Adv. Neural. Inf. Process. Syst. **34**, 23818–23830 (2021)
2. Raghu, M., Unterthiner, T., Kornblith, S., Zhang, C., Dosovitskiy, A.: Do vision transformers see like convolutional neural networks? Adv. Neural. Inf. Process. Syst. **34**, 12116–12128 (2021)
3. Jin, Y., Han, D., Ko, H.: Trseg: transformer for semantic segmentation. Pattern Recogn. Lett. **148**, 29–35 (2021)
4. Heo, J., Wang, Y., Park, J.: Occlusion-aware spatial attention transformer for occluded object recognition. Pattern Recogn. Lett. **159**, 70–76 (2022)
5. Vaswani, A., et al.: Attention is all you need. Adv. Neural Inform. Process. Syst. **30** (2017)
6. Hassani, A., Walton, S., Shah, N., Abuduweili, A., Li, J., Shi, H.: Escaping the big data paradigm with compact transformers. arXiv preprint arXiv:2104.05704 (2021)
7. Lee, S.H., Lee, S., Song, B.C.: Vision transformer for small-size datasets. arXiv preprint arXiv:2112.13492 (2021)
8. Pan, J., et al.: Edgevits: competing light-weight cnns on mobile devices with vision transformers. In: European Conference on Computer Vision, pp. 294–311. Springer (2022). https://doi.org/10.1007/978-3-031-20083-0_18
9. Wang, J., et al.: Riformer: keep your vision backbone effective but removing token mixer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14443–14452 (2023)
10. Ryoo, M.S., Piergiovanni, A.J., Arnab, A., Dehghani, M., Angelova, A.: Tokenlearner: what can 8 learned tokens do for images and videos? arXiv preprint arXiv:2106.11297 (2021)
11. Liu, Z., et al.: Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 10012–10022 (2021)
12. Yuan, L., et al.:. Tokens-to-token vit: training vision transformers from scratch on imagenet. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 558–567 (2021)
13. Wu, H., et al.: Cvt: Introducing convolutions to vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 22–31 (2021)
14. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H.: Training data-efficient image transformers & distillation through attention. In: International Conference on Machine Learning, pp. 10347–10357. PMLR (2021)
15. d'Ascoli, S., Touvron, H., Leavitt, M.L., Morcos, A.S., Biroli, G., Sagun, L.: Convit: improving vision transformers with soft convolutional inductive biases. In: International Conference on Machine Learning, pp. 2286–2296. PMLR (2021)
16. Srinivas, A., Lin, T.-Y., Parmar, N., Shlens, J., Abbeel, P., Vaswani, A.: Bottleneck transformers for visual recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 16519–16529 (2021)
17. Guo, J., Han, K., Wu, H., Tang, Y., Chen, X., Wang, Y., Xu, C.: Cmt: convolutional neural networks meet vision transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision And Pattern Recognition, pp. 12175–12185 (2022)

18. Fan, X., Liu, H.: Flexformer: flexible transformer for efficient visual recognition. Pattern Recogn. Lett. **169**, 95–101 (2023)
19. Sun, P., et al.: Swformer: sparse window transformer for 3d object detection in point clouds. In: European Conference on Computer Vision, pp. 426–442. Springer (2022). https://doi.org/10.1007/978-3-031-20080-9_25
20. Chang, H., Zhang, H., Jiang, L., Liu, C., Freeman, W.T.: Maskgit: masked generative image transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11315–11325 (2022)
21. Wang, Y., Yan, L., Feng, Z., Xia, Y., Xiao, B.: Visual tracking using transformer with a combination of convolution and attention. Image Vis. Comput. **137**, 104760 (2023)
22. Mehta, S., Rastegari, M.: Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. arXiv preprint arXiv:2110.02178 (2021)
23. Islam, M.A., Jia, S., Bruce, N.D.B.: How much position information do convolutional neural networks encode? arXiv preprint arXiv:2001.08248 (2020)
24. Graham, B., et al.: Levit: a vision transformer in convnet's clothing for faster inference. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 12259–12269 (2021)
25. McMahan, B., Moore, E., Ramage, D., Hampson, S., Aguera y Arcas, B.: Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics, pp. 1273–1282. PMLR (2017)
26. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
27. Zhu, C., Chen, W., Peng, T., Wang, Y., Jin, M.: Hard sample aware noise robust learning for histopathology image classification. IEEE Trans. Med. Imaging **41**(4), 881–894 (2021)
28. 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)
29. Heo, B., Yun, S., Han, D., Chun, S., Choe, J., Oh, S.J.: Rethinking spatial dimensions of vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 11936–11945 (2021)
30. Wang, W., et al.: Pyramid vision transformer: a versatile backbone for dense prediction without convolutions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 568–578 (2021)
31. Wang, W., et al.: Pvt v2: improved baselines with pyramid vision transformer. Comput. Vis. Media **8**(3), 415–424 (2022)
32. Zhang, X., Zhang, Y.: Conv-pvt: a fusion architecture of convolution and pyramid vision transformer. Int. J. Mach. Learn. Cybern. **14**(6), 2127–2136 (2023)
33. Lin, T.Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
34. Lin, T.-Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2980–2988 (2017)
35. 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)
36. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, pp. 249–256. JMLR Workshop and Conference Proceedings (2010)

37. Huang, T., Huang, L., You, S., Wang, F., Qian, C., Xu, C.: Lightvit: Towards light-weight convolution-free vision transformers. arXiv preprint arXiv:2207.05557 (2022)
38. Yu, W., et al.: Metaformer is actually what you need for vision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10819–10829 (2022)
39. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ade20k dataset. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 633–641 (2017)
40. 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)
41. Zhang, W., et al.: Topformer: token pyramid transformer for mobile semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12083–12093 (2022)
42. Ali, A., et al.: Xcit: cross-covariance image transformers. Adv. Neural. Inf. Process. Syst. **34**, 20014–20027 (2021)

SCAD-Net: Spatial-Channel Attention and Depth-Map Analysis Network for Face Anti-spoofing

Nguyen Quoc Viet(✉), Le Minh Tri, Nguyen Thi Yen, and Nguyen Minh Giang

COMIT Corporation, TTC Tower No 19, Duy Tan Street, Ha Noi City, Cau Giay District, Vietnam
sales@comitcorp.com
https://www.comitcorp.com

Abstract. This work delves into the critical realm of Face Anti-Spoofing in face recognition systems. Previous research has explored Deep Learning methodologies, encompassing End-to-End binary cross-entropy supervision, Pixel-wise auxiliary supervision, and Generative models with pixel-wise supervision. In this study, we proposed a method that synergized two main approaches: End-to-End binary cross-entropy supervision and Pixel-wise auxiliary supervision. Specifically, our method is called SCAD-Net and is made up of a Binary Classification Module and a Depth-map Decoder for pixel analysis tasks. The findings showed the effectiveness of the proposed approach, surpassing the performance of both referenced architectures and current State-of-the-art methods (SOTA). This amalgamation of methodologies not only fortifies the robustness of Face Anti-Spoofing systems but also contributes significantly to advancing the understandings and the capabilities of countering emerging spoofing techniques. Remarkably, with only 9.8 million parameters and requiring only 10 milliseconds per inference, SCAD-Net demonstrates exceptional efficiency and potential for real-world application, highlighting its suitability for deployment in resource-constrained environments.

Keyword: Computer Vision, Deep Learning, Face anti-spoofing.

1 Introduction

Nowadays, facial recognition has become an essential component of biometric authentication systems, playing a significant role in various aspects of our lives, particularly in the area of security and personal verification. This technology offers convenience; however, it also brings along the potential risk of fraudulent activities. Spoofing techniques such as printed images, phones, and tablets demonstrated how easily it is to exploit facial recognition systems. Thus, the

---

Supported by COMIT Coporation.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 319–333, 2025.
https://doi.org/10.1007/978-3-031-78128-5_21

issue of face anti-spoofing (FAS) has become more and more critical. The primary challenge lies in accurately identifying instances where there are indicators of attempted deception aimed at circumventing facial recognition systems.

The primary challenge lies in accurately identifying instances where there are indicators of attempted deception aimed at circumventing facial recognition systems. All face spoofing strategies can be divided into the following categories: two-dimensional (2D) attacks and three-dimensional (3D) attacks. 2D attacks include cases that use a photo, a printed image, or a video containing a person's facial features to deceive the recognition system. Compared with traditional 2D attacks, 3D attacks are more realistic in terms of color, texture, and geometry structure. 3D masks are made from different materials: paper, resin, plaster, plastic, silicone or latex. In real-world use cases, 2D attacks occurred more frequently due to its significantly lower barrier of entry to attempt. Therefore, the method developed in this work focuses on countering forgery in 2D attack scenarios.

Most current models employ deep learning to counter these spoofing tactics, two of the most prominent methods are End-to-End Binary Cross-Entropy Supervision [7,26,36,37] and Pixel-wise Supervision [3,8,12,14,34,35].

End-to-end Binary Cross-Entropy Supervision utilizes deep learning methods to address the problem by treating it as a binary classification problem, where one class represents live images, and the other represents spoof images. The majority of methods employ CNN architectures to extract image features for classification. However, these supervisory cues only provide global information (spatial/temporal) for live/spoof embedding learning, potentially leading to weak generalization capabilities. Although the models achieve fantastic accuracy in the intra-testing set which is similar to the training set, they have poor results in the cross-testing sets which are in different domains. As a consequence, the models demonstrate ineffective performance with other different camera devices, illumination, and Presentation Attacks that were not seen in the training set.

Pixel-wise Supervision is a method that provides localized supervision at each pixel, offering detailed supervisory signals to detect signs of spoof. It is currently the most widely used method, often employing physical signals such as pseudo depth labels, binary mask labels, and reflection maps for the detailed description of local live/spoof cues and supporting the auxiliary learning process. Recent methods [3,8,17] frequently utilize depth maps for auxiliary learning in the model. These approaches typically depend on the local features of images to reconstruct the depth map of the face. The underlying concept is that human faces exhibit varying depths at different facial points, whereas spoof faces presented through devices like phones, tablets, or printed photos will have consistent depths at image points. Model [3], based on the Pixel-wise Supervision method, has achieved the best results on the majority of current benchmarks. Nevertheless, pixel-wise supervision often depends on high-quality training data (e.g., high resolution) to exploit forgery cues effectively. It may struggle to provide effective supervisory cues when training data contains a lot of noise and is of low quality.

Each approach has its advantages and disadvantages. Combining these two approaches to leverage their strengths and address each other's weaknesses was the motivation behind our research. We researched and selected two models [7,8] corresponding to the two approaches above. Subsequently, we integrated their architectures to create a novel baseline model. Furthermore, we introduced several enhancements to their structures compared to baseline models to enhance the model's ability to generalize. The depth map ground truth was chosen using a new strategy to improve the model's learning ability. Experimental results demonstrated the effectiveness of the proposed method, surpassing state-of-the-art models. This study has three main contributions:

- Proposing a novel anti-spoofing method that addresses the facial anti-spoofing challenge, outperforming the state-of-the-art methods on multiple benchmark tests, and demonstrating an applicability in the real-world system through its remarkable speed.
- Introducing a new approach to the problem by combining End-to-End Binary Cross-Entropy Supervision and Pixel-wise Supervision methods.
- The effectiveness of combining these methods demonstrated through experiments in this thesis will encourage further exploration of novel combined approaches in the future, potentially discovering unique and effective solutions.

2 Related Work

2.1 Traditional Hand-Crafted Feature

In the initial stages of Face Anti-Spoofing (FAS) development, when CNN [2] had not been developed yet, manual feature-based methods were employed as a feature extractor to extract hand-crafted features from a facial image, which were then input into binary classifiers like SVM. Some typical manual feature-based methods have been used such as LBP [18], HoG [19], SIFT [21], SURF [22], and DoG [23]. However, hand-crafted methods have poor results in FAS problems, because they were not specifically designed to capture spoof cues in facial images. Additionally, these methods were sensitive to various conditions, including differences in camera devices, illumination, and Presentation Attack (PA) instruments.

2.2 End-to-End Binary Cross-Entropy Supervision

The convolutional neural network (CNN) revolutionized the fields of computer vision and image processing. The architecture of CNN is designed to automatically and flexibly learn spatial hierarchies of features from input images through convolutional filters or kernels. In the context of Face Anti-Spoofing (FAS), CNN has replaced manual feature-based methods in feature extraction from input images. CNN-based methods treat FAS as a binary classification task. For instance, in [24], the input face image is passed through the VGG-face feature extractor (pre-trained), and then use SVM as the classifier. In another approach [25], a Resnet-50 network [6] pre-trained on the ImageNet [15] dataset is employed as the feature extractor, and a classifier is constructed from stacked probabilities. In particular, Qiao et al. [26] proposed a method that combines the architectures of CNN and Transformer [27] for feature extraction. The integration of self-attention modules [27] and convolution mapping enhances the model's ability to learn both general and local features. Although binary cross-entropy supervised methods provided promising performances, they struggled with detecting spoof signals which led to poor generalization.

2.3 Pixel-Wise Auxiliary Supervision

These methods pay attention to the information of pixel values and supervise the learning of auxiliary facial features, such as depth maps and reflection maps. The AENet model is introduced in the CelebA Spoof dataset [12] using depth maps for supervised learning. This approach uses two Conv3×3 layers, then up-sample the feature to 14×14 size to learn geometric information, and finally passes the feature map through three Fully Connected layers to learn semantic information. The PSPNet [8] model utilizes a Segmentation network architecture [9] for supervised learning based on facial depth maps. Notably, the TransFas method [3] achieves the highest results on several of the current benchmarks. TransFas uses the Transformer architecture as a feature extractor to capture global features of faces. To learn local features, TransFas uses a special architecture called Cross-layer Relation-aware Attention (CRA) [3]. Although the Pixel-wise Auxiliary Supervision method is more effective in learning global features than End-to-End Binary Cross-Entropy Supervision, its learning local signals capability is limited.

2.4 Foundation Models

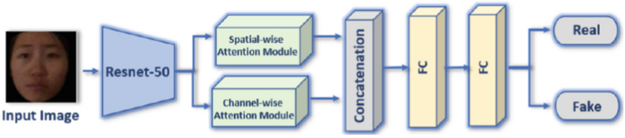


Fig. 1. The architecture of Face Fake Net. Resnet backbone and SCA module were used as a feature extractor and then passed the feature map through Fully Connected (FC) layers to classify the input images into live or spoof labels[7].

Face Fake Net. FACE FAKE NET [7] is an End-to-End Binary Cross-Entropy Supervision solution that combines CNN architecture and Spatial Channel-wise Attention Modules (SCA-CNN) [10]. Spatial-wise Attention Module and

Channel-wise Attention Module work at spatial and channel levels not only to enhance local features and global features but also to disregard unimportant ones. As presented in Fig. 1 The input image is fed into the 44 first layers of Resnet-50 which was trained on ImageNet dataset [15]. Then, the feature map is fed into both Spatial Attention Module and Channel Attention Module in parallel to extract spatial attention weight and channel attention weight. These attention weights are concatenated together and classified into 2 labels live or spoof by Fully Connected Layers. With the integration of these two modules, Face Fake Net exhibits enhanced results in anti-spoofing compared to other previous End-to-End Binary Cross-Entropy Supervision methods.

EfficientNet and EfficientNetV2. EfficientNet [29] and EfficientNetV2 [30] were introduced by Tan and Le, they are advanced convolutional neural network architectures known for their efficiency and performance in image classification tasks.

EfficientNet: EfficientNet is a series of structures of convolutional neural networks that systematically studies model scaling and identifies. EfficientNet carefully balances network depth, width, and resolution for optimal performance. This observation led to the development of a new scaling method that uniformly scales all dimensions of depth/width/resolution by using a simple yet highly optimal compound coefficient. EfficientNets achieves state-of-the-art accuracy on other datasets with significantly fewer parameters.

EfficientNetV2: focused on achieving faster training speed and better parameter efficiency compared to previous versions. New operations such as Fused-MBConv were incorporated into the architecture. EfficientNetV2 models are up to 6.8x smaller and train significantly faster than their predecessors.

Both architectures represent significant advancements in the development of efficient and powerful models for image classification, specifically addressing the Face Anti-spoofing problem. Bi-FPNFAS [38] used EfficientDet [32] which is a modified version of EfficientNet. Bi-FPNFAS employed a bi-directional feature pyramid network (BiFPN) that is used for convolutional multi-scaled feature extraction on the EfficientDet detection architecture to extract facial image feature maps and perform deep pixel-wise supervision.

3 Methodology

Although both Cross-entropy supervision and Pixel-wise Auxiliary Supervision have shown promising results, they also have their respective limitations. While Cross-entropy supervision methods have improved local feature learning, their ability to generalize has not seen significant enhancement. On the other hand, Pixel-wise Auxiliary Supervision methods heavily rely on depth map supervision for learning and reconstruction, making them highly dependent on the quality of the input images. This results in a substantial reduction in model accuracy when dealing with low-quality images. In this paper, we propose a solution called SCAD-Net, and the architecture is shown in Fig. 2. This method combines the

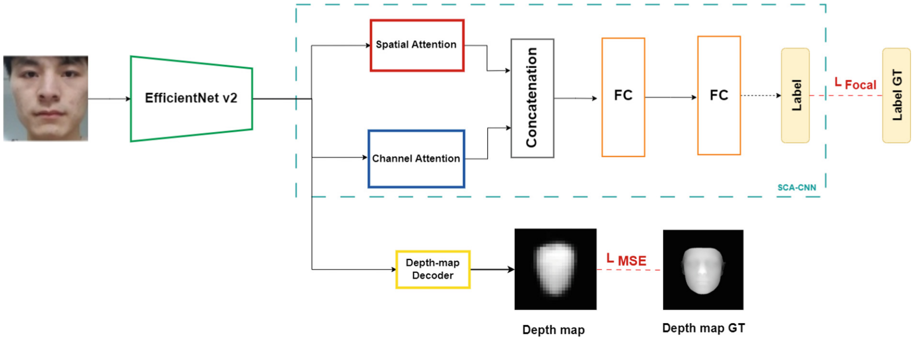


Fig. 2. The SCAD-Net model incorporates a Spatial-Channel Attention module to precisely predict live or spoof images and a Depth-map Decoder dedicated to auxiliary learning through facial depth map analysis. It utilizes EfficientNet-V2 for feature extraction and a Convolutional Network for depth map generation tasks.

architectures of the two aforementioned methods. The architecture of this solution was inspired by the structures of Face Fake Net [7] and further enhancements to advance its efficacy. The proposed method is developed in the hope of leveraging the advantages of the Cross-entropy supervision method and the Pixel-wise Auxiliary Supervision method and overcoming their weaknesses.

3.1 The Architecture

The feature extractor will process the input facial image to extract the feature map. After the feature map was obtained, it was passed independently through the Spatial-Channel Attention (SCA) [10] module and the Depth-map Decoder module. The SCA-CNN module uses the feature map to predict the label of the input image while the Depth-map Decoder reconstructs the facial depth map from the feature map. Focal Loss supervises the classification result and Mean Square Error Loss supervises the depth map output. Total Loss (1) is the sum of Focal Loss and Mean Square Error. The depth map provides the auxiliary information that aid the optimization of the feature extractor in the back-propagation step.

$$Loss_{Total} = Loss_{Focal} + Loss_{MSE} \tag{1}$$

3.2 Backbone

The input images are processed through EfficientNetV2's backbone [30]. In the experimental phase, we evaluated various architectures, including Resnet-18, Resnet-50, Resnet-101, EfficientNet, and EfficientNetV2. We determined that EfficientNetV2-M which was pre-trained by the ImageNet21k dataset offers the most effective results.

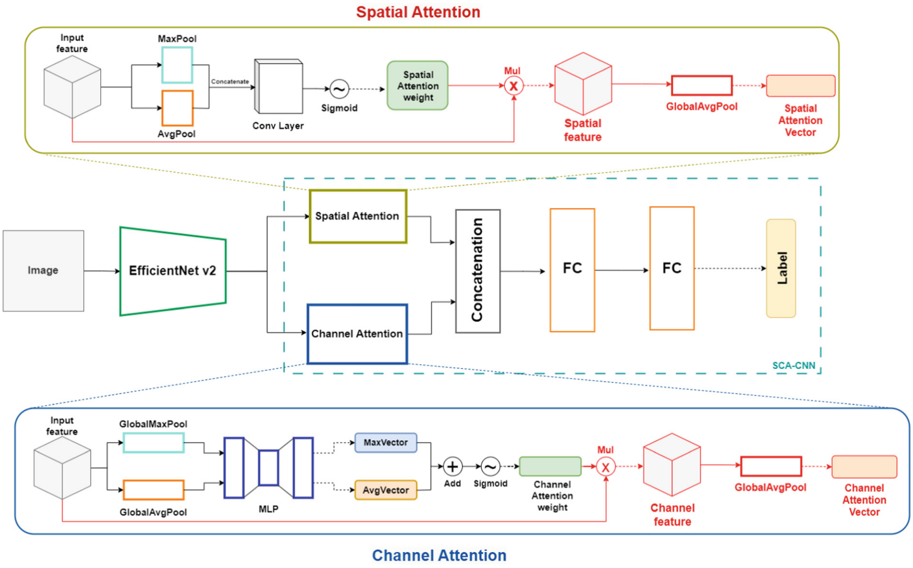


Fig. 3. The enhancements of the SCA block in the Face Fake Net model. The parts with arrows and blocks drawn in red indicate the additional improvements made in comparison to the original architecture. (Color figure online)

3.3 SCA-CNN Module

In the SCA block, there are several enhancements to the original model to improve its quality. In the original architecture, the feature map extracted by the feature extraction block is simultaneously passed through two attention modules: Channel Attention and Spatial Attention. The Spatial-Channel Attention block, as proposed in the Face Fake Net, only applies a sigmoid function to the attention weights and then concatenates them. This approach will potentially reduce the information from the original image. To address this, we experimented and added several steps before concatenating the attention feature vectors, as illustrated in Fig. 3. Specifically, the attention weight vectors are multiplied with the original feature map to extract information from the input data. In this way, we can extract the information which is important in the input image. Then, we pass the product through a global average pooling layer to aggregate the information contained within a feature vector. Finally, the feature vectors from both the Channel Attention and the Spatial Attention blocks are concatenated together. This concatenated vector is then fed into a multi-layer fully connected (FC) neural network for classification purposes.

3.4 Depth-Map Decoder Module

In this module, we employ a simple CNN network to estimate the depth map. The input to the module is the feature map extracted from the backbone, and the output is a 28 × 28 facial depth map. The depth map is a grayscale image, supervised by the Mean Squared Error loss function, with the ground truth generated from the PRNet network.

3.5 Focal Loss

A prevalent challenge in assessing models for the Face Anti-Spoofing (FAS) problem is imbalanced data. Within the common Face Anti-Spoofing datasets, the majority are images with spoof labeling due to the diversity in presentation attacks, environmental conditions, illumination, and background scenes. For example, the number of spoof labels is 5 times more than the number of live labels, so the model tends to overly focus on learning from the majority classes, while it learns very few features from the minority classes.

In this case, employing fundamental binary loss can lead to unsatisfactory prediction results. To address this issue, we employ Focal Loss instead of using the fundamental Cross-Entropy Loss in our classifier, aiming to improve the model's performance by effectively managing the majority class's effect on the model's weight.

Focal loss is an upgraded version of the cross-entropy loss, designed to address imbalanced data issues by focusing more on examples of the minority class and less on examples of the majority class. It was introduced in the context of object detection, where the imbalance between foreground and background classes can significantly affect performance. The innovation of focal loss lies in its ability to dynamically scale the loss value based on the confidence of the predictions, thereby reducing the relative loss for well-represented examples (majority class) and focusing the model's attention on less represented examples (minority class).

The formula for focal loss is designed to enhance the fundamental cross-entropy loss by adding an adjustment factor to focus more on the minority class, misclassified examples, and less on easy examples (majority class). The focal loss is given by (2):

$$\text{Focal Loss}(p_t) = -\alpha_t(1 - p_t)^\gamma \log(p_t) \tag{2}$$

p_t is the model's estimated probability for the class with the label y = 1, and α_t is a weighting factor that helps to balance the importance of majority/minority examples, γ is the focusing parameter that smoothly adjusts the rate at which majority examples are down-weighted, making the model focus more on minority examples.

4 Experiments

In this section, we used public datasets to evaluate our method, followed by the evaluations of our method's local feature learning ability and generalization learning ability.

4.1 Datasets and Metrics

Datasets. To evaluate the SCAD-Net method, we used three benchmark datasets: CelebA-spoof [12], OULU-NPU [11], and SiW [33]. Firstly, we used the CelebA-spoof dataset to compare with the result of Face Fake Net [7](our base method). Then, we used two high-resolution databases: OULU-NPU and SiW datasets to validate the generalization learning ability of our method. Finally, we used OULU-NPU and SiW databases for cross-dataset testing between them.

Metrics. In our research, we employed standard metrics to ensure a fair and consistent comparison with previous studies. These metrics are Attack Presentation Classification Error Rate (APCER), Bona Fide Presentation Classification Error Rate (BPCER), and Average Classification Error Rate (ACER), ACER is the mean of APCER and BPCER. Besides, False Positive Rate (FPR) and Recall (also known as Sensitivity or True Positive Rate) were used in CelebA-spoof to compare with previous relevant methods.

4.2 Implementation Details

Hardware. We used four RTX3090 GPUs, Intel Xeon Gold 6230 CPU, and 128GB RAM for training. Then we used RTX1650 GPU, AMD Ryren 7 4000 series, and 16GB RAM for inference and testing.

Data Preprocessing. We used the given face annotations to get faces and labels from datasets then used PRNet.[4] to generate the ground-truth depth maps for live images, and spoof images depth maps are set to black image (all pixels are zeros). The images's size was set to 224×224 (RGB) and the depth map's size was set to 28×28.

Training Setting. During the training period, we set the batch size to 64, the learning rate starts from e-4 and will reduce by half after every 10 epochs, and the weight decay is e-4. Our model was trained in 50 epochs. In the Focal Loss, α_t is 0.8 and γ is 2. Total Loss is the sum of Focal Loss and Mean Square Error. Our backbone is EfficientNetV2-M[30] which was pre-trained by the ImageNet21k[15] dataset to reduce training time and enhance the model's quality.

4.3 Ablation Study

To confirm the enhancement in the SCA-CNN module, we experiment with a model that truncated the Depth-map Decoder from the training phase. In this case, our model includes only the EfficientV2 backbone and SCA-CNN module. Table 1 demonstrates that our improvements in the foundational SCA-CNN module improved the model capabilities.

Then, we experiment with several backbones to choose the most suitable for our architecture. We evaluated various architectures, including Resnet-18,

Table 1. The comparison result between the original model Face Fake Net and our version which has been enhanced in the SCA-CNN module.

Method	APCER(%)↓	BPCER(%)↓	ACER(%)↓
Face Fake Net[7]	0.7	0	0.35
Face Fake Netenhanced in SCA-CNN module	0.29	0.37	**0.33**

Resnet-50, Resnet-101, EfficientNet, and EfficientNetV2. The results in the Table 2 show that EfficientNetV2-XL provides better accuracy. However, it has 235 million parameters so it is impossible to implement this version for the real-time system in reality. As a consequence, EfficientNetV2-M which was pre-trained by the ImageNet21k dataset offers the most suitable results.

Table 2. The comparison result between our versions in different backbone architectures.

Method	APCER(%)↓	BPCER(%)↓	ACER(%)↓
Resnet-50	0.29	0.37	0.33
Resnet-101	0.36	0.34	0.35
EfficientNetV2-M	0.26	0.34	0.3
EfficientNetV2-L	0.28	0.38	0.33
EfficientNetV2-XL	0.16	0.42	**0.29**

4.4 Result

In the testing phase, we truncate the Depth-map Decoder to optimize the model's speed. The inference time of 10 milliseconds per inference makes this model exceptionally well-suited for deployment in real-time systems.

Table 3. Comparison results on the testing set of the CelebA-Spoof.[12] dataset between our method and previous methods.

Method	Parameters(Million)	Recall (%)↑			APCER (%)↓	BPCER (%)↓	ACER (%)↓
		FPR = 1%	FPR = 0.5%	FPR = 0.1%			
Auxiliary [13]	22.1	97.3	95.2	83.2	5.71	1.41	3.56
BASN [14]	569.7	98.9	97.8	90.9	4	1.1	2.6
AENet [12]	42.7	98.9	97.3	87.3	2.29	0.96	1.63
Face Fake Net [7]	30.5	-	-	-	0.7	0	0.35
SCAD-Net (ours)	9.8	**99.4**	**99.0**	**96.4**	0.31	0.23	**0.27**

Table 4. Comparison results on protocols of the OULU-NPU[11] dataset between our method and previous methods.

Prot.	Method	APCER (%)↓	BPCER (%)↓	ACER (%)↓
1	PSPNet [8]	0.4	0.0	0.2
	TransFAS [3]	0.0	0.0	**0.0**
	SCAD-Net (Ours)	0.3	0.1	0.2
2	PSPNet [8]	0.8	1.4	1.1
	TransFAS [3]	1.5	0.5	1.0
	SCAD-Net (Ours)	0.5	0.3	**0.4**
3	PSPNet [8]	4.4±3.1	2.5±0.8	3.5±1.7
	TransFAS [3]	0.6±0.6	1.1±2.5	0.9±1.1
	SCAD-Net (Ours)	0.5±0.6	0.4±0.8	**0.5±0.4**
4	TransFAS [3]	3.3±3.7	1.7±2.4	2.5±2.9
	PSPNet [8]	2.1±2.2	2.5±4.2	2.3±2.3
	SCAD-Net (Ours)	1.9±1.8	2.7±2.1	**2.3±1.8**

Intra Testing. We execute intra-testing on both CelebA-spoof and OULU-NPU datasets. CelebA-spoof has only one train-test setting and OULU-NPU has four protocols to evaluate the generalization ability of face anti-spoofing methods.

Table 3 presents the comparison of results on the CelebA-spoof dataset between SCAD-Net and the previous methods. Our method achieved outstanding results compared with other methods with only 9.8 million parameters. This outcome shows that our developments in the SCA module enhanced the model's quality compared to the Face Fake Net [7] method (our base method).

Table 4 shows the test result on four protocols of the OULU-NPU dataset. This table compares the performance of our method with the performance of two other Pixel-wise supervision methods, specifically, TransFAS [3] is a state-of-the-art model. We perform the best results on protocols II, III, and IV of OULU-NPU. The results show that our method has a high accuracy in the effect of attacks created with different display devices, illumination conditions, and location. Moreover, the model demonstrated high accuracy in handling recording sensor variations, as evidenced by its performance on Protocol III.

Cross Testing. We proceeded to get cross-dataset testing between OULU-NPU and SiW. Specifically, we trained our model on SiW's training set and then evaluated the model in four protocol testing sets of OULU-NPU. Table 5 presents outstanding results of SCAD-Net in all four protocols of the OULU-NPU dataset. This outcome demonstrates the strong generalization learning ability of our method.

Table 5. The results of Cross-testing dataset between SiW and OULU-NPU. We used SiW's training set and the testing set of OULU-NPU and compared them to the results of previous methods.

Protocol	Method	APCER (%)↓	BPCER (%)↓	ACER (%)↓
1	CDCN [34]	0.6	21.7	11.2
	Auxiliary[13]	-	-	10.0
	FAS-SGTD [35]	1.7	13.3	7.5
	SCAD-Net (Ours)	2.0	3.0	**2.5**
2	CDCN [34]	10.3	15.3	12.8
	Auxiliary [13]	-	-	14.1
	FAS-SGTD [35]	9.7	14.2	11.9
	TransFAS [3]	3.9	11.9	7.9
	SCAD-Net (Ours)	8.0	4.8	**6.4**
3	CDCN [34]	17.5±4.6	11.7±12.0	14.6±4.8
	Auxiliary [13]	-	-	13.8±5.7
	FAS-SGTD [35]	11.4±5.8	11.9±10.4	11.7±3.6
	TransFAS [3]	7.8±4.4	7.2±5.3	7.5±2.6
	SCAD-Net (Ours)	1.9±3.9	2.3±2.8	**2.1±3.1**
4	CDCN [34]	0.8±1.9	10.0±11.6	5.4±5.7
	Auxiliary [13]	-	-	10.0±8.8
	FAS-SGTD [35]	0.8±1.9	15.8±20.3	8.3±10.0
	TransFAS [3]	2.1±3.0	6.7±6.9	4.4±4.7
	SCAD-Net (Ours)	1.9±3.5	2.3±2.6	**2.1±2.8**

5 Conclusion

In this work, we propose SCAD-Net for the Face anti-spoofing issue. Our method optimizes the Face Fake Net architecture by adding several layers and the EfficientNetV2 [30] backbone. By integrating Binary Cross Entropy and Pixel-wise Auxiliary Supervision, the model capitalizes on the strengths of both methods while mitigating their weaknesses. Its efficacy is validated by both intra-testing and cross-testing, demonstrating superior performance. Moreover, with only 9.8 million parameters—a significant reduction compared to prior models, this model is even capable of real world applications.

Acknowledgement. The authors would like to express their gratitude for the valuable feedback provided by the reviewers. This study was supported by COMIT Corporation.

References

1. Yu, Z., Qin, Y., Li, X., Zhao, C., Lei, Z., Zhao, G.: Deep learning for face anti-spoofing: a survey. IEEE Trans. Pattern Anal. Mach. Intell. **45**(5), 341–385 (2023)
2. O'Shea, K., Nash, R.: An Introduction to Convolutional Neural Networks. arXiv (2015)
3. Wang, Z., Wang, Q., Deng, W., Guo, G.: Face anti-spoofing using transformers with relation-aware mechanism. IEEE Trans, Biometrics, Behav. Identity Sci. **4**(3), 439–450 (2022)
4. Feng, Y., Wu, F., Shao, X., Wang, Y., Zhou, X.: Joint 3D face reconstruction and dense alignment with position map regression network. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) Computer Vision – ECCV 2018. LNCS, vol. 11218, pp. 557–574. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01264-9_33
5. Uhrig, R.E.: Introduction to artificial neural networks. In: Proceedings of IECON 1995 - 21st Annual Conference on IEEE Industrial Electronics, Orlando, FL, USA (1995)
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA (2016)
7. Alshaikhli, M., Elharrouss, O., Al-Maadeed, S., Bouridane, A.: Face-fake-net: the deep learning method for image face anti-spoofing detection. In: France, European Workshop on Visual Information Processing (EUVIP), Paris (2021)
8. Zhang, L., Sun, N., Wu, X., Luo, D.: Advanced face anti-spoofing with depth segmentation. In: International Joint Conference on Neural Networks (IJCNN), Padua, Italy (2022)
9. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: International Joint Conference on Neural Networks (IJCNN), Honolulu, HI, USA (2017)
10. Chen, L., et al.: SCA-CNN: spatial and channel-wise attention in convolutional networks for image captioning. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA (2017)
11. Boulkenafet, Z., Komulainen, J., Li, L., Feng, X., Hadid, A.: OULUNPU: a mobile face presentation attack database with real-world variations. In: IEEE International Conference on Automatic Face and Gesture Recognition, Washington, DC, USA (2017)
12. Zhang, Y., et al.: CelebA-spoof: large-scale face anti-spoofing dataset with rich annotations. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12357, pp. 70–85. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58610-2_5
13. Liu, Y., Jourabloo, A., Liu, X.: Learning deep models for face anti-spoofing: binaryor auxiliary supervision. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA (2018)
14. Kim, T., Kim, Y., Kim, I., Kim, D.: BASN: enriching feature representation using bipartite auxiliary supervisions for face anti-spoofing. In: 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), Seoul, Korea (South) (2019)
15. 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 (2009)

16. Zhang, Z.: Improved adam optimizer for deep neural networks. In: 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS) (2018)
17. Wang, Z., Wang, Q., Deng, W., Guo, G.: Learning multi-granularity temporal characteristics for face anti-spoofing. IEEE Trans. Inf. Forensics Secur. **17**, 1254–1269 (2022)
18. Määttä, J., Hadid, A., Pietikäinen, M.: Face spoofing detection from single images using micro-texture analysis. In: 2011 International Joint Conference on Biometrics (IJCB) (2011)
19. Komulainen, J., Hadid, A., & Pietikäinen, M.: Context-based face anti-spoofing. In: 2013 IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS) (2013)
20. Yang, J., Lei, Z., Liao, S., Li, S.Z.: Face liveness detection with component dependent descriptor. In: 2013 International Conference on Biometrics (ICB) (2013)
21. Patel, K., Han, H., Jain, A.K.: Secure face unlock: spoof detection on smartphones. IEEE Trans. Inf. Forensics Secur. **11**(10), 2268–2283 (2016)
22. Boulkenafet, Z., Komulainen, J., Hadid, A.: Face antispoofing using speeded-up robust features and fisher vector encoding. IEEE Signal Process. Lett. **24**(2), 141–145 (2017)
23. Peixoto, B., Michelassi, C., Rocha, A.: Face liveness detection under bad illumination conditions. In: 2011 18th IEEE International Conference on Image Processing (2011)
24. Li, L., Feng, X., Boulkenafet, Z., Xia, Z., Li, M., Hadid, A.: An original face anti-spoofing approach using partial convolutional neural network. In: 2016 Sixth International Conference on Image Processing Theory, Tools and Applications (IPTA) (2016)
25. Bresan, R., Pinto, A.S., Rocha, A., Beluzo, C.E., Carvalho, T.J.: FaceSpoof Buster: a Presentation Attack Detector Based on Intrinsic Image Properties and Deep Learning. ArXiv (2019)
26. Qiao, T., Wu, J., Zheng, N., Xu, M., Luo, X.: FGDNet: fine-grained detection network towards face anti-spoofing. IEEE Trans. Multimedia **25**, 7350–7363 (2023)
27. Vaswani, A., et al.: Attention Is All You Need. CoRR abs/ arXiv: 1706.03762 (2017)
28. Guo, J., Zhu, X., Yang, Y., Yang, F., Lei, Z., Li, S.Z.: Towards Fast, Accurate and Stable 3D Dense Face Alignment. CoRR abs/ arXiv: 2009.09960 (2020)
29. Tan, M., Le, Q.V.: EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. ArXiv abs/ arXiv: 1905.11946 (2019)
30. Tan, M., Le, Q.V.: EfficientNetV2: Smaller Models and Faster Training. ArXiv abs/ arXiv: 2104.00298 (2021)
31. Ming, Z., Yu, Z., Al-Ghadi, M., Visani, M., Luqman, M.M., Burie, J.-C.: Vitranspad: video transformer using convolution and self-attention for face presentation attack detection. In: 2022 IEEE International Conference on Image Processing (ICIP) (2022)
32. Tan, M., Pang, R., Le, Q.V.: EfficientDet: Scalable and Efficient Object Detection. arXiv:1911.09070 (2020)
33. Liu, Y., Jourabloo, A., Liu, X.: Learning Deep Models for Face Anti-Spoofing: Binary or Auxiliary Supervision. CoRR abs/ arXiv: 1803.11097 (2018)
34. Yu, Z., et al.: Searching Central Difference Convolutional Networks for Face Anti-Spoofing. CoRR abs/ arXiv: 2003.04092 (2020)
35. Wang, Z., et al.: Deep Spatial Gradient and Temporal Depth Learning for Face Anti-spoofing. CoRR abs/ arXiv: 2003.08061 (2020)

36. Wang, C.-Y., Lu, Y.-D., Yang, S.-T., Lai, S.-H.: PatchNet: A Simple Face Anti-Spoofing Framework via Fine-Grained Patch Recognition. arXiv:2203.14325 (2022)
37. Parkin, A., & Grinchuk, O.: Creating Artificial Modalities to Solve RGB Liveness. CoRR abs/ arXiv: 2006.16028 (2020)
38. Roy, K., et al.: Bi-FPNFAS: bi-directional feature pyramid network for pixel-wise face anti-spoofing by leveraging fourier spectra. Sensors (Basel, Switzerland) **21** (2021)

Next Generation Loss Function for Image Classification

Shakhnaz Akhmedova(✉) and Nils Körber

Robert Koch Institute, Nordufer 20, 13353 Berlin, Germany
`{AkhmedovaS,KoerberN}@rki.de`

Abstract. Neural networks are trained by minimizing a loss function that defines the discrepancy between the predicted model output and the target value. The selection of the loss function is crucial to achieve task-specific behaviour and highly influences the capability of the model. A variety of loss functions have been proposed for a wide range of tasks affecting training and model performance. For classification tasks, the cross entropy is the de-facto standard and usually the first choice. Here, we try to experimentally challenge the well-known loss functions, including cross entropy (CE) loss, by utilizing the genetic programming (GP) approach, a population-based evolutionary algorithm. GP constructs loss functions from a set of operators and leaf nodes and these functions are repeatedly recombined and mutated to find an optimal structure. Experiments were carried out on different small-sized datasets CIFAR-10, CIFAR-100 and Fashion-MNIST using an Inception model. The 5 best functions found were evaluated for different model architectures on a set of standard datasets ranging from 2 to 102 classes and very different sizes. One function, denoted as Next Generation Loss (NGL), clearly stood out showing same or better performance for all tested datasets compared to CE. To evaluate the NGL function on a large-scale dataset, we tested its performance on the Imagenet-1k dataset where it showed improved top-1 accuracy compared to models trained with identical settings and other losses. Finally, the NGL was trained on a segmentation downstream task for Pascal VOC 2012 and COCO-Stuff164k datasets improving the underlying model performance. The loss function code and an example of its implementation can be found on the project page https://github.com/ZKI-PH-ImageAnalysis/Next-Generation-Loss/tree/main.

Keywords: Loss function · Genetic Programming · Deep Learning · Classification · Segmentation

1 Introduction

Deep learning (DL) has received extensive research interest in developing new image processing algorithms, and resulting models have been remarkably suc-

---

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78128-5_22.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 334–349, 2025.
https://doi.org/10.1007/978-3-031-78128-5_22

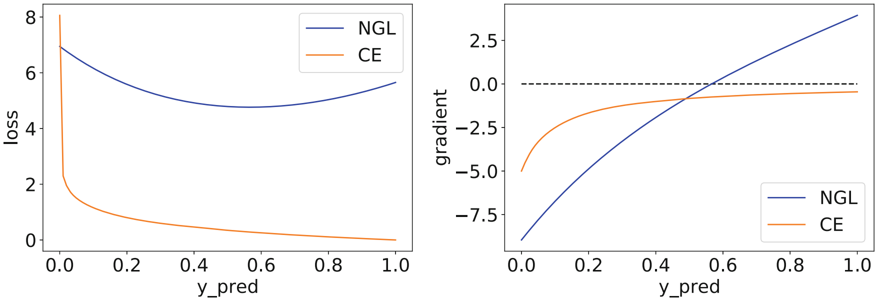


Fig. 1. NGL and CE functions (left) and their gradients (right) for $y_{real} = 1$.

cessful in a variety of image analysis tasks such as classification [20], segmentation [32], object detection [9] and more. Deep learning methods represented by (or partially based on) convolutional neural networks have become a well-known and constantly developing research topic in the field of image analysis. Various deep learning models, for example ResNet [12], Inception [37], U-Net [32], DeepLab [3], have been proposed over the years for different image processing tasks and domains. Vision transformers [6] constantly gain popularity among researchers due to their high training efficiency and scalability and are already outperforming convolutional neural networks on multiple benchmarks for classification, segmentation and image generation problems.

The loss function plays a crucial role in deep learning in general, as it shows the direction for model training and affects the performance of the model and training speed. There are well-known losses used for most image processing tasks, such as cross entropy (CE) [5], Focal Loss [23], or mean square error [43]. Recently, researchers started proposing loss functions developed for specific applications or tasks. Usually, that is done by hand and requires a lot of expert knowledge, time, parameter adjustments, and thus computational resources, due to the complexity of modern models.

The loss function design can be represented as a symbolic regression problem that can be solved by using Genetic Programming (GP) [1]. The GP approach is frequently used to design previously unknown heuristics in order to solve hard computational search problems [17]. Genetic programming is a domain-independent method, which was used in this study to genetically breed a population of loss functions for deep learning models to solve image classification problems. GP was applied to find a generally applicable loss function for DL classifiers independent of the dataset and model architecture. Several experiments with different objective functions using InceptionV3 [37] trained on the CIFAR-10 [19], Fashion-MNIST [45] and CIFAR-100 [19] datasets were conducted to find an optimized loss function.

The loss functions found were evaluated using ResNet50 and InceptionV3 models trained on CIFAR-10 [19], CIFAR-100 [19], Fashion-MNIST [45], Malaria [29], PCam [41], Colorectal Histology [16] and Caltech 101 [8] datasets.

In the next step, the best function, denoted as NGL, was used to train ResNet [12] models and Swin transformers [25] to classify the ImageNet-1k dataset. In the end, it was applied to train U-Net and DeepLab models on Pascal VOC 2012 [7] and COCO-Stuff164k [2] datasets to show the applicability for segmentation.

Among the evaluated functions, NGL demonstrated exceptionally good results outperforming baseline losses, such as cross entropy loss, focal loss, symmetric cross entropy and dice loss. Moreover, further investigation has showed that the reason for these good results may be self-regulation, which is inherently present in the NGL due to its mathematical definition.

Thus, the main contributions in this paper can be summarized as follows:

- Experimental results show that GP is capable to design a good loss function for deep neural networks without any prior knowledge neither about the model nor about the problem at hand;
- NGL, a new highly efficient loss function, applicable for any DL model to solve classification and segmentation problems, was found;
- Self-regularization naturally implemented in the NGL indicates a direction for loss function design in general;
- Models trained by the NGL achieved same or better performance on a variety of datasets including ImageNet-1k and COCO-Stuff164k datasets.

The rest of this paper is organized as follows. Section 2 briefly reviews related work of deep neural networks used for image analysis and loss function design. The Genetic Programming algorithm and proposed search procedure are described in Sect. 3, while the experimental settings and results are presented in Sect. 4. Obtained results are discussed in Sect. 5. Finally, the paper is concluded in Sect. 6.

2 Related Work

The progress in solving image processing and visual recognition tasks is related to the rapid development of convolutional neural networks (CNNs) and subsequently vision transformers, which have been already used for real-world applications in different scientific fields. A plethora of convolutional neural network architectures have been proposed, including the widely used Inception network [36] that explores the problem of multi-scale fusion in convolution calculations to better characterize image information, and ResNet [12] with residual blocks to solve the vanishing gradients problem.

Dosovitskiy et al. [6] proposed the Vision Transformer (ViT) by formulating image classification as a sequence prediction task of the image patch sequence similarly to [40], thereby capturing long-term dependencies within the input image. Transformer-based models for image processing have been rapidly developed, and the following models are considered as some of the most representative: Vision Transformers (ViT) [6], Data-efficient image Transformers (DeiT) [39] and Swin-Transformer [25].

Most deep learning models initially designed to solve classification problems can be applied also for segmentation tasks, as segmentation can be formulated as a classification problem of pixels with semantic labels (semantic segmentation) or partitioning of individual objects (instance segmentation). In the recent decade there were a lot of works presenting new and more powerful segmentation methods, U-Net [32], which was initially developed for biomedical image segmentation, is one of the most widely used architectures not only in the life sciences. Additionally, DeepLabv1 [3], DeepLabv2 [3] and DeepLabv3+ [4] are among some of the most popular image segmentation approaches, which use dilated convolution to address the decreasing resolution in the network (caused by max-pooling and striding) and Atrous Spatial Pyramid Pooling (ASPP), that probes an incoming convolutional feature layer with filters at multiple sampling rates.

Aside from developing deeper networks with more complex structures and features to get better performance, better loss functions have also been proven to be effective on improving the model performance in most recent works [15,38]. Many loss functions have been utilized for neural networks based on softmax activation function, such as Mean Square Error (MSE) loss function [43], Cross Entropy (CE) loss function [5] (Binary Cross Entropy loss function when images are divided into two classes, Categorical Cross Entropy for multiple classes), etc. In most state-of-the-art supervised learning problems, practitioners typically use large capacity deep neural networks together with cross-entropy loss. The latter can be explained by the fact that the traditional CE loss is supported by clear theory, easy training and good performance. Nowadays, there are many variants of CE that have been proposed in the past few years, e.g. symmetric CE (SCE) [11], or Focal Loss [23].

Loss functions used for segmentation can be divided into four groups described in [15]: distribution-based, region-based, boundary-based, and compounded losses. The most well-known and commonly used distribution-based loss functions used to train segmentation models are binary cross entropy [47] and focal loss [24]. Region-based loss functions such as dice loss [35], and Tversky loss [33] calculate the similarity between images. Hausdorff distance loss [31] is defined as boundary-based loss functions in [15], while the exponential logarithmic loss [44] is considered as the compounded loss.

Most of the mentioned loss functions can be considered as heuristic methods designed by experts by using theory and the domain knowledge for specific tasks and/or models. Moreover, loss function design requires great effort from experts to explore the large search space, which is usually sub-optimal in practice. Nevertheless, only several commonly used losses are usually implemented for model training, for example, categorical cross entropy for classification or dice loss for segmentation. Recently, the automated search of suitable loss functions without domain knowledge has received much attention from the computer vision community. Reinforcement learning algorithms were used to learn better loss functions with good generalization ability on different image analysis tasks [28,46]. However, loss functions found by using these approaches either have task-specific requirements, such as environment interaction in reinforcement learning, or remain fixed after training.

A lot of works on automatic loss search follow the search algorithms used in AutoML [13]. In [22] the authors explore the possibility of searching loss functions automatically from scratch for generic tasks, e.g., semantic and instance segmentation, object detection, and pose estimation. In studies such as [42] and [21] the main focus is on the search for particular hyper-parameters within the fixed loss formula. AutoML-Zero [30] proposes a framework to construct machine learning algorithms from simple mathematical operations, but the search space and search strategies are specialized, which limits its potential application. The authors of the study presented in [10] propose a framework for searching classification loss functions by using evolutionary algorithms but the searched loss poorly generalizes to large-scale datasets. These works motivated us to design loss functions by using the Genetic Programming algorithm, which uses a large set of primitive mathematical operations and terminal values, capable of training different models regardless of their structure or the task at hand.

3 Method

Genetic Programming (GP) is a population-based evolutionary optimization algorithm, which can be applied to design different heuristics, depending on the problem at hand, and, additionally, it is considered as a machine learning tool, as it can be used to discover a functional relationship between features in data. GP was initially inspired by the biological evolution, including natural processes such as mutation and selection. In GP, each solution is represented by a tree structure, and each of these trees are evaluated recursively to produce the resulting multivariate expression. There are two types of nodes used for tree-based GP: the terminal node, also called leaf, which is randomly chosen from the set of variables, and the tree node, which can be chosen from a predefined set of operators (an example of the generated trees is given in the Sect. 1 of the supplementary material).

The GP search process starts with random initialization of a set of potential solutions, which is also called a population of individuals, in the functional search space. The number of individuals is the predefined parameter n, which does not change through the whole search process. The overall GP search process is summarized in Algorithm 1. Thus, for this study a set of loss functions was randomly generated, and each loss function was represented as a tree, where the terminals were randomly chosen from the set $\{y_{pred}, y_{real}, \mathbb{R}\}$, while the operators were chosen from the set $\{+, -, \times, \div, \times(-1), \sqrt{}, \log, \exp, \sin, \cos\}$. To utilize these functions and terminals for loss function generation, several modifications were applied (mentioned modifications are described in the Sect. 1 of the supplementary material).

After initialization, the main search loop starts, which consists of the iteratively repeated steps crossover, mutation, fitness function evaluation and selection (the number of GP steps or generations is denoted as T). The fitness function

Algorithm 1. An overview of the implemented GP approach

```
Randomly initialize $n$ trees (generation 0), each representing one formula, e.g., loss function
Evaluate GP fitness function $F$ for each individual in the population
Determine the best individual
Create an empty external archive $A$
while generation number is less than $T$ do
    for each individual from population do
        Generate a child individual by applying the crossover operator
        if $rand_1 < M_{ST}$ then
            Apply subtree mutation to the generated child
        end if
        if $rand_2 < M_N$ then
            Apply one-point mutation to the generated child
        end if
    end for
    Evaluate GP fitness function $F$ for each generated child individual
    Update the best individual
    Create new population of size $2 \times n$ by concatenating children and parent individuals
    Select $n$ best individuals from the new population: create new generation
    Update the external archive $A$
end while
```

F determines the quality of the individual and is the crucial part of the optimization process, as our goal was to experimentally find a robust loss function that can be generalized to a variety of datasets.

The crossover operator is used to exchange the subtrees between two individuals (an example of the implemented crossover operator is demonstrated in the supplementary material). Mutation can be applied in various ways, but in this study two variants were used:

- a random subtree in the tree is chosen and replaced with another randomly generated subtree (subtee mutation);
- a random node in the tree is chosen and replaced with another randomly generated node (one-point mutation).

Both mutation operators as well as crossover have their own parameters, such as subtree and node mutation M_{ST}, M_N and crossover Cr rates, which determine how often individuals will be changed during the main search loop. Finally, all individuals and the ones generated after crossover and mutation steps are combined into one population, and then the selection operator is applied. In this study only n most fit individuals from the combined population are chosen for the next generation on each GP step.

Moreover, the success-history-based adaptation strategy was used to improve the efficiency of the GP approach. To be more specific, at the beginning of the main search loop an empty external archive A was created. The maximum size of this archive was set equal to the population size, $n_A = n$. All individuals not passing selection could have been saved in the archive with a given probability p_A. Individuals saved in the external set were used during crossover step with some probability Cr_A.

4 Experiments

4.1 Loss Function Search

As mentioned in the previous section, loss function search was performed by the GP algorithm and essentially can be described as an optimization process. A set of individuals, where each individual is a mathematical formula representing a loss function, is generated. These individuals change by crossover and mutation operators and then a new set of loss functions is selected. These actions are repeated a given number of times, called generations, and in the end the best individual or loss function is determined. In this implementation, the crucial part is selection, which in turn depends on how the fitness of the loss function is defined, since only the fittest individuals are transferred over to the next generation.

In this study, five experiments were conducted to search the loss function by GP, which only differed in how the fitness function F was defined. Specifically, the following definitions of F were used for each individual loss for each experiment:

- Train the InceptionV3 model from scratch one time on CIFAR-10 dataset, the validation error was used as the fitness value;
- Train the InceptionV3 model from scratch 3 times on Fashion-MNIST dataset, the averaged validation error was used as the fitness value;
- Fine-tune the pre-trained InceptionV3 model one time on CIFAR-10 dataset, the validation error was used as the fitness value;
- Fine-tune the pre-trained InceptionV3 model 3 times on Fashion-MNIST dataset, the averaged validation error was used as the fitness value;
- Train the InceptionV3 model from scratch on CIFAR-10, Fashion-MNIST and CIFAR-100 datasets (once each). The validation error for each dataset was compared to the respective validation error obtained by the same model trained using CE loss, and individual fitness was represented by the pair of numbers indicating the number of wins and the percentage of improvement compared to CE.

It should be noted that a loss function was discarded if its values were not in the range $[10^{-5}, 10^{5}]$ and a new one was generated instead of it. The fitness function evaluation procedure is described in details in the Sect. 2.1 of the supplementary material.

The model and datasets were selected to limit the training time required per generation while maintaining a reasonable evaluation of the selected loss functions. Regardless of the experiment, a batch size of 128, data augmentation including horizontal flip, width and height shift (0.1 for both), zoom (0.2), Adam optimizer [18] with reduction of the learning rate on plateau, and 50 epochs for CIFAR-10 and CIFAR-100 and 35 for Fashion-MNIST were used to train the network. The top layer of the InceptionV3 model was replaced by 9 new layers, consisting of Flatten, BatchNormalization (×3) [14], Dense (×3), Dropout (×2) [34], and a final Softmax layer.

The following parameters were used for GP regardless of the experiment: $n = 16$, $T = 100$, $M_{ST} = 0.3$, $M_N = 0.1$, $Cr = 0.7$, $p_A = Cr_A = 0.5$, minimum tree height and maximum tree size were set to 2 and 100, respectively.

The best performing functions for each of the five GP experiments were evaluated for different models and multiple datasets (all functions are listed in the Sect. 2.1 of the supplementary material).

4.2 Evaluation

Small Datasets for Classification. The evaluation of the five found functions was performed by training ResNet50 and InceptionV3 on seven datasets, which differed by the number of images, classes, by the type of images (grayscale and RGB), and their sizes. The brief description of datasets used in this study is given in Table 1 (N is the number of classes). ResNet50 was added to the initial experiments, as all the found functions were evaluated on InceptionV3 model during the GP search to rule out model specific properties. The top layers of both InceptionV3 and ResNet50 networks were not included, in both cases instead 9 new layers such as Flatten, BatchNormalization (×3), Dense (×3) and Dropout (×2) were added.

Table 1. Brief description of the small datasets used for classification.

Dataset	N	Dataset size	Class size	Image size	Image type
Malaria	2	27558	13779	edge lengths of $40 - 400$ pixels	RGB
PCam	2	327680	163840	96×96	RGB
Colorectal Histology	8	5×10^3	625	150×150	RGB
CIFAR-10	10	60×10^3	6×10^3	32×32	RGB
Fashion-MNIST	10	70×10^3	7×10^3	28×28	Grayscale
CIFAR-100	100	60×10^3	6×10^2	32×32	RGB
Caltech 101	102	9144	$40 - 800$	edge lengths of $200 - 300$ pixels	RGB

Parameter settings and data preprocessing varied for all datasets but were identical for all tested loss functions, specifics are shown in the Sect. 2.2 of the supplementary material. Classification accuracy on the test set after the last epoch was used as the evaluation metric.

Both InceptionV3 and ResNet50 models were trained five times for each dataset. Obtained results were averaged for both models and compared to the averaged CE (with and without $L2$ regularization), focal and SCE losses for the same models. In the end, the percentage of accuracy improvement or decrease compared to CE was determined. A final value from the range $[-0.5, 0.5]$ was considered as same performing, a value $> +0.5\%$ better performing and $< -0.5\%$ performing worse compared to CE.

Results obtained for all evaluated loss functions ($f_1, ..., f_5$) are given in Table 2 (results obtained for pre-trained models are presented in the Sect. 2.2 of the supplementary material). Loss function f_3 demonstrates the worst results compared to others. It showed particularly bad results on Malaria, Colorectal Histology, CIFAR-100, and Caltech 101 datasets, it should be noted that the last three mentioned datasets have less than 10^3 examples of each present class. Interestingly, no function outperformed CE loss on Malaria, CIFAR-10, and Fashion-MNIST, while 6 out of 9 functions showed significantly better results on PCam dataset.

Table 2. Average accuracy (%) for ResNet50 and InceptionV3 each trained 5 times for each dataset for cross entropy (CE) and percentual change of the tested loss functions compared to CE, best result per dataset is shown in bold.

Loss	Malaria	PCam	Colorectal Histology	CIFAR-10	Fashion-MNIST	CIFAR-100	Caltech 101	Mean
CE	94.0	69.4	88.9	92.8	94.0	68.2	72.5	±0
SCE	–0.06	+0.62	–0.45	–3.66	–2.42	–0.80	+2.71	–0.58
Focal	**+0.34**	+1.03	+ 2.89	-0.64	–0.02	–2.14	–0.78	+0.10
CE + $L2$	+0.11	–0.64	+0.88	**+0.32**	+ 0.09	+0.38	+3.67	+0.69
f_1	–0.27	+0.81	**+3.27**	-0.34	**+0.31**	-0.84	-3.67	-0.10
f_2	–0.09	+2.89	–7.56	-0.81	+0.04	+0.60	–3.61	–1.22
f_3	–27.98	+3.24	–62.45	–1.79	–0.48	–59.36	–41.67	-27.21
f_4	–0.21	–0.13	–1.98	-0.14	0.00	–0.32	–6.49	–1.32
$f_5(NGL)$	–0.27	**+7.07**	+1.77	+0.12	+0.07	**+1.01**	**+5.00**	**+2.11**

Overall, f_5 is the only function showing on average significantly better results than CE, outperforming it on four out of seven datasets with an improvement ranging from 1 to 7 %, while there was no significant difference on the remaining three datasets.

Given the overall good performance of the function found by an evolutionary method, we renamed function f_5 as Next Generation Loss (NGL) function, with the following formula:

$$f_{NGL} = \frac{1}{N}\sum_{i=1}^{N}\left[e^{(\alpha - y_{pred}^{(i)}\cdot(1+y_{real}^{(i)}))} - \cos(\cos(\sin(y_{pred}^{(i)})))\right], \qquad (1)$$

where $\alpha = 2.4092$, N is the number of classes.

To further test the generalizability of NGL, we tested its performance on ImageNet dataset and larger models.

ImageNet-1k. Experiments on ImageNet-1k dataset were performed using the ResNet [12] convolutional neural networks and Swin [25] transformer models using NGL, CE, SCE and focal losses with the same settings.

For ResNet, we used the AdamW optimizer [26], cosine annealing learning rate scheduler with a 30-epoch linear warm-up, while the learning rate was equal to 0.003. The number of epochs was set to 180, a batch size of 256, and a weight decay of 0.01 were used. For Swin-Transformers, the AdamW optimizer was employed for 400 epochs using a cosine decay learning rate scheduler and 60 epochs of linear warm-up. A batch size of 1024, an initial learning rate of 0.001, and a weight decay of 0.05 were used. All of the augmentation and regularization strategies were the same as in [25]. Results on ImageNet-1k are shown in Table 3. It should be noted, that for both ResNet and Swin models CE was applied with label smoothing and $L2$ regularization.

Table 3. Comparison of ImageNet results for models trained on CE, SCE NGL and focal losses. †denote results reported as 10-crop testing, all other results show single crop accuracy.

Model	top1-acc	Retrained (CE) top1-acc	SCE top1-acc	Focal top1-acc	NGL top1-acc
ResNet101 [12]	78.25 †	76.55	77.21	75.68	78.38
ResNet152 [12]	78.57 †	76.92	77.51	76.06	78.99
Swin-T [25]	81.3	81.19	78.79	77.68	81.25
Swin-S [25]	83.0	83.1	80.68	79.16	83.0

For ResNet models NGL shows a clear improvement over other losses, increasing the top-1 accuracy by $1-3\%$. For the Swin architecture NGL significantly outperforms SCE and focal losses, while it shows similar performance as regularized CE.

The training process for ResNet101 model is demonstrated in Fig. 2. It can be seen that NGL showed slower increase in accuracy compared to other losses, but converges later with a higher accuracy. Even though the NGL was found using only very small datasets, it shows astonishingly good performance on ImageNet-1k. For different architectures and model sizes NGL shows on average superior performance compared to mentioned loss functions.

Segmentation. Additionally, NGL was evaluated on segmentation problems. Specifically, it was used to train two segmentation models, DeepLabv2 with ResNet101 as backbone [27] and U-Net with ResNet34 as backbone. The U-Net model was trained and evaluated only on Pascal VOC 2012 [7] dataset, while the DeepLabv2 model was used for both Pascal VOC 2012 and COCO-Stuff164k [2] datasets.

The first experiments were conducted utilizing the U-Net model. The same parameter setting was used for CE, focal, dice and NGL losses: images were resized to 224 × 224, Adam optimizer was applied, the learning rate was 0.0001

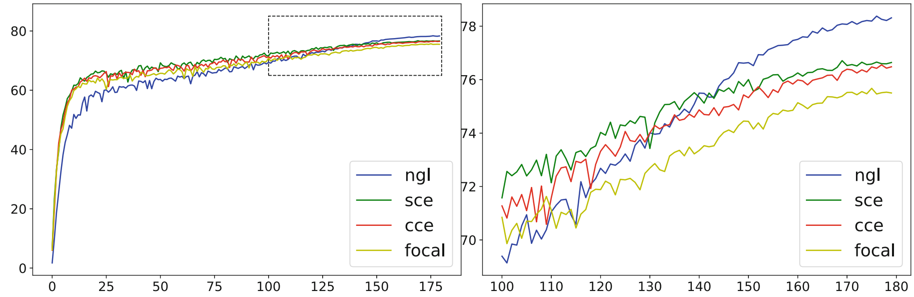


Fig. 2. Mean validation accuracy of ResNet101 model on each epoch during ImageNet-1k training.

and reduced on plateau with a factor of 0.2, patience 5 and $lr_{min} = 10^{-6}$ was used to change it according to the IoU metric value on validation dataset. The model was trained for 100 epochs with a batch size of 32. For DeepLabv2, Adam optimizer was chosen, the learning rate lr was set to 2.5×10^{-5} and the polynomial learning rate scheduler with weight decay 5×10^{-4} and power 0.9 was applied to decrease lr during 2×10^4 iterations. For both models the number of program runs was equal to 5 and mean IoU value was used to evaluate the performance of the model.

Results for Pascal VOC are presented in Table 4. Using the U-Net shows approximately $3 - 7\%$ improvement in terms of mean IoU when NGL is used for training compared to other losses, while showing the same results as CE and ouperforming dice and focal losses using DeepLabv2.

Table 4. Comparison of results (mIoU values) on Pascal VOC trained by CE, focal and dice and NGL loss functions.

Model	CE	Focal	Dice	NGL
U-Net	50.1	49.1	46.1	52.8
DeepLabv2	76.7	75.9	74.7	76.7

Finally, the DeepLabv2 model was trained on COCO-Stuff164k using mentioned loss functions. The parameter setting for these experiments was the same as for Pascal VOC 2012 dataset with the maximum number of iterations set to 3×10^5. A comparison of results is shown in Table 5, showing slight improvement for NGL loss compared to CE, and significant improvement compared to focal and dice losses.

The obtained results demonstrate that NGL is suitable for downstream segmentation tasks and showing improved performance compared to other commonly used losses.

Table 5. Comparison of results achieved by DeepLabv2 model on COCO-Stuff164k dataset.

	Pixel Accuracy	Mean IoU	Mean Accuracy
CE	66.9	39.8	51.8
Focal	67.7	39.5	51.5
Dice	39.7	17.3	26.9
NGL	67.8	39.9	51.6

5 Discussion

For qualitative analysis, the NGL function can be simplified to a binary classification problem, similar as in [10]. For a binary classification problem, NGL is as follows:

$$\begin{aligned} f_{NGL} = \frac{1}{2}\left[e^{(\alpha - y_{pred}^{(0)} \cdot (1+y_{real}))} + e^{(\alpha-(1-y_{pred}^{(0)})\cdot(2-y_{real}))}\right] \\ -\frac{1}{2}\left[\cos\left(\cos\left(\sin\left(y_{pred}^{(0)}\right)\right)\right) + \cos\left(\cos\left(\sin\left(1-y_{pred}^{(0)}\right)\right)\right)\right], \end{aligned} \tag{2}$$

where $\alpha = 2.4092$, $y_{pred} = (y_{pred}^{(0)}, y_{pred}^{(1)})$, $y_{pred}^{(1)} = 1 - y_{pred}^{(0)}$ and $y_{real} \in \{0, 1\}$.

Let us assume that the true labels y_{real} are either 0 or 1. The case where $y_{real} = 1$ is plotted in Fig. 1 for the CE loss and NGL functions. The cross entropy shows a monotonic decrease with y_{pred} converging to 1, the NGL shows a decrease in loss, resulting in a minimum at 0.57 and slightly increases when y_{pred} converges to 1. This increase in loss for y_{pred} approaching the true value seems to be counter-intuitive, but may be a factor for the overall good performance of the loss function. The mentioned increase of the loss value may prevent the model from becoming too confident in its output predictions and may provide an important advantage, as it lowers the probability of overfitting. Thus, this could provide an implicit form of regularization, enabling better generalization. A more detailed explanation is given in the Sect. 3 of the supplementary material.

While some of the functions found during GP showed better performance on certain datasets, NGL was the only function that performed better than CE (with and without regularization), SCE, focal and dice losses on average and was therefore used for ImageNet-1k and COCO-Stuff164k training, where it was able to show its generalizability to larger datasets and models. The NGL is independent of additional parameters, it is differentiable and has an implicit regularization. For larger datasets it could be observed that it converges slower than other mentioned losses, but at a higher maximum accuracy. The NGL was discovered experimentally and is not supported by theory, which in turn provides less information about the confidence of the model. Nonetheless, for a large proportion of applications the raw performance is the main goal to maximize. The search process did not cover the entire search space and in the future better performing functions may be found, thereby the NGL may point to a promising direction when searching for general suitable functions for classification tasks.

6 Conclusions

This study proposes to use Genetic Programming to search a generally applicable loss function for image classification tasks. During that process, a new function was found and shown to outperform other losses, commonly used for classification and segmentation tasks, on average on a variety of datasets demonstrating its general applicability. Moreover, it was shown that proposed loss function can be applied to train a variety of model architectures. Further analysis suggested that improvements provided by the new loss result from implicit regularization that reduces overfitting to the data.

References

1. Augusto, D., Barbosa, H.: Symbolic regression via genetic programming. In: Proceedings. Vol.1. Sixth Brazilian Symposium on Neural Networks, pp. 173–178 (2000)
2. Caesar, H., Uijlings, J., Ferrari, V.: Coco-stuff: thing and stuff classes in context. In: 2018 IEEE conference on Computer Vision and Pattern Recognition (CVPR). IEEE (2018)
3. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K.P., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE Trans. Pattern Anal. Mach. Intell. **40**, 834–848 (2016). https://api.semanticscholar.org/CorpusID:3429309
4. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: ECCV (2018)
5. Cover, T.M., Thomas, J.A.: Elements of information theory (2. ed.) (2006). https://api.semanticscholar.org/CorpusID:702542
6. Dosovitskiy, A., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. ArXiv abs/ arXiv: 2010.11929 (2020). https://api.semanticscholar.org/CorpusID:225039882
7. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge (VOC 2012) Results (2012). http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html
8. Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In: Computer Vision Pattern Recognition Workshop (2004)
9. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448 (2015)

10. Gonzalez, S., Miikkulainen, R.: Improved training speed, accuracy, and data utilization through loss function optimization. 2020 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8 (2019). https://api.semanticscholar.org/CorpusID:167217832
11. Han, B., Wu, Y.: A novel active contour model based on modified symmetric cross entropy for remote sensing river image segmentation. Pattern Recognit. **67**, 396–409 (2017). https://api.semanticscholar.org/CorpusID:2500127
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778 (2015). https://api.semanticscholar.org/CorpusID:206594692
13. He, X., Zhao, K., Chu, X.: Automl: a survey of the state-of-the-art. ArXiv abs/arXiv: 1908.00709 (2019). https://api.semanticscholar.org/CorpusID:199405568
14. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning (2015). https://api.semanticscholar.org/CorpusID:5808102
15. Jadon, S.: A survey of loss functions for semantic segmentation. In: 2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), pp. 1–7 (2020). https://api.semanticscholar.org/CorpusID:220128180
16. Kather, J.N., et al.: Multi-class texture analysis in colorectal cancer histology. Sci. Rep. **6**, 27988 (2016)
17. K.Burke, E., Hyde, M.R., Kendall, G., Ochoa, G., Ozcan, E., Woodward, J.R.: Exploring hyper-heuristic methodologies with genetic programming, pp. 177–201. Springer, Berlin (2009). https://doi.org/10.1007/978-3-642-01799-5_6
18. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7-9 May 2015, Conference Track Proceedings (2015). http://arxiv.org/abs/1412.6980
19. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
20. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Adv. Neural Inform. Process. Syst. **25** (2012)
21. Li, C., Lin, C., Guo, M., Wu, W., Ouyang, W., Yan, J.: Am-lfs: automl for loss function search. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 8409–8418 (2019). https://api.semanticscholar.org/CorpusID:158046703
22. Li, H., Fu, T., Dai, J., Li, H., Huang, G., Zhu, X.: Autoloss-zero: searching loss functions from scratch for generic tasks. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 999–1008 (2021). https://api.semanticscholar.org/CorpusID:232352674
23. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2980–2988 (2017)
24. Lin, T.Y., Goyal, P., Girshick, R.B., He, K., Dollár, P.: Focal loss for dense object detection. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 2999–3007 (2017). https://api.semanticscholar.org/CorpusID:47252984
25. Liu, Z., et al.: Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2021)
26. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization (2019)

27. Nakashima, K.: deeplab-pytorch (2018). https://github.com/kazuto1011/deeplab-pytorch
28. Oh, J., et al.: Discovering reinforcement learning algorithms. ArXiv abs/ arXiv: 2007.08794 (2020). https://api.semanticscholar.org/CorpusID:220633409
29. Rajaraman, S., et al.: Pre-trained convolutional neural networks as feature extractors toward improved malaria parasite detection in thin blood smear images. PeerJ **6**, e4568 (2018)
30. Real, E., Liang, C., So, D.R., Le, Q.V.: Automl-zero: evolving machine learning algorithms from scratch. In: International Conference on Machine Learning (2020). https://api.semanticscholar.org/CorpusID:212634211
31. Ribera, J., Güera, D., Chen, Y., Delp, E.J.: Weighted hausdorff distance: a loss function for object localization. ArXiv abs/ arXiv: 1806.07564 (2018). https://api.semanticscholar.org/CorpusID:49322181
32. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. ArXiv abs/ arXiv: 1505.04597 (2015). https://api.semanticscholar.org/CorpusID:3719281
33. Salehi, S.S.M., Erdoğmuş, D., Gholipour, A.: Tversky loss function for image segmentation using 3d fully convolutional deep networks. In: MLMI@MICCAI (2017). https://api.semanticscholar.org/CorpusID:732793
34. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(56), 1929–1958 (2014)
35. Sudre, C.H., Li, W., Vercauteren, T., Ourselin, S., Jorge Cardoso, M.: Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In: Cardoso, M.J., et al. (eds.) DLMIA/ML-CDS -2017. LNCS, vol. 10553, pp. 240–248. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-67558-9_28
36. Szegedy, C., et al.: Going deeper with convolutions. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–9 (2014). https://api.semanticscholar.org/CorpusID:206592484
37. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2818–2826 (2015). https://api.semanticscholar.org/CorpusID:206593880
38. Tian, Y., Su, D., Lauria, S., Liu, X.: Recent advances on loss functions in deep learning for computer vision. Neurocomputing **497**, 129–158 (2022). https://api.semanticscholar.org/CorpusID:248642142
39. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., J'egou, H.: Training data-efficient image transformers & distillation through attention. In: International Conference on Machine Learning (2020). https://api.semanticscholar.org/CorpusID:229363322
40. Vaswani, A., et al.: Attention is all you need. In: NIPS (2017). https://api.semanticscholar.org/CorpusID:13756489
41. Veeling, B.S., Linmans, J., Winkens, J., Cohen, T., Welling, M.: Rotation equivariant CNNs for digital pathology (Jun 2018)
42. Wang, X., Wang, S., Chi, C., Zhang, S., Mei, T.: Loss function search for face recognition. ArXiv abs/ arXiv: 2007.06542 (2020). https://api.semanticscholar.org/CorpusID:220496317
43. Willmott, C.J., Matsuura, K.: Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. Climate Res. **30**, 79–82 (2005). https://api.semanticscholar.org/CorpusID:120556606

44. Wong, K.C.L., Moradi, M., Tang, H., Syeda-Mahmood, T.F.: 3d segmentation with exponential logarithmic loss for highly unbalanced object sizes. ArXiv abs/arXiv: 1809.00076 (2018). https://api.semanticscholar.org/CorpusID:52157209
45. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms (2017)
46. Xu, Z., Hasselt, H.V., Silver, D.: Meta-gradient reinforcement learning. In: Neural Information Processing Systems (2018). https://api.semanticscholar.org/CorpusID:43966764
47. Yi-de, M., Qing, L., Zhi-bai, Q.: Automated image segmentation using improved pcnn model based on cross-entropy. In: Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing 2004, pp. 743–746 (2004). https://api.semanticscholar.org/CorpusID:11039022

NAOL: NeRF-Assisted Omnidirectional Localization

Muzhou Yu[1], Zhihan Cai[2], Kailu Wu[2], Dapeng Cao[1], and Kaisheng Ma[2](✉)

[1] School of Computer Science and Technology, Xi'an Jiaotong University,
710049 Xi'an, China
`{muzhou9999,dapengcao}@stu.xjtu.edu.cn`
[2] Institute for Interdisciplinary Information Sciences, Tsinghua University,
100080 Beijing, China
`{cai-zh21,wkl18,kaisheng}@mails.tsinghua.edu.cn`

Abstract. Visual localization is a challenging task involving precise camera position and orientation estimation from an image. Specifically, existing panorama datasets suffer from a limitation in the number of available omnidirectional images, resulting in a sparse distribution and insufficient diversity compared to the well-distributed dataset. This makes omnidirectional localization hard for conventional feature-based visual localization algorithms. In this paper, we introduce a novel and efficient approach named *NAOL*, specifically designed to address the omnidirectional localization problem. Our proposed pipeline unfolds in two pivotal stages: stage one employs a visual-based algorithm for preliminary coarse pose estimation. Acknowledging the number of omnidirectional images is limited and the distribution is sparse, in stage two, we take an innovative step to augment the dataset and refine the coarse pose estimations by stage one. We introduce the Depth-supervised Panorama Neural Radiance Fields (DP-NeRF), a novel approach designed for training on a single omnidirectional image with depth, thereby enriching the dataset while enabling an iterative algorithm grounded in DP-NeRF to enhance localization accuracy. Our experimental results validate the efficacy of the NAOL algorithm in performing visual localization in scenarios with sparse panorama datasets.

Keywords: Omnidirectional Localization · Visual Place Recognition · Neural Radiance Fields

1 Introduction

Visual place recognition stands out as a challenging taskand has received considerable attention in recent years, given its pivotal role in various practical applications in autonomous driving [36], virtual reality and augmented reality [31]. Visual-based localization algorithms [34,37] offer a straightforward and cost-effective means to locate a robot or a vehicle in a known environment, relying

---

M. Yu and Z. Cai—Equal contribution.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 350–366, 2025.
https://doi.org/10.1007/978-3-031-78128-5_23

solely on camera images as input without additional information. Typical methods in this field are often formulated as an image retrieval task [2,42]. These methods solve the problem through well-defined pipelines encompassing feature extraction, image retrieval, and pose optimization steps. This streamlined process ensures efficiency and practicality in solving the intricate problem of visual place recognition. The significance of these approaches is underscored by their relevance in addressing the demands of contemporary technologies.

Notably, we notice that visual-based localization is sensitive to environmental changes, illumination variations, alterations in appearance, and diverse perspectives, as well as to local geometric changes resulting from small scene changes [37,44]. In addition, due to the limited field of view of perspective cameras, some perspective images lack identifiable features and only exhibit ambiguous local structures [39,43], thereby hindering localization accuracy. In contrast, omnidirectional images, with their panoramic field of view, offer a comprehensive representation of the surroundings, making them resilient to object displacements and insignificant local features. The distinctive trait of omnidirectional images makes them promising solutions to alleviate the influence of outliers encountered in visual-based localization. Recognizing the potential of omnidirectional images serves as a driving force behind our exploration of their application in the visual-based localization task.

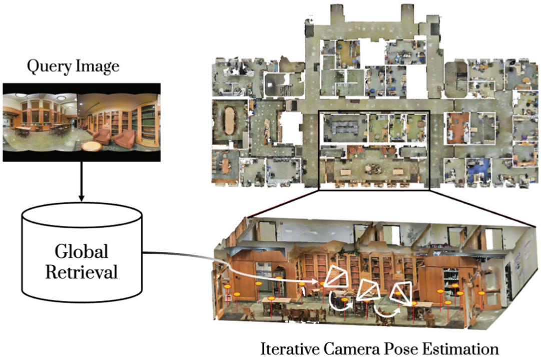


Fig. 1. The overview of our method. Given a query image, our approach initiates the global image retrieval phase, estimating several potential poses. Utilizing these estimated poses, we employ an iterative process to match the omnidirectional image rendered by the proposed DP-NeRF and the target image to optimize the estimated camera pose, refining and enhancing the accuracy of the localization process. This demo is performed on Stanford 2D-3D-S dataset [3].

However, existing panorama datasets are limited by the number of available omnidirectional images, resulting in a sparse distribution and insufficient diversity compared to the well-distributed dataset. This limitation adversely affects the accuracy of the image retrieval phase, which draws our attention. Neural

Radiance Fields (NeRF) [32], known for creating detailed 3D scene representations, is widely used in 3D modeling and has excelled in generating photorealistic new views of real-world scenes. By mapping 3D spatial information to color and density, and using ray tracing for image consistency, NeRF is ideal for enhancing sparse panorama datasets with synthetic, lifelike images.

Yet, the sparse nature of panorama datasets hinders conventional NeRF training, leading to suboptimal performance in the trained model. To address this issue, we introduce Depth-supervised Panorama NeRF (DP-NeRF), a novel approach tailored for training on a single omnidirectional image with depth. **Importantly, unlike DS-NeRF [10], which makes the density distribution on a sampling ray close to unimodal, we promote the density to match the real density. For sampling points where the distance from the camera is less than the measured ground truth depth, we impose a requirement that their density should be 0.** This strategic design addresses the inherent issues associated with sparse inputs, effectively resolving NeRF's inability to achieve comprehensive reconstruction under such conditions. The integration of depth supervision in DP-NeRF serves as a key enhancement to generate more accurate and robust reconstructions, particularly in scenarios characterized by limited and sparse panorama data.

Moreover, diverging from conventional pose optimization methods like feature matching, we employ an iterative algorithm based on our DP-NeRF to refine the rough results. Upon obtaining several potential matches from image retrieval, we compute the appearance difference between the panorama rendered from the DP-NeRF and the observed image. Subsequently, we leverage DP-NeRF to backpropagate the gradient of these residuals, employing gradient-descent optimization to iteratively refine the camera pose. This novel form of pose optimization stands out for its ability to navigate beyond the constraints imposed by a small dataset, contributing to a notable enhancement in localization accuracy. By incorporating the iterative refinement facilitated by DP-NeRF, our algorithm showcases increased resilience and adaptability, particularly in scenarios where traditional methods struggle with limited dataset size and diversity.

To summarize, our main contributions are as follows:

- We propose NAOL, a novel localization algorithm specifically tailored for panorama images. To tackle with the sparsity of omnidirectional images, we design DP-NeRF for training on a single omnidirectional image with depth to enrich the dataset.
- The proposed DP-NeRF enables our method to further refine the coarse localization based on the visual-based algorithm.
- The proposed NAOL excels in effectively performing better localization in sparse panorama dataset. Our algorithm showcases increased resilience and adaptability, particularly in scenarios where traditional methods may encounter limitations associated with dataset size and diversity.

2 Related Works

2.1 Visual Localization

Visual localization is the task of estimating the camera pose from an image. With the emergence of deep neural networks, learning-based methods emerge as prevalent solutions. One effective solution in this field is image retrieval [1,2,4,15,41,48], which predicts the image position by matching geographic markers with the most similar images from the database. This approach utilizes learned image features produced by a feature extraction backbone [2], which is further improved by spatial reranking [15] or feature reweighting [20]. In another direction, to learn the entire localization pipeline end-to-end, PoseNet [19] proposes to use a convolution neural network to regress the absolute camera pose from the input image directly and is later improved upon and studied in [6,45]. Direct pose regression's performance is closer to image retrieval and generally falls behind structure-based approaches. One common structure-based solution [11,35,36,39] is to leverage the 2D-3D correspondences between key points in the query image and a prebuilt 3D map. The camera pose is then calculated through optimization with RANSAC. Another common solution is scene coordinate regression. Shotton et al. [38] first propose to regress the scene coordinates using Random Forests. The other related works [5,25,46] leverage the convolutional neural network to predict scene coordinates directly from an input image and obtain the camera pose using the predicted correspondences.

2.2 Omnidirectional Localization

There have been relatively few techniques for visual localization on omnidirectional images due to the unique visual distortion [12,17,49] caused by the spherical projection equation. Conventional techniques solve omnidirectional image-based localization without using deep learning methodologies. For example, Caruso et al. [7] propose the calculation of direct image match for the whole omnidirectional model and corresponding omnidirectional simultaneous localization. In recent years, the advent of deep learning has spurred innovative solutions to tackle visual localization challenges with omnidirectional images. Zhang et al. [47] contribute by training neural networks that learn rotation equivariant features to process omnidirectional images effectively. Spherenet [9] proposes a deep learning framework that explicitly encodes distortion invariance into convolutional neural networks, allowing for effective learning of omnidirectional image representations. Another approach, PICCOLO [21], proposes a point cloud-centric omnidirectional localization algorithm that bypasses spherical distortion. However, it is noteworthy that PICCOLO's applicability is constrained by its reliance on high-quality dense point clouds, limiting its effectiveness in general scenarios. The growing body of work in recent years reflects an increasing interest and effort in developing effective deep learning-based solutions for omnidirectional visual localization.

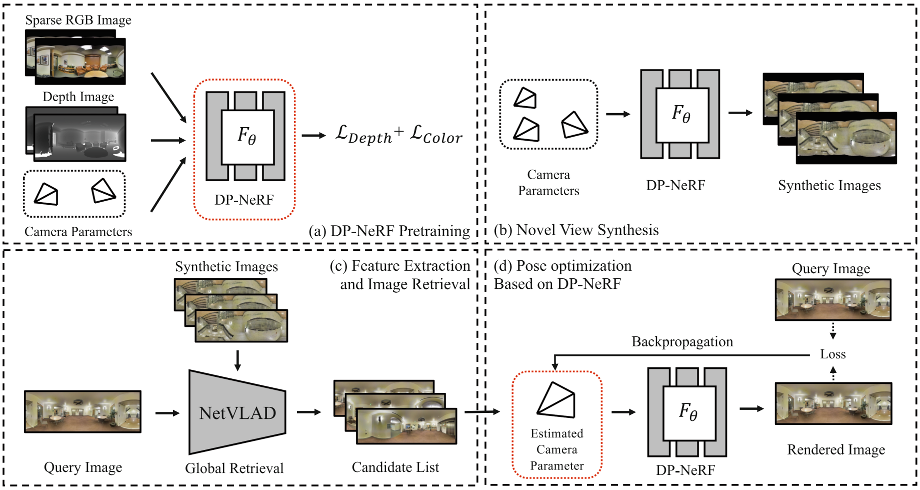


Fig. 2. The pipeline of NAOL: (a) First, we train the proposed DP-NeRF with real images. (b) With trained DP-NeRF, novel view synthesis is performed to generate additional synthetic images. (c) Then, real and synthetic images are combined to train a coarse pose estimator. (d) Given potential poses, fine pose estimator further improves the localization accuracy by iterative optimization. Only the parameters in the red box are optimized and updated during the training. (Color figure online)

2.3 Novel View Synthesis in Localization

Novel View Synthesis (NVS) [40], generating new camera views from scene images, is a longstanding challenge in computer graphics. Recent progress in NVS has demonstrated great success in achieving realistic view synthesis [29,30], enhancing accuracy for visual localization methods. Recently, NeRF has shown excellent performance in view synthesis by using a neural network to implicitly learn a static 3D scene for arbitrary new view synthesis of complex scenes. Direct-PoseNet [8] achieves photo-realistic pose prediction by using an additional loss between the query image and the NeRF synthetic image for camera pose regression. PoseGan [28] performs joint learning of pose regression and view synthesis, improving localization accuracy. NeRF-Loc [27] utilizes a pre-trained NeRF model along with camera views as inputs to produce labeled and oriented 3D bounding boxes for objects as outputs. LocoNeRF [33] develops the local SfM method that employs NERF as a substitute for image databases to serve as reference images. iNeRF [26] presents gradient-based optimization to realize pose estimates via the differentiability of NeRF but is limited to pose estimation of small objects due to the ray sampling strategy. Instead, our approach preserves the viewpoint's global information, extending iNeRF-based iterative methods for large scenes with sparse panoramas. Besides, unlike OmniNeRF [16], which synthesizes hundreds of maps from depth maps to train a NeRF, we use the depth maps directly for training, resulting in faster and better performance than OmniNeRF. DS-NeRF utilizes sparse deep supervision to regularize the geometric information and achieve faster training. Unlike DS-NeRF, which makes the density distribution on a sampling ray close to unimodal, we promote the density to match the real density.

3 Methodology

We now introduce our method, illustrated in Fig. 2. Our localization method comprises two interconnected stages. Prior to these two localization stages, a crucial preliminary step is taken: *DP-NeRF Pretraining* to augment the panorama dataset. This preparatory process ensures that our dataset is enriched with synthetic yet realistic images, setting the stage for more robust and accurate localization in subsequent stages. After dataset augmentation, in the first stage, termed *Feature Extraction and Image Retrieval*, we leverage a global image retrieval algorithm for coarse pose estimation. Subsequently, in the second stage, denoted as *Pose Optimization Based on DP-NeRF*, we deploy an iterative algorithm based on our DP-NeRF. This iterative refinement serves to intricately enhance the precision of position estimation, resulting in a significant improvement in the overall accuracy of the localization process.

3.1 DP-NeRF Pretraining

Traditional NeRF training, as presented in [32], revolves around minimizing the photometric loss across a sampled set of rays. Given a 3D point $x \in \mathbb{R}^3$ and a viewing direction $d \in \mathbb{R}^3$, we use common NeRF to mimic the implicit representation F of the density σ and RGB color c as $F(x, d) = (\sigma, c)$. However, when dealing with image localization, training a traditional NeRF becomes challenging due to the inherent sparsity of the data. Drawing inspiration from DS-NeRF [10], we use both RGB and depth information as training supervision. Consequently, the loss function of DP-NeRF comprises two parts:

$$\mathcal{L} = \mathcal{L}_{Color} + \lambda_D \mathcal{L}_{Depth}, \tag{1}$$

where $\mathcal{L}_{Color}$ denotes the color supervision loss, $\mathcal{L}_{Depth}$ is the depth supervision loss, and λ_D is the hyper-parameter to balance the color and depth supervision.

As for RGB-color supervision, the NeRF model samples the implicit radiance field at coordinates along each ray r to approximate the RGB-color $\hat{C}(r)$:

$$\hat{C}(r) = \int_{t_n}^{t_f} T(t)\sigma(t)c(t)dt, \tag{2}$$

where t parameterizes the ray as $r(t) = o + td$ with o denoting the origin and we utilize $T(t) = exp(-\int_{t_n}^{t} \sigma(s)ds)$ to integrate the differential density $\sigma(s)$ over points between t_n to t in order to calculate the occlusions. Then, the rendered color is reconstructed by:

$$\mathcal{L}_{Color} = \sum_{r \in \mathbb{R}(P)} \|\hat{C}(r) - C(r)\|_2^2, \tag{3}$$

where $\mathbb{R}(P)$ represents the set of rays rendered by NeRF and $C(r)$ is the ground truth color of ray r.

Then, as for depth image supervision, we use the NeRF model to render depth images from specific poses in a similar manner to rendering RGB color pixels, and the formulation from the original NeRF paper. Given a ray parameterized as $r(t) = o + td$:

$$\hat{D}(r) = \int_{t_n}^{t_f} T(t)\sigma(t)tdt, \tag{4}$$

where $\hat{D}(r)$ is the predicted depth along the axis of ray r. Therefore, the depth image supervision loss can be formulated as:

$$\mathcal{L}_{Depth} = \sum_{r \in \mathbb{R}(P)} |\hat{D}(r) - D(r)|^2, \tag{5}$$

where the $D(r)$ is the ground-truth depth image.

In contrast to DS-NeRF, which necessitates unimodal density sampling results for each ray, our approach only requires that the predicted depth agrees with ground truth. For sampling points where the distance from the camera is less than the measured ground truth depth, we impose a requirement that their density should be 0. Without generating hundreds of maps from depth map to train a NeRF, the proposed DP-NeRF uses the depth maps directly for training. The above design facilitates our method of faster and high-quality training.

We train our DP-NeRF for each training image and then use all DP-NeRF to represent the whole scene. After that, we can augment the panorama datasets with synthetic images generated from the trained DP-NeRF.

3.2 Stage One: Feature Extraction and Image Retrieval

Our coarse pose estimation leverages the original NetVLAD [2] network architecture, which employs the VLAD [18] approach to generate the image representation by aggregating the hidden feature maps embedded from a pretrained CNN. Formally, given an image I, the base architecture is denoted as $f : I \rightarrow \mathbb{R}^{H \times W \times D}$, producing an $H \times W \times D$ dimensional output map. This map comprises a set of D-dimensional features extracted at $H \times W$ spatial locations. The NetVLAD architecture clusters these D-dimensional features into a $K \times D$ dimensional matrix by adding the residuals between each feature $\mathbf{x}_i \in \mathbb{R}^D$ and K learned soft allocation weighted clustering centers. Specifically, given N D-dimensional local image descriptors, the output image representation is $K \times D$-dimensional. The VLAD aggregation layer $f_{\text{VLAD}} : \mathbb{R}^{N \times D} \rightarrow \mathbb{R}^{K \times D}$ is given by

$$f_{\text{VLAD}}(j, k) = \sum_{i=1}^{N} a_k\left(\mathbf{x}_i\right)\left(x_i(j) - c_k(j)\right), \tag{6}$$

where $x_i(j)$ and $\mathbf{c}_k(j)$ are the j-th element of the i-th descriptor and k-th cluster center, and a_k denotes the soft-assignment function. After VLAD aggregation, the resulting matrix is normalized column by column, converted into a single vector, L2 global normalization is performed on it, and finally PCA [13] with whitening is applied (learned on the training set).

Given a query image, we use this original NetVLAD descriptor to retrieve the top-k most likely matches. Then, we adopt several strategies to generate more accurate top-k estimated positions:

(1) We perform weighted averages for top-3, top-5, and top-10 positions, assigning weights based on the similarity scores from the NetVLAD descriptor.
(2) We cluster top-5 and top-10 positions to obtain cluster centers.
(3) Based on the distance from the cluster center to the top-1 position, we reset the weight of the top-1 position and re-calculate the weighted average.

3.3 Stage Two: Pose Optimization Based on DP-NeRF

In this stage, we further improve the localization accuracy based on the potential poses estimated by stage one. Inspired by iNeRF [26], we update the estimated pose by gradient-based optimization. As is shown in Subsect. 3.1, for each training image, we train one DP-NeRF model. Given a coarse pose T_0 consisting of viewing point and direction (x_0, d_0) of an observed image I from stage one, we choose the DP-NeRF F_θ nearest to the coarse pose T_0 for optimization. We freeze the parameters θ in the selected DP-NeRF F_θ, and then we need to determine the camera location $\hat{T}$ of the targeted image I. The optimization problem can be formulated as follows:

$$\hat{T} = \underset{T \in SE(3)}{\arg\min}\, \mathcal{L}(F_\theta(T), I), \tag{7}$$

where $\mathcal{L}(F_\theta(T), I)$ is L2-Loss of pixel difference between $F_\theta(T)$, the panorama rendered from the DP-NeRF F_θ at each step, and the observed image I. Similar to iNeRF, we adopted a gradient based process for optimizing $SE(3)$ and used relative transformations to ensure the estimated attitude $T = T_i$ during the optimization process continuously located on the $SE(3)$ manifold, where T_i is the estimated attitude at optimization step i.

We find that the global spatial distribution and semantic relation are important for pose estimation. However, the rays sampling strategies in iNeRF lead to incomplete information for optimization. Thus, we adopt a downsampling strategy to render low-resolution synthetic images. Different from ray sampling strategies, which sample a set of pixels from a rendered image to update the pose, we define a lower resolution for NeRF to render the image and select all the pixels for backpropagation. See detailed experiment analysis in ablation study c. Let the reduction factor of the resolution be r, and the original height and width of the high-resolution image be H and W. Thus, the reduced resolution of the image rendered by NeRF is $H' \times W'$, where $H' = r \times H$ and $W' = r \times W$. In the meantime, the resolution of the ground truth image is downsampled as the same resolution of $H' \times W'$ for calculating loss function $\mathcal{L}$. In this way, we not only save the global spatial information for omnidirectional localization but also reduce the memory storage to speed up the optimization. Obtaining potential coarse poses for the observed image I from stage one, we do the optimization process for each coarse pose sequentially to calculate the final estimated poses. We select the estimated pose with the smallest loss $\mathcal{L}(F_\theta(T), I)$ as our final fine estimated camera pose.

4 Experiments

4.1 Dataset and Implementation Details

Dataset. Stanford2D-3D-S [3] is a benchmark that has been widely recognized for its significant role in omnidirectional localization. It has become a standard for assessing the performance of various localization methods, and it is often used as a reference point in the field of computer vision. This benchmark provides a comprehensive set of data, concluding both panoramic images and depth maps, and evaluation metrics. Most importantly, **it consists of six large-scale indoor areas, covering over 6,000 m^2, which provides various localization scenes meanwhile the sparse distribution of panoramic images makes localization task realistically challenging.** In contrast, other datasets typically focus on small scenes with well-distributed images, which are not realistic in practice. Considering the unique challenges presented by this dataset and its provision of depth maps, a feature not commonly found in other datasets for omnidirectional localization, we utilize the Stanford2D-3D-S dataset to evaluate the localization performance of our method in comparison to existing approaches.

DP-NeRF Training. We train DP-NeRF with batch-size 1400 and 30k training steps in total. We adopt the Adam optimizer with an initial learning rate of 5e-3, which is decayed by 10 times at step 20k. The number of sampling points for coarse MLP is 64 and 128 for fine MLP. Other parameters follow the standard NeRF [32] training. Each DP-NeRF training session takes an average of 15 min. During synthesis, ten random positions within the DP-NeRF's visible range are taken as camera positions, and it takes less than 1 min to synthesize a panorama image.

Stage One Training. Real and synthetic datasets are combined together for training. We resize all panorama images to 1024 by 512 pixels and center crop them to 1024 by 340 pixels to avoid the adverse effect of black edges of images. All hyper-parameters are the same as NetVLAD [2]. Top-10 retrieval results are recorded for fine pose regression.

Stage Two Training. The total number of optimization steps is 150, and the initial learning rate is 0.1, which is multiplied by 0.2 at step 30 and step 60, respectively. As for the downsampling strategy, we set the downsampling factor as 64. Other parameters follow the iNeRF [26].

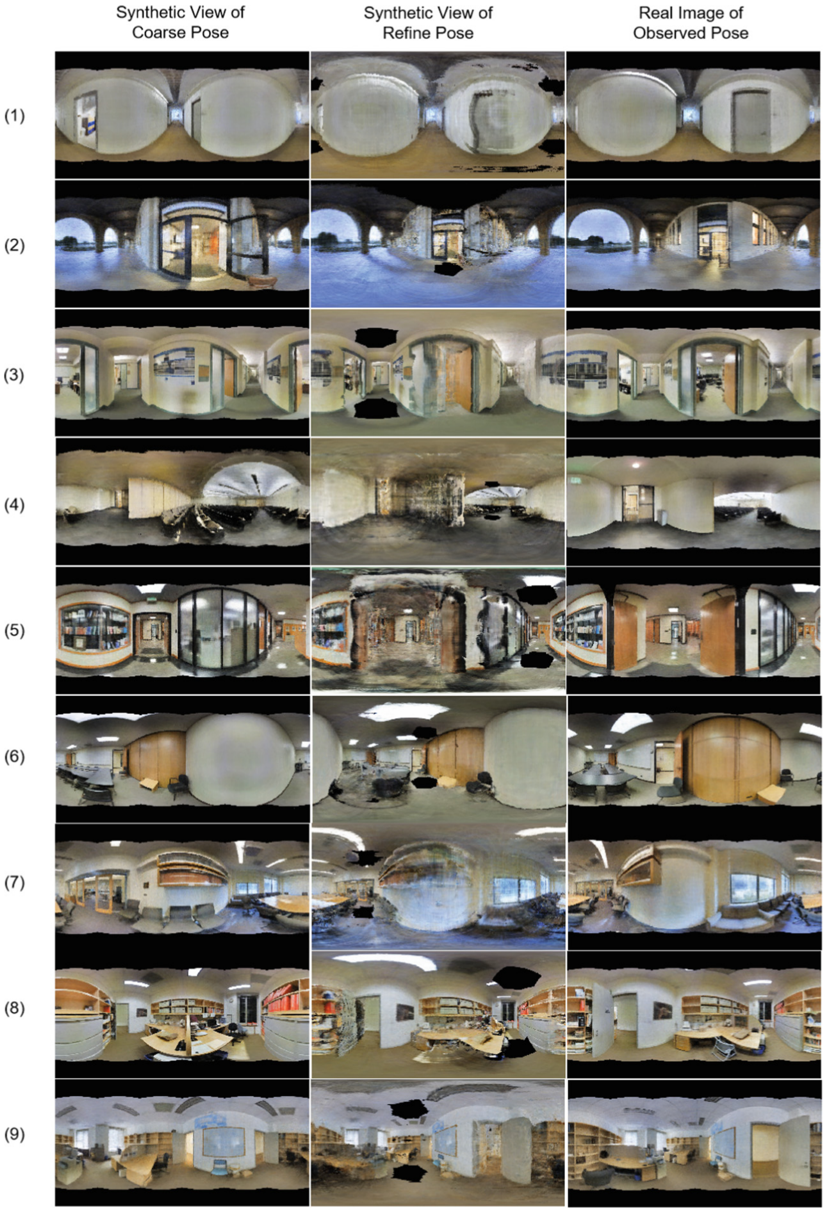

Synthetic View of Coarse Pose
Synthetic View of Refine Pose
Real Image of Observed Pose
(1)
(2)
(3)
(4)
(5)
(6)
(7)
(8)
(9)

Fig. 3. Qualitative results of our omnidirectional localization. *Recommended zoom in for observation.* We can observe that (1) our trained DP-NeRF can generate synthetic but realistic images, assisting our method for excellent localization accuracy; (2) by iteratively optimizing from the coarse pose, we can get the fine pose which is much closer to the pose of the observed image.

Table 1. Quantitative results of omnidirectional localization evaluated on Stanford2D-3D-S dataset. Note that the results are averaged across **all six large-scale scenes** in Stanford2D-3D-S.

Method	Translation Error (m)	Rotation Error (°)
NetVLAD [2]	7.28	N/A
SFRS [14]	7.79	N/A
PoseNet [19]	4.59	40.6
SphereNet [9]	3.86	38.5
Zhang et al. [47]	3.07	18.2
PICCOLO [22]	5.45	109.1
CPO [24]	5.00	108.7
LDL [23]	2.68	53.6
Ours (Coarse)	**6.73**	N/A
Ours (Fine)	**1.28**	**8.55**

Table 2. Quantitative results of omnidirectional localization evaluated on different areas of Stanford2D-3D-S dataset. Note that *T-Error* represents Translation Error and *R-Error* represents Rotation Error.

	Area1	Area2	Area3	Area4	Area5	Area6	Total
Area (m2)	965	1100	450	870	1700	935	6020
Volume (m3)	2850	3065	1215	2780	5370	2670	17,360
Scene Num.	190	299	83	258	373	208	1411
Time Cost of Data Aug. (h)	47	74	21	64	93	52	351
Ours Coarse T-Error (m)	10.06	5.97	4.29	3.59	8.41	17.14	6.73
Ours Fine T-Error (m)	1.89	0.65	0.50	0.91	1.42	1.55	1.28
Ours R-Error (°)	13.32	5.07	34.52	10.33	8.67	5.88	8.55

4.2 Results

In this section, we present a comparative analysis of the traditional localization methods and our proposed method in the context of the omnidirectional localization task, particularly focusing on scenarios characterized by sparse data distribution. We compare our method with eight state-of-the-art methods, namely NetVLAD [2], SFRS [14], PoseNet [19], SphereNet [9], Zhang et al. [47], PICCOLO [22], CPO [24], and LDL [23].

Quantitative Results. The quantitative localization results performed on the Stanford2D-3D-S dataset are summarized in Table 1 and Table 2. Initially, our method's coarse location result demonstrates an improvement in performance over the original NetVLAD, facilitated by the incorporation of DP-NeRF synthetic, but it still falls below that of other competing methods. Subsequently, we refine the coarse pose estimation results through the algorithm detailed in Subsect. 3.3. This refinement process results in a significant performance boost, surpassing other methods. Our method achieves lower translation and rotation errors, outperforming all existing methods by a large margin.

NAOL addresses the sparse panorama data challenges in the Stanford2D-3D-S, which traditional feature-based localization algorithms struggle with. Unlike these approaches, DP-NeRF generates additional images and iteratively refines pose estimation, effectively managing sparse data limitations. This realistic approach enhances omnidirectional localization in data-scarce environments.

Qualitative Results As shown in Fig. 3, we present the qualitative results of omnidirectional localization to demonstrate the effectiveness of our method. We can observe that (1) our trained DP-NeRF demonstrates the capability to generate synthetic yet realistic images, assisting our method in bolstering the performance; (2) by iteratively optimizing from the coarse pose, we can achieve a refined pose that is significantly closer to the pose of the observed image. These visualizations provide compelling evidence of our method's effectiveness in handling omnidirectional localization tasks, showcasing its ability to generate realistic synthetic data and iteratively optimize poses for improved accuracy.

Table 3. Ablation study on translation error (m) depending on panorama image or not and adding additional synthetic dataset or not. Localization results are evaluated on **all six areas** of Stanford2D-3D-S dataset.

Method	Perspective	Panorama	Panorama + DP-NeRF
NetVLAD [2]	13.01	7.28	6.73
SFRS [14]	13.20	7.79	5.76

Table 4. Ablation study on the effectiveness of different sampling strategies. Localization results are evaluated on Area 4 (≈1,000 m^2, 260 test images) of Stanford2D-3D-S.

Sampling Strategy	T-error (m)	R-error (°)
Random Sampling	1.79	13.61
Interest Point	2.12	18.67
Interest Region	1.42	12.12
Downsamping	**0.91**	**10.33**

Table 5. Ablation study on the time of pose optimization and translation error of different downsampling factor r on Area4 ($\approx$1,000 m^2, 260 test images) of Stanford2D-3D-S dataset. Note that considering the real world applicability of our method, we choose the **bold** scheme in the paper.

Downsampling Factor	16	32	**64**	128	256
Optimization Time (h)	52.50	17.50	**5.25**	2.63	0.88
T-error (m)	0.82	0.84	**0.91**	1.65	3.76

4.3 Ablation Study and Discussion

Panorama Image. To analyze the effectiveness of panorama images, we compare the coarse pose estimation module with a variant that uses the perspective images in place of the panorama images. We conduct this experiment on NetVLAD [2] and SFRS [14] to investigate the universal significance of panorama images. Both training parameters and training setup for these two methods are the same. Results are shown in Table 3. As expected, coarse pose estimation with panorama images demonstrates better performance, achieving at least $4.09m$ better than perspective images.

DP-NeRF Synthesis. We compare the improvement of localization accuracy on the coarse pose estimation module to ablate the effect of the additional DP-NeRF synthetic dataset. For a fair comparison, synthetic images are only used in the training stage. As shown in Table 3, with the additional synthetic dataset, both NetVLAD and SFRS demonstrate relatively better performance than only using the raw dataset.

Sampling Strategy. We compare different ray sampling strategies in our DP-NeRF with the proposed downsampling strategy to show the effectiveness of pose optimization in our method. As shown in Table 4, the downsampling strategy lower the translation and rotation error, showing a significant performance gain over other strategies.

Pose Optimization Time and Translation Error with Different Downsampling Factor. In our research, we analyze the impact of the downsampling factor r on our algorithm's performance, in which downsamples the rendered image and ground truth. Table 5 reveals that a lower r increases localization accuracy, but also demands more computation, presenting a trade-off between accuracy and runtime. After comprehensive evaluation, we choose a reduction factor of 64 as optimal, striking a balance between computational efficiency and localization accuracy. This selection effectively reduces the algorithm's complexity while maintaining sufficient accuracy, enhancing overall efficiency.

Discussion: Why do we choose NetVLAD as our feature extractor? The primary role of the feature extractor is to provide coarse locations for DP-NeRF to further improve in the second stage. Benefiting from the success of NeRFs, the DP-NeRF can be well-optimized as long as the coarse position is within the visible range. The classic, simple, and effective method, NetVLAD, can provide such coarse localization. Thus, we adopt it as our coarse localization method.

5 Conclusion

In this work, we propose NAOL, an efficient visual localization algorithm intricately crafted for panorama images. To overcome the challenges posed by the limited size and sparsity of panorama dataset, we design the DP-NeRF model and leverage it to augment the existing datasets. This augmentation proves instrumental in addressing the inherent limitations of small-sized and sparse datasets. Besides, the iterative optimization of DP-NeRF enhance the localization performance, both on translation error and rotation error. Our experiments corroborate that the proposed NAOL excels in effectively performing visual localization, particularly in scenarios characterized by sparse panorama datasets. This more realistic approach better simulates real-world conditions, which markedly enhance omnidirectional localization accuracy in data-scarce environment.

Acknowledgments. The work was supported by the Wafer-scale silicon optical switching interconnected on-chip computing system (2022YFB2804100) and the National Natural Science Foundation of China (20211710187).

References

1. Ali-bey, A., Chaib-draa, B., Giguère, P.: Mixvpr: feature mixing for visual place recognition. In: WACV, pp. 2997–3006. IEEE (2023)
2. Arandjelovic, R., Gronát, P., Torii, A., Pajdla, T., Sivic, J.: Netvlad: CNN architecture for weakly supervised place recognition. In: CVPR, pp. 5297–5307. IEEE Computer Society (2016)
3. Armeni, I., Sax, S., Zamir, A.R., Savarese, S.: Joint 2d-3d-semantic data for indoor scene understanding. CoRR arxiv:1702.01105 (2017)
4. Berton, G.M., Masone, C., Caputo, B.: Rethinking visual geo-localization for large-scale applications. In: CVPR, pp. 4868–4878. IEEE (2022)
5. Brachmann, E., Krull, A., Nowozin, S., Shotton, J., Michel, F., Gumhold, S., Rother, C.: DSAC - differentiable RANSAC for camera localization. In: CVPR, pp. 2492–2500. IEEE Computer Society (2017)
6. Brahmbhatt, S., Gu, J., Kim, K., Hays, J., Kautz, J.: Geometry-aware learning of maps for camera localization. In: CVPR, pp. 2616–2625. Computer Vision Foundation/IEEE Computer Society (2018)
7. Caruso, D., Engel, J., Cremers, D.: Large-scale direct SLAM for omnidirectional cameras. In: IROS, pp. 141–148. IEEE (2015)
8. Chen, S., Wang, Z., Prisacariu, V.: Direct-posenet: absolute pose regression with photometric consistency. In: 3DV, pp. 1175–1185. IEEE (2021)

9. Coors, B., Condurache, A.P., Geiger, A.: SphereNet: learning spherical representations for detection and classification in omnidirectional images. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11213, pp. 525–541. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01240-3_32
10. Deng, K., Liu, A., Zhu, J., Ramanan, D.: Depth-supervised nerf: fewer views and faster training for free, pp. 12872–12881 (2022)
11. Dusmanu, M., et al.: D2-net: a trainable CNN for joint description and detection of local features, pp. 8092–8101 (2019)
12. Frossard, P., Khasanova, R.: Graph-based classification of omnidirectional images. In: ICCV Workshops, pp. 860–869. IEEE Computer Society (2017)
13. Pearson, K.: Liii. on lines and planes of closest fit to systems of points in space. Lond. Edinburgh Dublin Phil. Maga. J. Sci. **2**(11), 559–572 (1901). https://doi.org/10.1080/14786440109462720
14. Ge, Y., Wang, H., Zhu, F., Zhao, R., Li, H.: Self-supervising fine-grained region similarities for large-scale image localization. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12349, pp. 369–386. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58548-8_22
15. Hausler, S., Garg, S., Xu, M., Milford, M., Fischer, T.: Patch-netvlad: multi-scale fusion of locally-global descriptors for place recognition. In: CVPR, pp. 14141–14152. Computer Vision Foundation/IEEE (2021)
16. Hsu, C., Sun, C., Chen, H.: Moving in a 360 world: Synthesizing panoramic parallaxes from a single panorama. CoRR arxiv:2106.10859 (2021)
17. Jayasuriya, M., Ranasinghe, R., Dissanayake, G.: Active perception for outdoor localisation with an omnidirectional camera. In: IROS, pp. 4567–4574. IEEE (2020)
18. Jégou, H., Douze, M., Schmid, C., Pérez, P.: Aggregating local descriptors into a compact image representation. In: CVPR, pp. 3304–3311. IEEE Computer Society (2010)
19. Kendall, A., Grimes, M., Cipolla, R.: Posenet: a convolutional network for real-time 6-dof camera relocalization. In: ICCV, pp. 2938–2946. IEEE Computer Society (2015)
20. Kim, H.J., Dunn, E., Frahm, J.: Learned contextual feature reweighting for image geo-localization. In: CVPR, pp. 3251–3260. IEEE Computer Society (2017)
21. Kim, J., Choi, C., Jang, H., Kim, Y.M.: PICCOLO: point cloud-centric omnidirectional localization. In: ICCV, pp. 3293–3303. IEEE (2021)
22. Kim, J., Choi, C., Jang, H., Kim, Y.M.: Piccolo: point cloud-centric omnidirectional localization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 3313–3323 (2021)
23. Kim, J., Choi, C., Jang, H., Kim, Y.M.: Ldl: Line distance functions for panoramic localization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 17882–17892 (October 2023)
24. Kim, J., Jang, H., Choi, C., Kim, Y.M.: Cpo: change robust panorama to point cloud localization. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 176–192 (2022)
25. Li, X., Wang, S., Zhao, Y., Verbeek, J., Kannala, J.: Hierarchical scene coordinate classification and regression for visual localization. In: CVPR, pp. 11980–11989. Computer Vision Foundation/IEEE (2020)
26. Lin, Y., Florence, P., Barron, J.T., Rodriguez, A., Isola, P., Lin, T.: inerf: inverting neural radiance fields for pose estimation. In: IROS, pp. 1323–1330. IEEE (2021)
27. Liu, J., Nie, Q., Liu, Y., Wang, C.: Nerf-loc: visual localization with conditional neural radiance field. In: ICRA, pp. 9385–9392. IEEE (2023)

28. Liu, K., Li, Q., Qiu, G.: Posegan: a pose-to-image translation framework for camera localization. CoRR arxiv:2006.12712 (2020)
29. Liu, L., Gu, J., Lin, K.Z., Chua, T., Theobalt, C.: Neural sparse voxel fields. In: NeurIPS (2020)
30. Lombardi, S., Simon, T., Saragih, J.M., Schwartz, G., Lehrmann, A.M., Sheikh, Y.: Neural volumes: learning dynamic renderable volumes from images. ACM Trans. Graph. **38**(4), 65:1–65:14 (2019)
31. Middelberg, S., Sattler, T., Untzelmann, O., Kobbelt, L.: Scalable 6-DOF localization on mobile devices. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8690, pp. 268–283. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10605-2_18
32. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: representing scenes as neural radiance fields for view synthesis. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12346, pp. 405–421. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58452-8_24
33. Nenashev, A., Kurenkov, M., Potapov, A., Zhura, I., Katerishich, M., Tsetserukou, D.: Loconerf: a nerf-based approach for local structure from motion for precise localization. In: SMC, pp. 641–646. IEEE (2023)
34. Piasco, N., Sidibé, D., Demonceaux, C., Gouet-Brunet, V.: A survey on visual-based localization: on the benefit of heterogeneous data. Pattern Recognit. **74**, 90–109 (2018)
35. Sarlin, P., Cadena, C., Siegwart, R., Dymczyk, M.: From coarse to fine: robust hierarchical localization at large scale. In: CVPR, pp. 12716–12725. Computer Vision Foundation/IEEE (2019)
36. Sarlin, P., et al.: Back to the feature: learning robust camera localization from pixels to pose. In: CVPR, pp. 3247–3257. Computer Vision Foundation/IEEE (2021)
37. Sattler, T., et al.: Benchmarking 6dof outdoor visual localization in changing conditions. In: CVPR, pp. 8601–8610. Computer Vision Foundation/IEEE Computer Society (2018)
38. Shotton, J., Glocker, B., Zach, C., Izadi, S., Criminisi, A., Fitzgibbon, A.W.: Scene coordinate regression forests for camera relocalization in RGB-D images. In: CVPR, pp. 2930–2937. IEEE Computer Society (2013)
39. Taira, H., et al.: Inloc: indoor visual localization with dense matching and view synthesis. In: CVPR, pp. 7199–7209. Computer Vision Foundation/IEEE Computer Society (2018)
40. Tewari, A., et al.: State of the art on neural rendering. Comput. Graph. Forum **39**(2), 701–727 (2020)
41. Torii, A., Arandjelovic, R., Sivic, J., Okutomi, M., Pajdla, T.: 24/7 place recognition by view synthesis. In: CVPR, pp. 1808–1817. IEEE Computer Society (2015)
42. Uy, M.A., Lee, G.H.: Pointnetvlad: deep point cloud based retrieval for large-scale place recognition. In: CVPR, pp. 4470–4479. Computer Vision Foundation/IEEE Computer Society (2018)
43. Walch, F., Hazirbas, C., Leal-Taixé, L., Sattler, T., Hilsenbeck, S., Cremers, D.: Image-based localization using lstms for structured feature correlation. In: ICCV, pp. 627–637. IEEE Computer Society (2017)
44. Wald, J., Sattler, T., Golodetz, S., Cavallari, T., Tombari, F.: Beyond controlled environments: 3D camera re-localization in changing indoor scenes. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12352, pp. 467–487. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58571-6_28

45. Wang, B., Chen, C., Lu, C.X., Zhao, P., Trigoni, N., Markham, A.: Atloc: attention guided camera localization. In: AAAI, pp. 10393–10401. AAAI Press (2020)
46. Yang, L., Bai, Z., Tang, C., Li, H., Furukawa, Y., Tan, P.: Sanet: scene agnostic network for camera localization. In: ICCV, pp. 42–51. IEEE (2019)
47. Zhang, C., Budvytis, I., Liwicki, S., Cipolla, R.: Rotation equivariant orientation estimation for omnidirectional localization. In: Ishikawa, H., Liu, C.-L., Pajdla, T., Shi, J. (eds.) ACCV 2020. LNCS, vol. 12625, pp. 334–350. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-69538-5_21
48. Zhang, J., Cao, Y., Wu, Q.: Vector of locally and adaptively aggregated descriptors for image feature representation. Pattern Recognit. **116**, 107952 (2021)
49. Zhao, Q., Zhu, C., Dai, F., Ma, Y., Jin, G., Zhang, Y.: Distortion-aware cnns for spherical images. In: IJCAI, pp. 1198–1204. ijcai.org (2018)

EdgeConvFormer: An Unsupervised Anomaly Detection Method for Multivariate Time Series

Jie Liu, Qilin Li(✉), Senjian An, Bradley Ezard, and Ling Li

School of Electrical Engineering, Computing and Mathematical Sciences,
Curtin University, Perth, WA 6102, Australia
`qilin.li@curtin.edu.au`

Abstract. In this paper, EdgeConvFormer is introduced as a novel approach to unsupervised anomaly detection in multivariate time series, combining the strengths of graph convolutions and Transformers within a hierarchical structure. The model utilizes Time2Vec to generate temporal embeddings and subsequently constructs dynamic graphs from point embeddings in the 2D sensor-time space, effectively extracting the most significant spatiotemporal relationships. A novel parallel sensor-specific attention mechanism has been developed to enhance the model's ability to exploit sensor-specific temporal dependencies, significantly boosting its detection capabilities. Extensive experiments have been conducted across six benchmark datasets for anomaly detection, including the expansive Exathlon dataset. The experimental results demonstrate that EdgeConvFormer outperforms state-of-the-art methods across various evaluation metrics.

Keywords: multivariate time series · anomaly detection · graph CNN · Transformer

1 Introduction

In the modern manufacturing and engineering sectors, a vast array of sensors are usually used to continuously monitor the status and behaviors of complex systems. Each sensor contributes a variate, defined as its individual time series signal, resulting in a substantial volume of multivariate time series data. The primary goal in managing these systems is to identify anomalies promptly, ensuring that potential issues are addressed in a timely manner. This proactive approach can significantly reduce ongoing maintenance expenses [1].

However, anomaly detection in multivariate time series data presents numerous challenges. Firstly, the lack of a universally accepted definition of what constitutes an anomaly makes the detection process inherently complex [2]. Furthermore, multivariate time series data often contain intricate temporal and spatial dependencies that are difficult to discern and accurately analyze [3]. The relationships within the data are also typically nonlinear and nonstationary, adding

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 367–382, 2025.
https://doi.org/10.1007/978-3-031-78128-5_24

another layer of complexity to their modeling [4]. Lastly, the process of labeling data for the training of anomaly detection models is both costly and labor-intensive, requiring specialized expertise that may not always be available. This limitation poses a significant barrier to the practical application of these models [5].

Recent advancements in deep learning have substantially enhanced anomaly detection capabilities, outperforming traditional methods such as ARMA, LOF, Isolation Forest, and OC-SVM [6,7]. Among these advancements, Transformers stand out for their efficiency in capturing long-term dependencies within sequential data through self-attention mechanisms [8–10]. However, the application of Transformers to time series analysis is impeded by several issues, such as their reliance on unsuitable sinusoidal positional encodings and a lack of inherent inductive biases, which necessitate extensive training datasets. These issues are often compounded by their limitations in dynamically capturing relationships between different sensors [11]. On the other hand, Graph Neural Networks (GNNs) have been investigated as a means to model the correlations between sensors in time series data, yet they tend to overlook long-term temporal patterns and depend on static, pre-defined graph structures. These limitations render them ineffective for analyzing multivariate time series data associated with dynamic systems [12,13].

In response to these challenges, we introduce the EdgeConvFormer, a novel architecture that seamlessly integrates dynamic graph CNNs with an advanced Transformer model, featuring a novel parallel sensor-spefic attention mechanism. By dividing the analysis of multivariate time series data into spatial and temporal correlations and tackling them through graph modeling and an attention mechanism alternately, the EdgeConvFormer markedly enhances its ability to identify anomaly patterns. The main contributions of this research are detailed as follows:

1. The development of EdgeConvFormer, a novel approach for unsupervised anomaly detection that integrates Time2Vec encoding, graph edge convolution, and parallel sensor-specific attention mechanisms. This approach enhances the detection of complex spatiotemporal patterns in multivariate time series by utilizing both global and local contexts.
2. The introduction of a tailored multi-head self-attention mechanism that processes the time series of each sensor in parallel. This allows for precise detection of sensor-specific patterns and interactions.
3. The adoption of Time2Vec for positional encoding, replacing the conventional approach. This modification allows EdgeConvFormer to better capture both periodic and aperiodic temporal patterns from the initial stages of analysis.
4. Through rigorous testing on a variety of benchmark datasets, we demonstrate that EdgeConvFormer outperforms current leading models in anomaly detection, showcasing its exceptional effectiveness and efficiency in identifying anomalies.

2 Related Work

Deep learning advancements have significantly enhanced anomaly detection capabilities, surpassing traditional techniques with the deployment of CNNs, RNNs/ LSTMs, Autoencoders, generative models, and deep one-class models [14]. Notable developments include the OmniAnomaly model [15], which combines GRUs with a VAE framework to handle diverse scenarios, and the LSTM-VAE by Park et al. [16], which merges LSTMs with VAEs to improve dynamic anomaly detection. MSCRED [17] enhances this approach by incorporating convolutional and LSTM layers with an attention mechanism, and Beat-GAN [18] uses a GAN framework to simulate normal operational data. The field has also evolved to better capture temporal dependencies, exemplified by TCN-AE [20] that integrates Temporal Convolutional Networks within an autoencoder setup.

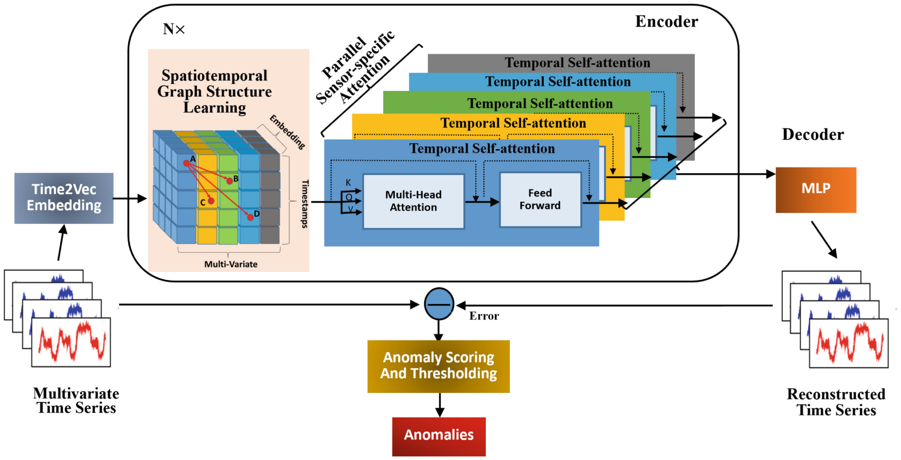
Transformer-based models are increasingly prominent for anomaly detection in multivariate time series, offering robust solutions for complex datasets [19]. The Anomaly Transformer [8] utilizes temporal patterns and a minimax strategy to distinguish anomalies, while TranAD [10] enhances detection through self-conditioning and adversarial training.

Graph Neural Networks (GNNs) have become pivotal in detecting anomalies within sensor networks by elucidating complex sensor interrelations. The MTAD-GAT model by Zhao et al. [21] uses graph attention layers to adaptively learn sensor influences, enhancing detection precision. Deng et al. [13] introduce the GDN model, which refines network topology understanding through similarities in sensor embeddings, improving anomaly identification. Further advancing this field, Chen et al. [22] develop the GTA model, which automates graph structure learning using the Gumbel-Softmax technique and influences propagation convolution to effectively model information flow and detect anomalies.

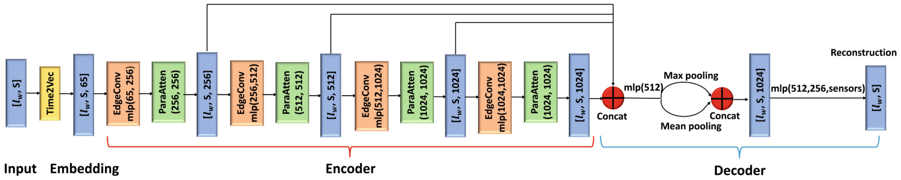
Our approach constitutes a significant advancement beyond current models by integrating the unique capabilities of GNNs and Transformers. It innovatively models both spatial and temporal correlations within multivariate time series data via a novel, modular approach, which we detail subsequently.

3 Methodology

In multivariate time series anomaly detection, given historical training data $X_{\text{train}} \in \mathbb{R}^{T_1 \times S}$, consisting of S variates (e.g. the signals from different sensors) each with T_1 timestamps, and assume this training data is free of anomalies, the objective is to determine the presence of anomalies at each timestamp t within a new, unseen test time series $X_{\text{test}} \in \mathbb{R}^{T_2 \times S}$. The idea is to train a neural network for the reconstruction of the anomaly-free data X_{train} so that its reconstruction error can be used as a quantification of the anomaly degree. The data point at any given timestamp t is denoted as x_t, where $x_t \in \mathbb{R}^{1 \times S}$ is a row vector, and $X_s \in \mathbb{R}^{T_1}$ represents the time series data for the s-th sensor. To facilitate analysis, both X_{train} and X_{test} are segmented into overlapping windows of length l_w with a stride of l_{str}.



(a) Overview of the EdgeConvFormer Model



(b) The Backbone of the EdgeConvFormer Architecture

Fig. 1. The Proposed EdgeConvFormer Architecture. (1) Input Embedding: Time2Vec extracts time patterns for each sensor. (2) Encoder: integrating EdgeConv and parallel temporal attention for each sensor (ParaAtten) in an alternative, hierarchical manner, enhancing information capture across sensors and time. EdgeConv refines spatial-temporal embeddings from neighbors, while ParaAtten focuses on temporal dynamics. (3) Decoder: An MLP layer aggregates these features, projecting them onto the sensor dimension.

3.1 Network Architecture

The proposed EdgeConvFormer is constructed as an encoder-decoder model tailored for anomaly detection in multivariate time series, as illustrated in Fig. 1 (a), and the architecture backbone is detailed in Fig. 1 (b).

Time2Vec Embedding. In our model, Time2Vec [23] is utilized to enhance the embedding process to better capture periodic and aperiodic behaviors. Each sensor's data, X_s, is transformed into a $(m+1)$-dimensional embedding using linear and sinusoidal functions. Specifically, the first dimension is linearly transformed $(\omega_0 X_s + \phi_0)$, and the remaining m dimensions $(m = 64)$ are sinusoidally transformed $(\sin(\omega_i X_s + \phi_i)$ for $1 \leq i \leq m)$.

These embeddings are aggregated into a unified structure with dimensions $[l_w, S, d_E]$, where l_w is the sliding window length, S is the number of sensors, and d_E ($m+1$) is the embedding depth. This structure feeds into the EdgeConvFormer's encoder, enhancing its ability to recognize sequences in diverse time series datasets.

Encoder. The encoder of the EdgeConvFormer model, depicted in Fig. 1 (b), consists of four layers, each integrating an EdgeConv [24] module with a parallel attention for each variate (ParaAtten) module. The EdgeConv module, designed for dynamic graph structuring in the sensor-time 2D dimension, collaborates with the ParaAtten to independently and concurrently capture long-range temporal dependencies across various sensors, thus enhancing data representation at each stage. Initiated by inputs from the Time2Vec module, this layered configuration gradually refines the features to improve the detection of spatiotemporal patterns in multivariate time series data.

EdgeConv module: In the EdgeConvFormer model, for each overlapping window, each of the four layers uses an EdgeConv module to dynamically update embeddings based on spatial-temporal relationships. Inputs to each layer l are reshaped to matrices with dimensions $[l_w \times S, d_{in}^{(l)}]$, where $d_{in}^{(0)} = 65$, the dimension of the Time2Vec output, and $d_{in}^{(l)}$ are the dimensions of the outputs from their previous layers.

At each layer, the EdgeConv module constructs a k-nearest neighbor graph, updating the embeddings $h_p^{(l)}$ for each point p by combining the difference between each point and its neighbors in the sensor-time 2D space with the point's original features. This process is described by:

$$h_p^{(l+1)} = \max_{j \in N(p)} \left(\mathrm{ReLU} \left(\Theta^{(l)} \cdot (h_j^{(l)} - h_p^{(l)}) + \Phi^{(l)} \cdot h_p^{(l)} \right) \right), \tag{1}$$

where $N(p)$ is a set of points including the k-nearest neighbors of the point p, and $\Theta^{(l)}$ and $\Phi^{(l)}$ are layer-specific transformations. This operation merges local and global contexts, emphasizing significant features through max-pooling [24].

The resulting tensor from each layer has dimensions $[l_w \times S, d_{out}^{(l)}]$ where $d_{out}^{(l)}$ are set to be 256, 512, 1024, and 1024 respectively for $l = 0, 1, 2, 3$, enhancing the encoder's ability to detect complex patterns and anomalies in the multivariate time series data.

ParaAtten Module: In the EdgeConvFormer architecture, each layer incorporates a parallel attention module (ParaAtten) for each variate, as shown in Fig. 2. This module modifies the traditional Transformer architecture to process signals from each sensor separately and in parallel, keeping the sensor and feature dimensions separate and independent. This approach contrasts with traditional Transformer models, which typically flatten these dimensions into a single combined one. This configuration is crucial for preserving the unique temporal characteristics of each sensor's data.

For each sensor s, the output from the EdgeConv module, denoted by $\widetilde{x}$, is reshaped into a matrix $\check{x}_s$ with dimension $l_w \times d_{out}^{(l)}$, as demonstrated in Fig. 2.

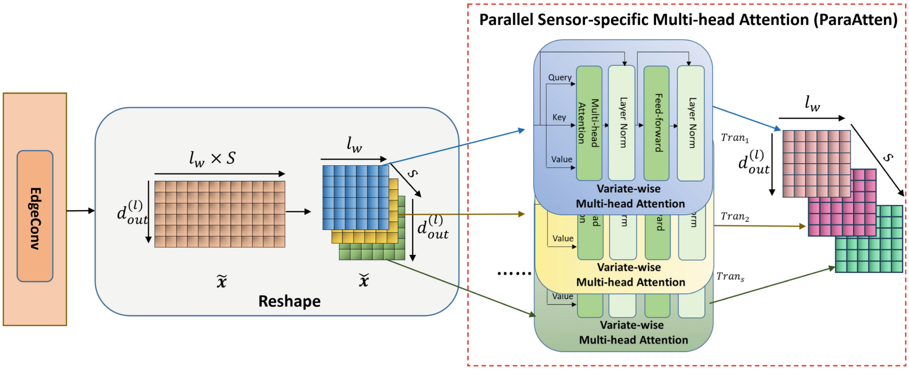


Fig. 2. ParaAtten module: parallel sensor-specific attention

Here, l_w is the window length, and $d_{out}^{(l)}$ is the dimension of features per sensor. This structure ensures independent yet parallel processing for each variate.

For each sensor s, the Transformer calculates Query (Q), Key (K), and Value (V) matrices using specific weights:

$$Q_s^{(l)} = \check{x}_s \times w^{Q_s(l)}, \quad K_s^{(l)} = \check{x}_s \times w^{K_s(l)}, \quad V_s^{(l)} = \check{x}_s \times w^{V_s(l)}. \tag{2}$$

The attention mechanism independently processes features for each sensor, employing Multi-Head Scaled Dot-Product Attention:

$$\begin{aligned} MHA_s^{(l)} &= Concat(Head_{1,s}, \dots, Head_{N,s}) \\ &\text{where each } Head_{n,s} = softmax\left(\frac{Q_{n,s}^{(l)} K_{n,s}^{(l)^T}}{\sqrt{d_k}}\right) V_{n,s}^{(l)}, \end{aligned} \tag{3}$$

This process is followed by a layer normalization and a residual connection, enhancing feature integration:

$$Tr_s^{(l)} = LayerNorm(\check{x}_s + MHA_s^{(l)}), \quad Tran_s^{(l)} = LayerNorm(Tr_s^{(l)} + FFN(Tr_s^{(l)})). \tag{4}$$

Decoder. The decoder in the EdgeConvFormer architecture, as shown in Fig. 1(b), concatenates the multi-channel outputs (across multiple sensors) from the four-layer EdgeConv-ParaAtten modules ($Tran^{(0)}$ to $Tran^{(3)}$). These outputs are reshaped into a single feature vector and fed into a Multi-Layer Perceptron (MLP) network to reconstruct the signals of sensors.

We measure the model's reconstruction accuracy using the mean squared error (MSE) between reconstructed and actual values:

$$L_{MSE} = \frac{1}{l_w} \sum_{t=0}^{l_w - 1} \|\hat{x}_t - x_t\|_2^2. \tag{5}$$

3.2 Computational Complexity

The computational complexity of the EdgeConvFormer model is primarily determined by the EdgeConv and Transformer layers. Given the number of sensors S, window size n, feature dimension d, number of layers L, and number of attention heads h, the time complexity is $\mathcal{O}(L \cdot S \cdot n \cdot d^2 \cdot h)$. This complexity can be reduced through parallel processing and distributed training methods.

4 Anomaly Detection and Evaluation

Once the model is trained based on normal sensor signals, we can identify anomalies by computing the reconstruction error Er_t^s for each sensor s at time t. This error, which represents the absolute difference between the reconstructed and the original values, is defined as:

$$Er_t^s = \|\hat{x}_t^s - x_t^s\|_1. \tag{6}$$

However, relying solely on reconstruction errors can lead to misclassifications. To increase accuracy, specialized scoring functions and thresholding techniques are employed to refine anomaly scores to reduce false positives and negatives and improve detection precision.

4.1 Anomaly Scoring Function

***Gauss_D*:** The *Gauss_D* scoring function employs a dynamic Gaussian model that fits distributions per sensor in real-time, adjusting the mean μ_t^i and variance $(\sigma_t^i)^2$ to evolving data trends, crucial for environments with variable characteristics [25].

***Gauss_D_K*:** The *Gauss_D_K* scoring applies Gaussian kernel smoothing to the *Gauss_D* model, smoothing anomaly scores across sensors to reduce noise impacts and better align anomaly detection across different sensors. This technique enhances detection accuracy while minimizing false positives [26].

4.2 Thresholding Method

***Best-F-score*:** This method identifies the threshold that maximizes the F score, crucial for achieving the best balance between precision and recall, which enhances the accuracy of anomaly detection assessments [8].

***Top-k*:** This strategy marks the k instances with the highest anomaly scores as anomalies, adapting to the actual anomaly count in the dataset [26].

***Tail-p*:** Suitable for streaming data, this method dynamically sets thresholds based on negative log probabilities across multiple sensors. It optimizes detection by testing various small tail probability values [26].

4.3 Evaluation Metrics

F_1* *score: A traditional metric that calculates the harmonic mean of precision and recall, balancing the rates of false positives and negatives [18].

Fpa_1* *score: Adjusts the F_1 score for time series by considering the detection within an anomalous segment, favoring timely over constant alerts [15].

Fc_1* *score: Integrates point-wise precision and segment-wise recall, offering a balanced assessment of detection performance [27].

F_1-$P_T R_T$* *score: Focuses on range-based anomalies, combining range-based precision and recall. This metric evaluates the detection's timing, coverage, and accuracy within each anomalous segment, reflecting its practical utility in real-world scenarios [28].

Additionally, AU-ROC and AU-PRC scores evaluate the model's discriminative performance across different thresholds, which is crucial for handling imbalanced datasets [29].

5 Experiments

Datesets. To evaluate EdgeConvFormer's performance, we use five prominent multivariate time series datasets from varied real-world applications. These include the SMD dataset from server systems [15], MSL and SMAP from NASA space missions [30], SWaT from a water treatment facility [31], and PSM from eBay's server metrics [32]. MSL, SMAP, and SMD are multi-entity datasets, containing data from various units, while SWaT and PSM are single-entity. We train individual models for each entity in the multi-entity datasets and combine the results to evaluate overall performance.

Additionally, we use the Exathlon dataset [28], comprising high-dimensional and large data, which offers a comprehensive testing ground for the proposed EdgeConvFormer, allowing us to evaluate its effectiveness with detailed, range-based metrics against the complexities and demands of real-world anomaly detection.

Baselines To evaluate anomaly detection models, we compare EdgeConvFormer against several leading models identified by Garg et al. [26], including LSTM-VAE [16], MSCRED [17], OmniAnomaly [15], BeatGAN [18], TCN-AE [33], UAE [26], and two notable Transformer-based models in anomaly detection: TranAD [10] and Anomaly Transformer [8].

5.1 Quantitative Evaluation

Comparison with models in [26] Table 1 demonstrates that our EdgeConvFormer model consistently outperforms other leading models, including UAE, LSTM-VAE, MSCRED, BeatGan, TCN-AE, and OmniAnomaly, as identified in [26]. These models were assessed across various thresholding methods and metrics using a dynamic Gaussian scoring function, except for OmniAnomaly which

Table 1. Performance comparison of EdgeConvFormer with models from [26] using dynamic Gaussian scoring across five datasets. Configurations are consistent with those reported in [26]. The best results are highlighted in bold.

Metric		F_1			Fpa_1			Fc_1			AU-ROC	AU-PRC	AvgRank
Thresholding		Top-K	Best-F_1	Tail-P	Top-K	Best-Fpa_1	Tail-P	Top-K	Best-Fc_1	Tail-P			
SMD	**OmniAnomaly**	0.4211	0.3519	0.2433	**0.7670**	0.9548	0.4471	0.5429	0.8233	0.3885	0.8067	0.4166	4.1
	LSTM-VAE	0.3977	0.2979	0.3149	0.7321	0.9546	0.8772	0.5241	0.7844	**0.6213**	0.7565	0.3816	4.9
	MSCRED	0.4111	**0.3537**	0.2653	0.6484	0.8068	0.1623	0.3989	0.5197	0.2711	0.8103	0.4174	5.5
	BeatGan	0.4056	0.2664	0.2924	0.7539	0.9592	0.6940	0.5387	0.8069	0.4372	0.8282	0.4014	4.7
	TCN-AE	**0.4477**	0.2847	**0.3625**	0.7628	0.9810	**0.8893**	**0.5818**	**0.8597**	0.5823	0.8153	**0.4268**	**2.0**
	UAE	0.4131	0.2379	0.3107	0.7545	0.9815	0.8395	0.5590	0.8459	0.5944	**0.8283**	0.3955	3.6
	EdgeConvFormer	0.4163	0.2728	0.3276	0.7546	**0.9816**	0.8678	0.5564	0.8595	0.6068	0.8039	0.3967	3.2
MSL	**OmniAnomaly**	0.2308	0.1845	0.0824	0.7319	0.9174	**0.5251**	0.3330	0.5231	0.2823	0.6476	0.2201	3.8
	LSTM-VAE	0.2192	0.0973	0.0335	0.7290	0.9467	0.1899	0.3178	0.6106	0.0352	0.5841	0.2166	5.5
	MSCRED	0.2332	0.2159	0.0339	0.7305	0.7696	0.0454	0.3433	0.5055	0.0352	0.6225	0.2556	4.5
	BeatGan	0.2737	0.1421	0.1940	0.7330	0.9359	0.1901	0.3857	0.5570	0.1940	**0.6515**	0.2516	3.2
	TCN-AE	0.2561	0.1295	0.0339	0.7426	0.9642	0.0454	0.3645	0.6608	0.0352	0.6057	0.2513	3.7
	UAE	0.1831	0.0638	0.1941	0.7156	0.9592	0.1903	0.2925	0.4596	0.1941	0.6178	0.1741	5.2
	EdgeConvFormer	**0.3307**	**0.3404**	**0.2280**	**0.7364**	**0.9669**	0.1928	**0.4329**	**0.7520**	**0.4653**	0.5997	**0.3247**	**1.6**
SMAP	**OmniAnomaly**	0.2272	**0.1881**	0.1406	0.7896	0.9494	0.2193	0.2968	0.5881	0.1934	0.6136	0.2269	4.2
	LSTM-VAE	0.1391	0.0646	0.1934	0.7405	0.9783	0.2272	0.2165	0.4868	0.1934	0.5353	0.1382	5.6
	MSCRED	0.2423	0.1487	0.1933	0.7640	0.8365	**0.2297**	0.3347	0.5921	0.1933	0.6168	0.2449	3.3
	BeatGan	0.2393	0.1106	0.1928	0.7664	0.9796	0.2280	0.3318	0.6243	0.1928	0.6124	0.2267	4.1
	TCN-AE	0.2396	0.1861	0.1927	0.7889	**0.9845**	0.2272	0.3237	0.5861	0.1927	0.6233	0.2208	3.9
	UAE	0.1678	0.1475	0.1929	0.7420	0.9728	0.2276	0.2530	0.5107	0.1929	0.5678	0.1590	5.4
	EdgeConvFormer	**0.3490**	0.1766	**0.1942**	**0.7912**	**0.9845**	0.2294	**0.4759**	**0.6877**	**0.1942**	**0.7220**	**0.3124**	**1.3**
SWaT	**OmniAnomaly**	0.2154	0.2425	0.2259	0.5505	0.6358	0.5707	0.3256	0.4437	0.3502	0.5386	0.2135	6.5
	LSTM-VAE	0.4008	0.3693	0.4036	0.5501	0.6685	0.5583	0.4752	0.5876	0.4930	0.6829	0.3417	4.5
	MSCRED	0.2324	0.1319	0.2691	0.6945	0.8359	0.8168	0.3572	0.5436	0.4033	0.6558	0.2298	5.3
	BeatGan	0.4901	0.2655	0.3652	0.7523	0.9006	0.8926	0.5905	0.6645	0.3645	0.8569	0.5049	2.8
	TCN-AE	0.3315	0.2541	0.3399	0.6926	0.8884	0.8757	0.3315	0.2541	0.3399	0.7583	0.3462	4.8
	UAE	0.4735	0.0834	0.4683	0.7821	**0.9616**	**0.9464**	0.6239	**0.7809**	0.6690	0.8441	0.4344	2.5
	EdgeConvFormer	**0.7412**	**0.7275**	**0.7325**	**0.8209**	0.8833	0.8703	**0.6966**	0.7258	**0.7169**	**0.9045**	**0.7068**	**1.6**
PSM	**OmniAnomaly**	0.4315	0.4106	0.3575	0.7694	0.9300	0.9259	0.5479	0.5872	0.5775	0.6462	0.4623	4.5
	LSTM-VAE	0.4216	0.2132	0.4421	0.7749	0.9276	0.9211	0.5599	0.6088	0.5948	0.6431	0.4291	4.7
	MSCRED	0.4399	**0.5353**	0.4688	0.7486	0.8028	0.7903	0.4489	0.5213	0.4645	0.7172	0.4303	5.0
	BeatGan	0.3579	0.0552	0.4075	0.7561	0.9641	0.9596	0.5007	0.6948	0.6941	0.5893	0.3819	4.7
	TCN-AE	**0.4965**	0.3041	**0.5092**	**0.7981**	0.9402	0.9344	**0.6261**	0.6840	0.6801	**0.7268**	0.5132	2.5
	UAE	0.3627	0.0413	0.3942	0.7810	**0.9741**	0.9650	0.5128	0.6863	0.6846	0.5746	0.3649	4.5
	EdgeConvFormer	0.4655	0.3265	0.4803	0.7592	0.9740	**0.9701**	0.6043	**0.7504**	**0.7465**	0.6835	**0.5161**	**2.1**

uses a predefined scoring approach. The EdgeConvFormer model achieved the highest average rank on the MSL, SMAP, SWaT, and PSM datasets and the second highest on the SMD dataset, as shown in Table 2. Specifically, for the MSL and SMAP datasets, which contain multiple short-length entities, EdgeConvFormer was trained separately for each entity, excelling despite the Transformer architecture's typical need for more extensive data. This underscores the EdgeConv module's crucial role in enhancing performance with limited data, as confirmed by an ablation study detailed in Table 5, which shows a performance drop when the module is removed.

The algorithm performance ranking from the best to worst is EdgeConvFormer, TCN-AE, BeatGan, UAE, OmniAnomaly, MSCRED, and LSTM-VAE. The TCN-AE model excels by capturing long-term temporal patterns with its

Table 2. Average rank of algorithms across all five datasets.

Algorithms	Avg-rank on SMD	Avg-rank on MSL	Avg-rank on SMAP	Avg-rank on SWaT	Avg-rank on PSM	Overall Avg-rank
OmniAnomaly	4.1	3.8	4.2	6.5	4.5	4.62
LSTM-VAE	4.9	5.5	5.6	4.5	4.7	5.04
MSCRED	5.5	4.5	3.3	5.3	5.0	4.72
BeatGan	4.7	3.2	4.1	2.8	4.7	3.90
TCN-AE	**2.0**	3.7	3.9	4.8	2.5	3.38
UAE	3.6	5.2	5.4	2.5	4.5	4.24
EdgeConvFormer (ours)	3.2	**1.6**	**1.3**	**1.6**	**2.1**	**1.96**

dilated convolutional layers, performing exceptionally when not restricted to the first sensor's data. BeatGan's third-place benefits from its GAN architecture. However, its CNN-based generator may not fully capture sensor interrelationships, affecting its ability to handle long-term dependencies unless filter sizes are optimally tuned. The UAE model excels on the SWaT dataset by independently analyzing each sensor, avoiding the need to account for inter-sensor relationships. This method is effective because the actuators in the SWaT dataset only provide binary outputs, which are simpler than the continuous data from sensors. By focusing on the temporal dynamics of individual sensors, UAE adeptly identifies anomalies within each data stream. LSTM-VAE and OmniAnomaly, both based on RNN structures, vary in performance. OmniAnomaly integrates GRU and VAE, enhancing its ability to model complex time-series variabilities. However, its static scoring function limits adaptability, which is critical in dynamic environments. MSCRED struggles possibly due to its multi-scale signature matrices that might not adequately capture the full scope of the system's dynamics, leading to information loss.

Evaluation on the Exathlon Dataset. The Exathlon dataset provides a rigorous test for anomaly detection models through a structured series of evaluation levels from AD1 to AD4, where each level increases in difficulty. This setup tests capabilities from simple anomaly detection to precise identification and timing of anomalies.

Our evaluation used three specific Exathlon applications to test a wide range of anomaly types, allowing a comprehensive assessment of our models against diverse anomaly behaviors. According to the results in Table 3, the EdgeConvFormer model generally outperformed other algorithms, demonstrating its robustness across various types of anomalies and all AD levels. Key findings include: 1) Superior Performance of EdgeConvFormer: Across all evaluation metrics, particularly the F_1-$P_T R_T$ score, this model demonstrated robust accuracy in pinpointing anomalies. 2) Reliable Detection Across Complexity Levels: EdgeConvFormer consistently excelled, even at higher AD levels like AD4, showcasing its ability to accurately segment time anomalies amidst increasing evaluation demands.

Comparison with the Two Transformer-Based Approaches. Our analysis in Table 4 evaluates EdgeConvFormer against two Transformer-based mod-

Table 3. Anomaly detection results on Exathlon dataset. AD1 (Anomaly Existence), AD2 (Range Detection), AD3 (Early Detection), and AD4 (Exactly-Once Detection). P_T: Range-based Precision (as %), R_T: Range-based Recall (as %), F_1-P_TR_T: Range-based F_1 score (as %). The best results are highlighted in bold.

AD1	**P_T for Anomaly Types T1→T5**					**R_T for Anomaly Types T1→T5**					**F_1-P_TR_T for Anomaly Types T1→ T5**				
OmniAnomaly	0.4543	0.4310	0.3913	0.3500	0.3667	1.0	1.0	1.0	0.3333	1.0	0.6247	0.6024	0.5625	0.3415	0.5366
LSTM-VAE	0.9193	0.9411	0.5556	0.1259	0.6047	1.0	1.0	0.75	**1.0**	1.0	0.9580	0.9744	0.6383	0.2237	0.7537
MSCRED	0.2973	0.9167	0.4802	**0.6347**	0.7925	1.0	1.0	1.0	0.6667	0.8	0.4583	0.9565	0.6488	0.6503	0.7962
BeatGan	0.9119	0.9474	0.2714	0.3250	0.6000	1.0	1.0	0.5	**1.0**	0.8	0.9539	0.9730	0.3518	0.4906	0.6857
TCN-AE	0.5571	0.8571	0.3115	0.2857	0.7460	1.0	1.0	0.5	0.3333	1.0	0.7156	0.9231	0.3839	0.3077	0.8545
UAE	0.8776	0.9697	0.5000	0.5000	**0.8699**	1.0	1.0	0.25	0.1667	1.0	0.9348	0.9846	0.3333	0.2500	**0.9304**
EdgeConvFormer	**0.9347**	**0.9847**	**0.5667**	0.5870	0.8349	**1.0**	**1.0**	**1.0**	0.8333	**1.0**	**0.9662**	**0.9923**	**0.7234**	**0.6888**	0.9100
AD2	**P_T for Anomaly Types T1 → T5**					**R_T for Anomaly Types T1 → T5**					**F_1-P_TR_T for Anomaly Types T1 → T5**				
OmniAnomaly	**0.9313**	**1.0**	0.2034	0.2439	0.2394	0.2902	0.0120	0.0031	0.0493	0.0154	0.4425	0.0238	0.0061	0.0820	0.0290
LSTM-VAE	0.8125	0.3755	0.5556	0.1259	0.7248	0.7827	0.6889	0.2608	0.3365	0.4100	0.7973	0.4860	0.3549	0.1833	0.5237
MSCRED	0.4268	0.5890	0.3076	0.6667	0.5261	**0.9316**	0.9801	0.8698	0.3884	0.3148	0.5854	0.7358	0.4546	0.4908	0.3939
BeatGan	0.7894	0.8793	0.2498	0.3250	0.4233	0.6120	0.9115	0.4611	**0.7907**	0.1868	0.6895	0.8951	0.3241	0.4606	0.2593
TCN-AE	0.6667	0.8166	0.2185	0.1197	0.5795	0.8650	0.9687	0.3791	0.7521	0.2424	0.7530	0.8862	0.2772	0.2065	0.3418
UAE	0.7844	0.6154	0.2961	0.0915	**0.8610**	0.5964	0.9151	0.3207	0.7550	0.3261	0.6776	0.7369	0.3079	0.1633	0.4730
EdgeConvFormer	0.7600	0.8904	**0.5621**	**0.7622**	0.4551	0.8529	**0.9898**	**0.9170**	0.5744	**0.9549**	**0.8038**	**0.9375**	**0.6969**	**0.6551**	**0.6164**
AD3	**P_T for Anomaly Types T1 → T5**					**R_T for Anomaly Types T1 → T5**					**F_1-P_TR_T for Anomaly Types T1 → T5**				
OmniAnomaly	**0.9285**	0.0008	0.2034	0.2439	0.2394	0.1821	0.0016	0.0021	0.0151	0.0125	0.3045	0.0102	0.0042	0.0285	0.0237
LSTM-VAE	0.8125	**0.9929**	0.5556	0.1259	0.3810	0.5448	0.3266	0.2545	0.2455	0.5349	0.6523	0.4915	0.3491	0.1665	0.4450
MSCRED	0.3733	0.5890	0.3077	0.5000	0.5261	**0.9277**	**0.9801**	0.7608	0.3670	0.2943	0.5324	0.7358	0.4382	0.4233	0.3775
BeatGan	0.7894	0.8793	0.2499	0.3250	0.4233	0.4521	0.7520	0.3964	**0.7303**	0.1732	0.5749	0.8106	0.3065	0.4498	0.2487
TCN-AE	0.6667	0..8166	0.2185	0.1197	0.2277	0.5825	0.6360	0.3437	0.6753	0.4868	0.6218	0.7151	0.2672	0.2033	0.3103
UAE	0.4984	0.6154	0.2961	0.0915	**0.8610**	0.9183	0.5642	0.2891	0.7110	0.2901	0.6461	0.5887	0.2925	0.1622	0.4340
EdgeConvFormer	0.5751	0.8904	**0.5621**	**0.6097**	0.4551	0.9229	0.8813	**0.8613**	0.5744	**0.8952**	**0.7086**	**0.8858**	**0.6802**	**0.5915**	**0.6034**
AD4	**P_T for Anomaly Types T1 → T5**					**R_T for Anomaly Types T1 → T5**					**F_1-P_TR_T for Anomaly Types T1 → T5**				
OmniAnomaly	**0.9445**	**1.0**	0.2619	0.2439	0.3667	0	0	0	0.0002	0.0059	0	0	0	0.0005	0.0116
LSTM-VAE	0.4993	0.9929	**0.5556**	0.3333	0.3766	**0.9453**	0	0.2345	0.0253	0.3774	0.6535	0	0.3298	0.0471	0.3770
MSCRED	0.3733	0.5890	0.1457	0.3913	0.5261	0.9277	0.9801	**0.9358**	0.2860	0.2836	0.5324	**0.7358**	0.2521	0.3304	0.3685
BeatGan	0.3742	0.8793	0.2499	0.3224	0.4233	0.9441	0	0.2390	0.5000	0.1287	0.5360	0	0.2443	0.3920	0.1974
TCN-AE	0.3995	0.8166	0.2185	0.1189	0.5795	0.6308	0	0.2376	0.3258	0.1851	0.4892	0	0.2277	0.1743	0.2806
UAE	0.4984	0.8553	0.2961	0.0910	**0.8610**	0.6307	0	0.2418	0.5000	0.1842	0.5568	0	0.2662	0.1540	0.3034
EdgeConvFormer	0.5751	0.5759	0.4877	**0.6097**	0.4169	0.9229	**0.9999**	0.7565	**0.5343**	**0.8022**	**0.7086**	0.7309	**0.5930**	**0.5695**	**0.5487**

els, TranAD and Anomaly Transformer, which enhance sensitivity to anomalies through adversarial training and a minimax strategy, respectively. EdgeConvFormer consistently outperforms these models, particularly excelling in Fpa_1 scores across multiple datasets like SMD, MSL, SMAP, and PSM, demonstrating robust anomaly detection in complex multivariate contexts and with limited data. While it ranks second on the SWaT dataset, its high precision illustrates its effectiveness at minimizing false positives. In competitive settings like the PSM dataset, EdgeConvFormer closely rivals Anomaly Transformer, affirming its capability in intricate anomaly detection scenarios.

5.2 Qualitative Evalution

The performance of the EdgeConvFormer model, shown in Fig. 3, highlights its proficiency in anomaly detection across diverse datasets. The model consistently aligns predicted anomalies (red) closely with the ground truth (blue), minimizing false positives and recognizing patterns of consecutive anomalies, which indicates

Table 4. Performance comparison of EdgeConvFormer with SOTA on the five public datasets. P: Precision (as %), R: Recall (as %), Fpa_1: point-adjusted F_1 score (as %). The best results are highlighted in bold. Except for EdgeConvFormer, all other model results are cited from TranAD [10] and Anomaly Transformer [8].

Dateset	SMD			MSL			SMAP			SWaT			PSM		
Metric	P	R	Fpa_1	P	R	Fpa_1	P	R	Fpa_1	P	R	Fpa_1	P	R	Fpa_1
Anomaly Transformer	89.40	95.45	92.33	92.09	95.15	93.59	94.13	99.40	96.69	91.55	**96.73**	**94.07**	96.91	**98.90**	**97.89**
TranAD	92.62	**99.74**	96.05	90.38	**99.99**	94.94	80.43	**99.99**	89.15	97.60	69.97	81.51	68.13	90.18	77.62
EdgeConvFormer(ours)	**97.83**	98.50	**98.16**	**95.78**	97.61	**96.69**	**97.33**	99.61	**98.45**	**97.81**	80.52	88.33	**98.13**	96.68	97.40

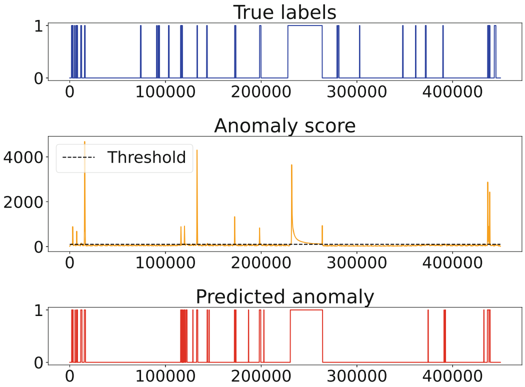


Fig. 3. Qualitative evaluation of anomaly detection on SWaT dataset using the proposed EdgeConvFormer model. For each dataset, the blue line is the ground truth anomaly, the orange line is the predicted anomaly score, and the red line is the predicted anomaly label by applying the best-Fc_1 threshold (black dash line) to the anomaly score. (Color figure online)

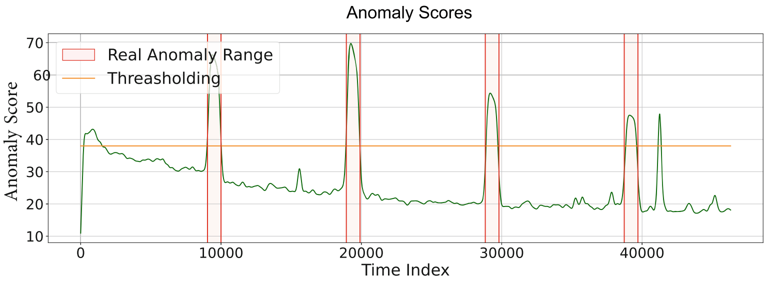


Fig. 4. Qualitative evaluation of anomaly detection on Exathlon dataset using the proposed EdgeConvFormer model. The pink rectangle is the ground truth anomaly, the green line is the predicted anomaly score (with Gauss_D_K scoring), and the yellow line is the tail-p threshold to the anomaly score. (Color figure online)

a grasp of temporal context. This performance is consistent across all datasets, where it effectively marks at least one point within nearly every anomaly segment, optimizing detection while maintaining low false positives.

Further insights from Fig. 4 demonstrate the model's precision on the Exathlon dataset using *Gauss_D_K* scoring and *Tail-p* thresholding. The model effectively delineates true anomaly boundaries and closely matches detected anomalies with the actual ones, affirming EdgeConvFormer's capability to handle complex anomaly detection tasks with high efficiency.

5.3 Ablation Studies

Ablation studies on the EdgeConvFormer model evaluate the individual contributions of its Time2Vec embedding, EdgeConv module, and ParaAtten module.

Removing Time2Vec while using the standard positional encoding [34] significantly reduced performance: F_1 score fell by 89.71%, highlighting Time2Vec's effectiveness in capturing temporal dynamics. Removing the EdgeConv module led to a dramatic decrease in F_1 score by 90.02% and similarly large declines across other metrics, underscoring its crucial role in integrating spatial-temporal data. Conversely, excluding the ParaAtten module resulted in smaller reductions (F_1 score down by 6.00%), indicating its importance in processing long-range dependencies, though less critical than the EdgeConv module.

Replacing the ParaAtten module with the traditional Flattened Transformer architecture resulted in decreases of 30.76% in the F1 score, 6.13% in Fpa1, 37.97% in Fc1, 13.59% in AUROC, and 27.68% in AUPRC. These results underscore EdgeConvFormer's effectiveness in processing sensor-specific data streams, in contrast to the Flattened Transformer, which may merge variate and embedding dimensions, potentially leading to feature confusion.

Table 5. Ablation study on the EdgeConvFormer model on the PSM dataset, assessing the effects of removing Time2Vec, EdgeConv, and ParaAtten modules, as well as replacing ParaAtten with flatted self-attention. The best results are highlighted in bold.

Metric	F_1	Fpa_1	Fc_1	AU-ROC	AU-PRC
w/o Time2Vec	0.0336	0.9047	0.6007	0.6037	0.3795
w/o EdgeConv	0.0326	0.9044	0.6030	0.6112	0.3867
w/o ParaAtten	0.3069	0.9165	0.6850	0.6503	0.4464
replace ParaAtten	0.2497	0.9177	0.5439	0.6017	0.4042
EdgeConvFormer	**0.3265**	**0.9740**	**0.7504**	**0.6835**	**0.5161**

6 Conclusion

This paper presents the EdgeConvFormer model, an unsupervised anomaly detection method in multivariate time series, integrating Time2Vec, dynamic graph CNNs (EdgeConv), and parallel sensor-specific attention mechanisms for improved detection performances. Time2Vec enriches input embeddings with both positional and time-specific characteristics, while EdgeConv and parallel sensor-specific attentions are combined in a multi-layered structure to refine spatiotemporal features alternatively. This configuration enhances detection accuracy and has proven effective across various real-world datasets, outperforming existing state-of-the-art anomaly detection methods. Future work could explore more efficient data representations within graph CNNs and attention mechanisms, potentially at the sub-series level to optimize computational and memory demands.

References

1. Zope, K., Singh, K., Nistala, S.H., Basak, A., Rathore, P., Runkana, V.: Anomaly detection and diagnosis in manufacturing systems: a comparative study of statistical, machine learning and deep learning techniques (2019)
2. Darban, Z.Z., Webb, G.I., Pan, S., Aggarwal, C.C., Salehi, M.: Deep learning for time series anomaly detection: a survey (2022)
3. Wang, Y., et al.: Fully-connected spatial-temporal graph for multivariate time-series data. Proc. AAAI Conf. Artif. Intell. **38**(14), 15715–15724 (2024). https://doi.org/10.1609/aaai.v38i14.29500
4. Feremans, L., Vercruyssen, V., Cule, B., Meert, W., Goethals, B.: Pattern-based anomaly detection in mixed-type time series. In: Brefeld, U., Fromont, E., Hotho, A., Knobbe, A., Maathuis, M., Robardet, C. (eds.) Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2019, Würzburg, Germany, September 16–20, 2019, Proceedings, Part I, pp. 240–256. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-46150-8_15
5. Ho, T.K.K., Karami, A., Armanfard, N.: Graph-based time-series anomaly detection: a survey (2023)
6. Liu, F.T., Ting, K.M., Zhou, Z.-H.: Isolation forest. In 2008 Eighth IEEE International Conference on Data Mining, pp. 413–422. IEEE (2008)
7. Chalapathy, R., Menon, A.K., Chawla, S.: Anomaly detection using one-class neural networks. arXiv:1802.06360 (2018)
8. Xu, J., Wu, H., Wang, J., Long, M.: Anomaly transformer: time series anomaly detection with association discrepancy. arXiv preprint arXiv:2110.02642 (2021)
9. Doshi, K., Abudalou, S., Yilmaz, Y.: TiSAT: time series anomaly transformer. arXiv:2203.05167 (2022)
10. Tuli, S., Casale, G., Jennings, N.R.: TranAD: deep transformer networks for anomaly detection in multivariate time series data. arXiv:2201.07284 (2022)
11. Wen, Q., et al.: Transformers in time series: a survey. arXiv:2202.07125 (2023)
12. Gu, J., Jia, Z., Cai, T., Song, X., Mahmood, A.: Dynamic correlation adjacency-matrix-based graph neural networks for traffic flow prediction. Sensors **23**, 2897 (2023)
13. Deng, A., Hooi, B.: Graph neural network-based anomaly detection in multivariate time series. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 4027–4035 (2021)
14. Pang, G., Shen, C., Cao, L., Hengel, A.V.D.: Deep learning for anomaly detection: a review. ACM Comput. Surv. (CSUR) **54**(2), 1–38 (2021)

15. Su, Y., Zhao, Y., Niu, C., Liu, R., Sun, W., Pei, D.: Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In: Proceeding of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2828–2837 (2019)
16. Park, D., Hoshi, Y., Kemp, C.C.: A multimodal anomaly detector for robot-assisted feeding using an LSTM-based variational autoencoder. IEEE Robot. Autom. Lett. **3**(3), 1544–1551 (2018)
17. Zhang, C., et al.: A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 1409–1416 (2019)
18. Zhou, B., Liu, S., Hooi, B., Cheng, X., Ye, J.: BeatGAN: anomalous rhythm detection using adversarially generated time series. In: IJCAI, pp. 4433–4439 (2019)
19. Zerveas, G., Jayaraman, S., Patel, D., Bhamidipaty, A., Eickhoff, C.: A transformer-based framework for multivariate time series representation learning. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 2114–2124 (2021)
20. Thill, M., Konen, W., Bäck, T.: Time series encodings with temporal convolutional networks. In: Filipič, B., Minisci, E., Vasile, M. (eds.) Bioinspired Optimization Methods and Their Applications: 9th International Conference, BIOMA 2020, Brussels, Belgium, November 19–20, 2020, Proceedings, pp. 161–173. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-63710-1_13
21. Zhao, H., et al.: Multivariate time-series anomaly detection via graph attention network. In Proceedings of the 2020 IEEE International Conference on Data Mining (ICDM), pp. 841–850. IEEE (2020)
22. Chen, Z., Chen, D., Zhang, X., Yuan, Z., Cheng, X.: Learning graph structures with transformer for multivariate time series anomaly detection in IoT. IEEE, IEEE Internet Things J (2021)
23. Kazemi, S.M., et al.: Time2Vec: learning a vector representation of time (2019)
24. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph CNN for learning on point clouds. ACM Trans. Graph. (TOG) **38**(5), 1–12 (2019)
25. Ahmad, S., Lavin, A., Purdy, S., Agha, Z.: Unsupervised real-time anomaly detection for streaming data. Neurocomputing **262**, 134–147 (2017)
26. Garg, A., Zhang, W., Samaran, J., Savitha, R., Foo, C.S.: An evaluation of anomaly detection and diagnosis in multivariate time series. IEEE Trans. Neural Netw. Learn. Syst. **33**(6), 2508–2517 (2021)
27. Li, Z., Zhao, Y., Han, J., Su, Y., Jiao, R., Wen, X., Pei, D.: Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 3220–3230 (2021)
28. Jacob, V., Song, F., Stiegler, A., Rad, B., Diao, Y., Tatbul, N.: Exathlon: a benchmark for explainable anomaly detection over time series. arXiv:2010.05073 (2020)
29. Fernández, A., García, S., Galar, M., Prati, R.C., Krawczyk, B., Herrera, F.: Learning from Imbalanced Data Sets, vol. 10. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98074-4
30. Hundman, K., Constantinou, V., Laporte, C., Colwell, I., Soderstrom, T.: Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 387–395 (2018)

31. Goh, J., Adepu, S., Junejo, K.N., Mathur, A.: A dataset to support research in the design of secure water treatment systems. In: Havarneanu, G., Setola, R., Nassopoulos, H., Wolthusen, S. (eds.) Critical Information Infrastructures Security, pp. 88–99. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-71368-7_8
32. Abdulaal, A., Liu, Z., Lancewicki, T.: Practical approach to asynchronous multivariate time series anomaly detection and localization. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 2485–2494 (2021)
33. Bai, S., Kolter, J.Z., Koltun, V.: An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv:1803.01271 (2018)
34. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)

Lighten CARAFE: Dynamic Lightweight Upsampling with Guided Reassemble Kernels

Ruigang Fu[1(✉)], Qingyong Hu[2(✉)], Xiaohu Dong[1], Yinghui Gao[2], Biao Li[1],
and Ping Zhong[1]

[1] National University of Defense Technology, Changsha 410073, China
{`furuigang08,dongxiaohu16,libiao,zhongping`}`@nudt.edu.cn`
[2] Chinese Academy of Military Sciences, Beijing 100000, China
{`huqingyong15,yhgao`}`@nudt.edu.cn`

Abstract. As a fundamental operation in modern machine vision models, feature upsampling has been widely used and investigated in the literatures. An ideal upsampling operation should be lightweight, with low computational complexity. That is, it can not only improve the overall performance but also not affect the model complexity. Content-aware Reassembly of Features (CARAFE) is a well-designed learnable operation to achieve feature upsampling. Albeit encouraging performance achieved, this method requires generating large-scale kernels, which brings a mass of extra redundant parameters, and inherently has limited scalability. To this end, we propose a lightweight upsampling operation, termed Dynamic Lightweight Upsampling (DLU) in this paper. In particular, it first constructs a small-scale source kernel space, and then samples the large-scale kernels from the kernel space by introducing learnable guidance offsets, hence avoiding introducing a large collection of trainable parameters in upsampling. Experiments on several mainstream vision tasks show that our DLU achieves comparable and even better performance to the original CARAFE, but with much lower complexity, *e.g.,* DLU requires 91% fewer parameters and at least 63% fewer FLOPs (Floating Point Operations) than CARAFE in the case of 16× upsampling, but outperforms the CARAFE by 0.3% mAP in object detection. Code is available at https://github.com/Fu0511/Dynamic-Lightweight-Upsampling.

Keywords: Upsampling operation · Dynamic convolution · Lightweight design · Deep learning

1 Introduction

Effective scene parsing and object recognition are key abilities required by most intelligent vision systems to perceive and interact with the real world [19,30]. Recently, deep learning-based approaches have attracted increasing attention of the computer vision community [23,29] for their strong capability in representation learning. Despite remarkable research progress has been achieved,

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 383–399, 2025.
https://doi.org/10.1007/978-3-031-78128-5_25

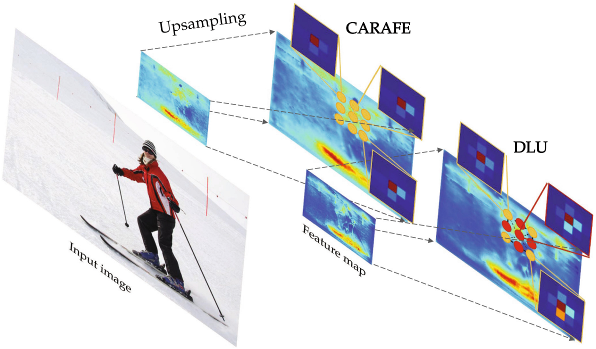


Fig. 1. Intuitive comparison of CARAFE [21] and DLU. Unlike CARAFE needs to generate independent kernels (yellow ones) for all the positions of the upsampling output, our DLU generates a portion (yellow ones), and samples others (red ones) with learnable offsets through interpolation. (Color figure online)

it remains challenging to deploy large neural models in practical applications, especially under resource-constrained conditions.

To this end, it is desirable to further explore the lightweight design of neural architecture and operation. In particular, feature upsampling, as one of the fundamental operations in modern deep learning architectures [19,20], has been under-explored compared with other basic operations such as convolution and pooling. Therefore, we dive deep into this area and focus on the lightweight upsampling design in this paper.

Traditional interpolation-based upsampling approaches such as nearest neighbor and bilinear interpolation, have been extensively adopted in classical models for their simplicity and the nature of easy-to-implement [8]. However, interpolation operations fundamentally only consider the spatial adjacency relationship, while failing to capture the rich semantic information encoded in the feature map of CNNs [21]. This limitation has motivated researchers to explore learnable upsampling techniques that can introduce additional information through the upsampling operation in a data-driven manner [24]. The most commonly used is deconvolution [16], which can be viewed as an inverse operation of convolution. Albeit intuitive, the deconvolution operation has not considered the local variations explicitly in the images, since it applies the same kernel across different locations.

Compared with the existing interpolation-based and most learning-based upsampling such as deconvolution [16] and pixel shuffle [18], a learnable upsampling operation named CARAFE (Content-aware Reassembly of Features) [21,22] achieves promising performance. However, this operation needs to generate a corresponding reassembly kernel for all the positions of the upsampling output, leading to the training parameters being quadratically related to the product of the reassembly kernel size and the upsampling ratio (detail calculation can be found in Table 1 in Sect. 3.3). This inherently limits the scalability of the proposed method, especially under large upsampling ratios.

To this end, we proposed a novel upsampling approach termed Dynamic Lightweight Upsampling (DLU) in this paper. As shown in Fig. 1, different from CARAFE generates an independent kernel for each position of the upsampling output, the proposed DLU first constructs a "source kernel space" with a portion of kernels, and then samples the kernels with learnable offsets from the kernel space by interpolation. Extensive experiments on several mainstream vision tasks justify the superior performance of the proposed DLU.

The key contributions of our paper are highlighted as follows:

- We propose a novel learnable upsampling operation called DLU by generating large-scale expected kernels from sampling in a "source kernel space". It is lightweight, with low computational complexity, and can be seamlessly plugged into various modern neural architectures.
- The proposed DLU operation is notably lightweight than the original CARAFE operation, but achieve comparable or even better task performance. For example, DLU needs 90% fewer parameters and at least 63% fewer FLOPs than the original CARAFE in the case of 16× upsampling, but outperforms the CARAFE by 0.3% mAP in object detection.
- Extensive experiments on three fundemental vision tasks with four representative neural networks verify the availability and scalability of our DLU. For example, by integrating our DLU, the detection performance of FPN [8] and Libra RCNN [17] can be improved by 1.2% and 0.7% mAP respectively, compared to their original versions.

2 Related Work

Feature upsampling is widely used in vision tasks such as object detection, semantic segmentation and instance segmentation, to increase the resolution of the feature map. In general, it can be divided into interpolation-based and learning-based upsampling.

Interpolation-Based Upsampling. Representative methods include nearest-neighbor interpolation, bilinear interpolation, and bicubic interpolation. To perform interpolation at a specified position, these methods usually first identify a neighborhood of the specific pixel, and then calculate a weighted sum of the specific neighborhood. In general, these methods are fast and easy-to-implement [8]. However, the main drawback of the interpolation-based methods is that the output of upsampling is totally derived from the input. That is, no additional information was introduced during the upsampling operation.

Learning-Based Upsampling. Different from interpolation-based upsampling, learning-based upsampling methods aim to introduce additional information during the upsampling operation, hence attracting increasing attention in recent years. Deconvolution operation [16] is proposed to perform zero-padding of the input feature map first, and then increase the feature resolution by executing standard convolutions. However, the effective receptive field of deconvolution is rather small. Pixel shuffle [18] achieves upsampling by first increasing the feature channels through convolution, and then reshaping to a specific resolution. This method can achieve a larger receptive field, but with the price of massive extra parameters. To improve the upsampling efficiency, Mazzini et al. [14] proposed guided upsampling, which enlarges the input using bilinear interpolation, but sampling pixels with learnable offsets. However, this method also suffers from small effective receptive fields. Building upon guided upsampling, DySample [10] delves deeper into the generation of the offsets, exploring superior methods to achieve improved performance.

A handful recent works have started to explore the upsampling with learnable kernels to dynamically enlarge the receptive field [12,13,21]. Among them, CARAFE [21] is an effective learnable upsampling operation. Given the upsampling ratio as σ, CARAFE would generate σ^2 kernels for each pixel of the input, then one pixel would output σ^2 pixels through convoluting the σ^2 different generated kernels. Later, CARAFE++ [22] is further proposed to endow the learnable downsampling capacity, following the similar concepts of CARAFE. FADE [12] and SAPA [13] refined the framework of CARAFE to a double-input framework. Their input requires not only the low-resolution feature to be upsampled but also a high-resolution guiding feature as an additional input. Consequently, their application scenarios is limited (high-resolution features must be available). Our goal is to contribute a lightweight, scalable and effective upsampling operation. Since CARAFE-based techniques require a large collection of parameters in generating kernels, we lighten the kernel generation module of CARAFE and propose a dynamic lightweight upsampling operator.

3 Method

3.1 Revisit CARAFE

Given a feature map $\mathcal{F}$ of size $H \times W \times C$, and the upsampling ratio as σ, the expected size of outputted feature map after upsampling is $\sigma H \times \sigma W \times C$. Specifically, CARAFE achieves feature upsampling through the following steps: 1) Predicting a tensor of size $H \times W \times k_{up}^2 \sigma^2$ by convoluting the input feature map, where $k_{up} \times k_{up}$ is the receptive field (kernel size) for upsampling. This tensor is further reshaped to $\sigma H \times \sigma W \times k_{up}^2$ according to pixel shuffle [18], and each spatial location in the tensor forms a $k_{up} \times k_{up}$ kernel, resulting in $\sigma H \times \sigma W$ kernels in total (known as reassemble kernels). 2) All these kernels are normalized by softmax function, respectively. 3) For each location of the expected output (i, j, c) (position (i, j) in the c-th feature map), taking the predicted kernel at position (i, j) from the above kernel tensor, and finding the corresponding neighbourhood of $k_{up} \times k_{up}$ around the input pixel $(\lfloor i/\sigma \rfloor, \lfloor j/\sigma \rfloor, c)$. The final response at (i, j, c) of the upsampled feature maps is the inner product of the kernel and the pixel neighborhood.

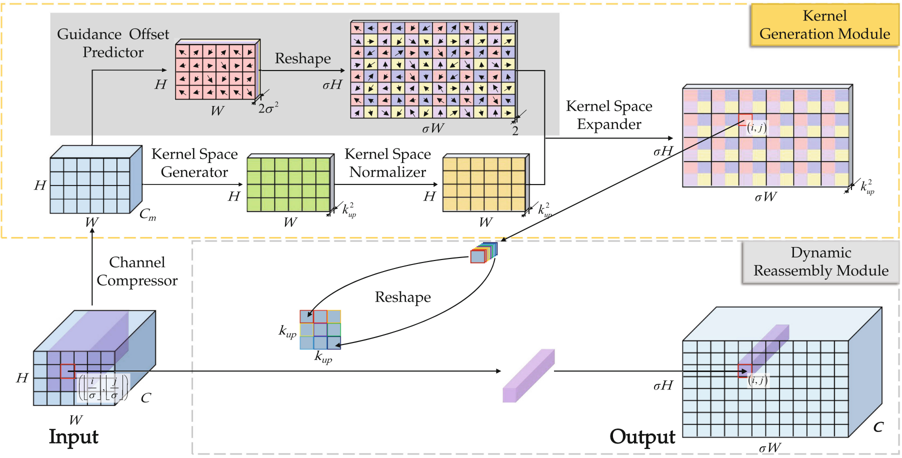


Fig. 2. The overall framework of our DLU.

Limitation of CARAFE. CARAFE uses a convolution layer to predict a tensor of size $H \times W \times k_{up}^2 \sigma^2$ in the branch of reassemble kernel generation, where the parameters in this convolution layer are proportional to $k_{up}^2 \sigma^2$. This comes with a large collection of trainable parameters, resulting in memory-inefficient, especially when a large receptive field (*i.e.*, large k_{up}) or aggressive upsampling ratio (*i.e.*, large σ) is used. Additionally, too many parameters also lead to a high risk of overfitting.

Solutions. It is important to note that the kernels predicted by CARAFE are independent of each other. However, it is argued that using independent kernels to convolute an input may lead to an issue of so-called "checkerboard artifacts"[1], since there is no direct relationship between adjacent pixels in the upsampled output [4]. As a result, the large collection of trainable parameters in CARAFE are likely to be redundant. To address this issue, it may be more feasible and effective to explore the interdependence of adjacent kernels.

Inspired by the idea of deformable kernels [3], that is, kernels can be augmented by learning free-form offsets on kernel coordinates in the original kernel space, we further proposed a lightweight upsampling operation by avoiding generating a large collection of independent kernels. Specifically, we first construct a small-scale source kernel space, and then sample large-scale expected kernels from the source kernel space by introducing learnable guidance offset. It is noted that the trainable parameters can be greatly reduced in this way.

3.2 The Proposed DLU

As shown in Fig. 2, the proposed DLU mainly consists of two modules: kernel generation and dynamic reassembly module. Given a scaling ratio σ for feature

---

[1] https://distill.pub/2016/deconv-checkerboard/.

upsampling, the kernel generation module is used to predict $\sigma H \times \sigma W$ kernels, where H and W determine the shape of the input feature map. The dynamic reassembly module is responsible for generating the upsampling output with shape of $\sigma H \times \sigma W$, each location of the input is convoluted with location-specific generated kernels [6]. That is, the convolution is dynamic and content-aware.

Kernel Generation Module. Different from the original CARAFE [21], the kernel generation module in our framework is composed of: channel compressor, kernel space generator, kernel space normalizer, guidance offset predictor, and kernel space expander.

Channel Compressor. Similar to CARAFE, this step aims to reduce the number of input feature channels, hence further saving the overall computational cost. Specifically, we compress the input feature map from C channels to C_m channels through a 1×1 convolution layer.

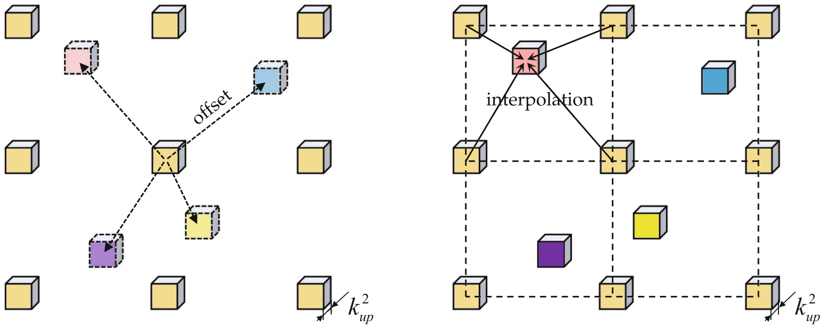
Kernel Space Generator. Next, a $k_{encoder} \times k_{encoder}$ convolution layer is used to encode the compressed feature maps to generate a tensor of size $H \times W \times k_{up}^2$. Each location in the tensor can be reshaped as a $k_{up} \times k_{up}$ kernel, where k_{up} determined the size of the receptive field for upsampling. Up to here, the tensor with $H \times W$ kernels forms a “source kernel space”.

Kernel Space Normalizer. Here, the kernel space is normalized by performing softmax operation spatially. As a result, the elements for each kernel in the space are forced to be positive, with a summation of 1. This step is motivated by the classical interpolation-based upsampling methods. Specifically, our DLU can be fundamentally regarded as a linear weighted method, *i.e.,* each position of the upsampled feature is a weighted sum of the input feature and the generated kernels represent the weights. Different from traditional deconvolution operation, softmax normalization of the kernels is used to make the upsampled feature have the same order of magnitudes as the input feature, which provides clearer physical interpretability.

Guidance Offset Predictor. Here, the proposed guidance offset predictor is used to predict the sampling coordinates for enlarging the kernel space. Specifically, a $k_{encoder} \times k_{encoder}$ convolutional layer is used to convolute the compressed feature map to generate a tensor of size $H \times W \times 2\sigma^2$. This tensor can be further reshaped to $\sigma H \times \sigma W$ 2-D offsets (in x and y dimensions respectively), as shown in Fig. 2.

Kernel Space Expander. For each location in the normalized source kernel space, we assign σ^2 2-D offsets and sample σ^2 newborn kernels by applying bilinear interpolation channel-wisely. Specifically, Fig. 3 illustrates the expansion of the source kernel space.

Overall, the kernel generation module results in $\sigma H \times \sigma W$ final kernels, as shown in Fig. 2. Note that it can be proved that arbitrary kernel in the expanded kernel space is also softmax normalized after the normalization of the source kernel space (The detailed proof can be found in our appendix).



a) Locate the newborn kernels using learned offsets

b) Determine the newborn kernels using bilinear interpolation channel-wisely

Fig. 3. Illustration of the expansion of source kernel space. In this figure, each kernel in the source space (orange) generates 4 newborn kernels (pink, blue, purple, and yellow) in the case of 2× upsampling, and the newborn kernels are sampled from the source kernel space using bilinear interpolation channel-wisely. (Color figure online)

Dynamic Reassembly Module. Following the original CARAFE [21], a dynamic reassembly module is used to output the upsampled feature map which is expected to have the size of $\sigma H \times \sigma W \times C$. To get the response at position (i, j) in the c-th feature map of the expected output, we first take the predicted kernel at position (i, j) from the expected expanded kernel space, and then dig out a neighborhood of $k_{up} \times k_{up}$ around the input pixel at position $(\lfloor i/\sigma \rfloor, \lfloor j/\sigma \rfloor)$ in the c-th feature map of the input, then, the specific response in the output is equal to the inner product of the kernel and the pixel neighborhood.

3.3 Comparison with CARAFE

As described above, the key difference between CARAFE and our DLU is the way to generate reassemble kernels, which further leads to different number of trainable parameters and computational costs.

Based on [15] and [21], given the task of upsampling a feature map with input channel C_{in} by a ratio of σ, the details of the required parameters and computation in the kernel generation module of CARAFE [18] and our DLU can be found in Table 1. Table 1 shows that both the parameters and computational complexity of CARAFE are primarily proportional to $\sigma^2 k_{up}^2$, while in our DLU, the parameters and computational complexity are primarily proportional to $k_{up}^2 + 2\sigma^2$. This indicates that our DLU model generally has fewer parameters and lower computational complexity than CARAFE when $k_{up} \geq 3$ and $\sigma \geq 2$ (general cases). We also noticed that this advantage of our approach is more distinguished if dense upsampling with aggressive upsampling ratios or large receptive fields are used.

Note that, our DLU operator introduces bilinear interpolation in its kernel space expander, which results in a term of $9\sigma^2 k_{up}^2$ being included in the total computation. However, this term is typically negligible compared to the compu-

Table 1. Details of the required parameter and computation in the kernel generation module of CARAFE [21] and our DLU.

# Trainable Parameters	Kernel Generation Module
CARAFE	$\underbrace{(C_{in}+1)C_m}_{Channel\ Compressor} + \underbrace{(C_m k_{encoder}^2+1)\boldsymbol{\sigma^2 k_{up}^2}}_{Kernel\ Generator}$
DLU	$\underbrace{(C_{in}+1)C_m}_{Channel\ Compressor} + \underbrace{(C_m k_{encoder}^2+1)\boldsymbol{k_{up}^2}}_{Kernel\ Space\ Generator} + \underbrace{(C_m k_{encoder}^2+1)\boldsymbol{2\sigma^2}}_{Guidance\ Offset\ Predictor}$
# Per-pixel FLOPs	Kernel Generation Module
CARAFE	$\underbrace{2(C_{in}+1)C_m}_{Channel\ Compressor} + \underbrace{2(C_m k_{encoder}^2+1)\boldsymbol{\sigma^2 k_{up}^2}}_{Kernel\ Generator} + \underbrace{\boldsymbol{\sigma^2} \times (k_{up}^2\text{–D sm})^1}_{Kernel\ Normalization}$
DLU	$\underbrace{2(C_{in}+1)C_m}_{Channel\ Compressor} + \underbrace{2(C_m k_{encoder}^2+1)\boldsymbol{k_{up}^2}}_{Kernel\ Space\ Generator} + \underbrace{\boldsymbol{1} \times (k_{up}^2\text{–D sm})}_{Kernel\ Normalization}$ $+ \underbrace{2(C_m k_{encoder}^2+1)\boldsymbol{2\sigma^2}}_{Guidance\ Offset\ Predictor} + \underbrace{\boldsymbol{9\sigma^2 k_{up}^2}}_{Kernel\ Space\ Expander}$

(k_{up}^2-D sm) represents the computation of softmax normalization for a k_{up}^2-D vector.

tation of CARAFE. This is because the kernel generator of CARAFE requires $2(C_m k_{encoder}^2 + 1)\sigma^2 k_{up}^2$ FLOPs, and $C_m k_{encoder}^2$ is typically a large value.

4 Experiments and Results

In this section, we mainly evaluate the performance of different upsampling methods on multiple mainstream vision tasks. Specifically, these upsampling methods are integrated into several typical deep-learning models to replace the original upsampling operators.

4.1 Experimental Setup

- **Object Detection.** For object detection, we integrate different upsampling operations into the FPN [8] framework, and then compare the performance on the MS COCO dataset [9]. Results are evaluated based on the standard COCO metric, *i.e.* mAP of IoUs from 0.5 to 0.95. Besides, we also have experiments on Libra CNN [17], evaluating the performance of the original CARAFE and our proposed methods with different upsampling ratios.
- **Semantic Segmentation.** We take Semantic FPN [7] as the baseline framework and then compare the semantic segmentation performance of different upsampling operations on the ADE20K dataset [28]. Results are measured with mean IoU (mIoU) which indicates the average IoU between predictions and ground truth masks.
- **Instance Segmentation.** For instance segmentation, we take the Mask RCNN [5] as the baseline framework and then compare the segmentation performance of different upsampling operations on the MS COCO dataset. Due to the page limits, please refer to our appendix for the experimental results on instance segmentation.

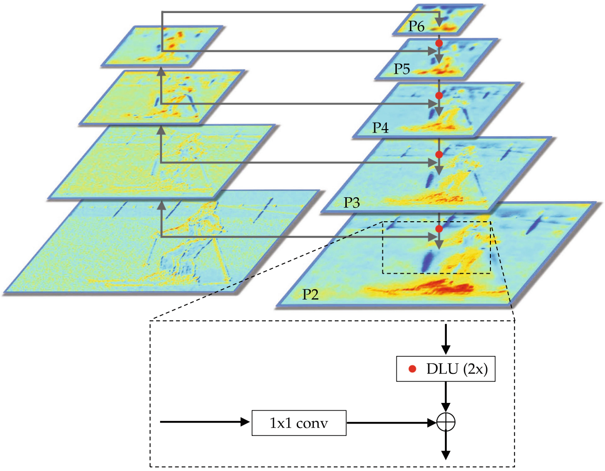


Fig. 4. FPN architecture with the upsampling operator as DLU.

- **Implementation Details.** All the deep models are trained on the training subset, and the final performance is evaluated on the validation set. Experiments of object detection and instance segmentation are implemented based on the mmdetection toolbox [1] and follow the 1× training schedule settings on four NVIDIA Quadro P5000 GPUs. Specifically, for training our models, we employed stochastic gradient descent for 12 epochs with a batch size of 2 examples per GPU, weight decay of 0.0001, momentum of 0.9, and a learning rate of 0.0025. The initial learning rate decayed by a factor of 10 at the 8th and 11th epochs. Similarly, for our experiments on semantic segmentation, we used the mmsegmentation toolbox [26] and followed the 40k training schedule settings on four NVIDIA Quadro P5000 GPUs. Our models were trained using stochastic gradient descent for 40k iterations with a batch size of 4 examples per GPU, weight decay of 0.0005, momentum of 0.9, and a learning rate of 0.01. We utilized the poly learning rate policy with a power factor of 0.9 to decay the initial learning rate.

Unless noted, our DLU shares the same hyper-parameters with CARAFE in all the experiments. Specifically, $C_m = 64$ for the channel compressor, $k_{encoder} = 3$, $k_{up} = 5$ for the kernel space generator. The ablation studies of hyperparameters impact can be found in the appendix. We initialized the weights of the kernel space generator from a zero-mean Gaussian distribution with standard deviation of 0.001, since a large standard deviation may lead to small gradients in back-propagation for the following kernel space normalizer (softmax function). Then, the weights of the guidance offset predictor are initialized by zero-initializing, while other layers follow adaptive Xavier initialization.

Table 2. Detection results on MS COCO 2017 validation set with FPN [8]. Various upsampling operations are used. Please note that "Params" represents the introduced additional parameters per upsampling operation, "FLOPs" represents the FLOPs of each upsampling operation corresponding to each pixel of the input feature map. Above the dotted line are interpolation-based methods, while below the dotted line are learning-based methods. Best performance is in boldface, and second best is underlined.

Method	mAP	mAP_{50}	mAP_{75}	mAP_S	mAP_M	mAP_L	Params	FLOPs	FPS
Nearest	37.4	58.1	40.6	20.9	41.2	48.9	0	0	20.4
Bilinear	37.5	58.5	40.5	21.3	41.1	48.8	0	9K	20.3
double-input flow									
FADE [12]	38.5	59.6	**41.8**	**23.1**	42.2	49.3	49K	–	–
SAPA-B [13]	37.8	59.2	40.6	22.4	41.4	49.1	**25K**	–	–
single-input flow									
Deconv [16]	37.2	57.8	40.2	20.5	41.0	48.0	262K	1.2M	18.8
Pixel shuffle [18]	37.6	58.5	40.6	21.1	41.4	48.3	2.4M	4.7M	18.2
CARAFE [21]	**38.6**	**59.9**	41.6	23.0	**42.3**	49.5	74K	199K+4×(25-D sm)[1]	**19.4**
DLU(Ours)	**38.6**	**59.9**	**41.8**	22.6	42.2	**50.1**	35K	**123K+1×(25-D sm)**	18.8

(25-D sm) represents the computation of softmax normalization for a 25-D vector.

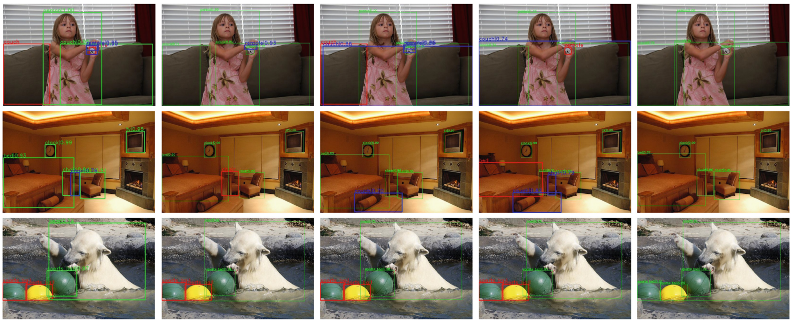
4.2 Object Detection

FPN. We take the FPN [8] as the backbone. Specifically, FPN constructs a feature pyramid to exploit multi-scale features with a top-down pathway and several lateral connections. As shown in Fig. 4, the lateral connections first are used to align the feature channels by applying a 1 × 1 convolution. Then, features of different scales are fused in the top-down pathway by upsampling the low-resolution features by 2×.

In our experiments, we adopted a feature pyramid of 5 feature levels: $\{P2, P3, P4, P5, P6\}$ with strides $\{4, 8, 16, 32, 64\}$, and 4 upsampling operators with a scaling ratio of 2 are used totally, as shown in Fig. 4. To compare the impact of different upsampling operations on object detection performance, we first replace the original nearest-neighbor interpolation with different upsampling operators in FPN, and then evaluate the detection performance. Table 2 shows the quantitative detection results achieved by FPN with different upsampling operators on the MS COCO 2017 validation set. We analyse the results from the following aspects:

(1) **Effectiveness**. By simply replacing the original nearest-neighbor interpolation with the proposed DLU in the framework of baseline FPN, the best detection performance is achieved, with a 1.2% improvement on mAP. We can also see that the proposed operation is beneficial for the detection performance with various object scales, since the AP_S, AP_M, AP_L (small, medium, large) are all improved. More quantitative results can be seen in Fig. 5.

(2) **Scalability (Memory Consumption)**. FADE [12] and SAPA [13] are methods with double-input flow, that is, their input not only comprises the



(a) BI (b) Deconv (c) Pixel shuffle (d) CARAFE (e) DLU

Fig. 5. Comparison of detection results of FPN [8] with different upsampling methods on the validation set of COCO 2017. The true positives, false positives, and false negatives are indicated by green, blue, and red rectangles, respectively. "BI" represent for Bilinear Interpolation. (Color figure online)

low-resolution feature to be upsampled, but also needs a high-resolution guiding feature. While they also successfully lighten CARAFE, their dependency on the availability of the high-resolution guiding features restricts their application scenarios.

For methods with single-input flow, identical detection performance is achieved when the original CARAFE or the proposed DLU is used in the FPN framework. However, the proposed DLU is much more lightweight compared with the original CARAFE. The detailed quantitative results in our case are shown in Table 2, 74K additional parameters are required per CARAFE, while our DLU only requires 35K additional parameters, up to half fewer parameters, but with comparable performance. This further indicates that the parameters of the original CARAFE are redundant. The reduction of redundancy in parameters makes the computation complexity of DLU lower than the original CARAFE in terms of FLOPs ($\approx$ 123k vs. 199k in our case).

(3) **Efficiency**. DLU has lower FLOPs than CARAFE, however, we also found that DLU's actual FPS is not improved compared to CARAFE. We attribute this to the inferior degree of parallelism in the interpolation operation in a GPU environment. Nevertheless, we are confident that with increased parallelism and code optimization in the future work, our method will undergo significant improvements in terms of efficiency.

Libra RCNN. Libra RCNN [17] is another representative neural architecture in object detection. It explores better utilization of the multi-level features of FPN [8] to generate more discriminative pyramid representations. Specifically, Libra RCNN proposed a BFP (Balanced Feature Pyramid) module, which first integrates the multi-level outputs of FPN by using multiple rescaling operations and

Table 3. Detection results achieved by Libra CNN [17] on MS COCO 2017 validation set with different upsampling operations. Note that the default upsampling operation (untick) used in the BFP module is the nearest neighbor interpolation.

Method	×2	×4	×8	×16	mAP	Params
Baseline					37.7	0
CARAFE	✓				37.9	74K
		✓			37.8	247K
			✓		38.1	939K
				✓	38.1	3.7M
	✓	✓	✓	✓	38.1	4.9M
DLU	✓				37.9	35K
		✓			38.0	49K
			✓		38.2	104K
				✓	38.3	326K
	✓	✓	✓	✓	**38.4**	515K

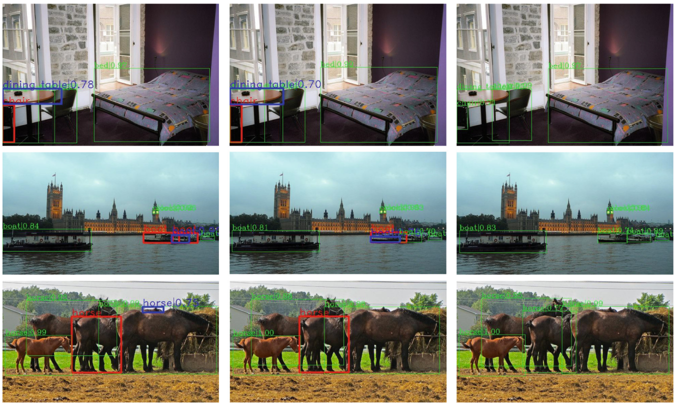
element-wise summation. Then, it refines the so-called balanced semantic features by several refinement operations (*e.g.,* convolution). Finally, the obtained feature is rescaled back to the pyramid resolutions and then added to the original FPN features. In our experiments, we build the BFP on a FPN with 5 output feature maps and rescale all features to the highest resolution. That is, 4 upsampling operations with ratio of 2×, 4×, 8× and 16× are needed in our BFP.

To fairly compare the performance of CARAFE and our DLU under different upsampling ratios, we only replace the upsampling layer in BFP, while keeping the rest of FPN unchanged. As shown in Table 3, the proposed DLU improve the baseline by 0.7% mAP, and is consistently better than CARAFE under different settings. Furthermore, our DLU requires far fewer parameters than CARAFE, which is more visible at large sampling rates (*e.g.,* 326K vs. 3.7M in the case of 16× upsampling). This further verifies that the parameters in CARAFE are indeed redundant, and excessively redundant parameters also increase the risk of model overfitting (*i.e.,* the model parameters increase but the performance decreases).

We conducted a qualitative comparison of the detection performance achieved by Libra RCNN by replacing all the upsampling operations in the BFP with different methods. The results are shown in Fig. 6.

4.3 Semantic Segmentation

To further evaluate the effectiveness of our proposed DLU, we conducted experiments on the task of semantic segmentation. These experiments provide additional insights into the applicability and potential benefits of DLU in other computer vision tasks beyond object detection.



(a) NI (b) CARAFE (c) DLU

Fig. 6. Comparison of detection results of Libra RCNN [17] with different upsampling methods on the validation set of COCO 2017. The true positives, false positives, and false negatives are indicated by green, blue, and red rectangles, respectively. "NI" represent for Nearest-neighbor Interpolation. (Color figure online)

For the task of semantic segmentation, we adopted the representative Semantic FPN model [7] as our baseline network. The architecture of Semantic FPN is based on the FPN model [8]. Specifically, in its module of semantic segmentation branch, each FPN level is upsampled using convolutions and 2× bilinear upsampling progressively until it reaches a 1/4 scale of the input. These outputs are then fused using summation operation and finally decoded to a pixel-wise output using a 4× bilinear upsampling.

In the above processing, upsampling operations are used for two modules: FUSE (to fuse feature maps with different spatial resolution) and Decoder (to decode the feature map to a pixel-wise output). In our experiments, we replaced the original bilinear interpolation used in these two modules with different upsampling operators and compared their segmentation performance.

Table 4. Segmentation results on ADE20K val with Semantic FPN [7]. Various upsampling methods are used in its FUSE and Decoder modules. The default upsampling operation (untick) is the bilinear interpolation.

Method	FUSE(×2,×4,×8)	Decoder(×4)	mIoU	Params
Baseline			35.53	0
CARAFE	✓		36.32	1.2M
		✓	36.57	239K
DLU	✓		**37.07**	165K
		✓	36.58	41K

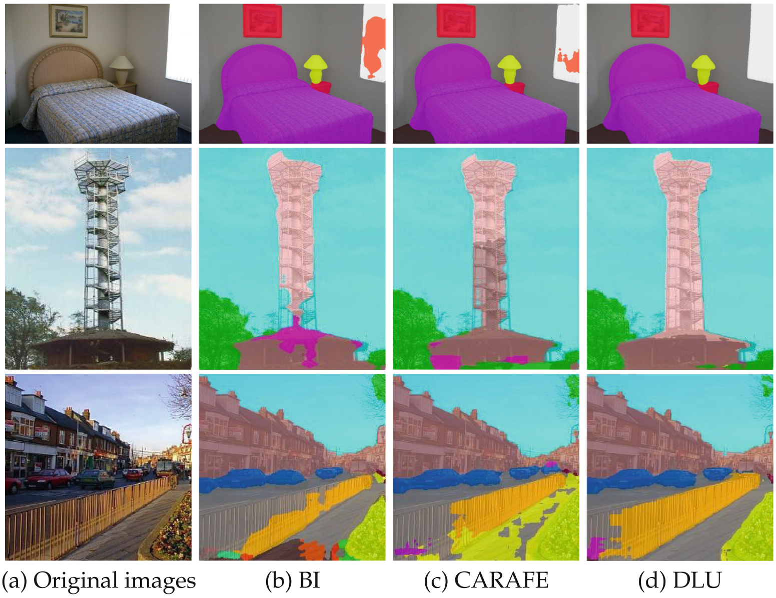

(a) Original images (b) BI (c) CARAFE (d) DLU

Fig. 7. Comparison of segmentation results of Semantic FPN [7] on ADE20K val. Different upsampling methods are used in its FUSE module.

To compare the performance of CARAFE and our proposed DLU under different upsampling ratios in the context of semantic segmentation, we made a simple modification to the FUSE module. Specifically, instead of enlarging the size of feature maps using convolutions and 2× upsampling progressively, we used one convolution layer and one upsampling operation with a rather large scaling ratio. For example, for feature maps that are 1/32 scale of the input, to reach 1/4 scale of the input, we first applied a 3×3 convolution layer on the feature maps and then resized the feature maps using an 8× upsampling operation directly.

Our experiments utilized semantic FPN with four feature levels, and we used three upsampling operations with ratios of 2×, 4×, and 8× in the FUSE module. Table 4 presents the quantitative semantic segmentation performance achieved by semantic FPN with different upsampling operations. The results show that, by replacing the baseline bilinear interpolation with DLU in either the FUSE or the Decoder module, the overall performance of semantic segmentation can be significantly improved. For instance, in the FUSE module, DLU outperformed the bilinear interpolation by 1.5% in terms of mIoU scores and also achieved better task performance than CARAFE, despite having fewer parameters. Figure 7 provides a qualitative comparison of segmentation performance achieved by semantic FPN with different sampling operations in its FUSE modules.

4.4 Discussion

Albeit much lower complexity compared with CARAFE, we also noticed that the overall framework of our DLU is still similar to the CARAFE. However, this does not mean that our DLU has only limited significance. Instead, the practical significance of our DLU lies in:

- **Versatility**. As a fundamental operation, upsampling is widely used in most deep neural networks. By replacing the original nearest-neighbor interpolation or bilinear interpolation operation, our DLU can be easily and seamlessly integrated into existing deep neural networks with different purposes, leading to better performance, but with few additional parameters.
- **Scalability**. As an upsampling operation with strong scalability, the proposed DLU is quite suitable for deep networks with aggressive upsampling ratios which are becoming increasingly common. For example, Upernet [25], Libra RCNN [17] utilizes 16×, 8× upsampling when combining feature maps with different resolutions in object detection; FCN [11] uses 32×, 16×, 8× upsampling in its FCN-8 s version in semantic segmentation; RCAN [27], SAN [2] uses 8× upsampling in their upscale module for super-resolution.

We hope that our DLU can be widely adopted and utilized in different tasks, not only improving the overall performance but also keeping the model complexity stable, and making a positive contribution to the field.

5 Conclusions

In this paper, we propose a lightweight upsampling operation termed Dynamic Lightweight Upsampling (DLU). The key idea is to avoid generating a large collection of independent kernels with massive trainable parameters, but sample in a source kernel space to reduce redundancy. Extensive experimental results on three mainstream vision tasks with four representation backbones show that the proposed DLU can be seamlessly integrated into existing networks, and effectively improve the performance of downstream tasks.

Acknowledgements. This work was supported by the National Natural Science Foundation of China under Grant 62001482 and Hunan Provincial Natural Science Foundation of China under Grant 2021JJ40676.

References

1. Chen, K., Wang, J., et al.: MMDetection: open mmlab detection toolbox and benchmark. arXiv preprint arXiv:1906.07155 (2019)
2. Dai, T., Cai, J., Zhang, Y., Xia, S.T., Zhang, L.: Second-order attention network for single image super-resolution. In: CVPR, pp. 11065–11074 (2019)
3. Gao, H., Zhu, X., Lin, S., Dai, J.: Deformable kernels: adapting effective receptive fields for object deformation. In: ICLR (2019)

4. Gao, H., Yuan, H., Wang, Z., Ji, S.: Pixel transposed convolutional networks. TPAMI **42**(5), 1218–1227 (2019)
5. He, K., Gkioxari, G., Dollar, P., Girshick, R.: Mask r-cnn. In: ICCV (2017)
6. Jia, X., De Brabandere, B., Tuytelaars, T., Gool, L.V.: Dynamic filter networks. NeurIPS **29**, 667–675 (2016)
7. Kirillov, A., Girshick, R., He, K., Dollár, P.: Panoptic feature pyramid networks. In: CVPR, pp. 6399–6408 (2019)
8. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: CVPR, pp. 2117–2125 (2017)
9. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
10. Liu, W., Lu, H., Fu, H., Cao, Z.: Learning to upsample by learning to sample. In: ICCV, pp. 6027–6037 (2023)
11. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR, pp. 3431–3440 (2015)
12. Lu, H., Liu, W., Fu, H., Cao, Z.: Fade: fusing the assets of decoder and encoder for task-agnostic upsampling. In: ECCV 2022, pp. 231–247. Springer, Heidelberg (2022). https://doi.org/10.1007/978-3-031-19812-0_14
13. Lu, H., Liu, W., Ye, Z., Fu, H., Liu, Y., Cao, Z.: Sapa: similarity-aware point affiliation for feature upsampling. NeurIPS **35**, 20889–20901 (2022)
14. Mazzini, D.: Guided upsampling network for real-time semantic segmentation. In: BMVC (2018)
15. Molchanov, P., Tyree, S., et al.: Pruning convolutional neural networks for resource efficient inference. arXiv preprint arXiv:1611.06440 (2016)
16. Noh, H., Hong, S., Han, B.: Learning deconvolution network for semantic segmentation. In: ICCV, pp. 1520–1528 (2015)
17. Pang, J., Chen, K., Shi, J., Feng, H., Ouyang, W., Lin, D.: Libra r-cnn: towards balanced learning for object detection. In: CVPR, pp. 821–830 (2019)
18. Shi, W., et al.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In: CVPR, pp. 1874–1883 (2016)
19. Vijayakumar, A., Vairavasundaram, S.: Yolo-based object detection models: a review and its applications. Multimedia Tools Appl. 1–40 (2024)
20. Wang, C.Y., Yeh, I.H., Liao, H.Y.M.: Yolov9: learning what you want to learn using programmable gradient information. arXiv preprint arXiv:2402.13616 (2024)
21. Wang, J., Chen, K., Xu, R., Liu, Z., Loy, C.C., Lin, D.: Carafe: content-aware reassembly of features. In: ICCV, pp. 3007–3016 (2019)
22. Wang, J., Chen, K., Xu, R., Liu, Z., Loy, C.C., Lin, D.: Carafe++: unified content-aware reassembly of features. TPAMI **44**, 4674–4687 (2021)
23. Wang, J., Yang, W., Guo, H., Zhang, R., Xia, G.S.: Tiny object detection in aerial images. In: ICPR, pp. 3791–3798. IEEE (2021)
24. Wang, Z., Chen, J., Hoi, S.C.: Deep learning for image super-resolution: a survey. TPAMI **43**(10), 3365–3387 (2020)
25. Xiao, T., Liu, Y., Zhou, B., Jiang, Y., Sun, J.: Unified perceptual parsing for scene understanding. In: ECCV 2018, pp. 418–434 (2018)
26. Xu, J., et al.: MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark (2020). https://github.com/open-mmlab/mmsegmentation
27. Zhang, Y., Li, K., Li, K., Wang, L., Zhong, B., Fu, Y.: Image super-resolution using very deep residual channel attention networks. In: ECCV 2018, pp. 286–301 (2018)

28. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ade20k dataset. In: CVPR, pp. 633–641 (2017)
29. Zhu, G., Wang, R., et al.: Rfnet: a refinement network for semantic segmentation. In: ICPR, pp. 670–676. IEEE (2022)
30. Zou, X., et al.: Segment everything everywhere all at once. NeurIPS **36** (2024)

Hand over Face Gesture Classification with Feature Driven Vision Transformer and Supervised Contrastive Learning

Kankana Roy[1,3], Aparna Mohanty[2,3](✉), and Rajiv Ranjan Sahay[3]

[1] Karolinska Institute, Solnavägen 1, Stockholm, Sweden
[2] School of Electronics Engineering, Vellore Institute of Technology, Vellore, Tamil Nadu 632014, India
`aparnamhnty@gmail.com`
[3] Indian Institute of Technology Kharagpur, Kharagpur, West Bengal 721302, India

Abstract. Hand gesture recognition and facial expression analysis are often treated as separate research problems in the existing literature. Although hand-over-face occlusions are commonly regarded as a limitation in facial expression recognition, they frequently serve as cues for mental states. In spontaneous conversations, hands over the face are frequent and informative about the participants' mental states, providing essential cues for understanding non-verbal communication.

This work systematically explores various hand-over-face gestures using recent machine learning algorithms. We establish a dataset comprising 30 hand-over-face gestures in laboratory settings, featuring 15 subjects mimicking subconscious scenarios. In our investigation, we propose a novel two-stage approach for hand-over-face gesture classification. Initially, an encoder is trained using contrastive learning, and subsequently, the trained encoder serves as a feature extractor for a feature-driven transformer classifier. This unique methodology explores the potential of contrastive learning for acquiring discriminative features for hand-over-face gesture classification.

To the best of our knowledge, our proposed model is the first to investigate contrastive learning for the hand-over-face gesture classification task. Our approach achieves new state-of-the-art results on the provided dataset, surpassing recent image classification methods in the literature, with an impressive accuracy of 88.68%. Furthermore, we evaluate the extensibility of our method on the widely-used facial expression dataset FER2013 and RAFDB, achieving a comparable performance of 71.30% and 85.74% on the test set, even though the proposed model is not specifically tailored to the facial expression recognition problem.

Keywords: hand-over-face · transformer · feature driven · contrastive learning · emotion classification

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 400–417, 2025.
https://doi.org/10.1007/978-3-031-78128-5_26

1 Introduction

Non-verbal communication plays a crucial role in human communication and expressing emotions. The ability to interpret non-verbal communication is essential in understanding emotion and predicting the intent and actions of others. The existing literature has extensively studied human emotion, expressed through facial expressions and bodily postures, for several decades [32]. Similarly, hand gestures have also been studied in the literature [22]. Further, simultaneous recognition of emotion and gestures, i.e., bi-modal emotion recognition, has also been addressed in previous research [36]. In facial expression recognition, the task is to classify sub-conscious spontaneous expressions, where hand occlusions are often considered unwanted and even removed [15]. However, humans often use body, face, and hand gestures while communicating, and hand-over-face (HoF) occlusions contain vital cues about the mental state of an individual.

Some existing work on HoF for emotion understanding exists in the literature [4,16,18,20,23]. The proposed approach in [18], detected, localized, and

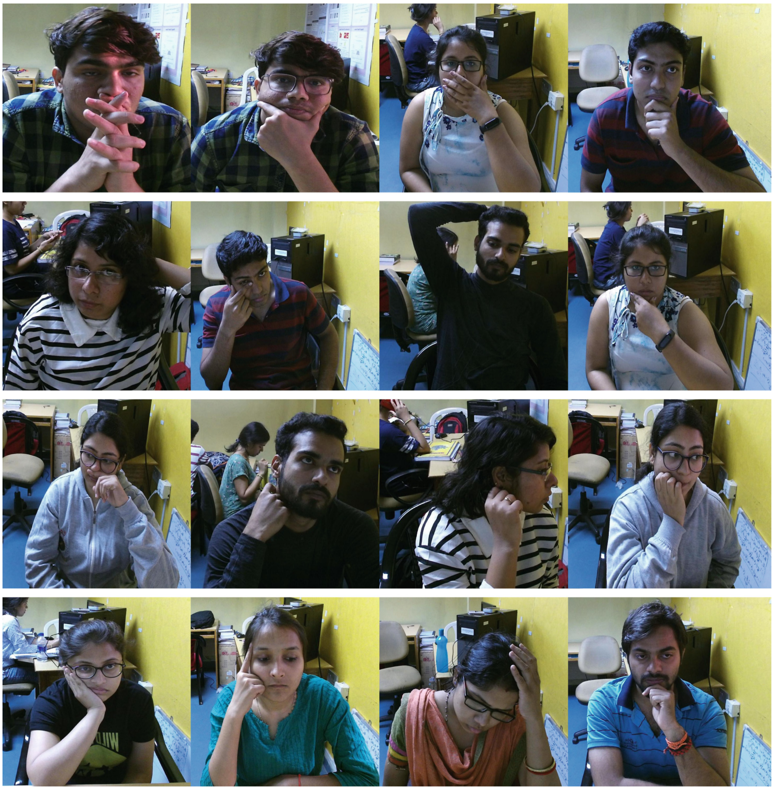


Fig. 1. Sample images of 30 classes in our proposed CVLab hand-over-face gesture dataset (HoFGD).

classified hand-over-face gestures from a video corpus using edge detectors and local binary patterns. The authors of [19] interpreted hand-over-face gestures using various cues such as hand action, hand gesture, and facial region occluded in terms of cognitive states. The work in [20] detects HoF occlusions and classifies HoF gestures using multimodal fusion with different spatial and spatiotemporal features. In [23] authors classify emotions despite faces being occluded by hands using HoG features and SVM classifier. In [4] HoF gestures are classified using a CNN. Mahmoud *et al.* [17,19] developed the first hand-over-face dataset focusing on emotions rather than gestures and attempted to classify moods of the person containing hand-over-face. The dataset is small with limited subjects and lacked structure. We propose a new hand-over-face gesture dataset and develop an approach to classify hand-over-face gestures.

Table 1. Comparative analysis of similar datasets and attributes of the proposed dataset HoFGD

Dataset	Number of classes	Annotation	Dimension	Spontaneity	Number of sample	Number of subjects
Cam3D [17]	12	weak	3D	spontaneous	108	7
FABO [6]	6	strong	2D	posed	210	24
HoFGD (This work)	30	strong	2D	posed	9000	15

The similar datasets in the HoF paradigm are summarized in Table 1. The first dataset is the Cam3D corpus [17], where the authors aimed to capture succinct natural emotion videos featuring hand-over-face events naturally occurring and associated with self-assessed mental states. The second dataset is the FABO dataset [6], which includes events with both hand and facial expressions. This corpus is posed. FABO dataset [6] is no longer available, and Cam3D dataset [17] has weak annotation and class and subject imbalance problems. Next is our proposed HoFGD dataset, and more details about this dataset can be found in Sect. 2.1. The proposed dataset is complex due to the presence of fine-grained gestures with small intra-class variation, background clutter, and multiple categories. We refer to this dataset as the CVLab Hand over Face Gesture dataset (HoFGD). A snapshot of this dataset, containing different gestures and subjects, is shown in Fig. 1.

In the earlier literature on HoF gestures, researchers predominantly relied on traditional approaches such as edge detectors, local binary patterns, HoG, SVM, and CNN. However, recent advancements have introduced more efficient models such as TResnet [28] for image classification tasks. With the advent of transformer-based models, exemplified by ViT [3], a recently proposed algorithm [35] aim to understand the importance of relative position encoding in influencing the sequence ordering of input tokens. Despite the potential benefits of this approach, it introduces additional complexity and enforced attention in the initial shallow layers of the transformer, which may not be optimal.

Self-supervised learning has emerged as a highly effective method for acquiring discriminative representations. With the introduction of contrastive learning [2], a multitude of studies [14,31] have harnessed its potential for additional training across various machine learning problems. Following the success of self-supervised contrastive learning, the development of supervised contrastive learning [9] ensued, wherein label information is effectively leveraged. In this study, our focus is on investigating task-specific discriminative features through the application of supervised contrastive learning [9]. This approach encourages the network to extract similar features from clusters of points belonging to the same class, while simultaneously pushing apart clusters of samples from different classes. The objective is to foster the learning of discriminative representations.

In addition, our contributions include

- Introducing the integration of supervised contrastive learning as an auxiliary task for HoF classification, employing TResnet [28] to extract superior feature maps.
- Utilizing feature maps derived from the contrastively trained TResnet in the subsequent training phase, choosing them over conventional image patches in the transformer architecture to enhance the efficacy in our classification task.
- Proposing a new Hand over Face Gesture Dataset (HoFGD), representing a significant stride forward in the realm of HoF analysis.
- Demonstrating the effectiveness of our proposed methodology in the HoF gesture classification task and providing additional results on the FER2013 and RAFDB [13] datasets for comprehensive validation.

2 Method

2.1 Datasets

To demonstrate its capability of HoF gesture classification, we conduct experiments over the HoFGD gesture dataset which was created for a basic understanding of non-verbal human communication. Inspired by Cam3D [17] dataset and nonverbal body language in communication [1,24] we selected 30 HoF gestures and asked our participants to enact with supervision by showing them sample images. The dataset is captured using a Kinnect V2 RGB-D sensor in the laboratory environment. A total of 15 subjects (8 males, 7 females) volunteered for capturing a total of 30 gesture classes. Each gesture is repeated 10 times for each right and left hand for a total of 20 times. Our experiments are conducted over a total of $9,000$ frames, exclusively focusing on RGB frames. The dimensions of the captured original RGB frames are $1,920 \times 1,080$. The dataset is divided into $2:1$ training and testing ratio containing 10 subjects for training and the remaining 5 subjects for testing. Images are cropped only to keep the subject's upper body to remove background clutter. This cropping is done manually. Because of manual cropping per subject, the frame size varies for each subject and it is less than the original frame size already mentioned. We resize the images to 384×384 for further processing into the proposed model. We introduce this dataset because it has a very small intra-class distance, making it harder for classification.

FER2013 stands out as an emotion recognition dataset that encapsulates challenging naturalistic conditions and difficulties. Unveiled at the International Conference on Machine Learning (ICML) in 2013, it swiftly emerged as a benchmark for assessing model proficiency in the realm of emotion recognition. It consists of a collection of 35,887 labeled images, categorized into seven different facial expressions: anger, disgust, fear, happiness, sadness, surprise, and a neutral expression. The images are relatively low-resolution grayscale images, originally sourced from various internet platforms. The dataset is widely utilized for evaluating and bench-marking the performance of facial expression recognition algorithms. Each image is annotated with a corresponding emotion label, making it a valuable resource for training and testing models aimed at understanding and classifying facial expressions in real-world scenarios.

The RAFDB [13] is a large-scale facial expression dataset comprising approximately 30,000 images. These images are annotated with seven basic emotion labels: happiness, sadness, surprise, fear, disgust, anger, and neutral. The dataset's diversity and high-quality annotations, obtained through reliable crowdsourcing, make it ideal for developing and evaluating facial expression recognition models in real-world conditions. We additionally experimented over FER2013 and RAFDB dataset to show the robustness and adaptability of our proposed method.

The metric used for comparison of different approaches is the average recognition rate that is defined by $\frac{\text{Total number of correctly identified gestures}}{\text{Total number of gestures}}$.

2.2 Proposed Model

As shown in Fig. 2, the following components comprise our framework:

- **Source data generation:** To create a positive pair $x_j \in \mathbb{R}^{B\times384\times384}$ for each image $x_i \in \mathbb{R}^{B\times384\times384}$, we employ the traditional image augmentation methods, which include shear, rotation, flip, and crop.
- **Feature extractor:** Our feature extractor is pretrained TResNet [28]. Specifically, we used the TResNet-L model [28] as a feature encoder which takes 384 × 384 sized images as input and the resulting feature is of size 2432. It should be noted that all data, including augmented data, share this feature extractor. It processes separately $x_i \in \mathbb{R}^{B\times384\times384}$ and $x_j \in \mathbb{R}^{B\times384\times384}$ and generates hidden features $h_i \in \mathbb{R}^{B\times2432}$ and $h_j \in \mathbb{R}^{B\times2432}$.
- **Projection head:** Following contrastive training strategy [9] we apply an MLP to convert features $h_i \in \mathbb{R}^{B\times2432}$ and $h_j \in \mathbb{R}^{B\times2432}$ to another representation $z_i \in \mathbb{R}^{B\times1024}$, and $z_j \in \mathbb{R}^{B\times1024}$. This step aid the final model to learn more discriminative feature.To obtain the projected representations $z_i \in \mathbb{R}^{B\times1024}$, and $z_j \in \mathbb{R}^{B\times1024}$, we apply an MLP to $h_i \in \mathbb{R}^{B\times2432}$ and $h_j \in \mathbb{R}^{B\times2432}$ with a single hidden layer, following SupCon [9] loss. We observe that this MLP projection aids our model in learning more accurate discriminative features, with the help of SupCon [9] loss.
- **Contrastive loss:** For contrastive learning, we use SupCon loss [9] to obtain the projected representations $z \in \mathbb{R}^{B\times1024}$. Details of the supervised contrastive learning are discussed in Sect. 2.2.1.

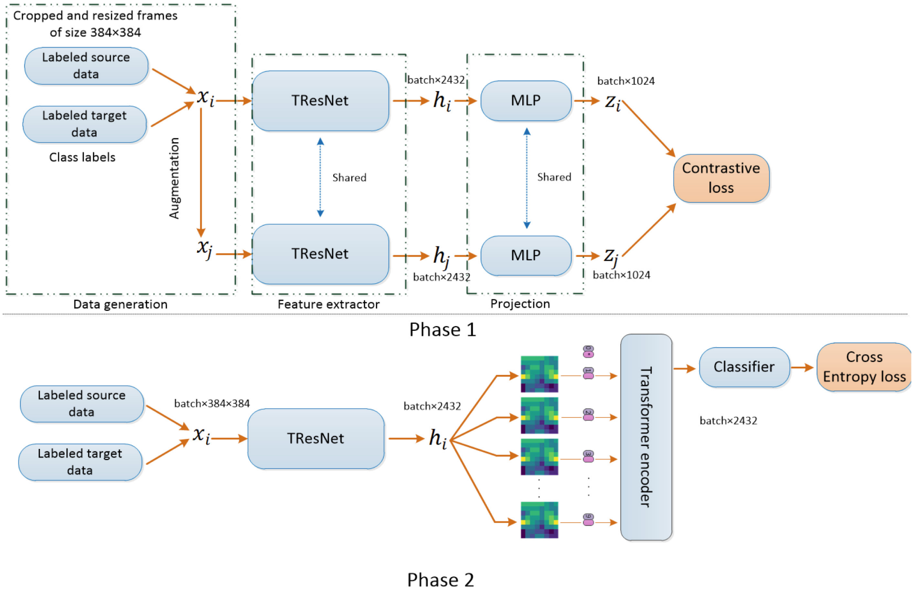


Fig. 2. The pipeline of the proposed method. The flow of labeled data from the dataset is indicated by the orange arrows. Phase 1 represents contrastive training and phase 2 represents supervised training for the transformer. (Color figure online)

- **Transformer classifier:** To perform the final classification we employ a transformer ViT along with cross-entropy loss to calculate HoF classification loss L_{cls}. Details can be found in Sect. 2.2.2.

To address the problem of HoF gesture classification, we propose our framework with contrastive learning for feature extraction in the first step. In the second stage, we replace the patch embedding layer of the transformer with the features extraction layer from the feature encoder.

2.2.1 Supervised Contrastive Training for HoF Gesture Classification

As shown in the phase 1 of Fig. 2, given an input batch X of N data points where each data point $x \in \mathbb{R}^{B\times384\times384}$ produces two instances, ($x_i \in \mathbb{R}^{B\times384\times384}$, $x_j \in \mathbb{R}^{B\times384\times384}$), by data augmentation. To make learning robust and invariant to noise, we use four different data augmentation techniques: flip, crop, shear, and rotation. To achieve a normalized embedding, the encoder TResNet $f(.)$ forward propagates both instances of data $x_i \in \mathbb{R}^{B\times384\times384}$, $x_j \in \mathbb{R}^{B\times384\times384}$. This intermediate representation, $h \in \mathbb{R}^{B\times2432}$, is further transmitted during training through an MLP, $g(.)$, which would be discarded during inference. The output $z \in \mathbb{R}^{B\times 1024}$ from MLP network ($g(.)$) are utilized to estimate contrastive loss. After discarding MLP in the next phase the intermediate representation $h \in \mathbb{R}^{B\times 2432}$ is utilized to train the transformer classifier. The intermediate features $h \in \mathbb{R}^{B\times 2432}$ for our method are the normalized activations of the final pooling layer of the TResNet encoder.

Given a set of N randomly selected samples and its corresponding label pairs $\{x_k, y_k\}_{k=1,\ldots,N}$, the training batch in consideration is composed of $2N$ pairs, $\{\tilde{x}_k, \tilde{y}_k\}_{k=1,\ldots,2N}$, where $\tilde{x}_{2k}$ and $\tilde{x}_{2k-1}$ are two random data augmentations of x_k and $\tilde{y}_{2k} = \tilde{y}_{2k-1} = y_k$. The supervised contrastive loss can be defined as follows:

$$\begin{aligned} L^{sup} &= \sum_{i=1}^{2N} L_i^{sup} \\ L_i^{sup} &= \frac{-1}{2N_{\tilde{y}_i}-1} \sum_{j=1}^{2N} \mathbb{1}_{[i\neq j]} \cdot \mathbb{1}_{[\tilde{y}_i \neq \tilde{y}_j]} \cdot \\ &\log \frac{\exp(\mathrm{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[} k \neq i] \exp(\mathrm{sim}(z_i, z_k)/\tau)} \end{aligned} \tag{1}$$

According to Eq. 1, the supervised contrastive loss expands the number of positive pairs of $\tilde{x}_i$ so that all sub-data with the same label are viewed as positive pairs. This shows that the supervised contrastive loss is an extension of the self-supervised contrastive loss.

2.2.2 Feature Driven Transformer for HoF Gesture Classification

As shown in the second phase of Fig. 2 a standard Transformer generally contains a patch embedding layer that converts input images into a 1D sequence of token embedding. In this work, we propose to replace the patch embedding layer by using a feature extraction layer from encoder f, and pass a 1D sequence of flattened features $h_i \in \mathbb{R}^{N\times(P^2\cdot C)}$, where $(P \times P)$ is the resolution of each feature, N is the number of features, which also acts as the Transformer's functional sequence length and C is the number of channels. $\mathbf{E}$ represents the learned embeddings of tokens or positional encodings, while $\mathbf{E_{pos}}$ typically represents positional encodings to incorporate position information into the model. To match latent vector size D across all of the Transformer's layers, we use a trainable linear projection layer (Eq. 5). The Transformer encoder constitutes alternating layers of multi-headed self-attention (MSA) and MLP blocks. A Layernorm (LN) layer is applied followed by a residual layer before each previously mentioned block. The MLP consists of two fully connected layers followed by GELU non-linearity. The functionality of the Transformer can be summarized as:

$$w_0 = [h_{class}; h_p^1\mathbf{E}; h_p^2\mathbf{E}; ...; h_p^N\mathbf{E}] + \mathbf{E}_{pos},$$
$$\mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times D}, \mathbf{E}_{pos} \in \mathbb{R}^{(N+1) \times D} \tag{2}$$
$$w_l^{'} = \text{MSA}(\text{LN}(w_{l-1})) + w_{l-1}, \quad l = 1, .., L \tag{3}$$
$$w_l = \text{MLP}(\text{LN}(w_l^{'})) + w_l^{'}, \quad l = 1, .., L \tag{4}$$
$$y = \text{LN}(w_L^0) \tag{5}$$

2.3 Implementation Details

Supervised Contrastive Learning: In contrast to classic contrastive learning [2], our approach utilizes data exclusively from the target dataset, omitting any auxiliary dataset. Consequently, we adopt supervised contrastive learning. In contrast to the former, which amalgamates unlabeled datasets based on their inter-distance on a feature plane, our method independently learns on the labeled dataset.

Hyperparameter Tuning: As baseline approaches, we select TResNet, TResNet + ML Decoder, and TResNet + ViT. Furthermore, we employ TResNet as the patch embedding layer for the proposed model. The activation function for the projection head is Leaky ReLU, and the MLP classifier for TResNet is chosen. Vision transformer activation functions include Layer Normalization and GeLU. In terms of the training approach, we train TResNet for 100 epochs using the SGD optimizer. A linear learning rate scheduler with an initial learning rate of 0.001 is adopted, incorporating warm-up steps constituting 0.1 of the total training steps. For ViT, we keep TResNet fixed, utilizing it as a patch embedding layer, and train ViT for 30 epochs with an initial learning rate of 0.001, following the same setting as TResNet. Regarding contrastive learning, we set the temperature value for SupCon loss at 0.07 and $1e^{-4}$ as the weight decay parameter for L2.

3 Results

To compare our proposed method with other approaches, we evaluate it using the HoFGD, FER2013, and RAFDB datasets consistently throughout the experiments, as described in Sect. 2.1.

3.1 Baselines

We compare our model with state-of-the-art image classification models VGG19, TResNet, and ViT and its variation:

- **VGG19:** VGG19 is a very well-established deep-learning classification technique. VGG19 model was initialized with the pre-trained ImageNet [30] weight and further finetuned on the training data and subsequently test on the testing data.

- **TResNet:** TResNet is a recent efficient variation of ResNet which retains the same hardware efficiency while improving accuracy. A similar evaluation strategy is applied as discussed as VGG19.
- **TResNet + ML Decoder:** We adopted TResNet with ML Decoder head [29] instead of last layers of MLP classification layers.
- **ViT:** Another baseline is a very recently proposed ViT model [3] for image classification.
- **TResNet + ViT:** We also evaluated our approach against TResNet + ViT without contrastive learning by using TResNet as a feature extractor instead of feeding image patches in ViT.
- **Proposed model:** As shown in Fig. 2, TResNet is the encoder in our model. In phase 1 we add one layer of MLP and in phase 2 we add a transformer layer as a classifier similar to TResNet + ML Decoder or TResNet + ViT. The main difference is with this baseline model that the training consists of phase 2 only without any contrastively pre-trained TResNet encoder which boosts the performance.

3.2 Comparison with Baselines

Table 2. HoF gesture classification accuracy on proposed HoFGD dataset and comparison against baseline approaches.

Method	Accuracyv(%)
VGG19	69.77
TResNet	82.28
TResNet + ML Decoder	84.06
ViT	83.54
TResNet + ViT	83.48
Proposed	**88.68**

Table 2 provides an overview of the experiments conducted on our proposed HoFGD dataset, incorporating various baseline methods and our proposed approach. We attained accuracy percentages of 69.77%, 82.28%, and 83.54% with VGG19, TResNet, and ViT, respectively. The inclusion of TResNet+ML Decoder and TResNet+ViT yielded improved results of 84.06% and 83.48%. Notably, the use of TResNet as an encoder in combination with a sophisticated classifier demonstrated enhanced performance. In our final proposed approach, we achieved an accuracy of 88.68%. Significantly, our algorithm consistently outperformed the TResNet baseline by an average margin of 6.4%. By leveraging the robust discriminative features of TResNet, our model also surpassed other baselines such as ViT and ML Decoder by a substantial margin. The introduction of contrastive training further elevated performance, enabling our method

Table 3. FER classification accuracy on FER2013 dataset and comparison against baseline approaches.

Method	Accuracy(%)
VGG19	63.63
TResNet	67.70
TResNet + ML Decoder	66.14
ViT	65.03
TResNet + ViT	67.45
Proposed	**71.30**

Table 4. FER classification accuracy on RAFDB dataset [13] and comparison against baseline approaches.

Method	Accuracy(%)
VGG19	63.20
TResNet	78.91
TResNet + ML Decoder	84.68
ViT	81.00
TResNet + ViT	66.72
Proposed	**85.74**

to surpass TResNet, ViT, and TResNet+ViT in HoF gesture classification on the HoFGD dataset. These results underscore the effectiveness of our model in providing discriminative features for classification tasks..

In Table 3, we present the outcomes of our experiments conducted on the FER2013 dataset, employing the same set of baseline methods alongside our proposed approach. The results indicate performance percentages of 63.63%, 67.70%, and 65.03% achieved with VGG19, TResNet, and ViT, respectively. When incorporating TResNet+ML Decoder and TResNet+ViT, we observed improved percentages of 66.14% and 67.45%. Notably, our proposed approach demonstrated superior performance, reaching 71.30%. Remarkably, our algorithm outperforms the TResNet baseline by an average margin of 4.4%.

In Table 4, we detail the results of our experiments on the RAFDB dataset [13], using the same baseline methods and our proposed approach. The results show performance rates of 63.20%, 78.91%, and 81.00% for VGG19, TResNet, and ViT, respectively. By incorporating TResNet+ML Decoder and TResNet+ViT, we observed improved rates of 84.68% and 66.72%. Notably, our proposed approach achieved superior performance, reaching 85.74%. Impressively, our method outperforms the TResNet baseline by an average margin of 6.9%. Capitalizing on the robust discriminative features of TResNet, our model also surpasses other baseline models like ViT and ML Decoder by a substantial margin. These findings underscore the effectiveness of our model in generalizing to

Table 5. HoF gesture classification accuracy on proposed HoFGD dataset and comparison against state-of-the-art approaches.

Method	Accuracy
CLIP [27]	86.2%
ALIGN [7]	86.8%
Florence [38]	87.61%
BASIC [25]	88.0%
CoCa [37]	88.44%
Proposed	88.68%

other related classification tasks, even without explicit modeling for the fine-grained task of facial expression recognition.

3.3 Comparison with State-of-the-Art Methods

The classification accuracy achieved by various state-of-the-art (SOTA) approaches [7,25,27,37,38] on the proposed HoFGD dataset is presented in Table 5. It's noteworthy that the studies in [7,27] showcase the learning of image and text representations for cross-modal alignment tasks using dual-encoder models, pre-trained with contrastive learning on noisy image-to-text pairs. Specifically, a classification accuracy of 86.2% and 86.8% is reported for [7,27], respectively. The Florence [38] approach aimed to generalize representations from entire scenes to fine objects, transitioning from images to videos, and extending from RGB to modalities like captions or depth. Furthermore, the Florence [38] model exhibits adaptability for various tasks such as image captioning, classification, retrieval, and action recognition. It attains a classification accuracy of 87.61% as detailed in Table 5. The BASIC framework [25] experimented with fine-tuning using contrastive loss after pre-training on a large-scale image annotation dataset [30] through cross-entropy. This effort resulted in a classification accuracy of 88.0% on the proposed HoFGD dataset. Recently, the work presented in [37] leveraged the advantages of both contrastive learning and image-to-caption generation through a straightforward architecture.

In Table 6, we present an overview of various SOTA approaches [5,8,21,26,33] for facial expression recognition on the FER2013 dataset. It's crucial to note that our primary objective in conducting these experiments on the FER2013 dataset is not to establish a new benchmark for facial expression recognition but to assess the generalizability of our proposed models to related classification problems. Unfortunately, there isn't an alternative dataset for the HoF problem that suits our needs. The previous dataset [20] is deemed insufficiently structured for our classification task. Our approach is not specifically tailored for facial expression recognition compared to other SOTA approaches outlined in Table 6. We begin by comparing our results with DeepEmotion [21], which achieved an accuracy of 70.02%, and then against CNNHypOpt [33] with an accuracy

Table 6. Facial expression recognition accuracy on FER2013 dataset and comparison against state-of-the-art approaches.

Method	Accuracy(%)
DeepEmotion [21]	70.02
CNNHypOpt [33]	72.16
VGGNet [8]	73.28
Residual Masking Network [26]	74.14
Ad-Corre [5]	72.03
Proposed	71.30

of 72.16%. Additionally, we compare with other methods such as VGGNet [8], Residual Masking Network [26], and Ad-Corre [5], achieving accuracies of 73.28%, 74.14%, and 72.03%, respectively. Our proposed method achieves an accuracy of 71.30%, surpassing DeepEmotion [21] and demonstrating competitive performance with most others. It's noteworthy that the top-performing approaches, including [26], incorporate specific modules to enhance the fine-grained features of facial expressions, contributing to their high performance levels.

Table 7. Facial expression recognition accuracy on RAFDB dataset [13] and comparison against state-of-the-art approaches.

Method	Accuracy(%)
RAN [34]	86.9
FaceBehaviorNet [11]	78.0
C-EXPR-NET [10]	87.5
C-MT-PSR [12]	84.8
C-MT-VGGFACE [12]	81.4
Proposed	85.74

In Table 7, we provide an overview of various SOTA approaches [10–12,34] for facial expression recognition on the RAFDB dataset [13]. Similar to our previous experiments on the FER2013 dataset, our objective is not to set a new benchmark for facial expression recognition but to evaluate the generalizability of our proposed models to related classification tasks. Our approach is not specifically optimized for facial expression recognition compared to other SOTA methods listed in Table 7. We first compare our results with RAN [34], which achieved an accuracy of 86.9%, followed by FaceBehaviorNet [11] with an accuracy of 78.0%. Additionally, we consider other methods such as C-EXPR-NET [10], C-MT-PSR [12], and C-MT-VGGFACE [12], which achieved accuracies of 87.5%, 84.8%, and 81.4%, respectively. Our proposed method attains an

Table 8. Ablation studies on HoFGD dataset.

Method	Single phase training			Data augmentation			Complete training
	Only phase 2	Only phase 2 No augmentation	Only phase 1	Horizontal flip	Crop	No augmentation	
Accuracy(%)	83.48	80.54	85.04	87.74	85.21	83.46	88.68

accuracy of 85.74%, demonstrating competitive performance with most of the existing approaches.

3.4 Ablation Studies

The ablation studies conducted for our methodology are summarized as follows. Firstly, we analyze the impact of solely conducting the phase 2 experiment without any prior phase 1 training. The accuracy obtained on HoFGD dataset is 83.48% as shown in the first column of Table 8. Secondly, we conduct a phase 2 experiment with no data augmentation to examine the effect of no phase 1 training and data augmentation. The accuracy obtained on the HoFGD dataset is 80.54% as shown in the second column of Table 8. Next, we scrutinize the consequences of exclusively performing phase 1 training without subsequent phase 2 experimentation. Accuracy obtained is 85.04% as shown in third column of Table 8. Next, we delve into the comparison between employing and omitting data augmentation techniques, shedding light on its effects in the fourth column of Table 8. Furthermore, we extend our ablation studies to investigate the specific impact of data augmentation during phase 2. These additional findings are meticulously presented in Table 8. Lastly, we also include the final result using both phases 1, and 2 and also data augmentations to show the full accuracy gained using this method. These comprehensive ablation studies allow us to discern the individual contributions and significance of each experimental component in our proposed methodology, contributing to a nuanced understanding of its performance.

4 Discussion

The utilization of contrastive learning methods to enhance the feature encoder in TResNet models has proven to be highly effective. By leveraging data-driven training techniques, we have been able to significantly enhance the robustness of our model, especially in the domain of HoF classification tasks. This approach not only enhances overall performance but also surpasses models relying solely on CNNs or transformers. A critical observation arises when exploring the impact of the two-stage training strategy on classification performance. There appears to be a trade-off between the overhead of the two-stage training process and the resultant classification outcomes. Striking this delicate balance is imperative for optimizing the model's performance. As anticipated, the integration of

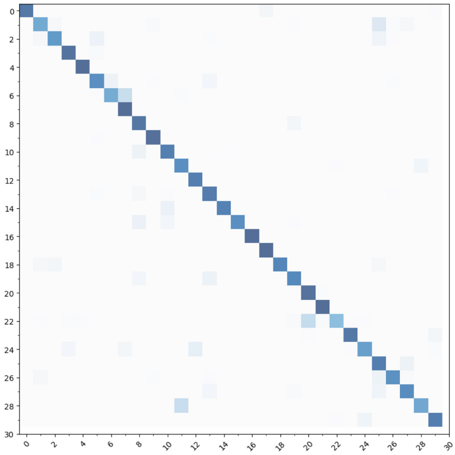


Fig. 3. Confusion matrix for HoFGD dataset with TResNet + transformer layer with supervised contrastive learning.

contrastive learning with transformers yields promising results in HoF classification. This amalgamation of methodologies, encompassing contrastive learning, CNN feature encoding, and transformers, equips the model with enhanced discriminative capabilities. Consequently, we observe a substantial improvement in classification accuracy in the HoF domain. The focus of this work is specifically on HoF gesture, it would be out of scope to showcase the results of the proposed approach on unrelated datasets to elaborate test on generalization. However, we do present the performance of our approach on similar topics facial expression datasets FER2013 and RAFDB. Larger datasets for HoF are not available in the literature. Moreover, it might work for natural image classification tasks, but it may be unrelated since the task is not streamlined with our problem. Biases may arise due to reliance on the dataset. However, to mitigate this, we strive to generalize the proposed approach. We achieve generalization by expanding our evaluation to encompass diverse datasets and real-world scenarios. However, when transitioning to FER, the observed performance gains do not quite reach the SOTA level. This discrepancy is likely attributable to the fine-tuned nature of the compared methodologies, tailored to address the unique intricacies of FER. Figure 3 illustrates the confusion matrix for the HoFGD dataset, employing the TResNet + ViT framework with contrastive learning. Meanwhile, Fig. 4, and Fig. 5 offers insights into the classification results for the FER2013 and RAFDB [13] dataset using the same framework. Notably, our model encounters challenges in distinguishing between the contempt and disgust classes, owing to the subtle inter-class differences inherent in these expressions.

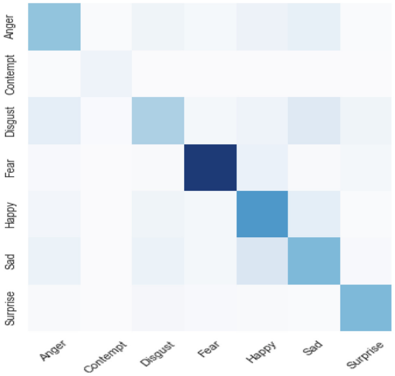


Fig. 4. Confusion matrix for FER2013 dataset with TResNet + transformer layer with supervised contrastive learning.

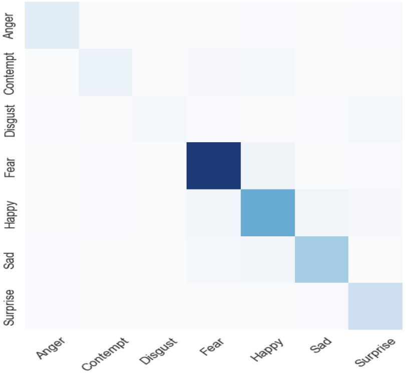


Fig. 5. Confusion matrix for RAFDB dataset [13] with TResNet + transformer layer with supervised contrastive learning.

5 Conclusion

In this paper, we approach the challenging task of HoF classification by framing it as a gesture recognition problem. Acknowledging the potential limitations of existing datasets in this domain, we took the initiative to create the HoFGD dataset, meticulously designed to represent sub-conscious HoF gestures which naturally occur when humans communicate or contemplate while encompassing a broader spectrum of classes and subjects. To address this complex problem, we devised a two-stage training paradigm. In the initial stage, we train a TResNet feature encoder via contrastive training. Building upon this foundation, the second stage introduces a new feature-driven vision transformer. Here, we integrate the TResNet features gleaned from the previous stage into the patch embedding layers of the vision transformer. The empirical results showcase the efficacy of our approach. Through various experiments, we demonstrate the superiority of our proposed methodology over contemporary deep learning algorithms in the realm of HoF gesture recognition. Moreover, we demonstrate the extensibility of our framework by extending its application to the domain of FER. Despite the distinct challenges posed by FER, our approach showcases remarkable adaptability, competing against the performance of other tailored deep learning frameworks. The source code is made publicly available on Github (https://github.com/kankanar/ICPR-2024). Furthermore, for researchers interested in the HoFGD dataset, it is available upon request.

References

1. https://www.indiabix.com/body-language/hand-to-face-gestures/ (2019). Accessed 19 July 2008
2. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International Conference on Machine Learning, pp. 1597–1607. PMLR (2020)
3. Dosovitskiy, A., et al.: An image is worth 16×16 words: transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
4. El Sayed, A.S., Tsuji, A., Terada, K.: Classification of hand over face gestures using deep learning. In: Proceedings of International Conference on Quality Control Artificial Vision, vol. 11172, pp. 319–324. SPIE (2019)
5. Fard, A.P., Mahoor, M.H.: Ad-corre: adaptive correlation-based loss for facial expression recognition in the wild. IEEE Access **10**, 26756–26768 (2022)
6. Gunes, H., Piccardi, M.: A bimodal face and body gesture database for automatic analysis of human nonverbal affective behavior. In: International Conference on Pattern Recogition, vol. 1, pp. 1148–1153. IEEE (2006)
7. Jia, C., et al.: Scaling up visual and vision-language representation learning with noisy text supervision. In: International. Conference on Machine Learning, pp. 4904–4916. PMLR (2021)
8. Khaireddin, Y., Chen, Z.: Facial emotion recognition: state of the art performance on FER2013. arXiv preprint arXiv:2105.03588 (2021)
9. Khosla, P., et al.: Supervised contrastive learning. Neur. Inf. Process. Syst. **33**, 18661–18673 (2020)
10. Kollias, D.: Multi-label compound expression recognition: C-EXPR database & network. In: IEEE International Conference on Computer Vision and Pattern Recogition, pp. 5589–5598 (2023)
11. Kollias, D., Sharmanska, V., Zafeiriou, S.: Distribution matching for heterogeneous multi-task learning: a large-scale face study. arXiv preprint arXiv:2105.03790 (2021)
12. Kollias, D., Sharmanska, V., Zafeiriou, S.: Distribution matching for multi-task learning of classification tasks: a large-scale study on faces & beyond. In: AAAI Conference on Artificial Intelligence, vol. 38, pp. 2813–2821 (2024)
13. Li, S., Deng, W., Du, J.: Reliable crowdsourcing and deep locality-preserving learning for expression recognition in the wild. In: IEEE International Conference on Computer Vision Pattern Recogition, pp. 2852–2861 (2017)
14. Li, T., Chen, X., Zhang, S., Dong, Z., Keutzer, K.: Cross-domain sentiment classification with contrastive learning and mutual information maximization. In: International Conference on Acoustics Speech Signal Processing, pp. 8203–8207. IEEE (2021)
15. Li, Y., Zeng, J., Shan, S., Chen, X.: Occlusion aware facial expression recognition using CNN with attention mechanism. IEEE Trans. Image Process. **28**(5), 2439–2450 (2018)
16. Mahmoud, M., Baltrušaitis, T., Robinson, P.: Automatic analysis of naturalistic hand-over-face gestures. ACM Trans. Interactive Intell. Syst. **6**(2), 1–18 (2016)
17. Mahmoud, M., Baltrušaitis, T., Robinson, P., Riek, L.D.: 3D corpus of spontaneous complex mental states. In: D'Mello, S., Graesser, A., Schuller, B., Martin, J.-C. (eds.) ACII 2011. LNCS, vol. 6974, pp. 205–214. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24600-5_24

18. Mahmoud, M., El-Kaliouby, R., Goneid, A.: Towards communicative face occlusions: machine detection of hand-over-face gestures. In: Kamel, M., Campilho, A. (eds.) ICIAR 2009. LNCS, vol. 5627, pp. 481–490. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02611-9_48
19. Mahmoud, M., Robinson, P.: Interpreting hand-over-face gestures. In: D'Mello, S., Graesser, A., Schuller, B., Martin, J.-C. (eds.) ACII 2011. LNCS, vol. 6975, pp. 248–255. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24571-8_27
20. Mahmoud, M.M., Baltrušaitis, T., Robinson, P.: Automatic detection of naturalistic hand-over-face gesture descriptors. In: Proceedings of International Conference on Multimodal Interaction, pp. 319–326 (2014)
21. Minaee, S., Minaei, M., Abdolrashidi, A.: Deep-emotion: facial expression recognition using attentional convolutional network. Sensors **21**(9), 3046 (2021)
22. Nunez, J.C., Cabido, R., Pantrigo, J.J., Montemayor, A.S., Velez, J.F.: Convolutional neural networks and long short-term memory for skeleton-based human activity and hand gesture recognition. Pattern Recogit. **76**, 80–94 (2018)
23. Patii, J., Patei, R., Kothiya, S.: Human mental states recognition under face occlusion. In: Proceedings of International Conference Innovations and Information Embedded Communication System, pp. 1–5. IEEE (2017)
24. Pease, B., Pease, A.: The definitive book of body language: The hidden meaning behind people's gestures and expressions. Bantam (2008)
25. Pham, H., et al.: Combined scaling for open-vocabulary image classification. arXiv preprint arXiv:2111.10050 (2021)
26. Pham, L., Vu, T.H., Tran, T.A.: Facial expression recognition using residual masking network. In: International Conference on Pattern Recongition, pp. 4513–4519. IEEE (2021)
27. Radford, A., et al.: Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning, pp. 8748–8763. PMLR (2021)
28. Ridnik, T., Lawen, H., Noy, A., Ben Baruch, E., Sharir, G., Friedman, I.: Tresnet: high performance gpu-dedicated architecture. In: Proceedings of IEEE/CVF Winter Conference on Application of Computer Vision, pp. 1400–1409 (2021)
29. Ridnik, T., Sharir, G., Ben-Cohen, A., Ben-Baruch, E., Noy, A.: Ml-decoder: scalable and versatile classification head. arXiv preprint arXiv:2111.12933 (2021)
30. Russakovsky, O., et al.: Imagenet large scale visual recognition challenge. Intl. J. Comput. Vis. **115**, 211–252 (2015)
31. Song, W., Han, J., Song, H.: Contrastive embedding learning method for respiratory sound classification. In: International Conference on Acoustics Speech Signal Processing, pp. 1275–1279. IEEE (2021)
32. Tian, Y., Cheng, J., Li, Y., Wang, S.: Secondary information aware facial expression recognition. IEEE Sig. Process. Lett. **26**(12), 1753–1757 (2019)
33. Vulpe-Grigoraşi, A., Grigore, O.: Convolutional neural network hyperparameters optimization for facial emotion recognition. In: International Symposium on Advanced Topics Electrical Engineering, pp. 1–5. IEEE (2021)
34. Wang, K., Peng, X., Yang, J., Meng, D., Qiao, Y.: Region attention networks for pose and occlusion robust facial expression recognition. IEEE Trans. Image Process. **29**, 4057–4069 (2020)
35. Wu, K., Peng, H., Chen, M., Fu, J., Chao, H.: Rethinking and improving relative position encoding for vision transformer. In: Proceedings of IEEE/CVF International Conference on Computer Vision, pp. 10033–10041 (2021)

36. Yan, J., Zheng, W., Xu, Q., Lu, G., Li, H., Wang, B.: Sparse kernel reduced-rank regression for bimodal emotion recognition from facial expression and speech. IEEE Trans. Multimedia **18**(7), 1319–1329 (2016)
37. Yu, J., Wang, Z., Vasudevan, V., Yeung, L., Seyedhosseini, M., Wu, Y.: Coca: contrastive captioners are image-text foundation models. arXiv preprint arXiv:2205.01917 (2022)
38. Yuan, L., et al.: Florence: a new foundation model for computer vision. arXiv preprint arXiv:2111.11432 (2021)

TabSeq: A Framework for Deep Learning on Tabular Data via Sequential Ordering

Al Zadid Sultan Bin Habib[1](✉), Kesheng Wang[2], Mary-Anne Hartley[3],
Gianfranco Doretto[1], and Donald A. Adjeroh[1]

[1] Lane Department of Computer Science and Electrical Engineering, West Virginia University, Morgantown, WV 26506, USA
ah00069@mix.wvu.edu,
{gianfranco.doretto,donald.adjeroh}@mail.wvu.edu
[2] College of Nursing, University of South Carolina, Columbia, SC 29208, USA
kesheng@mailbox.sc.edu
[3] Yale University School of Medicine, New Haven, CT 06510, USA
mary-anne.hartley@yale.edu

Abstract. Effective analysis of tabular data still poses a significant problem in deep learning, mainly because features in tabular datasets are often heterogeneous and have different levels of relevance. This work introduces TabSeq, a novel framework for the sequential ordering of features, addressing the vital necessity to optimize the learning process. Features are only sometimes equally informative, and for certain deep learning models, their random arrangement can hinder the model's learning capacity. Finding the optimum sequence order for such features could improve the deep learning models' learning process. The novel feature ordering technique, which we provide in this work, is based on clustering and incorporates both local ordering and global ordering. It is designed to be used with a multi-head attention mechanism in a denoising autoencoder network. Our framework uses clustering to align comparable features and improve data organization. Multi-head attention focuses on essential characteristics, whereas denoising autoencoder highlights important aspects by rebuilding from distorted inputs. This method improves the capability to learn from tabular data while lowering redundancy. Our research demonstrating improved performance through appropriate feature sequence rearrangement utilizing raw antibody microarray and two other real-world biomedical datasets validates the impact of feature ordering. These results demonstrate that feature ordering can be a viable approach to improved deep learning of tabular data.

Keywords: Deep Learning · Tabular Data · Feature Ordering

1 Introduction

Deep learning has transformed how we handle and comprehend diverse data types, resulting in unparalleled progress in numerous fields. Deep learning mod-

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 418–434, 2025.
https://doi.org/10.1007/978-3-031-78128-5_27

els have outperformed conventional techniques in audio analysis, picture identification, and Natural Language Processing (NLP), opening the door to new applications previously thought impractical. For example, Convolutional Neural Networks (CNNs) have emerged as the mainstay of image-processing applications, demonstrating exceptional performance in picture classification, object recognition, and other applications [15,22]. In NLP, Transformer-like models have established new benchmarks for text summarization, machine translation, and question-answering systems [11,36]. Additionally, deep learning has helped audio processing by advancing speech recognition and synthesis, greatly enhancing user interaction with technology [4,25,30]. These achievements demonstrate how deep learning is an essential tool for applications where standard feature engineering fails because it can grasp intricate patterns and relationships inside high-dimensional data.

The quest for an optimal deep learning architecture for tabular data, crucial in sectors like finance, healthcare, and retail, remains ongoing. Unlike image, text, and audio data, tabular data's structure, rows representing samples, and columns as features present distinct challenges, especially in modeling complex feature relationships that lack spatial or sequential correlation. Innovative model architectures and data representation methods are essential to address tabular data's unique aspects. Models such as TabNet [5], Neural Oblivious Decision Ensembles (NODE) [27], and TabTransformer [18] have emerged as practical solutions alongside popular gradient-boosting tree models. However, gaps remain in handling scenarios with high-dimensional features against smaller sample sizes, such as genomic or other medical data.

We introduce TabSeq, a framework for deep learning on tabular data, using the feature ordering to optimize tabular data utilization. Our approach is motivated by methods of band ordering often used in the efficient analysis of hyperspectral images. Adapting band ordering from hyperspectral images [32] to tabular data involves comparing dataset features to spectral bands, where features, like bands, vary in informational value. This approach uses statistical and machine learning methods to prioritize significant features and reduce redundancy, enhancing dataset efficiency similar to compression in hyperspectral imaging. The bandwidth minimization problem in communication networks [38] focuses on optimizing data transmission order or compressing data to meet bandwidth limits, akin to arranging features in tabular data for deep learning models. The novel contributions of our paper are as follows:

1. We present a novel feature ordering technique that combines local ordering and global ordering to optimize feature sequences and clustering to group comparable features. This innovative method systematically improves learning and significantly improves the model's performance on tabular datasets by prioritizing features according to their relevance and informative content.
2. Our framework enables a Denoising Autoencoder (DAE) architecture to incorporate the Multi-Head Attention (MHA) mechanism smoothly. This integration highlights important characteristics, eliminates redundancy by rebuilding

inputs from partially corrupted versions, and allows for dynamic attention to vital elements.
3. Our studies using raw antibody microarray and other datasets show that our feature ordering approach substantially improves the performance of deep learning models. The outcomes demonstrate how feature sequencing is crucial for training and validating the potential of feature ordering in tabular data processing inside deep learning frameworks.

These contributions collectively address the challenges of heterogeneous feature relevance in tabular data, setting a new precedent for data preprocessing and model optimization in deep learning applications.

2 Related Work

Feature ordering in tabular datasets is essential for improving machine learning models' interpretability, accuracy, and efficiency, particularly in deep learning. Models that recognize and rank important features can learn new information more quickly, require less training time, and exhibit better generalization on unobserved data [41]. While feature ordering is essential for all tabular data types, including numerical data, it influences models that use data structures, such as attention mechanisms or specific autoencoders.

Attention-Based Models: TabNet [5] employs an attention mechanism for feature selection in tabular data, enhancing performance and interpretability without rearranging features. TabTransformer [18] uses contextual embeddings to improve accuracy in handling categorical data, though it requires pre-training and fine-tuning. AutoInt [31] specializes in Click-Through Rate (CTR) prediction by learning feature interactions with a self-attentive network despite assuming unordered features. ASENN [26] predicts pavement deterioration with multi-dropout attention layers, offering efficient infrastructure maintenance solutions. Attention-based models might find it difficult to determine how important a particular feature is to the model's predictions; feature ordering can help with this problem by highlighting the elements with the most significant impact.

Tree-Based Models: The Tree Ensemble Layer (TEL) [14] by Hazimeh et al. enhances neural networks with the efficiency of tree ensembles through "soft trees" and sparse activations, improving performance. TEL, however, does not perform well in capturing complex feature interactions. NODE [27] by Popov et al. combined deep learning flexibility with gradient-boosted with the benefit of decision trees. They achieved superior outcomes via differentiable trees and entmax transformation, albeit with potential limitations in capturing nuanced feature interactions. Tree-based models might struggle to explain complex feature associations; feature ordering fills this gap by arranging features to clarify their relationships.

LLM-Based Models: TabLLM [16], developed by Hegselmann et al., leverages LLMs for few-shot categorization of tabular data by translating tables into natural language, showing superior performance over traditional techniques with limited data. MediTab [40] by Wang et al. introduced a "learn, annotate, refine" approach combined with LLMs for medical data predictions, achieving high performance and excellent zero-shot capabilities without fine-tuning. IngesTables [43], presented by Yak et al., creates scalable tabular foundation models, addressing key issues, such as large cardinality and semantic relevance, through an attention-based method with LLMs, offering cost-effective alternatives to conventional models for clinical trial predictions. Implicit data hierarchies may be a problem for LLM-based models; feature ordering might help by arranging data to represent underlying importance and relationships.

Graph-Based Models: Ruiz et al. introduced PLATO [28], leveraging an auxiliary knowledge graph for model regularization in MLPs. This improved learning on high-dimensional tabular datasets, reducing over-fitting by connecting features to knowledge graph nodes. T2G-FORMER [44], by Yan et al., enhances structured feature interactions through a Graph Estimator and a Transformer network, offering superior interaction modeling and prediction accuracy over conventional deep neural networks. Chen et al. proposed HyTrel [7], a model using hypergraphs to capture tabular data's structural properties, outperforming existing methods with minimal pretraining by integrating inductive biases about data structure. Graph-based models may miss linear feature correlations; feature ordering can address this limitation by better aligning features to depict linear trends and relationships. In general, graph-based approaches provide improvements in tabular data analysis by capturing possible interactions between features. However, they suffer from high computational costs, limiting their applicability.

Autoencoder-Based Models: ReConTab, developed by Chen et al., is a deep learning framework for automatic representation learning from tabular data, utilizing contrastive learning and an asymmetric autoencoder with regularization to enhance classification models like Random Forest and XGBoost [8]. ReMasker, introduced by Du et al., employs masked autoencoding for imputing missing values in tabular data. This improved the results through randomization in masking extra values and offering competitive performance, especially with increased missing data [12]. SwitchTab offers a self-supervised learning approach to identify less apparent dependencies in tabular data, using an asymmetric encoder-decoder to improve prediction tasks and provide interpretable insights via its embeddings [42]. Autoencoder-based models may need to be more efficient in prioritizing influential features, a limitation that feature ordering can overcome by arranging features to enhance model focus and interpretability.

Other Models: GrowNet introduces a method leveraging shallow neural networks within a gradient-boosting framework for various machine learning tasks, limited by static feature selection [6]. Using Scaled Exponential Linear Units (SELU), Self-Normalizing Neural Networks (SNNN) aim for automatic activation normalization to stabilize deep learning but face restrictions due to reliance on SELUs [20]. DCN V2 advances the integration of feature interactions, constrained by its static interaction framework [39]. Gorishniy et al.'s critique of deep learning models for tabular data, including the FT-Transformer, highlights the need for dynamic feature sequencing to enhance model performance [13]. Static feature integration is a potential problem for these models, which feature ordering can solve by dynamically modifying feature sequences to maximize learning and performance. Also see the transformer-based model, TabPFN, and the regularization-based model, TANGOS, in [17,19] respectively.

The literature review highlights innovative strategies for enhancing deep learning model performance on tabular data. Our feature ordering approach uniquely merges clustering with local ordering and global ordering in an MHA-augmented DAE framework, focusing on the ordered arrangement of features based on their importance and information content. This strategy contrasts with traditional methods that often do not consider the arrangement of features.

3 Methodology

This section presents our deep learning architecture, designed primarily to analyze tabular data effectively, as seen in Fig. 1. The process begins with feature clustering and then moves to local ordering and global ordering to improve input feature arrangements. These rearranged features are fed into an MHA mechanism and a DAE, ultimately leading to feature extraction and classification model decisions. Our methodology introduces a novel feature ordering technique to improve the analysis of tabular datasets. This methodology seeks to improve prediction accuracy and robustness for various applications by capturing intricate feature interactions and underlying patterns in the data.

3.1 Feature Ordering

Feature ordering is finding an optimal arrangement of features within and across clusters to minimize a defined cost function that reflects the disorganization of feature positioning. This involves computing permutations that best sequence the features according to their relationships. Given a dataset $X \in \mathbb{R}^{n \times m}$ with n samples and m features, we define a set of graphs $\{G_1, G_2, ..., G_k\}$, where k is the number of clusters, and each graph $G_c = (V_c, E_c)$ for cluster c has vertices $v_i \in V_c$ corresponding to features within that cluster. The edges $(v_i, v_j) \in E_c$ represent significant relationships between features i and j within the cluster. For each cluster c, the goal is to find a permutation π_c of its features that minimizes a local cost function F_c:

$$F_c(\pi_c) = \sum_{(v_i, v_j) \in E_c} |i - j|$$

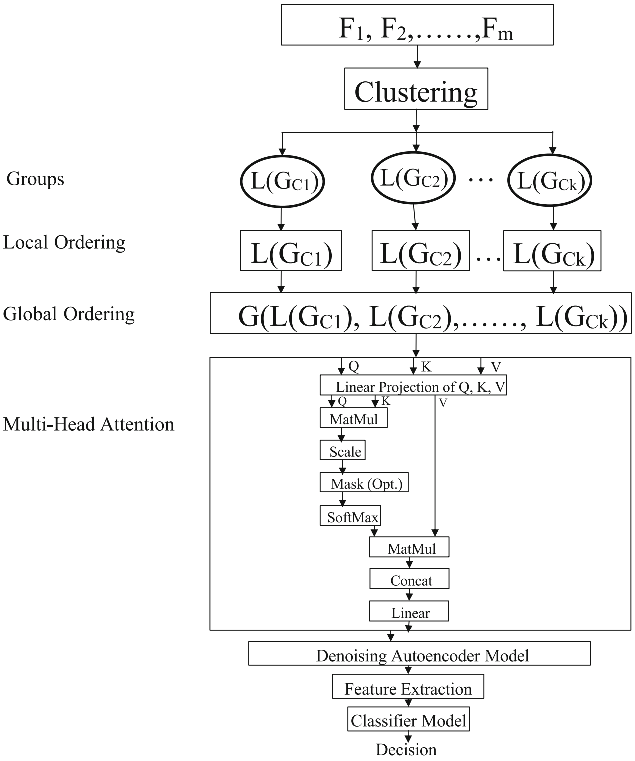


Fig. 1. Overview of the TabSeq framework.

where $i = \pi_c(v_i)$ and $j = \pi_c(v_j)$ represent the indices of features v_i and v_j in the permutation π_c, minimizing the disorganization within the cluster. The overall goal is to find a global permutation Π that integrates the local permutations π_c's and minimizes a global cost function F_G:

$$F_G(\Pi) = \sum_{c=1}^{k} \alpha_c \cdot F_c(\pi_c)$$

where α_c represents the weight or importance of cluster c in the global context with $\sum_c \alpha_c = 1$. The optimal permutation Π^* minimizes $F_G(\Pi)$:

$$\Pi^* = \arg\min_{\Pi} F_G(\Pi)$$

Local Ordering Function: Local ordering is the computation of a permutation that minimizes the sum of absolute differences in positions of related features within a cluster, thereby reducing feature dispersion.

$$L(G_c) = \text{argmin}_{\pi_c} \sum_{(v_i,v_j)\in E_c} |\pi_c(i) - \pi_c(j)|$$

Here, $L(G_c)$ outputs the permutation π_c for cluster c that minimizes the feature dispersion, where $\pi_c(i)$ is the position of feature i in the permutation π_c, and the sum quantifies the total dispersion of features in the cluster.

Global Ordering Function: Global ordering is the process of integrating local permutations from all clusters into a global permutation that minimizes the weighted sum of within-cluster feature dispersion to enhance the deep learning model's performance.

$$G(\{\pi_1, \pi_2, ..., \pi_k\}) = \text{argmin}_{\Pi} \sum_{c=1}^{k} \alpha_c \cdot D_{\pi_c}(G_c)$$

G integrates the local permutations π_c of all clusters into a global ordering Π, minimizing the weighted sum of feature dispersion $D_{\pi_c}(G_c)$ within each cluster G_c, where α_c represents the weight or importance of cluster c. In these functions, $D_{\pi_c}(G_c)$ represents a measure of feature dispersion within cluster c based on the permutation π_c, and Π is the global permutation that integrates these local orderings into a dataset-wide global feature ordering that aims to improve the performance of the model.

Feature Dispersion: In the context of feature ordering for a tabular dataset, the term "Feature Dispersion" describes the degree to which features with a strong relationship (or dependency) are placed far apart in the ordering. The goal would be to minimize this dispersion so that related features are positioned closer together, which could be advantageous for specific deep learning models that can benefit from the structure of the data (See statistical dispersion in [21,33]).

For instance, a generalized feature dispersion for a cluster G_c could be defined as:

$$D(\pi_c) = \sum_{(v_i, v_j) \in E_c} w_{ij} \cdot |\pi_c(i) - \pi_c(j)|$$

where, π_c is the permutation of features within cluster c, w_{ij} is a weight assigned to the edge between features i and j (which could be based on the strength of the relationship between the features e.g., correlation or mutual information), $|\pi_c(i) - \pi_c(j)|$ is the absolute difference in the ordered positions of features i and j within the permutation π_c.

Feature Dispersion and Variance: We adopt variance as a metric to guide the ordering of features locally [29]. This approach is based on the premise that organizing features to minimize their dispersion within clusters can enhance model performance by affecting the variance of these features in a beneficial manner. We understand that feature dispersion within a cluster, $D(\pi_c)$, reflects how spread out the features are in terms of their arrangement or ordering based on certain criteria (e.g., importance, similarity, etc.). Variance, $\text{Var}(X_i)$, measures the spread of values for a given feature i across the dataset or within clusters. The goal is to understand how minimizing $D(\pi_c)$ influences $\text{Var}_c(X_i)$ for features within the same cluster. A decrease in $D(\pi_c)$ (i.e., reduced dispersion or more closely arranged features based on their relationships) leads to an increase in the homogeneity of feature values within the cluster. This homogeneity, in turn, can lead to a more meaningful and possibly reduced variance ($\text{Var}_c(X_i)$) for the features within the cluster, as related features that behave similarly or have strong relationships are positioned closer together, thus reflecting their actual data distribution more accurately. The conceptual relationship can be summarized as:

$$\text{Var}_c(X_i) \propto \frac{1}{D(\pi_c)}$$

This expression suggests that as feature dispersion within a cluster decreases (making $D(\pi_c)$ smaller), the variance of features within that cluster ($\text{Var}_c(X_i)$) becomes more meaningful of the true data distribution. The inverse proportionality indicates that lower dispersion (closer grouping of related features) leads to a more stable or accurate variance representation, underlining the importance of thoughtful feature arrangement in enhancing model understanding and performance. Algorithm 1 captures the general procedure for our feature ordering, including both steps of local and global ordering.

3.2 MHA

MHA or Multi-Head Attention, inspired from [36], is the integration that serves as a cornerstone for enhancing the model's capacity to capture complex dependencies and interactions within the input data. This mechanism's key feature is its capacity to narrow down an input sequence through several attention heads at once, which enables the model to pay attention to data from various representation subspaces at various points in time. Formally, for each head h, we perform linear transformations on the input X to obtain queries Q_h, keys K_h, and values V_h using parameter matrices W_h^Q, W_h^K, and W_h^V, respectively:

$$Q_h = XW_h^Q, \quad K_h = XW_h^K, \quad V_h = XW_h^V$$

Subsequently, we compute the scaled dot-product attention for each head. The attention function operates on queries, keys, and values and scales the dot products of queries with all keys by $\frac{1}{\sqrt{d_k}}$, where d_k is the dimensionality of the keys and queries. This scaling factor helps stabilize the gradients during training. The attention scores are then passed through a softmax function to obtain the weights on the values:

$$\text{Attention}(Q_h, K_h, V_h) = \text{softmax}\left(\frac{Q_h K_h^T}{\sqrt{d_k}}\right) V_h$$

The final output of the MHA layer is created by concatenating and linearly transforming each head's output, each of which captures a unique feature of

Algorithm 1. Feature Ordering

- Preprocessed dataset $X_{\text{train}} \in \mathbb{R}^{n \times m}$ with n samples and m features.
- Clustering Algorithm: Choose from k-means, DBSCAN, HDBSCAN, or a Custom Algorithm.
- Sorting Order: Ascending or Descending.
- Number of clusters (num_clusters), required for k-means and optional for other algorithms.

Output:

- Reordered dataset $new\_training\_set \in \mathbb{R}^{n \times m}$.

Procedure

```
Initialize the clustering model based on the selected algorithm.
if Clustering Algorithm is k-means then
    Specify num_clusters.
else
    Use default or custom settings.
end if
Apply clustering to $X_{\text{train}}$ to get cluster labels.
Append cluster labels to $X_{\text{train}}$.
for each cluster, excluding noise do
    Select data for the current cluster.
    Calculate and order feature dispersion based on the sorting order.
    Record feature order for the cluster.
end for
Combine feature orders from all clusters into an overall feature order.
Reorder $X_{\text{train}}$ columns according to the overall feature order.
Assess model performance with the reordered dataset.
```

the incoming data. The information acquired by each head is combined by this concatenation procedure, maintaining the diversity of the attended features:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O$$

where, $\text{head}_h = \text{Attention}(Q_h, K_h, V_h)$, and W^O is the parameter matrix for the output linear transformation. Using this sophisticated attention mechanism, our model gains a more sophisticated capacity to identify and use the complex patterns in the data. The MHA mechanism provides insights into the diverse elements of the data that different heads focus on, hence improving the interpretability and expressive capacity of the model.

3.3 DAE

TabSeq leveraged an MHA layer in addition to DAE [37] architecture to overcome the difficulties associated with learning robust representations from high-dimensional tabular data. The DAE enhances the model's performance on ensuing tasks by lowering noise and extracting significant features. The DAE architecture comprises an encoder and a decoder, where the encoder maps the input data X to a latent space representation Z, and the decoder reconstructs the input from Z. The MHA layer improves the encoder's capacity to focus on relevant information by allocating different levels of attention to different data segments. Formally stated, the encoding procedure is as follows:

$$Z = f_{\text{encoder}}(X) = \text{ReLU}(W_e \cdot \text{MHA}(X) + b_e)$$

where X is the input data, W_e and b_e are the weights and bias of the encoding layer, respectively, and ReLU denotes the Rectified Linear Unit activation function. The MHA(X) function represents the output of the MHA layer applied to X. The decoder, aiming to reconstruct the input data from the latent representation Z, is given by:

$$\hat{X} = f_{\text{decoder}}(Z) = \text{Sigmoid}(W_d \cdot Z + b_d)$$

where, $\hat{X}$ is the reconstructed data, W_d and b_d are the decoder weights and bias, and Sigmoid is the activation function facilitating reconstruction. The loss function for the DAE, aiming to minimize the reconstruction error, is defined as the Mean Squared Error (MSE) between the original input X and its reconstruction $\hat{X}$:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N}(X_i - \hat{X}_i)^2$$

where N is the number of samples in the dataset. The MHA layer added to the DAE architecture improves the encoder's capacity to identify and highlight the most informative characteristics by utilizing the attention mechanism [9,34,35,45]. This leads to a less noisy and more discriminative representation in the latent space, which is essential for tasks that come after, like classification. Using a sequential network with dense layers, ReLU activation [23] for hidden layers, and sigmoid activation [10] for the output layer, optimized for classification tasks, the final classification model is trained on the encoded representations. Our methodology efficiently tackles the problem of learning from high-dimensional and noisy tabular data by incorporating the MHA layer within the DAE framework, significantly improving the model's predictive performance and robustness.

3.4 Feature Extraction and Classifier Model

In the TabSeq framework, the DAE is instrumental in preprocessing the input data by denoising and enhancing feature salience through its robust feature extraction process, where the encoder component transforms the corrupted input into a refined, lower-dimensional representation. These enhanced features are then utilized by the classifier, which is specifically configured with a softmax activation for multi-class scenarios to generate class probability distributions or a sigmoid activation for binary classification to yield a probability of class membership. This setup ensures that the classifier operates on high-quality features extracted post-DAE, thereby optimizing the model's accuracy and adaptability to different classification tasks. In a nutshell, the DAE processes the input data, which is then followed by a feature extraction process to extract robust, noise-reduced features, which are then utilized by the classifier to ensure precise predictions based on clean and relevant information, illustrating an essential sequential information flow where the classifier's efficacy is significantly enhanced by the high-quality features provided by the DAE.

4 Experimental Results

4.1 Datasets and Model Hyperparameters

In our research, the autoimmune diseases dataset used in [24] and publicly released in [3] contains 393 features targeting five disease classes of 316 samples, detailing each antibody's signal intensity. The ADNI dataset [1] includes 177 samples and 263 features with target attributes like AD123, ABETA12, and AV45AB12, representing various stages of Alzheimer's disease and captured through DTI analysis for white matter integrity. Lastly, the WDBC dataset [2] offers 32 features derived from breast mass images, aiming to differentiate between 357 benign and 212 malignant cases. Each dataset was partitioned into training, validation, and testing subsets following a 70:15:15 split, focusing on specific target attributes for comprehensive classification and analysis. TabSeq model with feature ordering integrates an MHA mechanism with four heads and dimensionality of 32 alongside a DAE comprising dense ReLU-activated layers. It was trained over 50 epochs with a batch size of 32 using the Adam optimizer, and the model employs MSE loss for the DAE and binary cross-entropy loss for the classifier. This configuration facilitates nuanced feature extraction and robust classification, as evidenced by the model's validation performance, optimizing computational efficiency and learning effectiveness. In our analysis, feature ordering was uniformly applied across baseline models using k-means clustering with 5 clusters in ascending order for the autoimmune diseases and ADNI datasets and 3 clusters for the WDBC dataset. This consistent methodology underscores the effectiveness of feature ordering in enhancing model performance across the board, with TabSeq demonstrating particularly notable improvements in accuracy and AUC with feature ordering.

4.2 Experiments with Autoimmune Diseases' Dataset

Two cases were included in our investigation (Table 1), which showed how feature ordering affected model performance. In Case 1, NODE's accuracy sharply declined, whereas TabSeq's increased from 85.42% to 87.23% with feature ordering. Case 2 demonstrated how feature ordering significantly improved TabNet's accuracy, raising it to 94.44%. TabSeq demonstrated its practical usage of feature

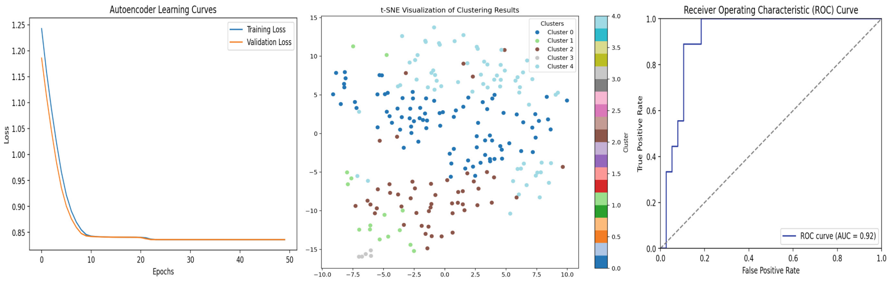
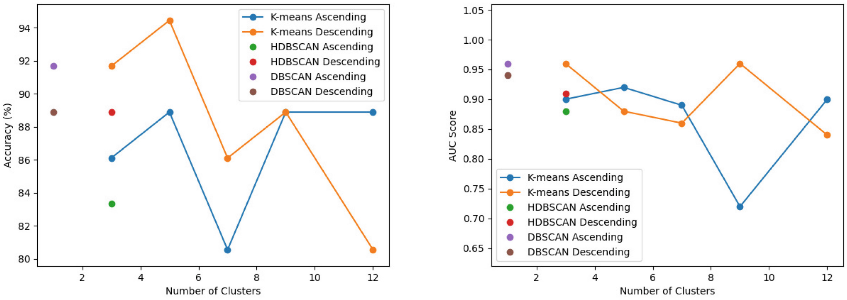


Fig. 2. Visualization of model performance on the autoimmune diseases dataset.

ordering with consistently good performance. These findings highlight the context and model-specificity of feature ordering's efficacy, with TabSeq consistently outperforming other models in the dataset on autoimmune disorders. Visualizations in Fig. 2 confirm effective clustering for feature ordering and model generalization and show the ROC curve's high binary classification accuracy. From Table 1, it is evident that feature ordering significantly affects model performance, especially in sequence-dependent architectures (e.g., autoencoders and LSTM), where the feature ordering aligns features in a meaningful sequence to improve learning. However, tree-based models, such as NODE, could not benefit as much because of their built-in feature selection methods. The local receptive field of 1D CNNs limits the effect of feature ordering. In contrast, TabNet's attention mechanism already prioritizes relevant features, which may cause inconsistent feature ordering performance. TabTransformer consistently enhances AUC across datasets; TANGOS shows marked improvements in AUC with feature ordering but does not demonstrate strong accuracy. These results underline TabSeq's capacity to discern complex patterns in high-dimensional genomic datasets, advocating their potential for data analysis. Based on ablation studies, we chose k-means with 5 clusters for feature ordering with features sorted in ascending order (Table 1 for Case 1–2 and Fig. 2 for Case 1).

4.3 Ablation Studies

We assessed the TabSeq model's performance using the autoimmune diseases' dataset, mainly how the clustering algorithms affected feature ordering. Various clustering algorithms affected AUC and model accuracy in different scenarios. For example, in Case 1, the maximum accuracy of 87.23% and the highest AUC of 0.92 were obtained using DBSCAN with a single cluster with the features sorted in ascending order. Case 2, on the other hand, achieved 94.44% accuracy and optimal performance using k-means at five clusters with features sorted in descending order. Surprisingly, DBSCAN achieved perfect accuracy and an AUC of 100% and 1.00 in Case 4, with features sorted in descending order. The significance of the number of clusters, specific clustering algorithm, and sorting order were also observed for these cases. Fig. 3 shows the assessment for Case 2,



(a) Case 2: Accuracy (b) Case 2: AUC

Fig. 3. Results' plots for TabSeq with different feature ordering configurations.

Table 1. Comparative results on autoimmune diseases' dataset for different models (# = without feature ordering, * = with feature ordering, Acc. = Accuracy).

Model	Acc.#	AUC#	Acc.*	AUC*
Case 1: H vs SLE+RA+SS+SV				
Linear SVM [24]	N/A	0.94	N/A	N/A
TabSeq (ours)	85.42%	0.92	87.23%	0.92
LSTM	82.11%	0.88	85.11%	0.91
DAE-LSTM	75.79%	0.80	82.98%	0.80
DAE	76.84%	0.80	80.85%	0.86
1D CNN	72.92%	0.84	79.17%	0.44
TabNet [5]	77.08%	0.85	83.33%	0.50
NODE [27]	89.58%	0.92	20.83%	0.40
TabTransformer [18]	82.98%	0.84	87.23%	0.75
TANGOS [19]	82.98%	0.61	82.98%	0.65
Case 2: SLE vs RA+SS+SV				
Linear SVM [24]	N/A	0.96	N/A	N/A
TabSeq (ours)	86.11%	0.87	91.67%	0.96
LSTM	81.94%	0.86	86.11%	0.93
DAE-LSTM	79.17%	0.81	86.11%	0.87
DAE	84.72%	0.84	86.11%	0.91
1D CNN	80.56%	0.81	80.56%	0.31
TabNet [5]	91.67%	0.60	94.44%	0.74
NODE [27]	83.33%	0.82	80.56%	0.68
TabTransformer [18]	86.11%	0.77	80.56%	0.54
TANGOS [19]	86.12%	0.39	86.12%	0.73

whereas the figures for other cases also looked similar. These results highlight the crucial role that clustering configurations in feature ordering play in improving the predictive power of the TabSeq model, indicating that the model's ability to distinguish intricate patterns of autoimmune diseases is greatly influenced by strategic cluster formation and feature ordering.

4.4 Experiments with ADNI and WDBC Dataset

The studies done on the ADNI dataset (Table 2) show how effective feature ordering is in several models, such as TabNet, NODE, and TabSeq. Across two target attributes, such as AD123 and ABETA12, the addition of feature ordering resulted in significant gains in accuracy and AUC scores; nonetheless, AD123 presented a multi-class classification challenge. For example, the TabSeq model's accuracy improved from 66.67% to 67.68% for the AD123 target. Similarly, substantial improvements were noted for the ABETA12 target respectively 64.44% to 75.13% for TabSeq. By efficiently selecting and rearranging features according to their informative contribution to the predictive goal, feature ordering improved the performance of deep learning models. These results highlight the importance of adopting feature ordering to refine model predictions for complex datasets. The studies conducted with the WDBC dataset (Table 2) demonstrate how feature ordering can improve model accuracy and AUC scores. The TabNet, NODE, and TabSeq versions performed better when feature ordering was used. Notably, compared to its excellent performance without feature ordering, the TabSeq model's accuracy increased, reaching 94.71% with feature ordering. TabTransformer boosts AUC; TANGOS shows lower accuracy; TabPFN excels in feature-limited datasets like WDBC but without feature ordering. We employed k-means clustering with 5 clusters for the ADNI dataset and 3 for the WDBC

Table 2. Performance on ADNI and WDBC datasets.

Dataset	Model	Acc.#	AUC#	Acc.*	AUC*
ADNI	**Target Attribute: AD123**				
	TabNet [5]	59.26%	0.68	66.67%	0.54
	NODE [27]	59.26%	0.68	66.67%	0.54
	TabSeq (ours)	66.67%	0.67	67.68%	0.61
	TabTransformer [18]	66.67%	0.40	74.07%	0.70
	TANGOS [19]	74.07%	0.70	74.07%	0.73
	Target Attribute: ABETA12				
	TabNet [5]	51.85%	0.52	74.07%	0.67
	NODE [27]	59.26%	0.57	59.26%	0.68
	TabSeq (ours)	64.44%	0.55	75.13%	0.71
	TabTransformer [18]	59.26%	0.50	59.26%	0.50
	TANGOS [19]	44.45%	0.50	44.45%	0.62
WDBC	TabNet [5]	91.86%	0.91	94.18%	0.99
	NODE [27]	91.86%	0.99	93.71%	0.98
	TabSeq (ours)	94.65%	0.91	94.71%	0.98
	TabTransformer [18]	87.08%	0.91	60%	0.37
	TabPFN [17]	94.19%	0.94	40.69%	0.50
	TANGOS [19]	60%	0.76	60%	0.85

dataset, ordering them ascendingly. As Fig. 3 shows, increasing cluster numbers initially enhanced performance for most of the cases but eventually declined after specific points. These results affirm that our feature ordering approach boosts model accuracy and AUC across datasets, noting its importance in advancing deep learning for tabular data.

5 Conclusion

This paper introduces TabSeq, a novel framework that employs feature ordering to enhance deep learning's performance on tabular datasets significantly. By integrating local ordering and global ordering within a DAE equipped with an MHA mechanism, our method systematically optimizes feature sequences to improve learning efficacy. Studies conducted on raw antibody microarray data and other medical datasets have underscored the capability of strategic feature sequencing to yield substantial performance gains. These empirical results reinforce feature ordering's potential as a game-changing technique in deep learning for tabular data. Its effectiveness is predominantly seen in sequence-based architectures, and its performance on low-dimensional datasets remains to be determined. Further research is needed to refine and extend the method's applicability to various architectures and datasets.

Acknowledgement. This work was supported in part by grants from the US National Science Foundation (Award #1920920, #2125872, and #2223793).

References

1. ADNI — Alzheimer's Disease Neuroimaging Initiative. https://adni.loni.usc.edu/. Accessed 03 July 2024
2. Breast Cancer Wisconsin (Diagnostic) - UCI Machine Learning Repository. https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic. Accessed 03 July 2024
3. Data_set_190503.xlsx. https://figshare.com/s/3bd3848a28ef6e7ae9a9. Accessed 03 July 2024
4. Amodei, D., et al.: Deep Speech 2: end-to-end speech recognition in English and Mandarin. In: Proceedings of International Conference on Machine Learning, pp. 173–182. PMLR (2016)
5. Arik, S.Ö., Pfister, T.: TabNet: attentive interpretable tabular learning. In: Proceedings of AAAI Conference on Artificial Intelligence, vol. 35, pp. 6679–6687 (2021)
6. Badirli, S., et al.: Gradient boosting neural networks: grownet. arXiv preprint arXiv:2002.07971 (2020)
7. Chen, P., et al.: HYTREL: hypergraph-enhanced tabular data representation learning. In: Advances in Neural Information Processing Systems, vol. 36 (2024)
8. Chen, S., et al.: ReConTab: regularized contrastive representation learning for tabular data. arXiv preprint arXiv:2310.18541 (2023)
9. Chen, Y., et al.: MAMA Net: multi-scale attention memory autoencoder network for anomaly detection. IEEE Trans. Med. Imaging **40**(3), 1032–1041 (2020)

10. DE, R.: Learning representations by back-propagation errors. Nature **323**, 533–536 (1986)
11. Devlin, J., et al.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
12. Du, T., et al.: ReMasker: imputing tabular data with masked autoencoding. arXiv preprint arXiv:2309.13793 (2023)
13. Gorishniy, Y., et al.: Revisiting deep learning models for tabular data. In: Advances in Neural Information Processing Systems, vol. 34, pp. 18932–18943 (2021)
14. Hazimeh, H., et al.: The tree ensemble layer: differentiability meets conditional computation. In: Proceedings of International Conference on Machine Learning, pp. 4138–4148. PMLR (2020)
15. He, K., et al.: Deep residual learning for image recognition. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
16. Hegselmann, S., et al.: TabLLM: few-shot classification of tabular data with large language models. In: International Conference on Artificial Intelligence and Statistics, pp. 5549–5581. PMLR (2023)
17. Hollmann, N., et al.: TabPFN: a transformer that solves small tabular classification problems in a second. In: NeurIPS 2022 First Table Representation Workshop (2022)
18. Huang, X., et al.: TabTransformer: tabular data modeling using contextual embeddings. arXiv preprint arXiv:2012.06678 (2020)
19. Jeffares, A., et al.: TANGOS: regularizing tabular neural networks through gradient orthogonalization and specialization. In: The Eleventh International Conference on Learning Representations (2023)
20. Klambauer, G., et al.: Self-normalizing neural networks. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
21. Kostal, L., et al.: Measures of statistical dispersion based on shannon and fisher information concepts. Inf. Sci. **235**, 214–223 (2013)
22. Krizhevsky, A., et al.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, vol. 25 (2012)
23. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: ICML 2010 (2010)
24. Ohlsson, M., et al.: Proteomic data analysis for differential profiling of the autoimmune diseases SLE, RA, SS, and ANCA-associated vasculitis. J. Proteome Res. **20**(2), 1252–1260 (2020)
25. Oord, A.V.D., et al.: WaveNet: a generative model for raw audio. arXiv preprint arXiv:1609.03499 (2016)
26. Philip, B., et al.: ASENN: attention-based selective embedding neural networks for road distress prediction. J. Big Data **10**(1), 164 (2023)
27. Popov, S., et al.: Neural oblivious decision ensembles for deep learning on tabular data. arXiv preprint arXiv:1909.06312 (2019)
28. Ruiz, C., et al.: High dimensional, tabular deep learning with an auxiliary knowledge graph. In: Advances in Neural Information Processing Systems, vol. 36 (2024)
29. Sá, C.R.: Variance-based feature importance in neural networks. In: Kralj Novak, P., Šmuc, T., Džeroski, S. (eds.) DS 2019. LNCS (LNAI), vol. 11828, pp. 306–315. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-33778-0_24
30. Senior, A., et al.: Deep neural networks for acoustic modeling in speech recognition. IEEE Signal Process. Maga. **29**, 82–97 (2012)
31. Song, W., et al.: AutoInt: automatic feature interaction learning via self-attentive neural networks. In: Proceedings of 28th ACM International Conference on Information and Knowledge Management, pp. 1161–1170 (2019)

32. Tate, S.R.: Band ordering in lossless compression of multispectral images. IEEE Trans. Comput. **46**(4), 477–483 (1997)
33. Th. Gries, S.: Analyzing Dispersion. In: A Practical Handbook of Corpus Linguistics, pp. 99–118. Springer (2021)
34. Tian, T., Fang, Z.F.: Attention-based autoencoder topic model for short texts. Procedia Comput. Sci. **151**, 1134–1139 (2019)
35. Tihon, S., Javaid, M.U., Fourure, D., Posocco, N., Peel, T.: DAEMA: denoising autoencoder with mask attention. In: Farkaš, I., Masulli, P., Otte, S., Wermter, S. (eds.) ICANN 2021. LNCS, vol. 12891, pp. 229–240. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86362-3_19
36. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
37. Vincent, P., et al.: Extracting and composing robust features with denoising autoencoders. In: Proceedings of 25th International Conference on Machine Learning, pp. 1096–1103 (2008)
38. Wang, C., et al.: Bandwidth minimization problem. In: MOSIM 2014, 10ème Conférence Francophone de Modélisation, Optimisation et Simulation (2014)
39. Wang, R., et al.: DCN V2: Improved deep & cross network and practical lessons for web-scale learning to rank systems. In: Proceedings of Web Conference 2021, pp. 1785–1797 (2021)
40. Wang, Z., et al.: MediTab: scaling medical tabular data predictors via data consolidation, enrichment, and refinement. arXiv preprint arXiv:2305.12081 (2023)
41. Wojtas, M., Chen, K.: Feature importance ranking for deep learning. In: Advances on Neural Information Processing System, vol. 33, pp.5105–5114 (2020)
42. Wu, J., et al.: SwitchTab: switched autoencoders are effective tabular learners. arXiv preprint arXiv:2401.02013 (2024)
43. Yak, S., et al.: IngesTables: scalable and efficient training of LLM-enabled tabular foundation models. In: NeurIPS 2023 Table Representation on Learning Workkshop (2023)
44. Yan, J., et al.: T2G-former: organizing tabular features into relation graphs promotes heterogeneous feature interaction. In: AAAI Conference on AI (2023)
45. Zhou, J.P., et al.: TAFA: two-headed attention fused autoencoder for context-aware recommendations. In: Proceedings of the 14th ACM Conference on Recommender Systems, pp. 338–347 (2020)

GraFix: A Graph Transformer with Fixed Attention Based on the WL Kernel

Lingfeng Zhang[1], Luca Cosmo[2], Giorgia Minello[2], Andrea Torsello[2], and Luca Rossi[1](✉)

[1] The Hong Kong Polytechnic University, Hung Hom, Hong Kong
`lingfeng.zhang@connect.polyu.hk`, `luca.rossi@polyu.edu.hk`
[2] Ca' Foscari University of Venice, Venice, Italy
{`luca.cosmo`,`giorgia.minello`,`andrea.torsello`}`@unive.it`

Abstract. In this paper we introduce GraFix, a novel graph transformer with fixed structural attention. Inspired by recent works 1) harnessing the link between (graph) kernels and the attention mechanism of transformers and 2) favouring simple fixed (non-learnable) attentive patterns over the standard attention mechanism, we propose to use graph kernels, specifically the WL kernel, to replace the learnable attention mechanism of a transformer with a fixed one capturing the structural similarity between substructures in the input graphs. The resulting graph transformer showcases an excellent performance on standard graph classification benchmarks, performing on-par with and in some instances outperforming a wide variety of alternative graph neural network and graph transformer-based approaches while at the same time benefiting from a reduced number of learnable parameters and learning runtime.

Keywords: Graph Neural Network · Graph Kernel · Graph Transformer

1 Introduction

The continuous advancement of hardware resources, particularly GPUs, of the past few years has led deep learning to become the poster child for artificial intelligence, fundamentally eclipsing longer established subfields of machine learning. This rise in popularity has been spearheaded by models such as Convolutional Neural Networks (CNNs) [12] and transformers [33], which have been applied with great success to fields such as computer vision and language modelling. The original versions of these models, however, are designed to operate on vectorial data and are therefore unable to effectively handle graph data, where structural features are often crucial to solve the task at hand. Indeed, a wide variety of objects, from images [11,28] to social networks [26] and chemical structures [23] are often more naturally represented as graphs, rather than an unstructured collection of features. The non-Euclidean nature of graphs, while bringing the advantage of high expressivity, also implies that it is difficult for traditional deep learning methods to be directly applied to graphs.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 435–450, 2025.
https://doi.org/10.1007/978-3-031-78128-5_28

For this reason, in recent years much of the attention of the deep learning community has focused on adapting successful deep learning architectures to operate on graph data. The most prominent example are Graph Neural Networks (GNNs) [1,2,13,15,27,30,35,40], which operate by transferring node feature information between adjacent nodes to encapsulate the structural characteristics of the graph. Prior to the introduction of GNNs, graph kernels [17–19,22,29,31] were considered the most popular approach to apply standard machine learning techniques to graphs, however GNNs have since demonstrated superior performance compared to traditional graph kernel-based learning methods. GNNs however suffer from their own limitations, such as the inability to learn long-range node dependencies, especially as the number of network layers increases and feature over-smoothing becomes an issue.

The attention mechanism, which allows transformers to model long-range dependencies [33], offers a compelling solution to the limitations of GNNs. While the standard transformer architecture does not take the graph structural information into account, but rather attempts to learn structural patterns by looking at the features alone, structure-aware positional encodings [9] and attention mechanisms [20] can allow a graph transformer to capture global graph information while preserving locality. The combination of GNNs and transformers can therefore enhance the representation learning capabilities of GNNs, leading to improved performance in tasks involving graph data.

The present study is motivated by recent works showing that replacing learnable attention with a fixed one does not negatively impact the model performance, on the contrary it can result in a slight performance improvement [5,24]. More specifically, in this paper we propose a novel Graph transformer with Fixed attention (GraFix), where the traditional (learnable) attention mechanism is a replaced by a fixed (non-learnable) attention mechanism based on graph kernels. We use the well-known WL to measure the similarity between nodes neighbourhoods and we let the transformer use this information when attending to the graph nodes. We compare our model with conventional GNNs methods as well as alternative graph transformers and we find that our method achieves a competitive performance in terms of graph classification accuracy while reducing the number of model parameters and training time.

The remainder of this paper is organized as follows. Section 2 reviews the related work, while Sects. 3 and 4 introduce the necessary background and our model, respectively. Finally, Sect. 5 presents the experimental results and Sect. 6 concludes the paper.

2 Related Work

While standard machine learning algorithms are not originally designed to deal with non-vectorial inputs such as graphs, the introduction of graph kernels allows to use kernel-based algorithms, such as support vector machines [14], for graph problems. This is achieved by defining appropriate kernel functions that can capture the structural characteristics of graphs when mapping them to an (implicit) embedding space. Well-known graph kernels include: 1) the graphlet kernel [18], which counts occurrences of small connected subgraphs called graphlets; 2) the random-walk kernel, which compares the behaviour of random walks [31] on pairs of input graphs; 3) the subgraph kernel [17], which compares graphs by enumerating common subgraphs; and 3) the Weisfeiler-Lehman (WL) kernel [29], which refines node labels iteratively based on neighborhood information and measures similarity by evaluating label consistency across iterations.

With the advent of deep neural networks, researchers turned their attention to end-to-end models, attempting to generalise existing architectures to deal with graph data. GNNs can be seen as an attempt to generalise the concept of convolution at the core of CNNs beyond the regularity of grids of pixels [6]. Most existing GNNs follow the message-passing model, where nodes interact through edges to update their features. Examples of well-known GNN models include GCN [15], GraphSAGE [13], DiffPool [38], DGCNN [40], and GIN [35]. As mentioned in the previous section, the main drawback of these models lies in their tendency to capture very well the local graph connectivity at the expense of long-range node dependencies.

Graph transformers address this issue by adding structural awareness to standard transformers [21], which are naturally well suited to model long-range information. This in turn can be achieved in three ways: 1) passing GNN-enriched features to a transformer, 2) using unique position encoding of graph structures, and 3) designing more sophisticated attention mechanisms. GraphTrans [34] extracts local information using a GCN which is then input to a transformer model. GraphiT [20] substitutes the GCN with a Graph Convolutional Kernel Network (GCKN) [3], where the GCKN uses kernels to extract information about local graph substructures. While GNNs can provide decent feature extractors for local structure, they need to be trained along with or before the transformer model, leading to a more complex search space. Approaches based on structure-aware positional encoding bypass this issue. For example, Dwivedi and Bresson [9] use the eigenvectors of the graph Laplacian as the position encoding. SAN [16] works in a similar way but it allows the use of the entire Laplace spectrum and learns the interactions between different frequencies, thus improving the model ability to understand the complex structure of the graph. In [25], a random walk positional encoding was proposed.

A number of graph transformer methods propose to directly modify the attention mechanism to account for structural information. The attention mechanism of the vanilla transformer can be regarded as propagating information over a fully connected graph, thus disregarding the underlying structure of the data. This however can be modified to take into account the graph structure. Dwivedi and Bresson [9] propose to use the adjacency matrix to restrict the attention mechanism to local information about the nodes. Inspired by Tsai et al. [32], who show how the original attention mechanism can be interpreted as the application of a kernel smoother over the input features, GraphiT [20] applies this idea to graphs and uses kernels between graph nodes to modulate the attention mechanism. Chen et al. [4] take this one step further and propose to replace the entire attention mechanism with a kernel smoother, where the attention between pair of nodes is a function of the similarity between the subgraphs rooted at those nodes. In particular, they propose to use an exponential kernel between the features extracted by a GNN on the local subgraphs around each node. Finally, other approaches have sought to bias the attention using the the shortest path distance [37] and the resistance distance [39] between the nodes.

Our model is closely related to [9] and [4], as we also formulate the attention mechanism between nodes in terms of graph kernels. Crucially, however, our attention is fixed, *i.e.*, it is not learned but rather precomputed as the graph kernel between subgraphs rooted on different nodes. Whilst [9] and [4] use GNNs to compute the embeddings of the nodes neighbourhoods, our framework allows to directly use a wide variety of well-known (non-differentiable) graph kernels. Our work is also inspired by [5,24], which have recently shown that fixing attention coefficients may have little to no negative impact on the model performance. On the contrary it can result in a slight performance improvement [24]. To the best of our knowledge, our work is the first to apply non-learnable attention patterns to graph transformers.

3 Background

Let $G = (V, E, \mathbf{X})$ denote a graph over n nodes with edge set E, where $\mathbf{X} \in \mathbb{R}^{n \times d}$ is the matrix of d-dimensional features of the nodes $v \in V$. In this section we introduce the two main components underpinning our model, 1) the WL graph kernel and 2) the transformer attention mechanism.

3.1 WL Graph Kernel

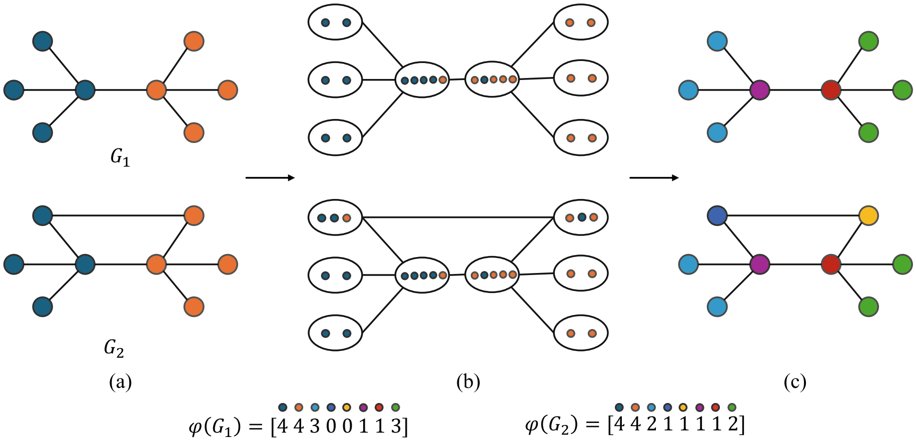
The WL kernel [29] is a graph kernel that operates based on concept of iterative label refinement. Let $\mathcal{N}(v)$ denote the set of nodes adjacent to v, where each node has an associated label $l(v)$, then the label is iteratively refined as

$$l^{i+1}(v) = f(l^i(v), g(l^i(u)|u \in \mathcal{N}(v))) , \tag{1}$$

where f is an aggregation function responsible for the generation of new labels, g is a permutation-invariant function, and i denotes the iteration number.

In the WL kernel each node is initially assigned a label. The labels are then iteratively refined by integrating information obtained from neighboring nodes according to Eq. 1. Figure 1 shows an example of WL label refinement, where the initial node labels of two graphs G_1 and G_2 (a) are transformed into the refined form (c) by aggregating the colors associated with the neighboring nodes (b). Following this aggregation process, a new node coloring is performed based on the updated features derived from the refined labels. A hash table enumerating the refined labels is then used to create the graph embeddings $\varphi(G_1)$ and $\varphi(G_2)$.

The example of Fig. 1 shows the outcome of a single iteration, but in practice the process is repeated until a maximum number of iterations h is reached. This


G_1
G_2
$\varphi(G_1) = [4\ 4\ 3\ 0\ 0\ 1\ 1\ 3]$
$\varphi(G_2) = [4\ 4\ 2\ 1\ 1\ 1\ 1\ 2]$

(a) (b) (c)

Fig. 1. One iteration of the WL label refinement.

in turn yields a set of graph embeddings $\{\varphi_i(G_1)\}_{i=1}^{h}$ and $\{\varphi_i(G_2)\}_{i=1}^{h}$, from which the WL kernel between G_1 and G_2 is computed as

$$k_{WL}^{h}(G_1, G_2) = \sum_{i=0}^{h} \left\langle \varphi_i(G_1), \varphi_i(G_2) \right\rangle , \tag{2}$$

where $\langle \cdot, \cdot \rangle$ indicates the dot product between each pair of histograms.

3.2 The Transformer Architecture

Transformers [33] rose to prominence due to their unmatched performance in language modelling tasks. At its core the transformer architecture revolves around the attention mechanism, which allows the model to learn contextualized representations of the input tokens encapsulating long-range dependencies within the input sequence. Beyond language modelling, transformers have been applied with success to fields including computer vision [8] and speech processing [7].

Given a matrix of input features $\mathbf{X}$, the transformer maps this to the matrices $\mathbf{Q}$(Query), $\mathbf{K}$(Key), and $\mathbf{V}$(Value) through three distinct linear transformations. With these matrices to hand, the attention is then defined as

$$\text{Attention}(\mathbf{X}) = \text{softmax}\Big(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\Big)\mathbf{V} \tag{3}$$

where $\sqrt{d_k}$ is a scaling factor that depends on the dimension of $\mathbf{Q}$. Equation 3 encapsulates the pairwise similarity between input tokens through the dot product $\mathbf{Q}\mathbf{K}^T$. This allows to compute the updated features

$$\mathbf{X}' = \text{Attention}(\mathbf{X}) + \mathbf{X} , \tag{4}$$

which can eventually be used to solve the downstream task at hand, *e.g.*, classification. Note that the common practice is to let $\mathbf{Q} = \mathbf{K}$ and to use multiple attention *heads* whereby multiple sets of matrices $\mathbf{Q}$ and $\mathbf{V}$ are learned to allow the network to model multiple contexts. Note also that in a vanilla implementation of the transformer model for graphs, where the input features are node features, the model inevitably ignores the crucial topological information intrinsic to the graph structure.

Tsai et al. [32] reformulate Eq. 3 through the lens of kernels, rewriting the attention mechanism applied to the ith input feature x_i as

$$\text{Attention}(x_i) = \sum_{x_j \in \mathbf{X}} \frac{k(x_i, x_j)}{\sum_{x_l} k(x_i, x_l)} f(x_j)\,, \tag{5}$$

where f denotes the value function (in the case of the original transformer, a linear function), k is a kernel function between input features, and we omitted the filtering function of the formulation in [32] for simplicity.

In the case of a graph $G = (V, E, \mathbf{X})$, given a structure-aware kernel function $k(x_v, x_u)$, where $x_v, x_u \in \mathbf{X}$ are the node features of $v, u \in V$, this yields

$$\text{Attention}(x_v) = \sum_{u \in V} \frac{k(x_v, x_u)}{\sum_{w \in V} k(x_v, x_w)} f(x_u)\,. \tag{6}$$

In Chen et al. [4] this is accomplished by using a GNN to create enriched features encapsulating the local neighbourhoods of the nodes and then using an exponential kernel function k on these enriched features. Dwivedi and Bresson [9] follow a similar but slightly more convoluted approach where two kernels, one capturing the similarity between the features and one capturing the similarity between the nodes positions with a graph, are multiplied together.

As explained in the next section, we differ from both these approaches in that we directly replace $k(x_v, x_u)$ with a graph kernel, specifically the WL kernel of Eq. 2. In other words, we propose to fix the attention coefficients rather than learning them end-to-end.

4 Our Model

Figure 2 shows the architecture of GraFix, the graph kernel transformer proposed in this paper. The two main components of our models are 1) the WL kernel attention computation and 2) the transformer layer(s).

In the WL kernel attention computation, we commence by extracting the the k-hop subgraph $S_{k,G}(v)$ around each node v, *i.e.*, the subgraph induced by selecting the nodes that can be reach from the central node by traversing at most k edges. We then compute the kernel matrix encapsulating the similarity between all pairs of subgraphs using Eq. 2, where all node features have been one-hot encoded. Finally, we define the entry $A_{v,u}$ of the attention matrix A corresponding to the attention coefficient between the node v and u as

$$A_{v,u} = k_{WL}^{h}(S_{k,G}(v), S_{k,G}(u))\,. \tag{7}$$

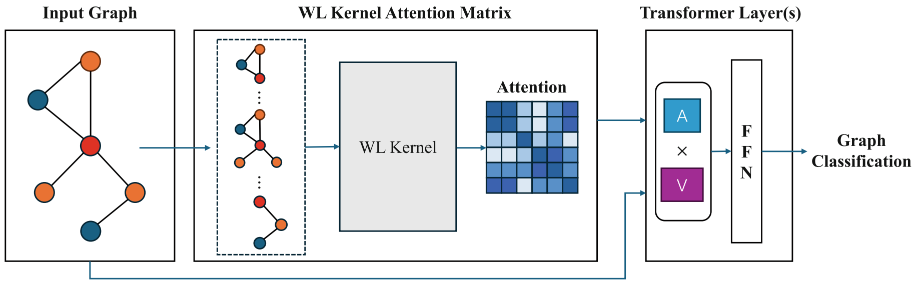


Fig. 2. The structure of the proposed graph transformer. The attention is precomputed and fixed by computing the WL kernel similarity between pairs of subgraphs extracted from the input graph. The node features are then fed to a series of transformer layers where the attention mechanism of Eq. 6 is used to enrich them.

The attention mechanism used to update the node representations in the transformer layer(s) is then

$$\text{Attention}(x_v) = \sum_{u \in V} A_{v,u} f(x_u) \,. \tag{8}$$

Note that the attention mechanism of Eq. 8 differs from that Eq. 5 in that we omit the normalization factor ensuring the row-wise stochasticity of the attention matrix. We experimentally observe that this leads to a better performance of the model while not impacting its convergence.

Our model also deviates from the standard practice of using multiple attention heads and relies on a single attention head instead. Raganato et al. [24] have shown that the conventional multi-head attention mechanisms predominantly encodes largely superfluous position information. Indeed, in our experiments we empirically observe that a single attention head is preferable to multiple ones, as also briefly demonstrated in the ablation study of Subsect. 5.6.

4.1 Computational Complexity

The computational complexity of the WL kernel can be expressed as $O(Nhm + N^2hn)$, where N is the number of subgraphs, h represents the iteration count, n corresponds to the maximum number of nodes and m denotes the maximum number of edges within the subgraphs. We propose an efficient implementation of the WL kernel that makes use of GPU hardware to speed up the computation of the attention matrix. The latter is precomputed separately from the model training, allowing us to speed up the training. Specifically, we use the `scatter` function of PyTorch Geometric to aggregate the neighboring labels in a computationally efficient way and then construct histograms to enumerate the labels for each graph, before comparing their similarity.

Table 1. Datasets statistics.

Dataset	Size	Classes	Avg. num. of nodes	Avg. num. of edges	Has node features
MUTAG	188	2	17.93	19.79	Yes
PTC	344	2	14.29	14.69	Yes
PROTEINS	1,113	2	39.06	72.82	Yes
NCI1	4,110	2	29.87	32.30	Yes
AIDS	2,000	2	15.69	16.20	Yes
IMDB-BINARY	2,000	2	19.77	96.53	No

5 Experimental Results

We compare our model against six state-of-the-art GNNs and five different graph transformer methods: GCN [15], DGCNN [40], DiffPool [38], ECC [30], GIN [35], GraphSAGE [13], vanilla transformer (node features only) [33], Structure-Aware Transformer (SAT) [4], GraphiT with diffusion coding, GraphiT with adjacent coding, GraphiT with GCKN coding [20]. Recall that GraphiT [20] and SAT [4] are the two most closely related architectures to ours, as they both employ some sort of kernel to replace the standard attention mechanism, although in both cases the attention is not fixed but learned.

We compare these methods on six commonly used datasets for graph classification tasks. Five datasets are biological/chemical datasets (MUTAG, PTC, PROTEINS, NCI1, AIDS) [23] and one is a social dataset (IMDB-BINARY) [36]. All node features are encoded using one-hot encoding, however since IMDB-BINARY does not have node features, we use the degree of each node as the node feature. Table 1 shows some of the graph dataset basic statistics, including number of graphs, classes, and average number of nodes and edges.

In our experiments we use two RTX3090 GPUs, each with a memory capacity of 24 GB, alongside two RTX2060 GPUs, each offering 12GB of memory. All our code is written in PyTorch and PyTorch Geometric[1].

5.1 Experimental Setup

We evaluate the performance of each model using 10-fold cross-validation, with a 90/10 training+validation/test split, further splitting train+validation into training (90) and validation (10).

We use the training/validation sets to select the optimal model configurations through grid search. To establish the hyperparameter search space, we follow the ranges specified in [10] for GNNs, while for the vanilla transformer and GraphiT we refer to the ranges outlined in [20]. For SAT [4], we vary: 1) the number of GCN layers from 1 to 3; 2) the number of transformer layers from 1 to 3; 3) the

---

[1] https://github.com/lcukyfuture/GraphT.

Table 2. Average mean classification accuracy ± standard error for our model (GraFix) and the competing ones on the 6 datasets considered in this study. The best model for each dataset is highlighted in bold, while the second best model is underlined.

Dataset	MUTAG	PTC	PROTEINS	NCI1	AIDS	IMDB-BINARY
GCN	75.99 ± 3.5	56.11 ± 2.8	71.60 ± 0.8	74.53 ± 0.7	94.55 ± 0.4	72.80 ± 1.8
DGCNN	85.03 ± 2.7	50.60 ± 2.2	73.86 ± 0.7	74.38 ± 0.7	99.40 ± 0.1	71.90 ± 1.4
DiffPool	81.35 ± 2.5	57.83 ± 2.3	71.25 ± 1.5	**79.32 ± 0.9**	99.15 ± 0.1	70.50 ± 1.2
ECC	80.82 ± 4.2	50.87 ± 2.8	71.17 ± 1.0	75.82 ± 0.5	96.05 ± 0.2	70.40 ± 1.8
GIN	80.26 ± 3.1	58.69 ± 1.6	74.76 ± 0.9	78.89 ± 0.6	99.35 ± 0.2	72.30 ± 1.2
GraphSAGE	72.22 ± 2.8	55.78 ± 2.7	71.25 ± 1.5	77.44 ± 0.7	97.40 ± 0.3	71.30 ± 2.3
Vanilla transformer	72.78 ± 3.6	51.74 ± 2.0	70.98 ± 1.1	65.67 ± 0.7	94.70 ± 0.5	72.40 ± 1.5
SAT	79.68 ± 3.4	56.72 ± 2.2	73.31 ± 1.10	76.27 ± 0.5	96.90 ± 0.4	72.90 ± 1.4
GraphiT + Diffusion	81.46 ± 2.6	**60.14 ± 2.1**	74.21 ± 1.2	76.69 ± 0.7	96.75 ± 0.5	70.60 ± 1.5
GraphiT + Adj	79.27 ± 1.7	57.84 ± 2.5	71.07 ± 1.4	77.96 ± 0.7	95.85 ± 0.4	71.90 ± 1.4
GraphiT + GCKN	81.93 ± 2.2	59.91 ± 3.2	70.98 ± 1.3	78.91 ± 0.7	98.40 ± 2.5	71.10 ± 1.5
GraFix (ours)	**85.09 ± 2.6**	57.57 ± 2.4	**75.11 ± 1.2**	71.70 ± 0.5	**99.45 ± 0.1**	**73.30 ± 1.4**

number of attention heads in [1, 4, 8]; and 4) the weight decay in [0.01, 0.001, 0.0001]. For our model, we vary: 1) the number of transformer layers from 1 to 3; 2) the number of hops k to build the subgraphs from 1 to 3, with the exception of IMDB-BINARY where the range is confined to 1 and 2; 3) the number of WL kernel iterations from 1 to 5, with the exception of the IMDB-BINARY dataset where the range is confined to 1 and 2.

Given the grid of hyperparameters, we select the model configuration with the lowest loss on the validation set and we measure its accuracy on the test set. We repeat this procedure for every fold and finally we report the average accuracy ± standard error over the 10 folds. During the training phase, stochastic gradient descent and Adam are employed as optimization algorithms, with initial learning rates set to 0.1 and 0.001, respectively. The learning rate is systematically reduced to half of its current value every 50 epochs.

We set the maximum number of epochs for all models to 1000, with an early stopping mechanism to minimize overfitting. To this end we use a predetermined patience parameter, which was set in alignment with the lowest validation loss recorded within the initial 200 epochs. More specifically, if the model validation loss failed to improve after the 200 epoch mark, this particular loss value was identified as the optimal validation loss.

5.2 Graph Classification Results

Table 2 shows that our model outperforms the competing ones, both traditional GNNs and transformer-based architectures, on four out of six datasets (MUTAG, PROTEINS, AIDS, and IMDB-BINARY), while achieving a higher mean accuracy than the vanilla transformer on all datasets.

We observe that GraFix seems to underperform on datasets characterized by more complex node features, *e.g.*, NCI1. In contrast, GraFix excels on the IMDB-

Table 3. Average per epoch training time ± standard error (measured in seconds ×10^{-}2). For each dataset, the fastest (second fasted) model is highlighted in bold (underlined).

Dataset	MUTAG	PTC	PROTEINS	NCI1	AIDS	IMDB-BINARY
Vanilla transformer	4.47±0.28	10.79±0.25	35.30±0.29	100.32±0.32	60.65±0.50	31.89±0.42
SAT	53.84±0.34	101.12±0.39	328.94±0.45	1193.23±0.88	578.34±0.38	294.15±0.39
GraphiT + Diffusion	6.74±0.28	11.26±0.23	43.06±0.22	131.82±0.32	76.95±0.22	39.68±0.23
GraphiT + Adj	7.30±0.18	11.61±0.23	44.52±0.23	111.67±0.31	76.72±0.24	40.39±0.25
GraphiT + GCKN	6.31±0.17	10.67±0.18	35.03±0.19	124.86±0.25	58.56±0.18	32.48±0.19
GraFix	**3.02±0.28**	**5.00±0.03**	**17.13±0.20**	**60.15±0.25**	**28.20±0.13**	**14.94±0.06**

BINARY dataset, where no node features are available, therefore highlighting the ability of GraFix to extract and leverage structural information.

5.3 Runtime Analysis

Table 3 shows that, in terms of training time, the proposed model is significantly faster than all the alternative graph transformers, including the vanilla transformer computed on the node features. Indeed, this is achieved by precomputing the attention coefficients rather than learning them end-to-end.

Remarkably, as observed in Table 2 this does not appear to result in a reduced classification accuracy. Note that the times shown here do not include the precomputation of the WL kernel (GraFix) or the GCKN features, diffusion kernel, and adjacency matrix positional encoding of GraphiT [20]. Finally, these training times refer to the best performing models selected during the grid search and used to generate the classification results of Table 2.

5.4 Number of Model Parameters

We compare the number of parameters of our model to that of the alternative graph transformers considered in these experiments. The results are shown in Table 4. As expected, by using a precomputed non-learnable attention matrix our model uses fewer trainable parameters, even when compared to a vanilla transformer. Our model uses less than 50% and about 65% of the number of parameters of SAT and GraphiT, respectively. Note that to compare the number of model parameters we fixed for all models the number of layers to 3 and the number of attention heads to 1.

Importantly, while fixing the attention coefficients reduces the number of trainable parameters within the model, it still maintains or improves the model accuracy. Therefore our model is able to optimize computational efficiency without sacrificing performance.

Table 4. Comparison of the number of model parameters for different configurations of GraphiT and our model. For each dataset, the model with the least number of parameters is highlighted in bold.

Dataset	MUTAG	PTC	PROTEINS	NCI1	AIDS	IMDB-BINARY
Vanilla transformer	105,154	105,858	104,898	107,074	107,138	113,410
SAT	142,594	143,298	142,338	144,514	144,578	150,850
GraphiT + Diffusion	104,386	105,090	104,130	106,306	106,370	112,642
GraphiT + Adj	104,386	105,090	104,130	106,306	106,370	112,642
GraphiT + GCKN	111,362	112,066	111,106	113,282	113,346	119,618
GraFix	**67,522**	**68,226**	**67,266**	**69,442**	**69,506**	**75,778**

5.5 Interpretability and Kernel Attention Visualization

Figure 3 shows a sample graph from the MUTAG dataset. The graph contains a core of 15 nodes (labeled in red in the figure), connected to 4 pairs of leaves (grey) through 4 nodes (orange). We visualise the attention matrices for this graph for different choices of the subgraph hops k in Fig. 4. In the latter, the yellow colour indicates a high value of the attention while blue indicates a low value. As expected, these heatmaps show the ability of attention, which is computed using the WL kernel, to discriminate between structurally dissimilar neighborhoods exiting within the graph.

The attention matrix for 1-hop subgraphs reveals distinct block diagonal patterns, clearly separating the core from the outer nodes. While the similarity between some subgraphs is close to 0 when $k = 1$, increasing k allows to progressively integrate broader structural elements of the graph, thus smoothing the similarity information.

5.6 Ablation Studies

To conclude our experimental analysis, we perform an ablation study of the network parameters on the MUTAG and PROTEINS datasets. Specifically, we study the influence of the number of subgraph hops and the number of WL iterations.

In Fig. 5, we show the average classification accuracy of our model as we vary the number of hops k from 1 to 3 and the number of WL iterations h from 3 to 5. On both MUTAG and PROTEINS our model achieves optimal performance with a limited number of iterations of the WL kernel. This can be explained by noting that as the number of iterations increases the attention matrix tends to be smoothed (as also observed in Fig. 4), consequently diminishing the discriminative power of structural features.

In terms of subgraph hops k, on the other hand, we observed different behaviours in MUTAG and PROTEINS. Notably, we observe that in the MUTAG dataset the model performance tends to increase when smaller subgraphs are used. Conversely, on the PROTEINS dataset the model tends to

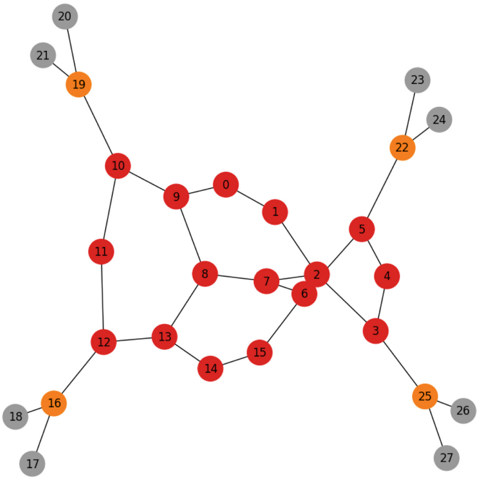
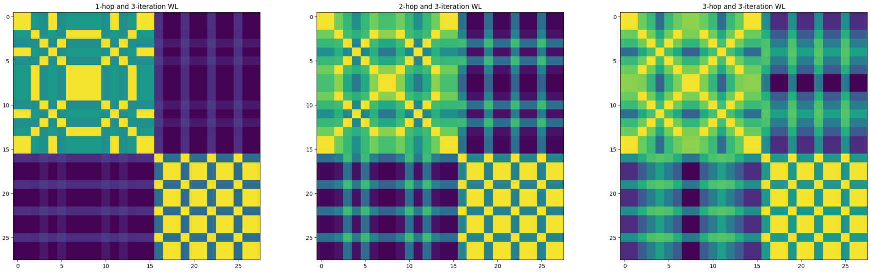


Fig. 3. Sample graph from the MUTAG dataset. Best seen in colour. (Color figure online)

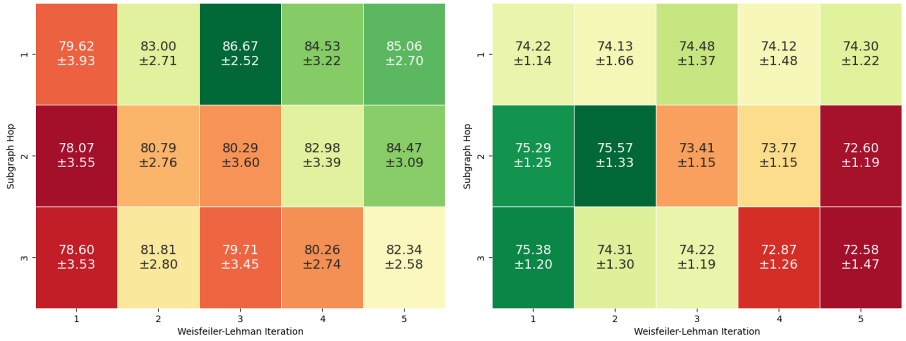


(a) 1-hop and 3-iteration (b) 2-hop and 3-iteration (c) 3-hop and 3-iteration

Fig. 4. WL kernel attention matrix for the sample graph of Fig. 3.

prefer subgraphs with an larger number of hops. This is in line with the size of graphs, smaller for MUTAG and larger for PROTEINS, as shown in Table 1.

Finally, we also compare the use of a single attention heads versus multiple ones. In Table 5 we define a simple model with two attentions heads, (i) one equivalent to extracting subgraphs with $k = 1$ hops and performing $h = 3$ WL iterations and (ii) one equivalent to extracting subgraphs with $k = 1$ hops and performing $h = 4$ WL iterations. We compare this model on the MUTAG dataset against two single-attention head models using (i) and (ii) as the head, respectively. As the results show, using multiple attention heads does not necessarily lead to a higher classification accuracy.



(a) MUTAG (b) PROTEINS

Fig. 5. Average classification accuracy ± standard error on the (a) MUTAG and (b) PROTEINS dataset as we vary both the subgraph hops k and the number of WL iterations h.

Table 5. Single- versus multi-head attention on MUTAG. k and h indicate the number of subgraph hops and the number of WL iterations, respectively.

Model	Avg. accuracy ± std. error
GraFix with single head ($k = 1, h = 3$)	**86.67±2.52**
GraFix with single head ($k = 1, h = 4$)	84.53±3.22
GraFix with two heads ($k = 1, h = 3$) + ($k = 1, h = 4$)	82.39±2.51

6 Conclusion

In this paper we introduced GraFix, a new graph transformer architecture where the attention is not learned but rather it is fixed and captures the similarity between substructures in the input graphs using graph kernels. We showed that by employing the WL kernel to precompute the attention coefficients between the nodes of the graphs the resulting model performs well when compared to alternative approaches, both GNNs and graph transformers, on standard graph classification tasks. At the same time, our model retains only a fraction of the parameters of other graph transformers and therefore benefits from a faster runtime. Future work will aim to improve the ability of the model to deal with graph datasets where features are highly discriminative. It would also be interesting to investigate the application of a similar model to time-varying graphs, with graphs themselves being interpreted as tokens.

Acknowledgments. L.C. is supported by the PRIN 2022 project n. 2022AL45R2 (EYE-FI.AI, CUP H53D2300350-0001). G.M. acknowledges financial support from the European Union - NextGenerationEU, in the framework of the iNEST - Interconnected Nord-Est Innovation Ecosystem (iNEST ECS_00000043 – CUP H43C22000540006). In this regard, the views and opinions expressed are solely those of the authors and do

not necessarily reflect those of the European Union, nor can the European Union be held responsible for them.

References

1. Bicciato, A., Cosmo, L., Minello, G., Rossi, L., Torsello, A.: Classifying me softly: a novel graph neural network based on features soft-alignment. In: Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR), pp. 43–53. Springer, Heidelberg (2022). https://doi.org/10.1007/978-3-031-23028-8_5
2. Bicciato, A., Cosmo, L., Minello, G., Rossi, L., Torsello, A.: Gnn-lofi: a novel graph neural network through localized feature-based histogram intersection. Pattern Recogn. **148**, 110210 (2024)
3. Chen, D., Jacob, L., Mairal, J.: Convolutional kernel networks for graph-structured data. In: International Conference on Machine Learning, pp. 1576–1586. PMLR (2020)
4. Chen, D., O'Bray, L., Borgwardt, K.: Structure-aware transformer for graph representation learning. In: International Conference on Machine Learning, pp. 3469–3489. PMLR (2022)
5. Child, R., Gray, S., Radford, A., Sutskever, I.: Generating long sequences with sparse transformers. arXiv preprint arXiv:1904.10509 (2019)
6. Cosmo, L., et al.: Graph kernel neural networks. IEEE Trans. Neural Netw. Learn. Syst. (2024)
7. Dong, L., Xu, S., Xu, B.: Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5884–5888. IEEE (2018)
8. Dosovitskiy, A., et al.: An image is worth 16x16 words: transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
9. Dwivedi, V.P., Bresson, X.: A generalization of transformer networks to graphs. arXiv preprint arXiv:2012.09699 (2020)
10. Errica, F., Podda, M., Bacciu, D., Micheli, A.: A fair comparison of graph neural networks for graph classification. arXiv preprint arXiv:1912.09893 (2019)
11. Gallagher-Syed, A., et al.: Multi-stain self-attention graph multiple instance learning pipeline for histopathology whole slide images. arXiv preprint arXiv:2309.10650 (2023)
12. Guo, J., et al.: CMT: convolutional neural networks meet vision transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12175–12185 (2022)
13. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
14. Hearst, M.A., Dumais, S.T., Osuna, E., Platt, J., Scholkopf, B.: Support vector machines. IEEE Intell. Syst. Appl. **13**(4), 18–28 (1998)
15. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
16. Kreuzer, D., Beaini, D., Hamilton, W., Létourneau, V., Tossou, P.: Rethinking graph transformers with spectral attention. In: Advances on Neural Information Processing System, vol. 34, pp. 21618–21629 (2021)
17. Kriege, N., Mutzel, P.: Subgraph matching kernels for attributed graphs. arXiv preprint arXiv:1206.6483 (2012)

18. Kriege, N.M., Johansson, F.D., Morris, C.: A survey on graph kernels. Appl. Netw. Sci. **5**, 1–42 (2020)
19. Liu, Y., Rossi, L., Torsello, A.: A novel graph kernel based on the Wasserstein distance and spectral signatures. In: S+SSPR 2022, pp. 122–131. Springer, Heidelberg (2022). https://doi.org/10.1007/978-3-031-23028-8_13
20. Mialon, G., Chen, D., Selosse, M., Mairal, J.: Graphit: encoding graph structure in transformers. arXiv preprint arXiv:2106.05667 (2021)
21. Min, E., et al.: Transformer for graphs: an overview from architecture perspective. arXiv preprint arXiv:2202.08455 (2022)
22. Minello, G., Rossi, L., Torsello, A.: Can a quantum walk tell which is which? a study of quantum walk-based graph similarity. Entropy **21**(3), 328 (2019)
23. Morris, C., Kriege, N.M., Bause, F., Kersting, K., Mutzel, P., Neumann, M.: Tudataset: a collection of benchmark datasets for learning with graphs. In: ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020) (2020). www.graphlearning.io
24. Raganato, A., Scherrer, Y., Tiedemann, J.: Fixed encoder self-attention patterns in transformer-based machine translation. In: Findings of ACL, vol. EMNLP 2020, pp. 556–568. Association for Computational Linguistics (2020)
25. Rampášek, L., Galkin, M., Dwivedi, V.P., Luu, A.T., Wolf, G., Beaini, D.: Recipe for a general, powerful, scalable graph transformer. In: Advances on Neural Information Processing System, vol. 35, pp. 14501–14515 (2022)
26. Rossi, L., Williams, M., Stich, C., Musolesi, M.: Privacy and the city: user identification and location semantics in location-based social networks. In: Proceedings of the International AAAI Conference on Web and Social Media, vol. 9, pp. 387–396 (2015)
27. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE Trans. Neural Netw. **20**(1), 61–80 (2008)
28. Senior, H., Slabaugh, G., Yuan, S., Rossi, L.: Graph neural networks in vision-language image understanding: a survey. Visual Comput., 1–26 (2024)
29. Shervashidze, N., Schweitzer, P., Van Leeuwen, E.J., Mehlhorn, K., Borgwardt, K.M.: Weisfeiler-lehman graph kernels. J. Mach. Learn. Res. **12**(9) (2011)
30. Simonovsky, M., Komodakis, N.: Dynamic edge-conditioned filters in convolutional neural networks on graphs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3693–3702 (2017)
31. Sugiyama, M., Borgwardt, K.: Halting in random walk kernels. In: Advances in Neural Information Processing Systems, vol. 28 (2015)
32. Tsai, Y.H.H., Bai, S., Yamada, M., Morency, L.P., Salakhutdinov, R.: Transformer dissection: a unified understanding of transformer's attention via the lens of kernel. arXiv preprint arXiv:1908.11775 (2019)
33. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
34. Wu, Z., Jain, P., Wright, M., Mirhoseini, A., Gonzalez, J.E., Stoica, I.: Representing long-range context for graph neural networks with global attention. In: Advances in Neural Information Processing Systems, vol. 34, pp. 13266–13279 (2021)
35. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? arXiv preprint arXiv:1810.00826 (2018)
36. Yanardag, P., Vishwanathan, S.: Deep graph kernels. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1365–1374 (2015)

37. Ying, C., et al.: Do transformers really perform badly for graph representation? In: Advances in Neural Information Processing Systems, vol. 34, pp. 28877–28888 (2021)
38. Ying, Z., You, J., Morris, C., Ren, X., Hamilton, W., Leskovec, J.: Hierarchical graph representation learning with differentiable pooling. In: Advances in Neural Information Processing Systems, vol. 31 (2018)
39. Zhang, B., Luo, S., Wang, L., He, D.: Rethinking the expressive power of gnns via graph biconnectivity. arXiv preprint arXiv:2301.09505 (2023)
40. Zhang, M., Cui, Z., Neumann, M., Chen, Y.: An end-to-end deep learning architecture for graph classification. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)

Multi-modal Deep Emotion-Cause Pair Extraction for Video Corpus

Qianli Zhao, Linlin Zong(✉), Bo Xu, Xianchao Zhang, and Xinyue Liu

Dalian University of Technology, Dalian, China
`llzong@dlut.edu.cn`
`https://www.dlut.edu.cn`

Abstract. Video corpus comprises multimedia information located in multiple modalities, which brings evident challenge to emotion-cause extraction tasks. Traditional approaches perform indifferently on multi-modal datasets due to the inadequate inter-modal alignment and fusion. We propose a Multi-modal Deep Emotion-Cause Pair Extraction algorithm, which introduces a novel inter-modal attention mechanism to effectively align and fuse text, audio, and video features. The proposed two-stage algorithm first extracts modal pairwise fusion features respectively for emotions and causes. Then, the pairwise features are jointly fused by the emotion-cause fusion module to mine the relationship between emotions and causes. Finally, we utilize a multi-modal fusion module and classifier to identify the emotion-pair relationship among utterances. Experiments show that the proposed algorithm improves the performance of the extraction of multi-modal emotion-cause pairs compared to baseline approaches on the current publicly available dataset.

Keywords: Deep emotion-cause pair extraction · Multi-modal Feature Fusion · Cross Attention alignment

1 Introduction

Nowadays, social media such as Twitter and TikTok generate huge amount of video data. Video corpora are retrieved from such data which contains multi modalities of textual, audio, and video information. Research on emotion identification has immense potential value on public opinion analysis, recommendation, and decision-making systems. Traditionally, emotion-cause extraction [13] has attracted much attention in natural language processing universe. The main goal [8] of this task is to extract the emotions and the causes that trigger the corresponding emotions from a given corpus. Recently, with the assistance of the deep neural networks, the Deep Emotion-Cause Pair Extraction [23] task aims to recognize emotion-cause pairs simultaneously based on deep learning technology.

The classic emotion-cause pair extraction algorithms work on textual dataset depending on emotion annotation. A growing body of research [5,7,21] has applied deep learning approaches to the task of emotional cause extraction in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 451–466, 2025.
https://doi.org/10.1007/978-3-031-78128-5_29

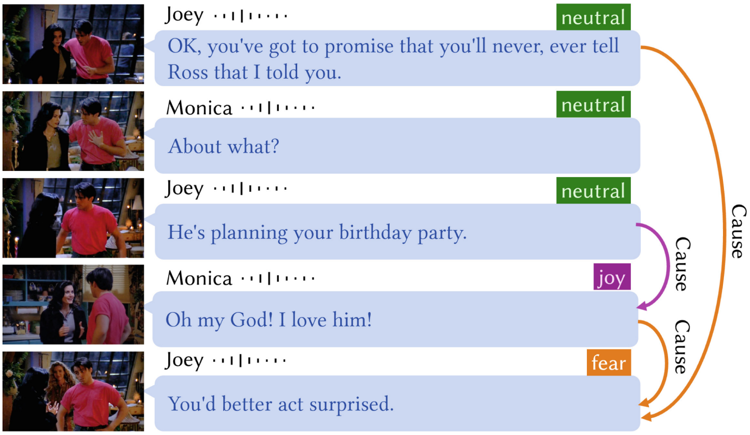

Joey
neutral
OK, you've got to promise that you'll never, ever tell Ross that I told you.
Monica
neutral
About what?
Joey
neutral
He's planning your birthday party.
Monica
joy
Oh my God! I love him!
Joey
fear
You'd better act surprised.
Cause
Cause
Cause

Fig. 1. An example of emotion-cause pair extraction

conversation scenarios. By complicated neural network models, it is possible to recognize precise emotions with interaction of multiple modalities.

Figure 1 illustrates a three modal emotion-cause pair extraction example excerpted from a video corpus. There are five utterances in the dialog, in which each utterance has an emotional label. We aim to identify the causes of specific utterance by ensemble all three modal features by the proposed algorithms. In this case, three emotion-cause pairs are extracted from the dialog example. It is clear that an utterance can simultaneously be the source and the target of cause relationships.

The research made in multi-modal learning have received widespread attention. The multi-modal approaches involve the study of emotional consensus, which effectively improves the accuracy of multimodal sentiment analysis. Multimodal emotion analysis identifies human emotional states using multiple different perceptual modalities (e.g., audio, text, images).

The task of multi-modal emotion-cause pair extraction proposes to extract possible emotions and their corresponding triggering causes from data corpus that contains modal combinations such as text-audio, text-video, or text-audio-video. The emotion data is more detailed, the expression of emotions is more diversified, and the recognition of emotions is more difficult. Therefore, extracting emotion-causes pairs also relies more on the learning and fusion methods of multi-modal feature representations, and the multimodal techniques need to be adapted to the extraction of emotion-cause pairs. Emotion-cause pair extraction methods applied in multimodal scenes can effectively establish the association between textual, audio and image emotion-cause information. Traditional approaches performs indifferently on multimodal datasets due to the inadequate inter-modal alignment and fusion.

In this paper, we propose an attention-based multimodal emotion-cause pair extraction algorithm for extracting structural units expressing emotion-cause in a multimodal dialog corpus. The algorithm introduces an inter-modal attention mechanism to effectively fuse text, audio and image features to improve the accuracy of extracting emotion-cause pairs.

The model consists of two stages. The first stage is multimodal emotion-cause information fusion, and the second stage is multimodal emotion-cause pair extraction. In the first stage, we first extract the three modalities of text (T), audio (A), and video (V) from the dialog dataset. High-dimensional feature extraction algorithms are applied to get the high-dimensional features, and then the proposed algorithm transfers the latent features to an attention fusion module to retrieve pairwise inter-modality features. Next, the fusion results of the multimodal feature fusion are fed to the extraction of the emotion-cause fusion module respectively. In the second stage, the predicted emotion-cause information from the first step is paired, and the matched emotion-cause pairs are labeled from the candidate pairs.

In summary, we list the contribution of this paper as follows.

(1) We propound that the inadequate alignment and fusion is the bottleneck of multi-modal emotion-cause pair extraction algorithms.
(2) We propose a novel attention-based multimodal emotion-cause pair extraction algorithm. The algorithm utilizes emotion-cause cross attention to align inter-modal features and a multimodal emotion-cause fusion network to fuse features.
(3) Experiments show that the algorithm proposed in this paper improves the performance of the extraction of inter-modal emotion-cause pairs compared to baseline approaches on the current publicly available corpus set.

2 Related Work

In recent years, many schemes addressing the task of emotion-cause pair extraction have been proposed. We firstly introduce the algorithms that work on textual datasets, then on multi-modal scenarios.

2.1 Textual Emotion Cause Extraction

In the early stages, the emotion-cause extraction task needs to be accomplished by manually annotating emotions. In a subsequent study, Gui [9] proposed a dataset of emotion cause extraction based on Sina News in 2016 and found that the sentence-level emotion-cause extraction task works better than the word-level emotion-cause extraction task. Based on the same goal, Li [16] proposed a CNN model with a common attention mechanism, and Chen [2] used a joint training approach for enhanced cause feature extraction. Li et al. proposed a graph convolutional network named ConGCN [15], which extract the features of e speakers and utterance contexts to build a heteromerous graph which is used to help solve the problem of conversational emotion recognition.

The emotion-cause pair extraction task [23] is proposed in 2019, which aims to extract the corpus parts with emotion information and the cause corpus parts that cause emotions from documents, getting rid of the dependence on manually labeled emotion labels.

Indep [23] is a two-step pipeline model that uses a bidirectional LSTM for encoding and extracting emotion-cause sentences, and final pairing for prediction. The Inter-EC model is based on the Indep model, but it extracts the emotion sentences first in a Bi-LSTM in the contextual semantic information of the learning clause, and then extracts the cause sentences based on the known emotion information.

The end-to-end model transforms the extraction task of emotion-causal pairs from the filtering of emotional and causal information at the sentence level to the categorization of sentence pairs $p_{ij} = (u_i, u_j)$. The end-to-end model focuses more on the learning of sentence-to-sentence relations, making emotion-cause extraction truly a relation discovery task rather than an information extraction task. The ECPE-2D [5] model uses a two-dimensional clause candidate pair approach to represent candidate emotion-cause pairs and proposes two variants of Transformer to model the relationship between emotion-cause pairs. The Pair-GCN [3] model serves as a candidate pair graph for nodes by constructing emotion-cause pairs between two clause window distances. The model learns the emotion-cause relationships between sentence nodes through a novel transformational parsing model with joint decoding. Trans ECPE [7] model learns the sentiment-cause relationships between clause nodes in a document through a novel Transformer parsing model with joint decoding. Rank-CP [21] model uses an end-to-end model to rank candidate pairs with scores, model the attentional relationships between sentences, then rank them by graphical attentional networks, and augment the ranking results by generating the relative position of kernel functions approach. The RSN [1] model uses Recurrent Synchronization Networks to jointly extract sentiment clauses, cause clauses, and sentiment-cause pairs. The PTF [22] model addresses all subtasks of emotion-cause pair extraction using global and unified clause pair labeling tasks instead of multitask learning.

2.2 Multimodal Emotion Cause Extraction

MM-DFM [10] uses a dynamic fusion module in its approach to multimodal fusion, which enables the fusion of multimodal contextual features. EmoCaps [17] extracts emotional tendencies and performs modal fusion with emotion capsules. Li et al. [14] proposed a two-stage multi-source multimodal fusion method based on graph attention to perform modal fusion and solve the heterogeneous gap problem. M2FNet [4] employs a fusion mechanism based on multiple heads of attention to fuse the emotion representations of the input data to obtain potentially rich emotion information. UniMSE [11] constructs a multi-modal knowledge sharing framework to jointly learn from different modalities. The current approach to solving the multimodal task is mainly to fuse multimodal high-dimensional features and predict the emotion-cause candidate pairs of the fused and unimodal data across different modalities by a classifier, which need more explorations and designs to unify knowledge from different modalities.

3 The Proposed Algorithm

The multi-modal deep emotion-cause pair extraction algorithm is proposed to extract emotion-cause pair $p_{ij} = \text{MMDECPE}(x_i, x_j)$ information from multi-modal corpus which consists of three modalities: text X_T, audio X_A, and video X_V, where p_{ij} is a binary value, x_i, x_j are utterances from the corpus. $p_{ij} = 1$ means that there exists an emotion-cause pair between utterance i and j, and vice versa. The proposed model for emotion-cause pair extraction for video corpus is shown in Fig. 2.

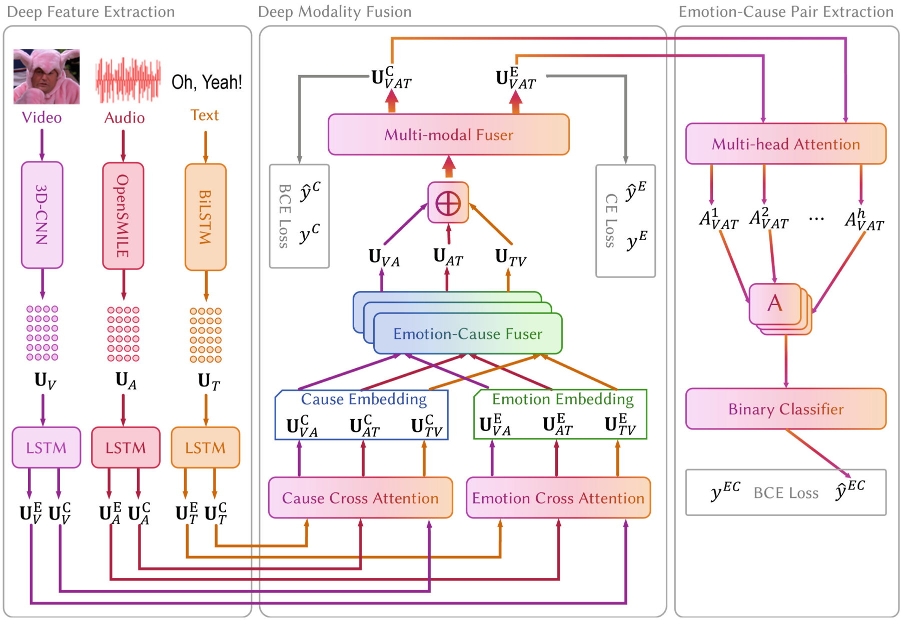


Fig. 2. The framework of the proposed algorithm

The proposed algorithm consists of two stages. The first stage is multimodal emotion-cause information fusion, and the second stage is multimodal emotion-cause pair extraction. In the first stage, we first extract the three modalities of text (T), audio (A), and video (V) from the dialog dataset. High-dimensional feature extraction algorithms are applied to get the high-dimensional features, and then the proposed algorithm transfers the latent features to an attention fusion module to retrieve inter-modality feature. Next, the fusion vectors of the multimodal feature fusion are subjected to the extraction of the emotion-cause fusion module respectively. In the second stage, the predicted emotion-cause information from the first step is paired, and the matched emotion-cause pairs are labeled from the candidate pairs.

3.1 High-Dimensional Deep Feature Extraction

In this paper, to learn the textual word-level contextual information, the feature modeling for textual modality (T) is generated by the bidirectional long and short-term memory network (BiLSTM). The sentence vectors are obtained by the textual modality encoder, the word vector length is set to 200:

$$U_T = \text{BiLSTM}(X_T) \tag{1}$$

The audio feature (A) is achieved by the now popular sound feature extraction tool openSMILE [6]. At the end of pre-training, the dimension size of the audio features obtained is 6373 dimensions. The audio feature vector is obtained by the sound coding tool as follows:

$$U_A = \text{openSMILE}(X_A) \tag{2}$$

For the video features (V), we use the 3D-CNN [12] network to learning the temporal features of the video. Only the temporal features labeled in the dataset are used for pre-training. We select 16 frames from each video with a resolution of 171×128, which were then provided to the 3D-CNN network to extract 4096-dimensional video features, followed by dimensionality reduction using a linear layer. After pre-training, 128-dimensional video features of each discourse are generated. The model extracts the video features for this dataset and obtains the discourse feature vector for the video modality:

$$U_V = \text{3D-CNN}(X_V) \tag{3}$$

3.2 Multi-modal Attentional Emotion-Cause Fusion

The proposed algorithm utilizes a cross-modal attentional fusion mechanism which employs an attentional mechanism to fuse features across modalities. Specifically, three independent contextual information learning LSTM networks are used to model contextual information for different modalities. In this way, representations of three modal with contextual information are obtained. Networks with the same structure are used for emotion and cause representations to obtain representations with contextual cause information:

$$\begin{aligned} U_T^E &= \text{LSTM}(U_T), \quad U_T^C = \text{LSTM}(U_T), \\ U_A^E &= \text{LSTM}(U_A), \quad U_A^C = \text{LSTM}(U_A), \\ U_V^E &= \text{LSTM}(U_V), \quad U_V^C = \text{LSTM}(U_V), \end{aligned} \tag{4}$$

where the parameters of the three contextual information learning networks comprising LSTM networks are independent of each other.

Once the representations are extracted, the emotion and cause information of pairwise modalities are retrieved by the cross-attention module to learn inter-modality contextual information:

$$\begin{aligned} U_{TV}^E &= \text{CR-ATTN}(U_T^E, U_V^E, U_V^E), \quad U_{TV}^C = \text{CR-ATTN}(U_T^C, U_V^C, U_V^C), \\ U_{AT}^E &= \text{CR-ATTN}(U_A^E, U_T^E, U_T^E), \quad U_{AT}^C = \text{CR-ATTN}(U_A^C, U_T^C, U_T^C), \\ U_{VA}^E &= \text{CR-ATTN}(U_V^E, U_A^E, U_A^E), \quad U_{VA}^C = \text{CR-ATTN}(U_V^C, U_A^C, U_A^C), \end{aligned} \tag{5}$$

where CR-ATTN is the cross-attention module. Here we exemplify the calculation of U_{AT}^{E} as

$$\begin{aligned} Q_A^E &= W^Q U_A^E, \\ K_T^E &= W^K U_V^E, \\ V_T^E &= W^V U_T^E, \\ U_{AT}^E &= \mathrm{softmax}\left(\frac{Q_A^E {K_T^E}^T}{\sqrt{d_k}}\right) V_T^E, \end{aligned} \tag{6}$$

where d_k is the feature dimension of K_T^E, $W^{\cdot}$ are parameters. The other five items in Eq. 5 are calculated similarly.

Next, with these representations and the pairwise modality representation, emotion-cause fusing mechanism is applied for emotion-cause fusion to learn casual relationships. This module applies the emotion-cause fusion to respective pairwise modalities features.

$$\begin{aligned} U_{TV} &= \text{EC-FUSER}(U_{TV}^E \oplus U_{TV}^C), \\ U_{AT} &= \text{EC-FUSER}(U_{AT}^E \oplus U_{AT}^C), \\ U_{VA} &= \text{EC-FUSER}(U_{VA}^E \oplus U_{VA}^C), \end{aligned} \tag{7}$$

where $\oplus$ indicates the concatenation of the two matrix. The EC-FUSER module consist multi-head self attention transforms. The algorithm learns emotion-cause alignment by the emotion-cause fusion.

After emotion-cause fusion, the pairwise modal latent features are used to generate the final three-modal fusion features. We use a siamese network as multimodal fusion module to generate final features. The multimodal fusion module works as follows.

$$U_{VAT}^{\prime E} = \mathrm{softmax}\left(W^E \left(U_{TV} \oplus U_{AT} \oplus U_{VA}\right) + B^E\right) \tag{8}$$

$$U_{VAT}^{E} = \sigma\left(W^E U_{VAT}^{\prime E} + B^E\right) \tag{9}$$

$$U_{VAT}^{\prime C} = \mathrm{softmax}\left(W^E \left(U_{TV} \oplus U_{AT} \oplus U_{VA}\right) + B^C\right) \tag{10}$$

$$U_{VAT}^{C} = \sigma\left(W^C U_{VAT}^{\prime C} + B^C\right) \tag{11}$$

where softmax and σ denotes the activation function, $W^{\cdot}$ and $B^{\cdot}$ denote the weight and bias parameters. U_{VAT}^{E} and U_{VAT}^{C} denote the emotion information and cause information extracted from the multimodal corpus. The multimodal feature representation extraction networks share the same parameter settings and network structure.

The multimodal emotion-cause features obtained after cross-attention fusion are classified by the classification network. The predicted emotional and cause label of utterance segments are obtained. The classifier consists of simple feed-forward layers

$$\hat{y}^C = \sigma(W^C U_{VAT}^C + b^C), \tag{12}$$

$$\hat{y}^E = \sigma(W^E U_{VAT}^E + b^E), \tag{13}$$

where $W^{\cdot}$ and $b^{\cdot}$ denote the weight and bias parameters,σ denotes the activation function. $\hat{y}_c \in \mathbb{R}^{2\times 1}$ denotes if the utterance is a cause of other utterance. $\hat{y}_e \in \mathbb{R}^{7\times 1}$ denotes the emotion label of the utterance.

Since the algorithm contains two stages that require independent optimization, at the first stage, the training losses for multimodal emotion extraction and cause extraction are shown as

$$L_1 = \alpha L^E + \beta L^C, \tag{14}$$

$$L^E = \text{CrossEntropy}(y^E, \hat{y}^E), \tag{15}$$

$$L^C = \text{BinaryCrossEntropy}(y^C, \hat{y}^C), \tag{16}$$

where y^E is the ground truth of emotion provided with the dataset, y^C is the ground truth of cause, which is extracted from the emotion-cause pair label and described in Sect. 4.1, α and β are user parameters which are used to regulate the loss of emotion and cause extraction. The L_1 loss guides the model to learn reasonable representations. Therefore, this model try to balance the impact of emotion and cause extraction tasks, thus improving the performance of the model.

In addition, the training of the first stage isolated with the second stage, and the parameter tuning in the first phase is independent of the subsequent training. When the training is completed, the parameters are fixed.

3.3 Multimodal Emotion-Cause Pair Extraction

In order to extract emotion-cause pairs, this paper pairs the above emotion representation U^E_{TAV} with the cause U^C_{TAV} representation by a Multi-head Attention layer. To get the candidate pairs $(u^E_{TAV}, u^C_{TAV})_i$, we concatenate the attention weight of all heads to form the emotion cause candidate pairs of the utterance matrix P . This matrix will help to infer the matching emotional cause pairs.

$$P = [AW_1(U^E_{TAV}, U^C_{TAV}); AW_2(U^E_{TAV}, U^C_{TAV}); ...; AW_h(U^E_{TAV}, U^C_{TAV})], \tag{17}$$

where AW means attention weight calculation module.

For the emotion-cause candidate pairs p_{ij} in P, the predicted emotion-cause pairs $\hat{y}^{EC}$ are obtained using an emotion-cause filter network, which consists of a fully connected layer :

$$\hat{y}^{EC} = \sigma(W^{EC} P_i + b^{EC}) \tag{18}$$

where σ denotes the activation function of the fully connected layer, and here the ReLU is used as the activation function, and W^{EC} and b^{EC} are the weight parameters and bias of the classifier network. $\hat{y}^{EC} \in \mathbb{R}^{2\times 1}$ is a one-hot binary value which indicates whether the pair is an emotion-cause pair.

The second stage is emotional-causal pair filtering, and the loss in the second stage is

$$L_2 = L^{EC} = \sum_i \sum_j y^{EC}_{ij} \log \hat{y}^{EC}_{ij}, \tag{19}$$

which is also a cross entropy loss.

As mentioned, the proposed algorithm consist of two stages. We first train the multi-modal attentional emotion-cause fusion part according to Eq. 14. After accomplishing the training of multi-modal attentional emotion-cause fusion, we train this part of model using Eq. 19.

4 Experiments

In the experiments section, we design three sets of experiments, validation on emotion-cause pair extraction accuracy, validation on emotion extraction accuracy, and ablation study.

4.1 Experimental Setup

Datasets. The dataset used in this chapter was compiled based on the MELD [18] multimodal sentiment analysis dataset by adding sentiment cause tags and removing dialogues with inconsistent timestamps. The MELD dataset is a multimodal video corpus that was collected and annotated from the TV series "Friends". MELD contains approximately 13,000 utterances from 1433 dialogues. Each utterance is labeled with emotion and sentiment tags, and includes audio, visual and text modals. The conversational utterance in the MELD dataset are all from the seven characters in the TV series.

The benchmark dataset Emotion-Cause-in-Friends (ECF) for the MECPE [19] task is obtained by annotating the emotion-cause labels. In the MECPE dataset, labels such as Emotional Cause Label Position and seven types of emotional categorization are added, which are Anger, Disgust, Fear, Joy, Sadness, Surprise, and Neutral. We divide the dataset into train set, dev set, and test set. Among all the labels of 13619 discourses in the ECF dataset, Neutral expressions meaning no special emotion accounts for 43.53%, Disgust and Fear have the lowest percentage, 3.04% and 2.74%, respectively; and Joy and Surprise have the highest percentage, 16.9% and 13.51%, respectively. The percentages of Anger and Sadness are 8.42% and 11.86%, respectively.

In the ECF dataset, the utterance index of the emotion-cause labels are marked, which enables us to more precisely identify the causes or events in emotional experiences. The cause label which is utilized in the first of the proposed algorithm are retrieved from the emotion-cause label as follows. It can be observed that not each utterance involves in Emotion-Cause relationship in the dataset, and the distribution of Emotion-Cause pair label is sparse. We extract all utterances that function as a Cause role in at least one Emotion-Cause pair from the corpus, then assign these utterances with Cause Label $y^C = 1$, other utterances that do not play as the Cause role with Cause Label $y^C = 0$.

Parameter Setting. In the training process, this paper uses the Adam optimizer for training, the learning rate l_1 is set to 0.005 The regularization factor l_2 is set to $1e - 5$. This paper sets the batch size of the experimental data batch to 32. For the parameters in the Loss, this paper uses the optimal settings recommended by it. For Bi-LSTM the vector size of the pre-trained sentence vectors is set to 200. α,β are set to 1.0 and 1.0 for the parameter settings in Loss respectively.

Baselines. In the experiments section, we design three types of experiments, validation on emotion-cause pair extraction accuracy, validation on emotion extraction accuracy on imbalance emotion label, and ablation study.

For the first validation, as there exist little arts on emotion-cause pair extraction, we pick the MECPE [19] algorithm as the main baseline. In addition, we design four heuristic models to validate the effectiveness of the algorithm, two emotion label prediction method and two cause distribution assumptions. In the four heuristic settings, The emotion label in the train set can be retrieved either by predicting the distribution of emotions or directly using the ground truth label.

We describe the difference between Bernoulli and multinomial distribution about cause label determination. The cause label y^C is a binary value if we choose Bernoulli distribution. We set $y^C = 1$ when an utterance works as the Cause role in at least one exist emotion-cause pair label, and vice versa. When multinomial distribution is selected, y^C is a continuous value in range $[0, 1]$. The variable p means the utterance involved in Q emotion-cause pair(s) as the Cause role, $Q \in \{0, 1, 2, 3+\}$. The parameter settings are based on the statistics of the whole dataset. We set $y^C = p(x|0 < u \leq Q)$ means that it accumulates the possibility of one utterance u which occurs as a cause role more than once.

By combine the different methods respectively, we obtain the four heuristic models:

1. Emotion prediction and Bernoulli model $E_P + C_B$;
2. Emotion label and Bernoulli model $E_L + C_B$;
3. Emotion prediction and multinomial model $E_P + C_M$;
4. Emotion label and multinomial model $E_L + C_M$.

For the second validation, we take ConGCN [15], MM-DFN [10], EmoCaps [17], EmpaGen [20] GA2MIF [14], MMTr [24] as baseline methods. They are not designed to detect emotion-cause pairs, nevertheless they can classify the utterance to emotional categories.

Evaluation Metrics. To assess the accuracy of the emotion-cause pair extraction, this paper uses the scores of Precision(P), Recall(R), and F1-score(F_1) as the evaluation metrics of the model's performance. Among all three measures, the values range from 0 to 1, and the higher value means the better performance.

In the second experiment set, we only use weighted F1-score as the criterion, as all selected state-of-art work reported this value. The weighted F1-score are calculated as follows

$$WF_1 = \frac{\sum_{k=1}^{K} n_k \cdot (F_1)_k}{\sum_{k=1}^{K} n_k},$$

where K mean the number of categories, n_k indicates number of the utterances in the k-th category.

4.2 Experiments on Emotion-Cause Pair Extraction Accuracy

The proposed algorithm mainly aims to improve the performance of extracting multimodal emotion-cause pairs, in which the multi-modal attention-based fusion mechanism contributes conspicuously to improvements. Through experimental comparisons, this model shows that the fusion model based on intermodal attention is superior and effective in dealing with emotional cause pair relations. Emotion-cause pair extraction experiments derive the data in Table 1.

Table 1. The performance results on emotion-pair extraction

Methods	Emotion Extraction			Cause Extraction			Emotion-Cause Pair Extraction		
	P	R	F1	P	R	F1	P	R	F1
$E_P + C_B$	0.7362	0.7968	0.7648	0.5491	0.5028	0.5244	0.3699	0.2677	0.3101
$E_L + C_B$	1	1	1	0.642	0.5485	0.5915	0.4958	0.3307	0.3967
$E_P + C_M$	0.7362	0.7968	0.7648	0.5488	0.5022	0.5239	0.3694	0.2671	0.3096
$E_L + C_M$	1	1	1	0.6417	0.5483	0.5913	0.4951	0.3303	0.3963
MECPE	0.7474	0.7957	0.77	**0.6522**	0.7776	0.7084	0.5483	0.5009	0.522
Ours	**0.7507**	**0.7996**	**0.7734**	0.6461	**0.7915**	**0.7109**	**0.5497**	**0.5025**	**0.525**

As displayed in Table 1, in the multimodal emotional cause pair extraction task, this model mainly outperforms the heuristic baseline models and the MECPE model. The table shows the accuracy rate of inter-modal feature fusion is improved compared to the MECPE model in comparison to the mechanism without any attentional fusion. The accuracy rate of intermodal attentional fusion is further improved. This suggests that the model of intermodal attentional multimodal feature fusion performs better in multimodal emotion inference and cause extraction tasks. Based on the comparison of these three experiments, the application of cross-modal inter-modal attention mechanisms between modalities is useful, and emotional cause contribute to the accuracy rate of extraction. Although the ground truth of emotion label in the test set is provided, without the adequate cause information, it is not able to improve the results single-handedly. Without aligning and fusing with each other, the experiment results are not as well as MECPE and the proposed algorithm.

In the MECPE algorithm, the learning of contextual information is performed after the multimodal feature fusion. However, in the attention-based multimodal feature fusion model, the learning of contextual information is performed by each modal feature separately in the attentional feature fusion. The attention model allows the data between different modalities to interact and intermingle, which improves the efficiency of emotion and cause pair extraction, and also enhances the model's ability to perceive semantic information. In the intermodal attention multimodal feature fusion model, the interaction of any two different feature sequences can be learned, and thus the model enhances the relevance of multimodal information from different modalities.

In summary, this experiment set verifies two issues. The first one is that the inter-modal attention mechanism can provide importance weights for features in different modalities, which makes it easier to discover and extract critical factors in emotion-cause pair extraction tasks. The second one is that distribution selection of cause did not influence the results obviously without the proposed aligning and fusing mechanism.

4.3 Experiments on Emotion Recognition

To evaluate the proposed model on emotion recognition task, we perform experiment on the ECF dataset with the proposed and existing methods. All models are multi-modal algorithms except for ConGCN which works on texts. We calculate the emotional label $\hat{y}^E$ of utterances compared with the baseline algorithms. We report the F1-score for each emotion category and weighted F1-score as the summary. We also illustrate the result of this settings on Fig. 3 which dropped the least two emotion utterances.

Table 2. Performance on multi-modal emotion recognition

Model	Neutral	Joy	Surprise	Anger	Sadness	Disgust	Fear	WF1
ConGCN	0.7670	0.5310	0.5030	0.4680	0.2850	0.1060	0.0870	0.594
EmoCaps	0.7712	0.5750	**0.6319**	**0.5754**	**0.4252**	0.0769	0.0303	0.64
MM-DFN	0.7776	0.5478	0.5069	0.4782	0.2294	–	–	0.5946
EmpaGen	0.792	0.618	0.556	0.502	0.276	0.109	0.095	0.596
GA2MIF	0.7692	0.5187	0.4908	0.4852	0.2718	–	–	0.5894
MMTr	0.7734	**0.6002**	0.5714	0.4824	0.3631	**0.2901**	**0.2048**	0.6441
Ours	**0.7865**	0.5964	0.589	0.573	0.3124	0.192	0.1	**0.6537**

The results in Table 2 indicate that the model proposed in this paper achieved an improvement in the overall weighted F1-score of the emotion recognition task on the MELD dataset, as the proposed framework aligns and adequately fuses the information among all modalities. The proposed framework with the early fusing and aligning mechanism also contributes to the best accuracy on the Neutral category. For other categories, the algorithm achieves competitive results with baseline algorithms, although it is the intermediate task.

The ConGCN does not accumulate the audio and video information, which leads to a weak performance. The Emocap produces best F1-score on the Surprise, Sadness and anger categories, probably because it employs the emotion capsules to maintain more information about emotion. The MMTr works best on Joy, Disgust and Fear categories, probably because it introduces L_2 norm on the loss function which regulates imbalance of categories.

From Fig. 3, we can observe that the imbalance in the ECF dataset is manifested in the different shares of data labels. As the share of emotion categories decreases, the accuracy of all algorithm decay which means that the lack of samples do harm to the model performance.

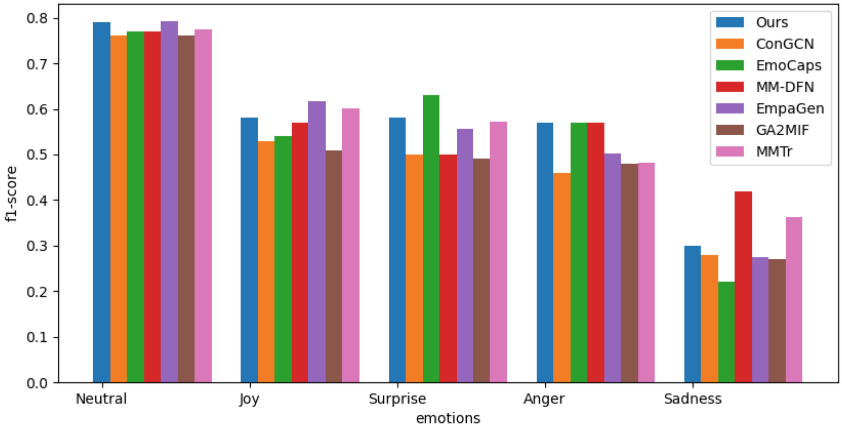


Fig. 3. Performance on different emotion labels.

In a word, the proposed algorithm can get competitive accuracy result on emotion recognition task with current baselines. The result shows that the inter-modality alignment and fusion enhance generalization of the model.

4.4 Ablation Study

In order to further explore the interactions between different modalities, we perform an ablation experiment on the emotion-cause pair extraction task. Specifically, the purpose of this experiment is to explore the contribution of different modalities.

The experimental result are displayed in Table 3. The first three columns are modality indentifiers, and only the modality with the ✓ mark is adopted in the experiment. When only one modality is adopted, the emotional-cause pair extraction framework omits the relevant multimodal fusion parts.

First, in the ablation experiments, fusing all modalities gives the best results for the emotional cause pair extraction task. When only text modalities are used, the model performs poorly on the emotional cause pair extraction task, while the performance of the model is significantly improved when other modalities are introduced into the model. Based on the experimental results, it is assumed that the relevant features in the video modality that express the emotion-cause

Table 3. Performance on Ablation Study

Text	Audio	Video	Precision	Recall	F1
✓			0.5412	0.4574	0.4957
	✓		0.4332	0.4014	0.42
		✓	0.3578	0.3195	0.3376
✓	✓		0.5376	0.4874	0.5112
✓		✓	0.5281	0.4839	0.505
✓	✓	✓	0.5546	0.5104	0.5315

information may not be sufficiently captured and extracted, which further affects the performance of the model. It is probably because the information entropy of audio and video modality is much less than that of text modality. Therefore, it can be concluded that using multi-modality may be an effective way to improve model performance.

Furthermore, the performance of multimodal sentiment analysis was investigated through ablation experiments, and it was found that using text and audio modalities trained together was better than using text or audio modalities alone. Based on the tabular data, it was found that the effect of training using text and video modalities was weaker relative to the effect of training using text and audio modalities. One of the reasons may be that video is not as good as other modalities in expressing emotions.

In addition, based on the mechanism of multi-modal fusion, following the modal arrangement of textual modality as the dominant modality and video and audio modalities as the supplementary modality, we can incorporate more effective information into the modal fusion and maintain a reasonable balance of modalities.

In summary, by conducting ablation experiments, we conclude that good performance using multi-modality in the emotion-cause pair extraction task, and some interesting phenomena are found. These conclusions can provide more information to the multimodal learning frameworks.

5 Conclusions

In this paper, we propose a multi-modal attention-based multimodal emotion-cause pair extraction algorithm, which processes the insufficient modal alignment and inadequate feature fusion which are encountered in the emotion-cause pair extraction task in multimodal corpus. With the application of pairwise modal attention fusion technique, the semantic information between modalities are effectively interacted to solve the problem of insufficient expression of emotions relying on single modalities, which in turn improves the accuracy of emotion-cause pair extraction in multimodal contexts. Specifically, the method uses the inter-modal attention mechanism to fuse this information to form a global representation, which complements the disadvantage of the insufficient expression of emotion-cause fusion among different modalities. Therefore, the proposed algorithm adopts adequate inter-modal alignment and fusion to successfully retrieve emotion-cause pairs from video corpora.

References

1. Chen, F., Shi, Z., Yang, Z., Huang, Y.: Recurrent synchronization network for emotion-cause pair extraction. Knowl.-Based Syst. **238**, 107965 (2022)
2. Chen, Y., Hou, W., Cheng, X., Li, S.: Joint learning for emotion classification and emotion cause detection. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 646–651 (2018)
3. Chen, Y., Hou, W., Li, S., Wu, C., Zhang, X.: End-to-end emotion-cause pair extraction with graph convolutional network. In: Proceedings of the 28th International Conference on Computational Linguistics, pp. 198–207 (2020)
4. Chudasama, V., Kar, P., Gudmalwar, A., Shah, N., Wasnik, P., Onoe, N.: M2fnet: multi-modal fusion network for emotion recognition in conversation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4652–4661 (2022)
5. Ding, Z., Xia, R., Yu, J.: ECPE-2D: emotion-cause pair extraction based on joint two-dimensional representation, interaction and prediction. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 3161–3170. Association for Computational Linguistics (2020). https://doi.org/10.18653/v1/2020.acl-main.288
6. Eyben, F., Wöllmer, M., Schuller, B.: Opensmile: the munich versatile and fast open-source audio feature extractor. In: Proceedings of the 18th ACM International Conference on Multimedia, pp. 1459–1462 (2010)
7. Fan, C., Yuan, C., Du, J., Gui, L., Yang, M., Xu, R.: Transition-based directed graph construction for emotion-cause pair extraction. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 3707–3717 (2020)
8. Gao, K., Xu, H., Wang, J.: Emotion cause detection for Chinese micro-blogs based on ECOCC model. In: Cao, T., Lim, E.-P., Zhou, Z.-H., Ho, T.-B., Cheung, D., Motoda, H. (eds.) PAKDD 2015. LNCS (LNAI), vol. 9078, pp. 3–14. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18032-8_1
9. Gui, L., Xu, R., Wu, D., Lu, Q., Zhou, Y.: Event-driven emotion cause extraction with corpus construction. In: Social Media Content Analysis: Natural Language Processing and Beyond, pp. 145–160. World Scientific (2018)
10. Hu, D., Hou, X., Wei, L., Jiang, L., Mo, Y.: Mm-dfn: multimodal dynamic fusion network for emotion recognition in conversations. In: ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 7037–7041. IEEE (2022)
11. Hu, G., Lin, T.E., Zhao, Y., Lu, G., Wu, Y., Li, Y.: UniMSE: towards unified multimodal sentiment analysis and emotion recognition. In: Goldberg, Y., Kozareva, Z., Zhang, Y. (eds.) Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing. pp. 7837–7851. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (Dec 2022)
12. Ji, S., Xu, W., Yang, M., Yu, K.: 3d convolutional neural networks for human action recognition. IEEE Trans. Pattern Anal. Mach. Intell. **35**(1), 221–231 (2012)

13. Lee, S.Y.M., Chen, Y., Huang, C.R.: A text-driven rule-based system for emotion cause detection. In: Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text, pp. 45–53 (2010)
14. Li, J., Wang, X., Lv, G., Zeng, Z.: Ga2mif: graph and attention based two-stage multi-source information fusion for conversational emotion detection. IEEE Trans. Affective Comput. (2023)
15. Li, S., Yan, H., Qiu, X.: Contrast and generation make bart a good dialogue emotion recognizer. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, pp. 11002–11010 (2022)
16. Li, X., Song, K., Feng, S., Wang, D., Zhang, Y.: A co-attention neural network model for emotion cause analysis with emotional context awareness. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 4752–4757. Association for Computational Linguistics, Brussels (2018)
17. Li, Z., Tang, F., Zhao, M., Zhu, Y.: EmoCaps: emotion capsule based model for conversational emotion recognition. In: Muresan, S., Nakov, P., Villavicencio, A. (eds.) Findings of the Association for Computational Linguistics: ACL 2022, pp. 1610–1618. Association for Computational Linguistics, Dublin (2022)
18. Poria, S., Hazarika, D., Majumder, N., Naik, G., Cambria, E., Mihalcea, R.: MELD: a multimodal multi-party dataset for emotion recognition in conversations. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, pp. 527–536 (2019)
19. Wang, F., Ding, Z., Xia, R., Li, Z., Yu, J.: Multimodal emotion-cause pair extraction in conversations. IEEE Trans. Affect. Comput. **14**(3), 1832–1844 (2023)
20. Wang, L., Li, R., Wu, Y., Jiang, Z.: A multiturn complementary generative framework for conversational emotion recognition. Int. J. Intell. Syst. **37**(9), 5643–5671 (2022)
21. Wei, P., Zhao, J., Mao, W.: Effective inter-clause modeling for end-to-end emotion-cause pair extraction. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 3171–3181 (2020)
22. Wu, Z., Dai, X., Xia, R.: Pairwise tagging framework for end-to-end emotion-cause pair extraction. Front. Comput. Sci. **17**(2) (2022)
23. Xia, R., Ding, Z.: Emotion-cause pair extraction: a new task to emotion analysis in texts. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 1003–1012. Association for Computational Linguistics, Florence (2019)
24. Zou, S., Huang, X., Shen, X., Liu, H.: Improving multimodal fusion with main modal transformer for emotion recognition in conversation. Knowl.-Based Syst. **258**, 109978 (2022)

Author Index

A
A. Adjeroh, Donald 418
Akhmedova, Shakhnaz 334
An, Senjian 367

B
Bano, Arju 144
Bengtsson Bernander, Karl 16
Bougourzi, Fares 243
Bylander, Karl 16

C
Cai, Zhihan 350
Cao, Chunjie 259
Cao, Dapeng 350
Chandgothia, Shreyas 97
Chen, Menglong 130
Cheng, Maggie 48
Cheng, Yinan 259
Chu, Lingyang 114
Cosmo, Luca 435
Cristóforis, Pablo De 64
Cui, Jianbin 114

D
Das, Monidipa 144
Dong, Xiaohu 383
Doretto, Gianfranco 418
Dornaika, Fadi 243
Drew, Steve 195
Duan, Miao 290

E
Ezard, Bradley 367

F
Fleuret, François 1
Fu, Baochen 290
Fu, Ruigang 383

G
Gao, Yinghui 383
Goel, Tripti 81

H
Habib, Al Zadid Sultan Bin 418
Hang, Lin 275
Hartley, Mary-Anne 418
Hou, Haojun 227
Hu, Qingyong 383
Huang, Yihua 227

J
Jing, Liwei 195

K
Karwasra, Prashant 32
Körber, Nils 334

L
Li, Biao 383
Li, Ling 367
Li, Qilin 367
Li, Shuohao 163
Li, Zhen 290
Lian, Zhichao 163
Liu, Jie 367
Liu, Xinyue 451
Lv, Xingfeng 275

M
Ma, Kaisheng 350
Minello, Giorgia 435
Minh Giang, Nguyen 319
Minh Tri, Le 319
Mohanty, Aparna 400

© The Editor(s) (if applicable) and The Author(s), under exclusive license
to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15304, pp. 467–468, 2025.
https://doi.org/10.1007/978-3-031-78128-5

Murugan, R. 81

N
Niu, Xiaoguang 195
Nyström, Ingela 16

P
Pilli, Raveendra 81

Q
Qiao, Tonghua 259
Qiao, Xu 290

R
Ranjan Sahay, Rajiv 400
Raverta Capua, Francisco 64
Ren, Qianqian 275
Rossi, Luca 435
Roy, Kankana 400

S
Schandin, Juan 64
Sekhar, Ardhendu 97
Sethi, Amit 97
Sharma, Neha 32
Sharma, Paritosh 32
Sinha, Atul Kumar 1
Sun, Jingzhang 259
Sun, Yukuan 130

T
Tahir, Md Arbaz 32
Taleb-Ahmed, Abdelmalik 243
Tanveer, M. 81
Tao, Fangjian 259
Thi Yen, Nguyen 319
Torsello, Andrea 435
Truong Hoang, Vinh 243

V
Viet, Nguyen Quoc 319

W
Wang, Fengyi 227
Wang, Jianming 130
Wang, Kesheng 418
Wang, Panpan 163
Wang, Shujuan 163
Wu, Kailu 350
Wu, Shaowu 195
Wu, Xiao-Jun 211

X
Xiao, Zhitao 130
Xu, Bo 451
Xu, Chenhao 180
Xu, Kaiye 180
Xu, Tianyang 211
Xu, Yuanzhuo 195

Y
Yang, Jie 195
Yi, Kai 302
Yu, Huiyuan 48
Yu, Muzhou 350
Yuan, Chunfeng 227

Z
Zhang, Bowei 302
Zhang, Lingfeng 435
Zhang, Qi 259
Zhang, Xianchao 451
Zhang, Yi 302
Zhang, Yudian 180
Zhang, Yusu 211
Zhao, Qianli 451
Zhong, Ping 383
Zhu, Guanghui 227
Zhu, Haijiang 180
Zong, Linlin 451
Zou, Binghui 259
Zuo, Xiuli 290