

LNCS 14864

De-Shuang Huang
Zhanjun Si
Yijie Pan (Eds.)

Advanced Intelligent Computing Technology and Applications

20th International Conference, ICIC 2024
Tianjin, China, August 5–8, 2024
Proceedings, Part III

3
Part III



 Springer

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.

De-Shuang Huang · Zhanjun Si · Yijie Pan
Editors

Advanced Intelligent Computing Technology and Applications

20th International Conference, ICIC 2024
Tianjin, China, August 5–8, 2024
Proceedings, Part III



Springer

Editors

De-Shuang Huang
Eastern Institute of Technology
Ningbo, China

Zhanjun Si
Tianjin University of Science and Technology
Tianjin, China

Yijie Pan
Eastern Institute of Technology
Ningbo, China

ISSN 0302-9743

ISSN 1611-3349 (electronic)

Lecture Notes in Computer Science

ISBN 978-981-97-5587-5

ISBN 978-981-97-5588-2 (eBook)

<https://doi.org/10.1007/978-981-97-5588-2>

© The Editor(s) (if applicable) and The Author(s), under exclusive license
to Springer Nature Singapore Pte Ltd. 2024

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 Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

If disposing of this product, please recycle the paper.

Preface

The International Conference on Intelligent Computing (ICIC) was started to provide an annual forum dedicated to emerging and challenging topics in artificial intelligence, machine learning, pattern recognition, bioinformatics, and computational biology. It aims to bring together researchers and practitioners from both academia and industry to share ideas, problems, and solutions related to the multifaceted aspects of intelligent computing.

ICIC 2024, held in Tianjin, China, August 5–8, 2024, constituted the 20th International Conference on Intelligent Computing. It built upon the success of ICIC 2023 (Zhengzhou, China), ICIC 2022 (Xi'an, China), ICIC 2021 (Shenzhen, China), ICIC 2020 (Bari, Italy), ICIC 2019 (Nanchang, China), ICIC 2018 (Wuhan, China), ICIC 2017 (Liverpool, UK), ICIC 2016 (Lanzhou, China), ICIC 2015 (Fuzhou, China), ICIC 2014 (Taiyuan, China), ICIC 2013 (Nanning, China), ICIC 2012 (Huangshan, China), ICIC 2011 (Zhengzhou, China), ICIC 2010 (Changsha, China), ICIC 2009 (Ulsan, South Korea), ICIC 2008 (Shanghai, China), ICIC 2007 (Qingdao, China), ICIC 2006 (Kunming, China), and ICIC 2005 (Hefei, China).

This year, the conference concentrated mainly on the theories and methodologies as well as the emerging applications of intelligent computing. Its aim was to unify the picture of contemporary intelligent computing techniques as an integral concept that highlights the trends in advanced computational intelligence and bridges theoretical research with applications. Therefore, the theme for this conference was “Advanced Intelligent Computing Technology and Applications”. Papers that focused on this theme were solicited, addressing theories, methodologies, and applications in science and technology.

ICIC 2024 received 2189 submissions from 15 countries and regions. All papers went through a rigorous single-blind peer-review procedure and each paper received at least three review reports. Based on the review reports, the Program Committee finally selected 863 high-quality papers for presentation at ICIC 2024, included in twenty-one volumes of proceedings published by Springer: thirteen volumes of Lecture Notes in Computer Science (LNCS), six volumes of Lecture Notes in Artificial Intelligence (LNAI), and two volumes of Lecture Notes in Bioinformatics (LNBI).

In addition, this year we selected 134 Poster papers from the remaining papers, which will be made accessible on the open access website <http://poster-openaccess.com/>.

This volume of LNCS_14864 includes 42 papers.

The organizers of ICIC 2024, including Eastern Institute of Technology, Ningbo, China; Tianjin University of Science and Technology, China; China University of Mining & Technology (Beijing), China; China University of Mining and Technology (Xuzhou), China; and North China University of Science and Technology, China, made an enormous effort to ensure the success of the conference. We hereby would like to thank the members of the Program Committee and the referees for their collective effort in reviewing and soliciting the papers. In particular, we would like to thank all the authors for contributing their papers. Without the high-quality submissions from the authors, the

success of the conference would not have been possible. Finally, we are especially grateful to the International Neural Network Society and the National Science Foundation of China for their sponsorship.

De-Shuang Huang
Fuping Lu

Organization

General Co-chairs

De-Shuang Huang	Eastern Institute of Technology, China
Fuping Lu	Tianjin University of Science and Technology, China

Program Committee Co-chairs

Prashan Premaratne	University of Wollongong, Australia
Xiankun Zhang	Tianjin University of Science and Technology, China
Chuanlei Zhang	Tianjin University of Science and Technology, China
Wei Chen	China University of Mining and Technology, China
Jair Cervantes Canales	Autonomous University of Mexico State, Mexico
Yijie Pan	Eastern Institute of Technology, China
Qinhu Zhang	Eastern Institute of Technology, China
Jiayang Guo	Xiamen University, China

Organizing Committee Co-chairs

Zhanjun Si	Tianjin University of Science and Technology, China
Xiaoyue Liu	North China University of Science and Technology, China
Fan Zhang	China University of Mining and Technology (Beijing), China

Organizing Committee Members

Yarui Chen	Tianjin University of Science and Technology, China
Jing Su	Tianjin University of Science and Technology, China

Shuo Yang	Tianjin University of Science and Technology, China
Jing Han	Tianjin University of Science and Technology, China
Yiying Zhang	Tianjin University of Science and Technology, China
Jucheng Yang	Tianjin University of Science and Technology, China
Qian Long	Tianjin University of Science and Technology, China
Yongjun Ma	Tianjin University of Science and Technology, China
Lin Sun	Tianjin University of Science and Technology, China
Guoliang Gong	Tianjin University of Science and Technology, China

Award Committee Chair

Kang-Hyun Jo University of Ulsan, South Korea

Tutorial Co-chairs

Abir Hussain Liverpool John Moores University, UK
Michal Choras Bydgoszcz University of Science and Technology,
Poland

Publication Co-chairs

Jair Cervantes Canales Autonomous University of Mexico State, Mexico
Chenxi Huang Xiamen University, China

Special Session Co-chairs

Valeriya Gribova Far Eastern Branch of Russian Academy of
Sciences, Russia
M. Michael Gromiha Indian Institute of Technology Madras, India

Special Issue Co-chairs

Yu-Dong Zhang	University of Leicester, UK
Yoshinori Kuno	Saitama University, Japan
Phalguni Gupta	Indian Institute of Technology Kanpur, India

International Liaison Chair

Prashan Premaratne	University of Wollongong, Australia
--------------------	-------------------------------------

Workshop Co-chairs

Kyungsook Han	Inha University, South Korea
Laurent Heutte	Université de Rouen Normandie, France

Publicity Co-chairs

Chun-Hou Zheng	Anhui University, China
Dhiya Al-Jumeily	Liverpool John Moores University, UK
Han Huang	Nanjing University of Information Science and Technology, China

Program Committee Members

Antonio Brunetti	Polytechnic University of Bari, Italy
Bin Liu	Beijing Institute of Technology, China
Bin Qian	Kunming University of Science and Technology, China
Bin Yang	Zaozhuang University, China
Bing Wang	Anhui University of Technology, China
Bingqiang Liu	Shandong University, China
Binhua Tang	Hohai University, China
Bo Li	Wuhan University of Science and Technology, China
Caihong Mu	Xidian University, China
Changqing Shen	Soochow University, China
Chao Song	University of South China, China
Cheng Tang	Kyushu University, Japan

Chin-Chih Chang	Chung Hua University, Taiwan, RoC
Chuanlei Zhang	Tianjin University of Science and Technology, China
Chunhou Zheng	Anhui University, China
Chunmei Liu	Howard University, USA
Chunquan Li	University of South China, China
Cong Shen	Tianjin University of Technology, China
Daowen Qiu	Sun Yat-sen University, China
Delong Yang	First People's Hospital of Foshan, China
Dian Ding	Shanghai Jiao Tong University, China
Dong Wang	University of Jinan, China
Duo Chen	Nanjing University of Chinese Medicine, China
Eros Gian Alessandro Pasero	Politecnico di Torino, Italy
Fa Zhang	Beijing Institute of Technology, China
Fei Guo	Central South University, China
Fei Luo	Wuhan University, China
Fei Shen	Nanjing University of Science and Technology, China
Feng Liu	East China Normal University, China
Feng Zou	Huaibei Normal University, China
Fengfeng Zhou	Jilin University, China
Fudong Nian	Hefei University, China
Fuxue Li	Yingkou Institute of Technology, China
Gang Li	Qilu University of Technology, China
Gaoxiang Ouyang	Beijing Normal University, China
Guanghui Gong	Eastern Institute of Technology, Ningbo, China
Guohui Ding	Shenyang Aerospace University, China
Guoliang Li	Huazhong Agricultural University, China
Han Zhang	Nankai University, China
Hao Huang	Hubei University, China
Hao Lin	University of Electronic Science and Technology of China, China
Haodi Feng	Shandong University, China
Haodong Zhu	Zhengzhou University of Light Industry, China
Heng Li	Southern University of Science and Technology, China
Hoang-Anh Ngo	The University of Waikato, New Zealand
Hongjie Wu	Suzhou University of Science and Technology, China
Hongmin Cai	South China University of Technology, China
Hulin Kuang	Central South University, China
Jiahui Pan	South China Normal University, China

Jian Huang	University of Electronic Science and Technology of China, China
Jian Shen	Beijing Institute of Technology, China
Jiang Xie	Shanghai University, China
Jianrong Li	Tianjin University of Science and Technology, China
Jiawei Luo	Hunan University, China
Jiayang Guo	Xiamen University, China
Jing Chen	Suzhou University of Science and Technology, China
Jing Hu	Wuhan University of Science and Technology, China
Jintian Lu	Jishou University, China
Jin-Xing Liu	University of Health and Rehabilitation Sciences, China
Jipeng Wu	First People's Hospital of Foshan, China
Joaquin Torres-Sospedra	Universidade do Minho, Portugal
Juan Liu	Wuhan University, China
Junfeng Xia	Anhui University, China
Jungang Lou	Huzhou University, China
Junqing Li	Yunnan Normal University, China
Junyi Li	Harbin Institute of Technology (Shenzhen), China
Ka-Chun Wong	City University of Hong Kong, China
Kangning Zhang	Academy of Mathematics and Systems Science, CAS, China
Ke Niu	Beijing Information Science and Technology University, China
Laurent Heutte	Université de Rouen Normandie, France
Le Zhang	Sichuan University, China
Lei Wang	Guangxi Academy of Sciences, China
Lejun Gong	Nanjing University of Posts and Telecommunications, China
Liang Gao	Huazhong University of Science & Technology, China
Lida Zhu	Huazhong Agriculture University, China
Lin Wang	University of Jinan, China
Lin Yuan	Qilu University of Technology, China
Liqliang Liu	Xi'an Technological University, China
Li-Wei Ko	National Yang Ming Chiao Tung University, Taiwan, RoC
Long Shao	Beijing Institute of Technology, China
Long Xu	Ningbo University, China
Meiyuan Xu	Minnan Normal University, China

Meng Liu	National University of Defense Technology, China
Michael Gromiha	Indian Institute of Technology Madras, India
Michal Choras	Bydgoszcz University of Science and Technology, Poland
Mingyong Li	Chongqing Normal University, China
Mohd Helmy Abd Wahab	Universiti Tun Hussein Onn Malaysia, Malaysia
Nicola Altini	Polytechnic University of Bari, Italy
Nier Wu	Inner Mongolia University of Technology, China
Peipei Gu	Zhengzhou University of Light Industry, China
Peng Chen	Anhui University, China
Pengjiang Qian	Jiangnan University, China
Pengwei Hu	Xinjiang Technical Institute of Physics and Chemistry, CAS, China
Prashan Premaratne	University of Wollongong, Australia
Pu-Feng Du	Tianjin University, China
Qi Sun	Hangzhou Nuowei Information Technology Co., Ltd., China
Qi Zhao	University of Science and Technology Liaoning, China
Qifang Luo	Guangxi University for Nationalities, China
Qinhu Zhang	Eastern Institute of Technology, Ningbo, China
Qiuzhen Lin	Shenzhen University, China
Quan Zou	University of Electronic Science and Technology of China, China
Rong Wang	Sichuan Normal University, China
Rong-Qiang Zeng	Chengdu University of Information Technology, China
Rui Wang	National University of Defense Technology, China
Saiful Islam	Aligarh Muslim University, India
Shanfeng Zhu	Fudan University, China
Shitong Wang	Jiangnan University, China
Shixiong Zhang	Xidian University, China
Sungshin Kim	Pusan National University, South Korea
Taisong Jin	Xiamen University, China
Tian Wu	Nanchang University, China
Tieshan Li	University of Electronic Science and Technology of China, China
Valeria Gribova	Far Eastern Branch of Russian Academy of Sciences, Russia
Wangren Qiu	Jingdezhen Ceramic University, China
Waqas Haider Bangyal	Kohsar University Murree, Pakistan

Wei Chen	China University of Mining and Technology (Xuzhou), China
Wei Chen	Chengdu University of Traditional Chinese Medicine, China
Wei Jiang	Fujian Medical University, China
Wei Wang	Henan Normal University, China
Wei Xu	East China Normal University, China
Weichao Wu	Beijing Institute of Technology, China
Weiwei Kong	Xi'an University of Posts and Telecommunications, China
Weixiang Liu	Shenzhen University, China
Wen Jiang	Ctrip Computer Technology (Shanghai) Co., Ltd., China
Wen-Sheng Chen	Shenzhen University, China
Wenzheng Bao	Xuzhou University of Technology, China
Xiangtao Li	Jilin University, China
Xiaodi Li	Shandong Normal University, China
Xiaofeng Wang	Hefei University, China
Xiaoke Ma	Xidian University, China
Xiaolei Zhu	Anhui Agricultural University, China
Xiaoli Lin	Wuhan University of Science and Technology, China
Xiaoqing Li	Capital University of Economics and Business, China
Xin Zhang	Jiangnan University, China
Xingjian Xu	Inner Mongolia Normal University, China
Xingquan Cai	North China University of Technology, China
Xingtao Wang	Harbin Institute of Technology, China
Xinguo Lu	Hunan University, China
Xingyu Feng	City University of Hong Kong, China
Xinlu Li	Hefei University, China
Xinzheng Xu	China University of Mining and Technology (Xuzhou), China
Xiufen Zou	Wuhan University, China
Xiujuan Lei	Shaanxi Normal University, China
Xiwei Liu	Tongji University, China
Xiyuan Chen	Southeast University, China
Xizhao Luo	Soochow University, China
Xulong Zhang	Ping An Technology (Shenzhen) Co., Ltd., China
Yang Yang	Hubei University, China
Yansen Su	Anhui University, China
Yijie Pan	Eastern Institute of Technology, Ningbo, China
Yiming Tang	Hefei University of Technology, China

Yizhang Jiang	Jiangnan University, China
Yong Wang	Academy of Mathematics and Systems Science, CAS, China
Yong Wu	Anhui Normal University, China
Yonggang Lu	Lanzhou University, China
Yu Lu	Shenzhen Technology University, China
Yu Xue	Huazhong University of Science and Technology, China
Yunxia Liu	Zhengzhou Normal University, China
Yupei Zhang	Northwestern Polytechnical University, China
Yushan Qiu	Shenzhen University, China
Yuyan Zheng	Shandong Normal University, China
Zhan-Li Sun	Anhui University, China
Zhen Shen	Nanyang Institute of Technology, China
Zhendong Liu	Shandong Jianzhu University, China
Zhenran Jiang	East China Normal University, China
Zhenyi Shen	Zhejiang University, China
Zhi-Hong Guan	Huazhong University of Science and Technology, China
Zhi-Ping Liu	Shandong University, China
Zhong-Qiu Zhao	Heifei Institute of Technology, China
Zhuangzhuang Chen	Hong Kong University of Science and Technology, China
Zhuo Lei	City Cloud Technology China Co., Ltd., China
Zixiao Kong	University of International Relations, China

Contents – Part III

Neural Networks

Trust Evaluation with Deep Learning in Online Social Networks: A State-of-the-Art Review	3
<i>Zhiqi Li, Weidong Fang, Chunsheng Zhu, Wentao Chen, Tianpeng Hao, and Wuxiong Zhang</i>	
Deep Neural Network-Based Intrusion Detection in Internet of Things: A State-of-the-Art Review	13
<i>Zhiqi Li, Weidong Fang, Chunsheng Zhu, Wentao Chen, Zhiwei Gao, Xinhang Jiang, and Wuxiong Zhang</i>	
CNN-SENet: A Convolutional Neural Network Model for Audio Snoring Detection Based on Channel Attention Mechanism	24
<i>Zijun Mao, Suqing Duan, Xiankun Zhang, Chuanlei Zhang, Haifeng Fan, Bolun Zhu, and Chengliang Huang</i>	
Selecting Effective Triplet Contrastive Loss for Domain Alignment	36
<i>Changshi Li and Zhijian Yang</i>	
RCSnet——Flower Classification Network Design Based on Transfer Learning and Channel Attention Mechanism	48
<i>Zijun Mao, Tianyu Zhong, Mojieming Wei, Runjie Hu, and Jianzheng Liu</i>	
MDGCL: Message Dropout Graph Contrastive Learning for Recommendation	60
<i>Qijia Xu, Wei Li, and Jingxin Chen</i>	
Improved CNN Model Using Innovative Adaptive-DropMessage for Gomoku Game	72
<i>Kangjie Cao, Xiali Li, Jinyao Wu, Hu Yuan, Wentao Li, Jiayun Li, He Huang, Jueqiao Huang, and Weijun Cheng</i>	
A Survey: Feature Fusion Method for Object Detection Field	84
<i>Zhe Lian, Yanjun Yin, Jingfang Lu, Qiaozhi Xu, Min Zhi, Wei Hu, and Wentao Duan</i>	
Dual-Branch Collaborative Learning for Visual Question Answering	96
<i>Weidong Tian, Junxiang Zhao, Wenzheng Xu, and Zhongqiu Zhao</i>	

SCAI: A Spectral Data Classification Framework with Adaptive Inference for Rapid and Portable Identification of Chinese Liquors	108
<i>Yundong Sun, Yansong Wang, Xuguang Xu, Dongjie Zhu, and Zhaoshuo Tian</i>	
EfficientPose: A Lightweight and Efficient Model with Transformer for Human Pose Estimation	120
<i>Wei Liang, Zhang Cheng, Junjia Han, and Yanxia Wang</i>	
Enhanced Chinese Named Entity Recognition with Transformer-Based Multi-feature Fusion	132
<i>Xiaoli Zhang, Quan Zhang, Kun Liang, and Haoyu Wang</i>	
YOLO-Fire: A Fire Detection Algorithm Based on YOLO	142
<i>Bo Xu, Fengyou Hua, Qun Yang, and Tao Wang</i>	
From Vision to Sound: The Application of ViT-LSTM in Music Sequence	155
<i>Menghao Fang, Shuo Zhang, Xia Li, Liangbin Yang, and Zixiao Kong</i>	
Graph Convolution Recommendation Algorithm Integrating Multi-relationship Preferences	167
<i>Jing Su, Tianfeng Zhao, Jianghong Wu, and Peixuan Shi</i>	
Temporal Sequential Wave Neural Network for Solving the Optimal Cognitive Subgraph Query Problem	178
<i>Jiaqian Bi, Zhilei Xu, and Qing Yu</i>	
Joint Prior Relation Enhancement and Non-autoregressive Decoding for Document-Level Event Extraction	190
<i>Xue Kang, Yan-Ni Han, Wen Zhang, Han Jiang, Yi Xiang, and Shu-Guang Liu</i>	
Intrusion Detection System Based on ViTCycleGAN and Rules	203
<i>Menghao Fang, Xia Li, Yuanyuan Wang, Qiuxuan Wang, Xinlei Sun, and Shuo Zhang</i>	
DFT-3DLaneNet: Dual-Frequency Domain Enhanced Transformer for 3D Lane Detection	215
<i>Kaijiang Li, Yuling Liu, Peisen Wang, XiangQian Liu, Xichen Liu, ChunYi Guo, and Bing Zhou</i>	
Correlation Matters: A Stock Price Predication Model Based on the Graph Convolutional Network	228
<i>Chengkun Xin, Qian Han, and Gang Pan</i>	

Coformer: Collaborative Transformer for Medical Image Segmentation	240
<i>Yafei Gao, Shichao Zhang, Dandan Zhang, Yucheng Shi, Guohua Zhao, and Lei Shi</i>	
FRMM: Feature Reprojection for Exemplar-Free Class-Incremental Learning	251
<i>Hao Wang and Jing Chen</i>	
PCQ: Emotion Recognition in Speech via Progressive Channel Querying	264
<i>Xincheng Wang, Liejun Wang, Yinfeng Yu, and Xinxin Jiao</i>	
Federated Learning Cellular Traffic Prediction Based on Multi-time Scale Information	276
<i>Xingzhen Gao, Yuhong Zhao, Jingyu Wang, and Chuanting Zhang</i>	
BGMA-Net: A Boundary-Guided and Multi-attention Network for Skin Lesion Segmentation	290
<i>Cong Wu, Yao Li, Yuan Zhou, Haitao Gan, and Yi Han</i>	
Graph Pre-training for Reconnaissance Perception in Automated Penetration Testing	302
<i>Yunfei Wang, Shixuan Liu, Chao Zhang, Wenhao Wang, Jiandong Jin, Cheng Zhu, and Changling Zhou</i>	
Smart Contract Vulnerability Detection Based on Multi Graph Convolutional Neural Networks with Self-attention	319
<i>Jiale Li, Xiao Yu, Jie Yu, Haoxin Sun, and Mengdi Sun</i>	
PBIM: Paired Backdoor Injection Method for Change Detection	331
<i>Rui Huang, Mengjia Hao, Zongyu Guo, and Yifan Zhang</i>	
Smart Contract Vulnerability Detection Based on Multimodal Feature Fusion	344
<i>Jie Yu, Xiao Yu, Jiale Li, Haoxin Sun, and Mengdi Sun</i>	
Graph Neural Network-Based Structured Scene Graph Generation for Efficient Wildfire Detection	356
<i>Yanning Ye, Shimin Luo, MengMeng Jing, Yongqi Ding, Kunbin He, and Lin Zuo</i>	
Towards Highly Effective Moving Tiny Ball Tracking via Vision Transformer	368
<i>Jizhe Yu, Yu Liu, Hongkui Wei, Kaiping Xu, Yifei Cao, and Jiangquan Li</i>	

SDE-Net: Skeleton Action Recognition Based on Spatio-Temporal Dependence Enhanced Networks	380
<i>Qing Sun, Jiuzhen Liang, Zhou Xinwen, and Hao Liu</i>	
Lightweight Coal Flow Foreign Object Detection Algorithm	393
<i>Ru Nie, Xiaobing Shen, Zhengwei Li, Yanxia Jiang, Hongmei Liao, and Zhuhong You</i>	
SDF-Net: Enhanced Novel View Synthesis from Ultra-sparse Viewpoints via Multi-level Feature Fusion	405
<i>Qijun He, Jingfu Yan, Jiahui Li, and Yifeng Li</i>	
TUPocket: 3D Convolutional Neural Network Combined with Transformer for Ligand Binding Site Detection	419
<i>Yangtao Meng and Tianhao Yan</i>	
A Graph Neural Network-Based Multi-agent Joint Motion Prediction Method for Motion Trajectory Prediction	431
<i>Hongxu Gao, Zhao Huang, Jia Zhou, Song Cheng, Quan Wang, and Yu Li</i>	
xNN-SF: An Explainable Neural Network Inspired by Stochastic Frontier Model	444
<i>Shuangxue Zhao</i>	
Eliciting Offensive Responses from Large Language Models: A Genetic Algorithm Approach	456
<i>Zheng Chen, Jiachen Zhu, and Anlong Chen</i>	
PaSeMix: A Multi-modal Partitional Semantic Data Augmentation Method for Text-Based Person Search	468
<i>Xinpan Yuan, Jiabao Li, Wenguang Gan, Wei Xia, and Yanbin Weng</i>	
Multi-behavior Recommender Model Based on LightGCN	480
<i>Han Xueying and Yang Yan</i>	
SSDC-Net: An Effective Classification Method of Steel Surface Defects Based on Salient Local Features	490
<i>Qifei Hao, Qingsong Gan, Zhe Liu, Jun Chen, Qi Shen, Chengxuan Qian, and Yi Liu</i>	
Neural Network Verification Accelerated by a Novel Abstract Framework	504
<i>Jiaqi Liu and Meng Wang</i>	
Author Index	517

Neural Networks



Trust Evaluation with Deep Learning in Online Social Networks: A State-of-the-Art Review

Zhiqi Li^{1,2} , Weidong Fang^{1,2,3} , Chunsheng Zhu⁴ , Wentao Chen⁵ , Tianpeng Hao⁶ , and Wuxiong Zhang^{1,2}

¹ Science and Technology on Microsystem Laboratory, Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, Shanghai 201899, China
`{lizhiqi,weidong.fang,wuxiong.zhang}@mail.sim.ac.cn`

² School of Electronic, Electrical, and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100049, China

³ Shanghai Research and Development Center for Micro-Nano Electronics, Shanghai 201210, China

⁴ College of Big Data and Internet, Shenzhen Technology University, Shenzhen 518118, China

⁵ State Grid Xinjiang Company Limited Electric Power Research Institute, Urumqi 830011, China

⁶ Xinjiang Baiyang River Basin Administration, Urumqi 830000, China
`yimengbingfeng@sina.com`

Abstract. In the realm of online social networks (OSNs), it has become increasingly crucial to analyze user behavior, establish trustworthy relationships to mitigate social risks, enhance security, and safeguard privacy. Trust evaluation is widely acknowledged as an effective approach for detecting internal attacks and identifying compromised nodes, and deep learning technology can significantly enhance its performance. However, there remains a notable gap for a review paper focused on trust evaluation utilizing deep learning techniques within OSNs. Therefore, conducting a state-of-the-art review on this subject has become imperative. We analyze and compare some recent related research, summarizing prevalent challenges and open issues while proposing optimization strategies to address them. For instance, graph-based neural networks methods often grapple with exponentially increasing computational complexity as network size expands, and imbalanced datasets typically lead to reduced model accuracy and generalization. Lastly, it presents several promising avenues for future research in the field.

Keywords: Online Social Networks · Trust · Deep Learning · Cyber Security

1 Introduction

Social networks are intricate structures that arise from the interactions, connections, and collaborations among individuals using computer-based networked platforms, as highlighted in previous works [1]. For example, as of April 2024, Facebook and YouTube

each have over 3 billion and 2.5 billion monthly active users, respectively¹. This illustrates how online social networks have become essential tools in facilitating communication and connection between individuals, both with friends and strangers, in everyday human life. Nevertheless, online social networks remain vulnerable to a plethora of international threats, including Distributed Denial of Service (DDoS) attacks, phishing attempts, and malware threats. Traditional network security measures such as firewalls and intrusion detection systems frequently demonstrate inadequacies in safeguarding against specific internal network attacks. In addressing these security challenges, trust evaluation is widely recognized as a potent approach for detecting insider threats and identifying compromised nodes [2]. Trust is defined as the expectations and beliefs one party holds regarding another during interpersonal interactions, with these expectations subject to variation depending on the specific context [3].

Nevertheless, trust evaluation in OSNs, characterized by their ultra-large-scale nature, encounters a plethora of challenges. Utilizing traditional methods such as Bayesian probability and game theory can prove challenging in accurately assessing the trustworthiness of these new nodes. Fortunately, deep learning (DL) technology has shown promise in predicting future trust changes for nodes, effectively alleviating the cold start problem and enhancing accuracy and real-time performance [4]. Our contributions in this work are summarized as follows:

1. We conduct a thorough analysis and comparison of advanced works related to trust evaluation using deep learning techniques in OSNs.
2. We find that the GNN-based schemes encounter some formidable challenge of exponentially increasing computational complexity as network size expands. We delve into relevant literature focusing on other DL methods.
3. We summarize some prevalent challenges and open issues while proposing optimization strategies to address them.
4. We put forth a range of suggested future research directions to guide the academic and practitioner community in exploring novel solutions to these challenges.

The remaining structure of this paper is as follows. The analysis and comparison of some trust evaluation schemes are provided in Sect. 2. In Sect. 3, we analyze some open problems and challenges and propose some strategies to mitigate the issues. In Sect. 4, some suggested directions for future research are proposed. Finally, conclusion of this paper is drawn in Sect. 5.

2 Analysis and Comparison of Related Work

2.1 Graph-Based Neural Networks

The evaluation of node trustworthiness within a network has proven to be a valuable defense mechanism against internal attacks [5]. While DL technology has received extensive research attention in OSNs, most deep neural networks (DNNs) have been designed for Euclidean structured data, which doesn't fully leverage network topology. To address

¹ <https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users>.

this limitation, some research has proposed graph-based DL methods for non-Euclidean structured data in recent years.

Graph Neural Networks (GNNs) have represented a significant advancement in deep learning, enhancing the efficiency of neural networks by extracting and analyzing potential dependencies through message passing and neighborhood aggregation. In their work, C. Huo et al. introduced an innovative GNN-based trust evaluation method named Trust-GNN for social networks [6]. TrustGNN adeptly integrated the propagation and composability properties of trust graphs into a GNN framework, thereby enhancing trust evaluation processes. Besides, in OSNs, there has been a growing demand for user communication with strangers. However, when the network data was sparse, the accuracy of trust prediction could be greatly affected [7]. To tackle this challenge, S. Ghafari et al. leveraged and extended GraphSAGE [8], a method for computing node representations in an inductive manner, alongside the proposal of a deep context-aware trust prediction model called DCAT [9]. It targeted the analysis of textual information provided by users in comment-based OSNs. Additionally, X. Chen et al. putted forward a trust evaluation framework by categorizing user features into four distinct groups [10]. G. Liu et al. introduced a neural network (NN) called WalkNet within the NeuralWalk framework to simulate the process of single-hop trust propagation [11].

The problem of trust prediction could be approached through various methods, such as matrix factorization and propagated trust. However, there has been limited research considering the complementary nature of these methods. To address this gap, X. Gao et al. introduced a novel trust prediction scheme called iSim, which integrated three factors to evaluate user similarity [12]. iSim was a conventional approach for predicting trust typically rely on static graphs as input, disregarding the time-dependent nature of social interactions. W. Lin et al. proposed a model called Medley, which captured time-varying latent factors and predicts the evolution of social trust over time [13]. They utilized functional time encoding to capture continuous-time features and incorporates evolving topological structures to infer pairwise social trust from past interactions.

As discussed above, we compile and analyze these graph-based trust evaluation schemes in Table 1. They are categorized as direct experience (DE) and indirect experience (IE) based on the type of historical experience. Furthermore, the “Rep.” represents the model’s assessment of the node’s reputation, the “Loc.” signifies the evaluation of trust as local trust, and the “Glo.” denotes the evaluation of trust as global trust, and A, P, R represent accuracy, precision, and recall, respectively.

2.2 Other Deep Learning Method

The scale of OSNs is vast, and graph-based trust evaluation schemes often encounter the challenge of exponentially increasing computational complexity. The graph-based methods can aid in identifying and building trust relationships between users, thereby identifying potential unreliable users or harmful information. Nevertheless, graph-based schemes often face limitations in identifying all trustworthy paths, which may necessitate the imposition of certain restrictions during the path-finding process.

Instead of relying solely on graph-based search algorithms, N. Fatehi et al. employed a distributed learning automata (DLA) to discover all trustworthy relationships [17]. It was proved that the flexibility and scalability of models can be improved [18]. Trust

Table 1. Summary of the Related Work on Trust Evaluation with Graph-Based Deep Learning Methods in OSNs

Scheme	DE	IE	Trust	Rep.	Loc.	Glo.	Dataset	Result	Advantage	Disadvantage
TrustGNN [6]	✓	✓	✓	✗	✓	✓	Advogato, PGP, Ciao, Epinions	F1 = 74.4%, 87.2%, 72.8%, and 81.8%; MAE = 0.081, 0.083, 0.050, and 0.032	Compared to other benchmarks, it exhibits superior accuracy, minimal errors, and low time complexity	It is limited to static networks, and its modeling and computation processes are singular
DCAT [9]	✓	✓	✓	✗	✓	✓	Ciao, Epinions	MAE = 0.36 and 0.4	Strong context-awareness ability	Weak ability to dynamically capture trust relationships
CHEN [10]	✓	✓	✓	✗	✗	✓	Twitter Dataset	A = 96%	High accuracy by analyzing the overlapping area of positive and negative feature distributions	Weak context-awareness ability for different scenarios and time
NeuralWalk [11]	-	-	✓	✗	✓	✓	Advogato, PGP	F1 = 74.6% and 91.6%	It can accurately predict unknown trust relationships in an inductive manner	High time cost and computational consumption
iSim [12]	-	-	✓	✗	✓	✗	Advotago, RobotNet	MAE = 0.1473 and 0.1300	Low time complexity and strong context awareness-ability	Poor robustness and generalization performance
Medley [13]	✗	✓	✓	✗	✓	✗	Bitcoin-Alpha	A = 73.3%	Obtaining time features and using attention mechanism to assign weights	High energy consumption and time cost
GATrust [14]	✓	✓	✓	-	✓	✓	Advogato, PGP	F1 = 0.773 and 0.913, MAE = 0.076 and 0.079	Improving the social trust evaluation performance of users and predict trust	Facing challenges related to the spatial and temporal dependencies

reasoning quality has heavily relied on trust propagation, which was influenced by various factors, such as the length of the path between users. For instance, DLATrust [19],

proposed by M. Ghavipour et al., aimed to identify reliable paths, and the trust network was treated as a static graph. However, the trust weights often change dynamically over time, prompting researchers to introduce a dynamic trust propagation algorithm called DyTrust [20]. To tackle challenges like the cold start problem and data sparsity, TrustDL [21] was proposed to integrate various information sources, including trust ratings, into the recommendation model.

However, the attributes of trust networks with directed edges and nodes with in-links and out-links could not be effectively captured by most existing networks with embedded solutions. A scheme proposed by Q. Wang et al., AtNE-Trust, was capable of capturing high-quality user embeddings and making accurate trust predictions [22]. Nevertheless, most trust prediction methods tend to focus on specific aspects of trust, lacking comprehensive research on the development of user trust. They introduced C-DeepTrust to fill this gap [23]. Furthermore, existing methods primarily aim to improve predictive performance by using more available information, but the influence of user information authenticity and user subjectivity was frequently disregarded in trust studies. S-DeepTrust [24] was proposed to address this gap by employing transfer learning to derive emotion labels from user review data, resulting in an emotion polarity matrix. This matrix was subsequently weighted to produce a revised rating matrix, enhancing the integration of emotional cues into the trust evaluation process.

We summarize the related work mentioned above in Table 2. Some abbreviations are the same as in Table 1 and “Tech.” indicates the technique used by the model. From Table 2, it is evident that DL-based trust evaluation models exhibit a notable propensity for high time and energy consumption. This can be attributed to the inherent nature of DL, which necessitates extensive data processing and optimization tasks. Besides, there are several datasets commonly used for training models. For, example, the Trustlet² dataset from an online trust relationship network, including data from Epinions² and Ciao³, two e-commerce websites. It encompasses social relationships between users, user ratings of products, and associations between products on these datasets.

3 Challenges and Open Problems

Despite the substantial contributions made by researchers in this field, numerous unresolved problems and challenges persist. Here, we outline several key problems and challenges pertaining to trust evaluation in OSNs.

Firstly, the applicability of some trust management models has been restricted because of the diverse range of network structures [10, 16]. There isn't a universally applicable trust model, and it's imperative to devise tailored protocols for each specific trust evaluation approach [25]. Trust evaluation configurations may vary across different network topologies or application contexts. The integration of adaptive mechanisms allows for flexible adjustment of parameters and strategies within the trust model, facilitating accommodation of diverse network structures and environmental conditions.

Secondly, in OSNs, trust evaluation models based on GNN may encounter the problem of exponentially increasing computational complexity, especially as the network

² <http://trustlet.org/datasets/advogato>.

³ <http://www.cse.msu.edu/~tangjili/trust.html>.

Table 2. Summary of the Related Work on Trust Evaluation with Other Deep Learning Methods in OSNs

Scheme	DE	IE	Trust	Rep.	Loc.	Glo.	Tech.	Dataset	Result	Advantage	Disadvantage
DeciTTrustNET [16]	✓	✓	✓	✓	✗	✓	Graph-based	—	—	Evaluating trust in a more complete manner	Lacking context-awareness ability for different scenarios
FATEHI [17]	✓	✓	✓	✗	✗	✓	DLA	Epinions	A = 93.98%	The limitation of finding trusted paths	Long execution time
DLATrust [19]	✗	✓	✓	✗	✓	✓	DLA	Advogato	MAE = 0.1152; P = 96.32%; R = 97.38%	Identifying reliable trust paths with higher accuracy	It could not effectively predict dynamic trust
DyTrust [20]	✗	✓	✓	✗	✓	✓	DLA	Kaitiaki	MAE = 0.1152, P = 95.43%, R = 99.43%	Inferring dynamic trust accurately	High energy consumption and time cost
C-DeepTrust [23]	✓	✓	✓	✗	✓	✗	LSTM	Epinions, Ciao	F1 = 0.917 and 0.924	Achieving a high level of prediction accuracy	High time complexity
S-DeepTrust [24]	✓	✓	✓	✗	✓	✗	LSTM	Epinions, Ciao	F1 = 0.942 and 0.918	High predictive precision and robustness in dealing with the sparsity of data	It is complex, with high computational costs

size grows [6, 11–13]. This is because in a large-scale network, GNN need to handle a large number of nodes and edges, leading to a sharp increase in computational complexity. To mitigate this challenge, the random sampling techniques can be used to select a subset of nodes and edges for training. It can reduce the computational load, but may result in information loss. Besides, it can assist in selecting appropriate GNN structures and hyperparameters to optimize performance with learning automata.

Thirdly, the dependence of trust evaluation on historical data is emphasized. However, in large-scale social networks, interactions between nodes are often sparse [16, 19]. It presents a challenge in accurately assessing the initial trust of newly introduced nodes at the system's outset, a problem commonly known as the cold-start problem [26]. During the initial stage, lower accuracy in trust evaluation tends to result from

the limited availability of historical data. Consequently, accurately evaluating the initial trustworthiness in the early phases of the system proves challenging.

Lastly, the unequal distribution of samples among various classes or labels within a dataset is termed class imbalance [11, 20]. The performance of machine learning models is adversely affected, including reduced accuracy and generalization ability. To mitigate this issue, dataset balancing techniques are commonly employed. With dataset balancing techniques, the number of samples are augmented in the minority class and reduced in the majority class. Additionally, it often proves effective in handling imbalanced data by employing ensemble methods like random forests or gradient boosting trees, as they can amalgamate predictions from multiple models [27].

4 Research Direction

4.1 Trust Evaluation Based on Ensemble Learning with DNN in OSNs

Ensemble Learning (EL) is a powerful AI technique that combines multiple machine learning approaches to obtain the best possible solution. However, it is not without its drawbacks, which include long training hours and high computational overhead. In contrast, models based on deep neural networks show greater adaptability to the environment than traditional models. For these challenges, we propose to train a DNN-based selector to select a suitable set of classifiers. This DNN-based selector efficiently determines which ML method classifiers are appropriate for the specific problem at hand. This pre-screening process reduces training and computation overhead.

A selector model based on deep neural networks is constructed, which can effectively identify suitable machine learning methods based on input problem characteristics. The architecture of this selector may include CNNs, RNNs, Transformer, or other variants. Once the deep neural network selector model is trained, it can be applied to real-world scenarios. When faced with a new problem, its features are input into the selector model. The output of this model, a collection of applicable machine learning methods, helps with the initial screening. This screening process notably diminishes training and computational overhead, as it obviates the need to train and evaluate all potential methods, focusing instead on those recommended by the selector.

4.2 Cross-Origin Trust Evaluation Based on Deep Learning with a Hyperparameter Auto Optimizer in OSNs

Integrating a hyperparameter auto optimizer into cross-domain evaluation systems for OSNs is a promising approach because it could present a promising solution to the persistent challenge of effectively detecting a multitude of violations such as fraud and spam. The difficulty in achieving this effectiveness often arises from the constraints in selecting appropriate training datasets for these evaluation systems, rendering them inadequately equipped to adapt to the dynamic online landscape and the diverse range of violations encountered.

By incorporating a hyperparameter auto optimizer into these evaluation systems, researchers could enable the automatic selection of optimal detection subsystems and

hyperparameter configurations. The system is highly adaptable and can flexibly respond to the changing social network environment. Therefore, by deploying this model, various violations can be better detected, thus enhancing the security of the social network. In addition, the introduction of hyperparameter automatic optimizers brings several significant advantages to OSNs cross-domain evaluation systems. First, the adaptability of the system could be enhanced, because the model can automatically select the best configuration on different social networking platforms without manual adjustment. Second, by integrating multiple trained detection subsystems, violations more comprehensively can be identified, overcoming the limitations of relying on a single IDS.

Ultimately, through the proposed model, not only can the overall user experience be improved, but the credibility of the online platform can also be enhanced and the security of the social network can be improved. By making the social space safer and more conducive to positive interactions, the integration of hyperparametric automated optimizers into cross-domain evaluation systems represents a critical step in promoting trust and confidence in the digital realm.

5 Conclusion

We provide a review of research concerning trust evaluation employing deep learning within online social networks is presented. Trust evaluation has been proven to be a powerful method in mitigating internal attacks, while intrusion detection acts as a robust defense against external threats. Nonetheless, certain challenges arise with graph-based models due to their exponential computational complexity. Consequently, we delve into some other deep learning techniques and discuss the strategies to mitigate these challenges. Firstly, we conduct a comparison and analysis of the latest trust evaluation schemes, focusing on graph neural networks and other deep learning approaches. We emphasize the exponential increase in computational complexity associated with graph neural network methods as network scale expands. Subsequently, some possible solutions for the challenges and problems encountered in trust evaluation are proposed. These challenges include network structural diversity, high computational complexity, cold start problems, and sample imbalance. Then, we suggest corresponding optimization strategies to tackle these challenges effectively. Finally, we propose two future research directions, which include trust evaluation through ensemble learning and cross-source trust evaluation utilizing hyperparameter auto-optimization. In summary, it offers a review of deep learning-based trust evaluation in OSNs, highlighting its significant importance in enhancing understanding and improving trust evaluation within such networks.

Acknowledgments. This study was funded in part by the Key Research and Development Task Special Project of the Xinjiang Uygur Autonomous Region (grant number 2022B01009), in part by the Shanghai Science and Technology Innovation Action Plan (grant number 21DZ1200600), in part by the National Key Research and Development Program (grant number 2020YFB2104202), and in part by the Shanghai Natural Science Foundation (grant number 21ZR1461700).

References

1. Kumar, S., Revathy, S.: Review on social network trust with respect to big data analytics. In: 2020 4th International Conference on Trends in Electronics and Informatics, Tirunelveli, India, pp. 721–727. IEEE (2020)
2. Fang, W., Cui, N., Chen, W., Zhang, W., Chen, Y.: A trust-based security system for data collection in smart city. *IEEE Trans. Ind. Inf.* **17**(6), 4131–4140 (2021)
3. Li, Z., Fang, W., Zhu, C., Gao, Z., Zhang, W.: AI-enabled trust in distributed networks. *IEEE Access* **11**, 88116–88134 (2023)
4. Fang, W., Zhu, C., Yu, F.R., Wang, K., Zhang, W.: Towards energy-efficient and secure data transmission in AI-enabled software defined industrial networks. *IEEE Trans. Ind. Inf.* **18**(6), 4265–4274 (2022)
5. Niu, X., Liu, G., Yang, Q.: Trustworthy website detection based on social hyperlink network analysis. *IEEE Trans. Netw. Sci. Eng.* **7**(1), 54–65 (2018)
6. Huo, C., He, D., Liang, C., Jin, D., Qiu, T., Wu, L.: TrustGNN: graph neural network-based trust evaluation via learnable propagative and composable nature. *IEEE Trans. Neural Netw. Learn. Syst.*, 1–13 (2023)
7. Fang, W., Zhu, C., Guizani, M., Rodrigues, J.J.P.C., Zhang, W.: HC-TUS: human cognition-based trust update scheme for AI-enabled VANET. *IEEE Netw.* <https://doi.org/10.1109/MNET.2023.3320934>
8. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. *Adv. Neural. Inf. Process. Syst.* **30**, 1024–1034 (2017)
9. Ghafari, S.M., Joshi, A., Beheshti, A., Paris, C., Yakhchi, S., Orgun, M.: DCAT: a deep context-aware trust prediction approach for online social networks. In: Proceedings of the 17th International Conference on Advances in Mobile Computing & Multimedia, pp. 20–27. ACM (2019)
10. Chen, X., Yuan, Y., Lu, L., Yang, J.: A multidimensional trust evaluation framework for online social networks based on machine learning. *IEEE Access* **7**, 175499–175513 (2019)
11. Liu, G., Li, C., Yang, Q.: NeuralWalk: trust assessment in online social networks with neural networks. In: Proceedings of the IEEE Conference on Computer Communications, Paris, France, pp. 1999–2007. IEEE (2019)
12. Gao, X., Xu, W., Liao, M., Chen, G.: Trust prediction for online social networks with integrated time-aware similarity. *ACM Trans. Knowl. Discov. Data* **15**, 1–30 (2021)
13. Lin, W., Li, B.: Medley: predicting social trust in time-varying online social networks. In: Proceedings of the IEEE Conference on Computer Communications, Vancouver, BC, Canada, pp. 1–10. IEEE (2021)
14. Jiang, N., Jie, W., Li, J., Liu, X., Jin, D.: GATrust: a multi-aspect graph attention network model for trust assessment in OSNs. *IEEE Trans. Knowl. Data Eng.* **35**(6), 5865–5878 (2023)
15. Jain, L., Katarya, R., Sachdeva, S.: Opinion leaders for information diffusion using graph neural network in online social networks. *ACM Trans. Web* **17**(2), 1–37 (2023)
16. Ureña, R., Chiclana, F., Herrera-Viedma, E.: DeciTrustNET: a graph-based trust and reputation framework for social networks. *Inf. Fus.* **61**, 101–112 (2020)
17. Fatehi, N., Shahhoseini, H.S., Wei, J., Chang, C.T.: An automata algorithm for generating trusted graphs in online social networks. *Appl. Soft Comput.* **118**, Article no. 108475 (2022)
18. Fang, W., Zhang, W., Yang, W., Li, Z., Gao, W., Yang, Y.: Trust management-based and energy-efficient hierarchical routing protocol in wireless sensor networks. *Digit. Commun. Netw.* **7**(5), 470–478 (2021)
19. Ghavipour, M., Meybodi, M.R.: Trust propagation algorithm based on learning automata for inferring local trust in online social networks. *Knowl. Based Syst.* **143**, 307–316 (2018)

20. Ghavipour, M., Meybodi, M.R.: A dynamic algorithm for stochastic trust propagation in online social networks: learning automata approach. *Comput. Commun.* **123**, 11–23 (2018)
21. Khaledian, N., Nazari, A., Khamforoosh, K., Abualigah, L., Javaheri, D.: TrustDL: use of trust-based dictionary learning to facilitate recommendation in social networks. *Expert Syst. Appl.* **228**, Article no. 120487 (2023)
22. Wang, Q., Zhao, W., Yang, J., Wu, J., Zhou, C., Xing, Q.: AtNE-trust: attributed trust network embedding for trust prediction in online social networks. In: Proceedings of the 2020 IEEE International Conference on Data Mining (ICDM), pp. 601–610. IEEE (2020)
23. Wang, Q., et al.: C-DeepTrust: a context-aware deep trust prediction model in online social networks. *IEEE Trans. Neural Netw. Learn. Syst.* **34**(6), 2767–2780 (2021)
24. Wang, Q., et al.: S-DeepTrust: a deep trust prediction method based on sentiment polarity perception. *Inf. Sci.* **633**, 104–121 (2023)
25. Ahvar, E., Fathy, M.: BEAR: a balanced energy-aware routing protocol for wireless sensor networks. *Wirel. Sens. Netw.* **2**(10), 793–800 (2010)
26. Siau, K., Wang, W.: Building trust in artificial intelligence, machine learning, and robotics. *Cut. Bus. Technol. J.* **31**(2), 47–53 (2018)
27. Fang, W., Zhu, C., Zhang, W.: Toward secure and lightweight data transmission for cloud–edge–terminal collaboration in artificial intelligence of things. *IEEE Internet Things J.* **11**(1), 105–113 (2024)



Deep Neural Network-Based Intrusion Detection in Internet of Things: A State-of-the-Art Review

Zhiqi Li^{1,2} , Weidong Fang^{1,2,3}  , Chunsheng Zhu⁴ , Wentao Chen⁵, Zhiwei Gao⁶, Xinhang Jiang⁷, and Wuxiong Zhang^{1,2} 

¹ Science and Technology on Microsystem Laboratory, Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, Shanghai 201899, China

{lizhiqi,weidong.fang,wuxiong.zhang}@mail.sim.ac.cn

² School of Electronic, Electrical, and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100049, China

³ Shanghai Research and Development Center for Micro-Nano Electronics, Shanghai 201210, China

⁴ College of Big Data and Internet, Shenzhen Technology University, Shenzhen 518118, China

⁵ State Grid Xinjiang Company Limited Electric Power Research Institute, Urumqi 830011, China

⁶ Ceprei Certification Body, the Fifth Electronics Research Institute of Ministry of Industry and Information Technology, Guangzhou 511370, China

⁷ School of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan 430081, China

Abstract. Various security threats are faced by the Internet of Things (IoT) as it enriches people's daily lives. Intrusion detection is employed as an effective method to mitigate these threats, encompassing Botnet, DDoS, and Scan attacks. Due to the rapid development of machine learning technology in recent years, deep neural networks (DNNs) emerge as powerful models utilized to significantly enhance the accuracy performance of intrusion detection systems (IDSs) and to increase their adaptability to dynamic networks. In this paper, related works proposed in the last three years are collected and selected, considering both traffic-based and behavior-based intrusion detection. Subsequently, a study and analysis of these related works is conducted. Additionally, we compare their techniques utilized, results, advantages, and disadvantages. Finally, we analyze the existing challenges and open issues and suggest some insightful future research works.

Keywords: Intrusion Detection · Internet of Things · Deep Neural Network · Machine Learning

1 Introduction

The Internet of Things (IoT) is a network formed by various devices, sensors, and objects connected through the internet, enabling interconnectivity and intelligent management [1]. In IoT, a number of personal sensitive data is often collected by devices, including users' geographical and health information. Therefore, if hackers infiltrate IoT devices,

they may steal users' privacy for malicious purposes or even launch attacks on the entire network. Therefore, intrusions and attacks have become a significant security issue in IoT.

In the realm of IoT, the reliability of device connectivity hinges significantly on the security models implemented. These models used to be deployed in safeguarding user data and thwarting devices from partaking in nefarious activities. The IDSs stand out as one of the primary tools utilized to defend against malicious intrusions [2]. They often be designed based on methods such as Naive Bayes and fuzzy algorithms. However, IDSs designed based on traditional methods often struggle to adapt well to the dynamic network environment. Fortunately, intrusion and attack detection have been extensively studied in the fields of statistics and machine learning. Deep neural network (DNN) is a type of deep learning methods based on artificial neural network. They achieve high-level abstraction and processing of input data through the combination and learning of multiple layers of neurons, enabling tasks such as classification and regression [3]. Moreover, by incorporating techniques such as regularization and data augmentation, it demonstrates high generalizability and robustness based on DNNs.

Due to the numerous advantages of DNNs, a substantial amount of IDSs have been proposed based on DNNs in IoT. While some survey and review papers [4–6] on deep learning-based intrusion detection in IoT have been published, there still remains a gap for a review paper specifically focused on DNN-based intrusion detection. The main findings of this research are as follows:

1. Conducting a thorough analysis and comparison of advanced works related to trust evaluation using deep learning techniques in OSNs. Analyzing and comparing some advanced related works on intrusion detection in IoT with DNN.
2. Find that, in DDoS attacks, better performance is exhibited by network traffic-based systems. Additionally, higher accuracy is achieved by device behavior-based systems for internal threats and unknown types of attacks.
3. Analyzing challenges and open issues and suggesting some research directions.

This paper is organized as follows. The analysis and comparison of some advanced work is presented in Sect. 2. Current issues and challenges are discussed in Sect. 3, and several significant research directions worth studying are proposed in Sect. 4. Lastly, conclusions are drawn in Sect. 5.

2 Analysis and Comparison of Related Work

Due to the high generalization capability and robustness of DNNs, there has been a significant amount of research on DNN-based IDSs in IoT in recent years. We select some related works based on the influence and quality, ensuring diversity in terms of methods and techniques.

2.1 Network Traffic-Based Intrusion Detection

Real-time network communication data is analyzed by network traffic-based IDSs to detect network attacks, port scanning, and other potential intrusion behaviors. Distributed

Denial-of-Service (DDoS) attacks represent a form of network assault characterized by inundating a target system with an overwhelming volume of requests or traffic from numerous IP sources simultaneously. The flow of traffic exceeding the system's capacity leads to the target system being unable to provide services properly.

Recently, some network traffic-based IDSs have been proposed. L. Adi et al. proposed a neural network-based IDS to detect DDoS attacks [7]. It achieves higher accuracy compared to KNN-based schemes, yet its outcomes are heavily impacted by the learning rate and hidden layer's values. Besides, P. Varma et al. introduced a DDoS attack detection method based on the Enhanced Elman spike Neural Network (EENN), and a good accuracy in detecting DDoS attacks was demonstrated [8]. However, it incurred high computational overhead and time complexity.

Additionally, some researchers proposed intrusion detection strategies for DDoS based on recurrent neural network (RNN) [9, 10]. These approaches achieved an accuracy rate of up to 99%. Besides, some IDSs based on network traffic have been proposed using the DRNN [11, 12]. Dense random neural network (DRNN) was a neural network model based on random weights. Unlike traditional neural networks, the weights in the DRNN are randomly initialized and randomly updated at each training iteration. It permitted exploration of a broader weight space during training, thereby mitigating the risk of becoming ensnared in local optima and enhancing generalization capabilities. Z. Huma et al. introduced the Hybrid Deep Random Neural Network (HDRaNN), a hybrid approach designed for cyberattack detection within Industrial IoT (IIoT) environments [13]. K.-H. Le et al. proposed an intelligent IDS called IMIDS for cyber threats in IoT using CNN [14].

Machine learning technology in industrial networks have emerged as a prominent field, with IIoT being a common industrial network [15]. Intrusion often be characterized by complex patterns and features, which could be effectively modeled and captured by CNNs through multiple convolutional layers and activation functions. This facilitated the enhancement of intrusion detection accuracy [16]. However, in certain tasks involving sequential data or long-range temporal dependencies, they may struggle to effectively capture these long-term dependencies. The combination of CNNs and long short-term memory (LSTM) algorithm, known as CNN-LSTM, could effectively address the limitations mentioned above. G. Parra et al. introduced a cloud-based distributed DL framework aimed at detecting DDoS and botnet attacks [17]. This framework amalgamated two pivotal security mechanisms, operating synergistically: a Distributed Convolutional Neural Network (DCNN) model designed for detecting phishing attempts and application-layer DDoS attacks.

The practical deployment of an IDS based on DNNs has been impeded due to the constrained computing and storage capabilities of IoT devices. Researchers have attempted to design IDSs based on lightweight neural networks (LNNs) to overcome this challenge. LNN-based IDS maintenance has maintained a low model complexity and a reduced number of parameters, while achieving acceptable performance and accuracy. Additionally, R. Zhao et al. proposed a lightweight DNN-based method for network recognition in IoT [18]. It achieved high detection accuracy while having a compact system size and low time complexity. To reduce computational costs, an IDS proposed by M. Elaziz et al. with DNNs to extract optimal features from network data [19].

M. Elaziz et al. employed DNNs to extract optimal features from IoT network data and introduced an efficient feature selection technique leveraging the recently developed SI optimization Capuchin monkey search algorithm (CapSA) [20]. However, they found that the CapSA model exhibited poor generalization performance on imbalanced datasets. To enhance generalization and tackle class imbalance issues within intrusion detection datasets, A. Thakkar et al. employed an ensemble learning approach, employing label-class classifiers and utilizing DNNs as base estimators [21].

In Table 1, we summarize several proposed networks traffic-based IDSs using DNNs in IoT. The summary of the comparison on their results, advantages, and disadvantages is presented, where A, P, and R respectively represent accuracy, precision, and recall.

Table 1. Summary of Related Network Traffic-based Intrusion Detection Schemes Using DNNs in IoT

Scheme	Tech.	Dataset	Result	Advantage	Disadvantage
ADI [7]	DNN	DDoS dataset	A = 99.99%	Accuracy was significantly higher than KNN-based methods	The results were influenced greatly by the learning rate and hidden layers
VARMA [8]	EENN	CICDDoS 2018	A = 97.89%, P = 98.85%, R = 99.97%	High accuracy performance	High computational overhead and time complexity
ASWAD [9]	RNN, LSTM	CICIDS2017	A = 99.76%, P = 98.90%	Combining the advantages of three neural networks	Lacking of a realistic testing platform results in unstable reliability
YOUSUF [10]	RNN	NSL KDD	A = 99.98%, P = 99.9%, R = 99.9%	Excellent accuracy and throughput performance	A limited number of DDoS attack could be effectively detected
RED. [11]	DRNN	Synthetic	A = 98.28%	High accuracy and robustness	High time complexity
LATIF [12]	DRNN	DS2OS	A = 99.20%, P = 99.11%, R = 99.13%	High detection accuracy and small system size	The feasibility of the system has not been validated

(continued)

Table 1. (*continued*)

Scheme	Tech.	Dataset	Result	Advantage	Disadvantage
HUMA [13]	DRNN	DS2OS, UNSW-NB15	A = 98.56%, P = 98.25%, R = 98.36%	High accuracy in classifying the 16 types of attacks	High time complexity, and it has not been deployed on a real platform
LE [14]	CNN	UNSW-NB15	A = 96.69%	It could detect 9 types of network attacks with high f-score	The detection accuracy of IMIDS for the Backdoor attack was low
PARRA [17]	DCNN, LSTM	N_BaIoT	A = 94.3%, F1 = 93.58%	High detection rates for DDoS and Botnet	The detection of attack types was limited. High time complexity
ZHAO [18]	LNN	UNSW-NB15, Bot-IoT	A = 98.94%, 99.99%	High detection accuracy and low time complexity	The system has not been deployed and tested in a real platform
ASG. [19]	CNN	TON-IoT	A = 99.99%, P = 99.99%, R = 99.99%	Low computational costs and training expenses	Challenges still existed in practical deployment
ELAZIZ [20]	DNN	NSL-KDD, BoT-IoT, KDD99	F1 = 92.7%, 99.9%, 76.8%	Low computational costs and training expenses	Performing poorly in terms of accuracy and generalization

2.2 Device Behavior-Based Intrusion Detection

Device behavior-based IDS focuses on monitoring and analyzing the behavioral patterns of IoT devices, such as sensor data and device operation logs. By leveraging DNNs, device behavior-based IDSs can learn the normal behavior patterns of devices and identify anomalous behavior that deviates from the expected normal behavior.

DRNNs have also been used in designing behavior-based IDSs. S. Latif et al. proposed a lightweight DRNN for IoT intrusion detection [23]. It demonstrated good accuracy in binary classification and can effectively identify multiple attack types. However, it had high time complexity and energy consumption. The deep belief network (DBN) was an unsupervised learning algorithm based on a probabilistic graphical model. It consisted of multiple stacked restricted Boltzmann machines, which formed a directed

acyclic graph between layers. H. Bhor et al. proposed a detection method by combining Taylor series with spider monkey optimization algorithm and DBN [24].

Graph neural networks (GNNs) differed from traditional neural networks in that they can effectively model and learn from the nodes and edges of a graph, as opposed to handling vector and matrix data. Y. Zhang et al. proposed a framework based on a GNN for IoT to detection intrusion, but it had limitations in effectively detecting certain types of attacks [25]. A. Basati et al. introduced a lightweight architecture founded on parallel deep autoencoders, which capitalized on both local and contextual information of individual values within feature vectors [26]. On the other hand, S. Kasongo utilized various types of RNNs, such as LSTM, Gated Recurrent Unit (GRU), and Simple RNN, to develop an IDS framework [27]. Besides, P. Sanju presented an advanced heuristic algorithm with integrated RNNs to enhance intrusion detection.

We summarize several proposed device behavior-based IDSs using DNNs in IoT in Table 2. The summary of the comparison on their results, advantages, and disadvantages is presented. It shows that the majority of device behavior-based systems with DNN achieve high accuracy. However, the performance for other types may be limited.

Table 2. Summary of Related Device Behavior-based Intrusion Detection Schemes Using DNN in IoT

Scheme	Tech.	Intrusion handled	Dataset	Results	Advantage	Disadvantage
LATIF [23]	DRNN	DDoS, Scanning	TON_IOT	A = 99.14%	It exhibited good accuracy performance in binary classification	The model exhibited high time complexity and energy consumption
BHOR. [24]	DBN	CMRI, MPCI, RECO	KDD CUP'99	A = 90%, P = 90%, R = 92%	It was superior in terms of accuracy and has a low false alarm rate	When the number of hidden neurons was low, the false alarm rate was high (> 10%)
ZHANG [25]	GNN	FDIA, MATM, PF	Gas pipeline dataset	A = 97.2%, P = 98%, R = 99%	High performance in small training sets and unbalanced class data	The types of attacks that could be detected effectively are limited
BASATI [26]	CNN	DoS, Probe	UNSW-NB15, KDD CUP'99, CICIDS2017	A = 100%, 99.92%, and 99.43%	High detection accuracy performance	The system had a high time complexity, and it has not been deployed on a real IoT platform
KASONGO [27]	RNN	–	NSLKDD, UNSW-NB15	F1 = 99.47% and 80.84%	It was suitable for resource-constrained situations	The detection accuracy was low for a minority of certain types

(continued)

Table 2. (*continued*)

Scheme	Tech.	Intrusion handled	Dataset	Results	Advantage	Disadvantage
SANJU [28]	LSTM	Botnet	IoT-23, CICIDS2017	A = 98.12% and 99.98%	It could capture both local and global patterns in the data	Poor model generalization performance
SAHRMA [29]	DNN, GAN	DoS	UNSW-NB15	A = 84%	It could reduce the number of features before classification, thus lowering the cost	It had slightly lower accuracy and cannot predict trust values in real-time
ROUZ. [30]	Snapshot Ensemble DNN	–	DS2OS	A = 90.58%, P = 87.42%	The detection accuracy was acceptable	The lack of testing on a real platform prevented accurate classification of attacks
PACH. [31]	ANN	Flood attack for fog nodes	Self-created dataset	A = 97.5%, P = 98.4%, R = 98.9%	High detection rates, low false-positive alerts, and small time overhead	The types of attacks that could be detected effectively are limited

3 Challenges and Open Problems

Based on our analysis and comparison, it can be observed that the aforementioned related works exhibit excellent performance. However, challenges and issues still persist in this field. Some of them are outlined as follows.

Initially, it is necessary for IDSs to function within real-time or high-throughput network environments. However, DNNs are frequently associated with high computational complexity and storage demands, potentially hindering real-time performance. Strategies are devised to tackle this challenge, such as lightweighting the model, implementing hardware acceleration, and optimizing algorithms.

Besides, DNNs are susceptible to adversarial attacks. Attackers could deceive IDSs by adding subtle perturbations, making them unable to correctly identify intrusions. Adversarial attacks encompass attacks on both training data and the model itself, which include adversarial sample generation and adversarial training. To enhance the robustness of IDSs, adversarial training and defense mechanisms need to be implemented.

Moreover, intrusion detection data is typically highly imbalanced, with normal data far outnumbering the anomalous data. It could lead to models being biased towards overfitting to normal data and neglecting anomalous samples during DNN training. It is a challenge to dealing with imbalanced data distribution that requires appropriate sampling strategies or the use of specific loss functions to address this issue. Besides, during inference or training, a significant amount of matrix operations and activation function computations are required, placing high demands on computational resources and energy consumption [32]. Besides, during the actual deployment and updating of IDSs, challenges arise in continuously collecting, processing, and analyzing network data. So, there is a need for rapid model updates to adapt to new intrusions.

4 Research Direction

4.1 An IDS Based on Federated Learning and Transfer Learning

Federated learning and transfer learning can be combined to address some challenges, such as data imbalance and diverse intrusion behaviors. Federated learning, by simultaneously considering multiple tasks or data sources and jointly modeling them, enhances the generalization capability and robustness of the model. The advantage of FL integrated with DNNs is that information is stored on localized IoT devices for model training and only shared on a centralized federated learning server. In intrusion detection, FL can be utilized to improve detection performance by leveraging different network traffic datasets and knowledge of various intrusion types.

Besides, through the utilization of transfer learning, researchers can leverage knowledge acquired from one or multiple related tasks and transfer it to the target task, effectively addressing issues related to data imbalance and few-shot learning. It leverages datasets or pre-trained models from related domains to transfer knowledge to the intrusion detection task, thereby improving the performance.

4.2 An IDS Based on Explainable DNNs

To enhance the interpretability and transparency of IDSs, attention can be directed towards research on interpretable DNNs. Explainable artificial intelligence (XAI) is a methodology and set of techniques designed for artificial intelligence systems with the aim of making their decision-making and operational processes understandable and explainable. It focuses on the design of model architectures and learning algorithms that render the decision-making process of the model more comprehensible and explicable. For the design of IDSs, it requires the construction of an interpretable model structure that can explain why a certain behavior is classified as an intrusion. In addition, combining explainability with adversarial training can enhance a model's ability to withstand adversarial attacks and provide explanations.

In addition, by combining explainability with adversarial training, the model's ability to withstand adversarial attacks can be enhanced and explanations provided. The goal of these proposed research directions is to address the challenges faced by deep neural network-based IDS while providing novel solutions and insights. By combining the knowledge of multi-source data and tasks of FL and transfer learning, the generalization ability and robustness of the model can be enhanced. The aim is to improve the interpretability of IDS, making it more trustworthy and easier to understand.

5 Conclusion

We provide a review of DNN-based intrusion detection within the context of the IoT. Firstly, we categorize the related works into network traffic-based and device behavior-based schemes for analysis and comparison. We find that for DDoS attacks, network traffic-based IDS performed better, while device behavior-based systems achieved higher accuracy for internal threats and unknown attack types. Due to the high generalization capability and robustness of DNN, DNN-based intrusion detection strategies often

achieve detection rates of up to 99% for certain attack types. However, DNNs typically have multiple layers and lots of parameters, resulting in high computational complexity and energy consumption requirements. Additionally, the imbalance in datasets is also an issue that needs to be addressed, as it leads to significant performance variations across different attack types. Finally, we propose two suggested future work. Researchers can explore the combined use of federated learning and transfer learning, or utilize interpretable DNNs, to enhance intrusion detection performance in IoT.

Acknowledgments. This study was funded in part by the Key Research and Development Task Special Project of the Xinjiang Uygur Autonomous Region (grant number 2022B01009), in part by the Shanghai Science and Technology Innovation Action Plan (grant number 21DZ1200600), in part by the National Key Research and Development Program (grant number 2020YFB2104202), and in part by the Shanghai Natural Science Foundation (grant number 21ZR1461700).

References

1. Fang, W., Cui, N., Chen, W., Zhang, W., Chen, Y.: A trust-based security system for data collection in smart city. *IEEE Trans. Ind. Inf.* **17**(6), 4131–4140 (2021)
2. Li, Z., Fang, W., Zhu, C., Gao, Z., Zhang, W.: AI-enabled trust in distributed networks. *IEEE Access* **11**, 88116–88134 (2023)
3. Fang, W., Zhu, C., Guizani, M., Rodrigues, J.J.P.C., Zhang, W.: HC-TUS: human cognition-based trust update scheme for AI-enabled VANET. *IEEE Netw.* <https://doi.org/10.1109/MNET.2023.3320934>
4. Yadav, N., Pande, S., Khamparia, A., Gupta, D.: Intrusion detection system on IoT with 5G network using deep learning. *Wirel. Commun. Mob. Comput.* **2022**, Article no. 9304689 (2022)
5. Jayalaxmi, P.L.S., Saha, R., Kumar, G., Conti, M., Kim, T.H.: Machine and deep learning solutions for intrusion detection and prevention in IoTs: a survey. *IEEE Access* **10**, 121173–121192 (2022)
6. Khan, A.R., Kashif, M., Jhaveri, R.H., Raut, R., Saba, T., Bahaj, S.A.: Deep learning for intrusion detection and security of Internet of Things (IoT): current analysis, challenges, and possible solutions. *Secur. Commun. Netw.* **2022** Article no. 4016073 (2022)
7. Adi, L.W.P., Mandala, S., Nugraha, Y.: DDoS attack detection system using neural network on Internet of Things. In: 2022 International Conference on Data Science and Its Applications (ICoDSA), Bandung, Indonesia, pp. 41–46. IEEE (2022)
8. Ravi Kiran Varma, P., Sathiya, R.R., Vanitha, M.: Enhanced Elman spike neural network based intrusion attack detection in software defined Internet of Things network. *Concur. Comput. Pract. Exp.* **35**(2), Article no. e7503 (2023)
9. Firas Mohammed Aswad, F.M.A., Ali Mohammed Saleh Ahmed, A.M.S.A., Nafea Ali Majeed Alhammadi, N.A.M.A., Bashar Ahmad Khalaf, B.A.K., Mostafa, S.A., Mostafa, S.A.: Deep learning in distributed denial-of-service attacks detection method for Internet of Things networks. *J. Intell. Syst.* **32**, 1–13 (2023)
10. Yousuf, O., Mir, R.N.: DDoS attack detection in Internet of Things using recurrent neural network. *Comput. Electr. Eng.* **101**, Article no. 108034 (2022)
11. Reddy, D.K., Behera, H.S., Nayak, J., Vijayakumar, P., Naik, B., Singh, P.K.: Deep neural network based anomaly detection in Internet of Things network traffic tracking for the applications of future smart cities. *Trans. Emerg. Telecommun. Technol.* **32**(7), Article no. e4121 (2021)

12. Latif, S., Zou, Z., Idrees, Z., Ahmad, J.: A novel attack detection scheme for the industrial Internet of Things using a lightweight random neural network. *IEEE Access* **8**, 89337–89350 (2020)
13. Huma, Z.E., Latif, S., Ahmad, J., et al.: A hybrid deep random neural network for cyberattack detection in the industrial Internet of Things. *IEEE Access* **9**, 55595–55605 (2021)
14. Le, K.H., Nguyen, M.H., Tran, T.D., Tran, N.D.: IMIDS: an intelligent intrusion detection system against cyber threats in IoT. *Electronics* **11**(4), Article no. 524 (2022)
15. Fang, W., Zhu, C., Yu, F.R., Wang, K., Zhang, W.: Towards energy-efficient and secure data transmission in AI-enabled software defined industrial networks. *IEEE Trans. Ind. Inf.* **18**(6), 4265–4274 (2022)
16. Fang, W., Zhu, C., Zhang, W.: Toward secure and lightweight data transmission for cloud–edge–terminal collaboration in artificial intelligence of things. *IEEE Internet Things J.* **11**(1), 105–113 (2024)
17. Parra, G.D.L.T., Rad, P., Choo, K.K.R., Beebe, N.: Detecting Internet of Things attacks using distributed deep learning. *J. Netw. Comput. Appl.* **163**, Article no. 102662 (2020)
18. Zhao, R., et al.: A novel intrusion detection method based on lightweight neural network for Internet of Things. *IEEE Internet Things J.* **9**(12), 9960–9972 (2021)
19. Asgharzadeh, H., Ghaffari, A., Masdari, M., Gharehchopogh, F.S.: Anomaly-based intrusion detection system in the Internet of Things using a convolutional neural network and multi-objective enhanced Capuchin Search Algorithm. *J. Parallel Distrib. Comput.* **175**, 1–21 (2023)
20. Abd Elaziz, M., Al-qaness, M.A., Dahou, A., Ibrahim, R.A., Abd El-Latif, A.A.: Intrusion detection approach for cloud and IoT environments using deep learning and Capuchin Search Algorithm. *Adv. Eng. Softw.* **176**, Article no. 103402 (2023)
21. Thakkar, A., Lohiya, R.: Attack classification of imbalanced intrusion data for IoT network using ensemble learning-based deep neural network. *IEEE Internet Things J.* **10**(13), 11888–11895 (2023)
22. Wang, X., Wang, Y., Javaheri, Z., Almutairi, L., Moghadamnejad, N., Younes, O.S.: Federated deep learning for anomaly detection in the Internet of Things. *Comput. Electr. Eng.* **108**, Article no. 108651 (2023)
23. Latif, S., Huma, Z.E., Jamal, S.S., et al.: Intrusion detection framework for the Internet of Things using a dense random neural network. *IEEE Trans. Ind. Inf.* **18**(9), 6435–6444 (2021)
24. Bhor, H.N., Kalla, M.: TRUST-based features for detecting the intruders in the Internet of Things network using deep learning. *Comput. Intell.* **38**(2), 438–462 (2022)
25. Zhang, Y., Yang, C., Huang, K., Li, Y.: Intrusion detection of industrial internet-of-things based on reconstructed graph neural networks. *IEEE Trans. Netw. Sci. Eng.* **10**(5), 2894–2905 (2022)
26. Basati, A., Faghih, M.M.: Efficient network intrusion detection in IoT using parallel deep auto-encoders. *Inf. Sci.* **598**, 57–74 (2022)
27. Kasongo, S.M.: A deep learning technique for intrusion detection system using a Recurrent Neural Networks based framework. *Comput. Commun.* **199**, 113–125 (2023)
28. Sanju, P.: Enhancing intrusion detection in IoT systems: a hybrid metaheuristics-deep learning approach with ensemble of recurrent neural networks. *J. Eng. Res.* Article no. 100122 (2023)
29. Sharma, B., Sharma, L., Lal, C., Roy, S.: Anomaly based network intrusion detection for IoT attacks using deep learning technique. *Comput. Electr. Eng.* **107**, Article no. 108626 (2023)

30. Rouzbahani, H.M., Bahrami, A.H., Karimipour, H.: A snapshot ensemble deep neural network model for attack detection in industrial Internet of Things. In: Karimipour, H., Derakhshan, F. (eds.) AI-Enabled Threat Detection and Security Analysis for Industrial IoT, pp. 181–194. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-76613-9_10
31. Pacheco, J., Benitez, V.H., Felix-Herran, L.C., Satam, P.: Artificial neural networks-based intrusion detection system for Internet of Things fog nodes. *IEEE Access* **8**, 73907–73918 (2020)
32. Fang, W., Zhang, W., Yang, W., Li, Z., Gao, W., Yang, Y.: Trust management-based and energy-efficient hierarchical routing protocol in wireless sensor networks. *Digit. Commun. Netw.* **7**(5), 470–478 (2021)



CNN-SENet: A Convolutional Neural Network Model for Audio Snoring Detection Based on Channel Attention Mechanism

Zijun Mao¹, Suqing Duan³, Xiankun Zhang^{1(✉)}, Chuanlei Zhang^{1(✉)}, Haifeng Fan^{1,2(✉)}, Bolun Zhu¹, and Chengliang Huang⁴

¹ College of Artificial Intelligence, Tianjin University of Science and Technology, Tianjin, China
{zhxkun, 97313114, hffan}@tust.edu.cn

² Yunsheng Intelligent Technology Co., Ltd., Tianjin, China

³ College of Electronic Information and Automation, Tianjin University of Science and Technology, Tianjin, China

⁴ Toronto Metropolitan University, Toronto, Canada

Abstract. Snoring is recognised as an independent risk factor for cardiovascular disease, making its monitoring crucial for disease prevention and management. Existing technologies face challenges due to the diversity of signals caused by individual physiological differences, as well as the non-linearity and multidimensional complexity of the signals themselves, making accurate and robust snoring monitoring difficult. To address these issues, this study proposes a hybrid architecture combining Convolutional Neural Networks (CNN) and Squeeze-and-Excitation Networks (SENet). By employing multidimensional nonlinear modelling and Mel-spectrogram feature extraction techniques, this architecture significantly improves the accuracy of snoring and non-snoring signal detection in various complex noise environments. In addition, the introduction of a channel attention mechanism further enhances the model's focus on multidimensional feature weights, ensuring high robustness and excellent analysis efficiency in the face of environmental noise interference. Experimental results validate the effectiveness of the proposed model, achieving 100% snoring recognition accuracy in noiseless environments and maintaining an average accuracy of 97.17% in noisy conditions, significantly outperforming existing traditional methods and recent advanced baseline models. This research provides an efficient and accurate new method for snoring monitoring.

Keywords: Snoring Monitoring · Deep Learning · Mel Frequency Cepstral Coefficients (MFCC) · Channel Attention Mechanism · Signal Processing

1 Introduction

Snoring signals exhibit remarkable non-linearity and high-dimensional characteristics due to the complex interplay of fundamental and harmonic frequencies over time. Such signals are influenced by individual physiological variations - including upper

airway anatomy and soft tissue properties - resulting in distinctive spectral profiles. Intra-individual differences, such as changes in sleep position and stage, also contribute to signal variability. Collection of snoring data is challenged by environmental noise, which can obscure signal features and degrade the performance of detection algorithms. Analysis of these signals is critical for the diagnosis of sleep disorders, particularly obstructive sleep apnoea (OSA), which is associated with serious health conditions such as cardiovascular disease and hypertension [1]. Although polysomnography (PSG) is the diagnostic benchmark for OSA, its high cost and inaccessibility are barriers. There is a need for a reliable, universal snoring detection model that accounts for individual and environmental diversity [2].

Recent advances in deep learning have significantly improved snoring detection by overcoming the limitations of traditional signal processing techniques. Deep learning algorithms excel at handling non-linear data without the need for predefined data distribution assumptions. The workflow includes pre-processing, feature extraction - commonly using Mel Frequency Cepstral Coefficients (MFCC), bark sub-band and Linear Prediction Coding (LPC) [3] - followed by dimensionality reduction using Principal Component Analysis (PCA) and classification using Support Vector Machines (SVM) or neural networks [4]. The integration of attention mechanisms into neural networks has significantly improved model performance by prioritising salient features and relevant information, and has proven particularly effective in managing the complex, multidimensional data inherent in biomedical signal processing tasks such as snoring analysis.

The contributions of this study are summarized as follows:

1. Proposed a hybrid architecture combining Convolutional Neural Networks (CNN) and Squeeze-and-Excitation Networks (SENet) to improve the accuracy of audio snoring detection in different noise environments.
2. Design of a channel attention mechanism to improve the focus on multidimensional feature weights, ensuring high robustness and efficiency in noisy conditions.

2 Related Work

Deep learning has significantly advanced the field of sleep-disordered breathing diagnostics, particularly for obstructive sleep apnoea (OSA), demonstrating its ability to improve diagnostic accuracy while being cost effective [5]. Li et al. [6] have pioneered a hybrid model combining 1D and 2D CNNs that processes sleep respiratory sounds using non-contact methods, achieving 89.3% accuracy through an innovative conversion of raw signals into visualised 2D images, highlighting the scalability and cost-effectiveness of the method for OSA screening. Furthermore, Ding et al. [7] introduced a model that integrates pre-trained VGG19 with LSTM networks, using Mel spectrograms for snoring sounds, and achieved 99.31% accuracy in differentiating OSAHS patients from simple snorers, demonstrating the ability of deep learning to handle complex signals. Li et al. [8] created a multi-class temporal convolutional network (TCN) with an OSA detection rate of 96.7%, demonstrating the feasibility of using these algorithms in embedded devices for home monitoring. In addition, Li et al. [9] compiled a database of 80,000 clinician-annotated snoring clips and developed a 2D CNN that successfully identified

OSAHS with 92.5% accuracy, further demonstrating the practicality of deep learning in snoring sound classification.

When selecting features for snoring classification, the Mel Frequency Cepstral Coefficients (MFCC) technique stands out for its effectiveness. It includes procedures such as pre-emphasis, signal framing, power spectrum analysis, Mel filtering, and discrete cosine transform (DCT) [10]. Khan [11] used MFCC to develop a wearable device that directly feeds feature images into a deep learning model, achieving 96% accuracy in snoring detection via a smartphone app. This highlights the synergy of MFCC and deep learning in improving detection accuracy. In addition, Janott et al. [12] compiled a comprehensive snoring database, which facilitated the training of SVM classifiers with Low-Level Descriptors (LLDs) related to MFCC, resulting in superior test set performance. Lim et al. [13] showed that combining MFCC with STFT for feature extraction and using RNNs for classification could improve accuracy even on smaller datasets. He et al. [14] refined the MFCC approach by integrating a sound signal processing technique, PAF, with a CNN, simplifying the network complexity and achieving 99.25% validation accuracy. Wall et al. [15] extended audio classification by using BiLSTM, GRU and attention mechanisms within a deep learning framework, achieving up to 96.8% accuracy. These studies highlight the versatility of deep learning in audio processing, especially for temporally complex sound signals.

While existing methods have made significant progress, they also have several limitations. First, the reliance on large data sets and complex models limits the applicability of these methods in resource-constrained settings. Second, most research has focused on snoring monitoring in quiet environments, with relatively little exploration of practicality and accuracy in diverse real-world settings. To address these challenges, this study optimises the structure and algorithms of deep learning models to improve the efficiency of processing small datasets and enhance the ability to extract features from multidimensional signals, aiming at a more comprehensive and accurate diagnosis of snoring. At the same time, the generalisation ability and practicality of the model in real-world application scenarios will be further investigated to enhance its prospects for use in real-world settings.

3 Methods

Traditional signal processing methods are often limited in their ability to capture subtle changes in signals and achieve high-precision classification. In recent years, Convolutional Neural Networks (CNNs) have become a focus of research due to their demonstrated capabilities in image and audio signal processing. However, the direct application of traditional CNN models to snoring signal processing may not adequately capture and exploit the causal relationships and detailed features within the signals, thereby affecting the accuracy of the final diagnosis or classification.

In this context, this study proposes a novel deep learning architecture - the CNN-SENet structure. This structure incorporates the Squeeze-and-Excitation Networks (SENet) module, which enhances the model's ability to capture and represent signal features, particularly in identifying and weighting important features within signals. As illustrated in Fig. 1, the detailed configuration of the CNN- SENet network architecture

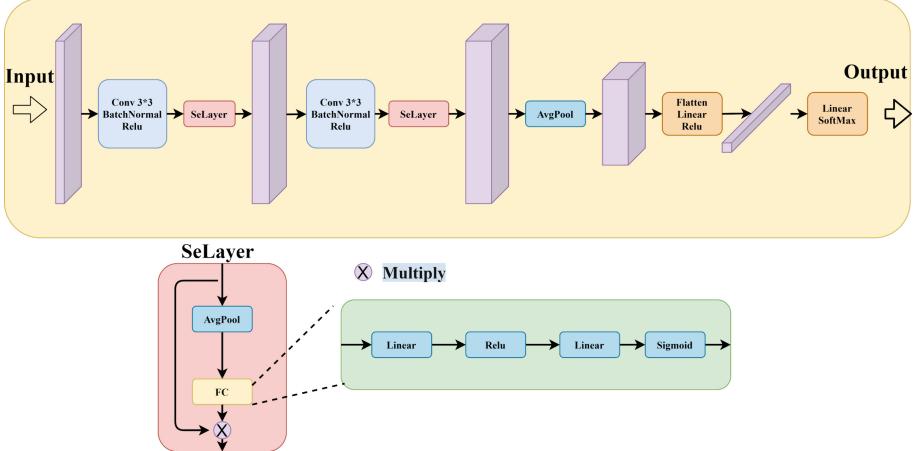


Fig. 1. CNN-SENet network and Compressive excitation network architecture.

is shown, including the functionality and interaction of each layer and how they enhance the model's learning and predictive capabilities through a serialised hierarchical structure. With a refined weight adjustment mechanism, it more accurately identifies and classifies snoring patterns.

3.1 Convolutional Neural Networks

The Convolutional Neural Network (CNN) is a multi-layer processing network model consisting of convolutional layers, pooling layers and fully connected layers. Through convolutional operations, convolutional layers can extract local features from images, and new layers of feature maps can be obtained according to activation functions. The calculation formula for the convolutional layer is as follows:

$$x_j^k = f \left(\sum_{i \in N_j} x_i^{k-1} * n_{ij}^k + b_j^k \right). \quad (1)$$

In the architecture, x_j^k represents the output of the j^{th} neuron in the k^{th} layer after applying the activation function. The activation function f is applied to the weighted sum of the inputs. Common examples include ReLU, sigmoid, and tanh. N_j denotes the index set of neurons in the $(l-1)^{th}$ layer that are connected to the j^{th} neuron. n_{ij}^k refers to the kernel or filter applied to the input of the i^{th} neuron, connecting to the j^{th} neuron in the k^{th} layer. These are the weights of the convolutional layer. b_j^k represents the bias term for the j^{th} neuron in the k^{th} layer.

Considering the effectiveness of the Rectified Linear Unit (ReLU) function in mitigating the vanishing gradient problem and accelerating the convergence speed during training, the ReLU function is chosen as the nonlinear activation function. The ReLU function is defined as follows:

$$f(x) = \max(0, x). \quad (2)$$

To normalise the feature maps, batch normalisation is applied immediately after the convolution operation and before activation, normalising the input of each mini-batch to a layer. This stabilises the learning process and significantly reduces the number of training epochs required to train deep networks. The batch normalisation transformation is formulated as follows:

$$\hat{X} = \frac{X - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}, \quad (3)$$

where \hat{X} is the input MFCC grayscale image to the Convolutional Neural Network, μ_B is the mean of the batched dataset, σ_B^2 is the variance of the batch, and ϵ is a very small number to avoid division by zero.

3.2 SENet

The introduction of the SENet module aims to improve the model's attention to important features in images by dynamically adjusting the response strength of channel features. The SE module consists of two main steps: Squeeze and Excitation. The squeeze step compresses the spatial information of each channel through global average pooling, generating a global feature descriptor. The Excitation step then uses this descriptor through a series of fully connected layers and the sigmoid activation function to learn the weight coefficients of each channel, thereby achieving dynamic recalibration of the channel feature responses. This process allows the network to adaptively emphasise features that are more important for the classification task while suppressing irrelevant information, thereby improving the accuracy and robustness of snoring signal detection in complex backgrounds.

The mathematical expression for the squeeze operation is as follows:

$$z_c = F_{sq}(U) = \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W u_c(i, j), \quad (4)$$

where U represents the image after feature extraction through convolution, u_c is the feature map of the c^{th} channel, and H and W denote the height and width of the image, respectively. z_c is the result after global average pooling.

The mathematical expression for the Excitation operation is as follows:

$$s = F_{ex}(z, W) = \sigma(g(z, W)) = \sigma(W_2 \delta(W_1 z)), \quad (5)$$

where σ represents the Sigmoid activation function, δ denotes the ReLU activation function, W_1 and W_2 respectively represent the weight matrices of two fully connected layers, and $g(z, W)$ denotes the operation of the two fully connected layers, s is the output after aggregating the weights of all channels.

4 Experiment

To comprehensively evaluate the performance of the Convolutional Neural Network-Squeeze-and-Excitation Network (CNN-SENet) model in the task of snoring recognition, this study conducted a detailed comparative analysis with recently published

cutting-edge snoring monitoring models. All experiments were completed under the PyTorch framework, utilizing a computing platform equipped with an RTX3060 graphics card to execute training tasks. To finely tune model performance, we set the batch size to 64 and used the Adam optimizer to adjust network parameters. The learning rate was set to $10e^{-3}$, and the cross-entropy loss function was employed to optimize the model.

4.1 Baseline

Our research compared the proposed CNN-SENet network with established models to evaluate its effectiveness in snoring detection. Notably, the CNN+RNN [16], an integrated deep neural model using a constant Q transform, achieved an accuracy of 95.3%. A CNN model [11] utilizing MEMS microphones and accelerometers as inputs, with MFCCs, reached 99.25% accuracy. MBCNN [17] improved classification by extracting multi-scale features using MFCC and asymmetric convolutional kernels, achieving 99.5% accuracy. Another CNN implementation [18] in wearable technology, employing an auditory detection module, achieved 96% accuracy. SnoreNet [14], a deep convolutional network, processed raw audio to detect snoring with a success rate of 81.82%. While these models have achieved notable results, they still have limitations in fully leveraging the multi-scale features of MFCCs, which affects their classification accuracy. Our study presents an effective model that quickly and accurately detects essential snoring signal features, offering a significant improvement in performance over existing methods.

4.2 Datasets

In this study, we used the public snoring dataset published by Khan (2021) on the Kaggle platform as the experimental basis [11]. The dataset contains a total of 1000 audio samples, categorised into two states: non-snoring (label 0) and snoring (label 1), with each state containing 500 audio samples. Each sample has a duration of 1 s. In order to thoroughly analyse and identify the signal characteristics of these two different states, we performed a Fast Fourier Transform (FFT) on all the audio samples, focusing on extracting and preserving their peak curves. This processing step revealed the differences between non-snoring and snoring signals, providing a clear visual reference for our subsequent analysis and model training, as shown in Figs. 2 and 3.

The snoring samples include 363 samples with snoring in the absence of background noise and 137 samples with non-snoring background noise. The snoring samples consist of snoring sounds from children, adult males, and adult females, with no background noise. The non-snoring samples are composed of various background noises such as baby crying, talking, and other potential noises near the snorer, making up 10 types.

The snoring dataset is divided into training, validation, and test sets in a 7:2:1 ratio. Each set contains an equal number of snoring and non-snoring samples, as shown in Table 1.

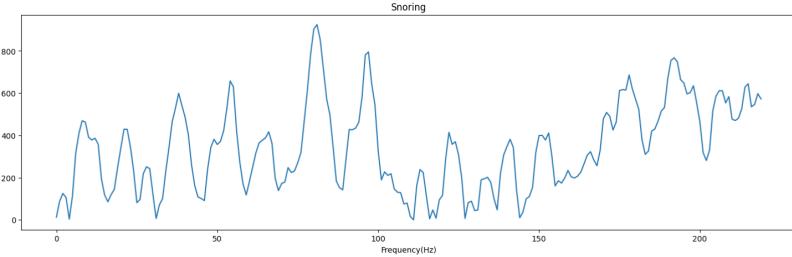


Fig. 2. FFT-Processed Snoring Signal Characteristics.

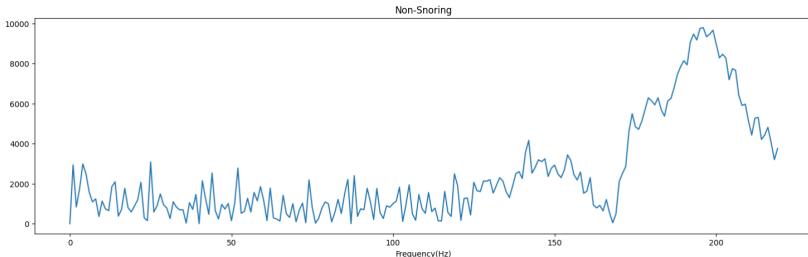


Fig. 3. FFT-Processed Non-Snoring Signal Characteristics.

Table 1. Snore dataset setup.

	Snoring	Non-snoring	Total
Training	350	350	700
Validation	100	100	200
Test	50	50	100

4.3 Data Preprocessing

This study employs a two-stage data processing workflow for audio recognition: initial Mel Frequency Cepstral Coefficients (MFCC) feature extraction from audio signals, followed by min-max normalization to optimize inputs for the model. From the dataset, 1000 audio samples with intact labels were selected for evaluation. Each audio signal $y(t)$ and its sampling rate sr underwent MFCC extraction, starting with pre-emphasis filtering $y'(t)$ to amplify high-frequency components, using a coefficient near 1. The signal was then segmented into frames with a 512-sample overlap, windowed to mitigate boundary discontinuities, and transformed via FFT. The frequency spectrum was mapped onto the Mel scale using Mel filters, reflecting the human ear's non-linear hearing, and the MFCC features were derived through DCT, resulting in 1000 images of size 400×400 .

For efficient processing and to minimize computational demands, these MFCC images were converted to grayscale and normalized using the formula:

$$X_n = \frac{X - X_{min}}{X_{max} - X_{min}}, \quad (6)$$

where X_n denotes the normalized MFCC image, X the original image, X_{min} and X_{max} represent the minimum and maximum pixel values across all images, facilitating uniform scaling. This normalization standardizes the dataset, enabling the model to effectively recognize patterns and ensures consistent image scale for learning.

This approach transforms one-dimensional audio signals into two-dimensional image representations, redefining the audio recognition challenge as an image classification task, thereby leveraging deep learning for complex pattern identification in audio signals with enhanced classification accuracy.

4.4 Evaluation Indicators

To assess the overall performance of the CNN-SENet model in detecting anomalies, we use accuracy, precision, recall, and F1 score as evaluation metrics:

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}, \quad (7)$$

$$Precision = \frac{TP}{TP + FP}, \quad (8)$$

$$Recall = \frac{TP}{TP + FN}, \quad (9)$$

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}. \quad (10)$$

True Positives (TP) refer to the count of anomalous samples that are accurately identified as anomalies. False Positives (FP) denote the instances where normal samples are incorrectly classified as anomalies. False Negatives (FN) represent the anomalous samples that fail to be detected as such. True Negatives (TN) are the normal samples that are correctly identified as normal.

Additionally, the cross-entropy loss function is employed to optimize the model. The cross-entropy loss for a single instance is defined as:

$$L_{CE} = -\sum_{i=1}^C y_i \log(\hat{y}_i), \quad (11)$$

where y_i is the true label (one-hot encoded) and \hat{y}_i is the predicted probability for class i . The overall cross-entropy loss for a batch of instances is the average of the individual losses.

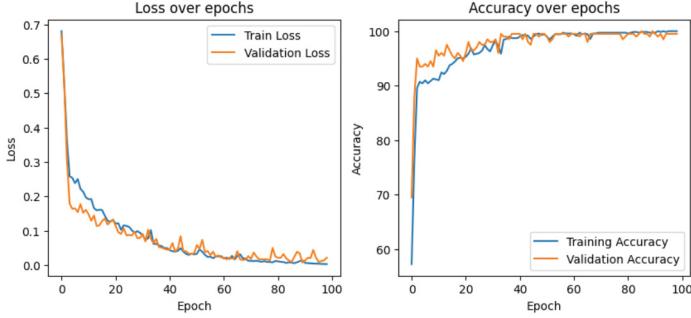


Fig. 4. Trends in training loss and validation loss (left), and training accuracy and validation accuracy (right) over 100 epochs for the snore recognition model based on CNN-SENet.

4.5 Experimental Results

Figure 4 shows a detailed view of the model's training loss, validation loss, training accuracy and validation accuracy trends over 100 training epochs. The results show that the model is in a good state of training, with both the loss and accuracy curves showing a stable trend, demonstrating the stability and convergence of the model during training.

Table 2 demonstrates the performance of the CNN-SENet model on the dataset, surpassing both baseline and recent models in terms of accuracy. This improvement stems from the integration of the Squeeze and Excitation (SE) layer, which introduces a channel-wise attention mechanism post-convolution, enabling dynamic recalibration of feature weights. Consequently, the model accurately prioritizes features critical to snoring detection, achieving a perfect accuracy rate of 100% and outperforming current benchmarks.

Table 2. Comparison with previous works for snoring detection.

Method	Features	Snoring Detection Accuracy(%)
CNN+RNN	CQT spectrum	95.30
CNN	Statistics analysis/MFCCs	99.25
MBCNN	MFCCs	99.50
CNN	MFCCs	96.00
SnoreNet	None	81.82
Baseline	MFCCs	98.00
CNN-SENet	MFCCs	100.00

Furthermore, in Table 3, to assess the effectiveness of the channel attention mechanism, an ablation study was performed under the same experimental conditions with a baseline network that removed the attention structure. Although the baseline network retained the core convolutional layers of the CNN architecture, it removed the attention

module. Comparing the CNN-SENet model with its baseline version and the SE module, it was found that the introduction of the channel attention mechanism significantly improved the model in terms of multi-scale extraction of MFCC features, improving accuracy, precision, recall and F1 score by 2%, 1.89%, 2.04% and 2% respectively, demonstrating the significant improvement in model performance due to the introduction of the channel attention mechanism.

Table 3. Ablation experiment

Method	Accuracy(%)	Precision(%)	Recall(%)	F1 Score(%)
Baseline	98.00	98.11	97.96	98.00
CNN-SENet	100.00	100.00	100.00	100.00

In order to explore the model's detection capabilities under more complex environmental conditions, this study introduced white noise and train noise as anomalous experimental conditions. By setting different levels of white noise (0.01, 0.02, 0.05) and train running noise (0.1, 0.2, 0.5), the performance of the model under these complex backgrounds was investigated, as shown in Table 4. The CNN- SENet model, by weighting the feature channels output by the convolutional layers, improved the model's response to important features. It extracted and learned more critical feature weights within the MFCC features, maintaining an average accuracy of 97.17% for snoring detection under complex noise conditions. The jitter observed in the training process curves can be attributed to interference in the model's judgement caused by the introduction of noise, but this phenomenon did not significantly affect the model's classification ability under complex judgement conditions, further demonstrating the model's robustness and applicability.

Table 4. Add different noise comparison

Noise_level	Accuracy(%)	Precision(%)	Recall(%)	F1 Score(%)
White_noise 0.01	99.00	98.86	99.12	98.98
White_noise 0.02	97.00	96.98	97.04	97.00
White_noise 0.05	97.00	97.46	96.59	96.93
Traffic_noise 0.1	99.00	99.15	98.81	98.97
Traffic_noise 0.2	98.00	98.11	97.96	98.00
Traffic_noise 0.5	93.00	93.23	92.87	92.97
Avg	97.17	97.30	97.07	97.14

5 Summary

This paper proposes a deep learning based convolutional neural network squeeze and excitation network (CNN-SENet) model. This model uses Mel spectrograms for feature extraction, and incorporates a channel attention mechanism to improve the model's ability to detect important features in the signals. The performance of our proposed model on the snoring dataset proves its superiority over models from recent years, effectively extracting key information from multi-scale features and accurately achieving snore sound prediction and classification.

Based on our findings, the high accuracy and robustness of the current model could enable the development of real-time snoring monitoring and sleep disorder diagnosis systems. Such systems would not only provide clinicians with accurate diagnostic data, but also allow patients to self-monitor at home, facilitating early detection of sleep problems. In addition, these systems could be further enhanced by incorporating multimodal data fusion techniques to include variables such as sleep posture, heart rate and blood oxygen saturation. This approach would lead to a more comprehensive model of sleep quality assessment, improving both the scope and accuracy of diagnoses.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (Grant No. 62377036), Tianjin Municipal Training Program of Innovation and Entrepreneurship for Undergraduates (202310057383).

References

1. Huang, Z.: Effects of demographic and sleep-related factors on snoring sound parameters. *Sleep Med.* **104**, 3–10 (2023)
2. Lechat, B.: Multi-night measurement for diagnosis and simplified monitoring of obstructive sleep apnoea. *Sleep Med. Rev.* (2023)
3. Sun, X.: Effective feature selection based on Fisher Ratio for snoring recognition using different validation methods. *Appl. Acoust.* **185**, 108429 (2022)
4. Erdenebayar, U.: Deep learning approaches for automatic detection of sleep apnea events from an electrocardiogram. *Comput. Methods Programs Biomed.* **180**, 105001 (2019)
5. Bi, W.L., Hosny, A., Schabath, M.B., et al.: Artificial intelligence in cancer imaging: clinical challenges and applications. *CA Cancer J. Clin.* **69**(2), 127–157 (2019)
6. Li, R.: Automatic snoring detection using a hybrid 1D–2D convolutional neural network. *Sci. Rep.* **13**(1), 14009 (2023)
7. Ding, L.: Automatically detecting apnea-hypopnea snoring signal based on VGG19+ LSTM. *Biomed. Signal Process. Control* **80**, 104351 (2023)
8. Luo, H.: Design of embedded real-time system for snoring and OSA detection based on machine learning. *Measurement* **214**, 112802 (2023)
9. Li, R.: Convolutional neural network for screening of obstructive sleep apnea using snoring sounds. *Biomed. Signal Process. Control* **86**, 104966 (2023)
10. Abdul, Z.K.: Mel frequency cepstral coefficient and its applications: a review. *IEEE Access* (2022)
11. Khan, T.H.: A deep learning model for snoring detection and vibration notification using a smart wearable gadget. *Electronics* **8**(9), 987 (2019)

12. Janott, C.: Snoring classified: the Munich-Passau snore sound corpus. *Comput. Biol. Med.* **94**, 106–118 (2018)
13. Lim, S.J.: Classification of snoring sound based on a recurrent neural network. *Expert Syst. Appl.* **123**, 237–245 (2019)
14. He, C.: A novel snore detection and suppression method for a flexible patch with MEMS microphone and accelerometer. *IEEE Internet Things J.* **9**(24), 25791–25804 (2022)
15. Wall, C., Zhang, L., Yu, Y., et al.: Deep recurrent neural networks with attention mechanisms for respiratory anomaly classification. In: 2021 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2021)
16. Dong, H., Wu, H., Yang, G., et al.: A multi-branch convolutional neural network for snoring detection based on audio. *Comput. Methods Biomed. Eng.* (2024)
17. Sun, J., Hu, X., Zhao, Y., Sun, S., Chen, C., Peng, S.: SnoreNet: detecting snore events from raw sound recordings. In: 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Berlin, Germany, pp. 4977–4981 (2019)
18. Xie, J., et al.: Audio-based snore detection using deep neural networks. *Comput. Methods Programs Biomed.. Methods Programs Biomed.* **200**, 105917 (2021)



Selecting Effective Triplet Contrastive Loss for Domain Alignment

Changshi Li¹ and Zhijian Yang²

¹ School of Computer Science, Wuhan University, Wuhan 430072, People's Republic of China

² School of Mathematics and Statistics, Wuhan University, Wuhan 430072,
People's Republic of China

zjyang.math@whu.edu.cn

Abstract. Deep Neural Networks (DNNs) have achieved great success in many applications. However, the performance of DNNs degrades significantly in unseen scenarios. Triplet contrastive loss as a domain alignment method has been used in domain generalization to solve above problems. However, there is still no method to guide how to select an effective triplet contrastive loss. In this paper, we derive three triplet contrastive losses as upper bounds for contrastive loss. We find an effective triplet contrastive loss by comparing the range of the three triplet contrastive losses and the rate of hard negative loss pairs. Then, the triplet contrastive loss is applied in domain alignment tasks to explore class invariant representations, demonstrating its effectiveness on five standard benchmarks.

Keywords: Domain Alignment · Contrastive Loss · Representation Learning

1 Introduction

Deep Neural Networks (DNNs) have achieved great success in many applications under the assumption that training and test data are independent and identically distributed. However, the performance of DNNs degrades significantly on out-of-distribution datasets [1], namely domain shift. Domain generalization is used to solve domain shift problems, which assumes the test datasets are not accessible during the training process.

Triplet contrastive loss [2] is defined by a triplet of data points: anchor points, positive points and negative points. Triplet contrastive loss aims to pull positive points together and push negative points apart. This characteristic is also employed to acquire domain invariant representations by pulling together points originating from the same domain. Points belonging to the same domain are categorized as positive points, whereas those originating from distinct domains are designated as negative points in the context of domain alignment [3]. While the pulling together of positive points has been thoroughly examined in domain alignment, the method of pushing away negative points remains an area of ongoing discussion. There are three schemes for calculating negative points distance:

1. Calculating the distance between anchor points and negative points only.

2. Calculating the distance between the negative points only.
3. Calculating both the distance between anchor points and negative points and the distance between the negative points.

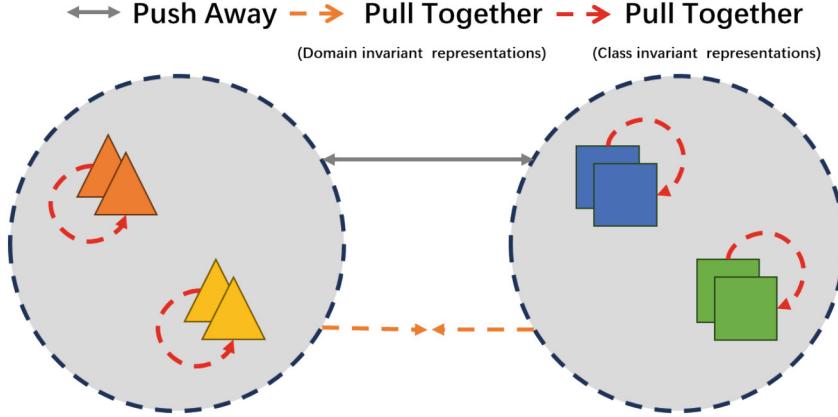


Fig. 1. The unexpected situation of some minibatches when learning domain invariant representations. Different colors represent different classes and different shapes represent different domains.

We analysis above schemes from both theorem and experiments. We verify the effectiveness of the losses from above schemes in domain alignment tasks, which are always used to improve the robustness of DNNs and mitigate the problem caused by domain shift. A model is typically trained on a sequence of minibatches sampled from various domains. It is possible that some of these minibatches may contain unexpected situations that can negatively impact the model's performance under domain invariant representations settings. These unexpected situations refer to points from two domains having different classes in some minibatches, we simply illustrated this in Fig. 1. We consider orange and grey arrows. Domain alignment aims to pull the blue points together with the triangle domain, and the classification task aims to push the blue points away from the triangle domain. As a result, minimizing the discrepancies from different domains would decrease the accuracy of classification models. Therefore, it is necessary to consider the information from labels and learn class invariant representations [4] to avoid such unacceptable situations. We consider red and grey arrows in Fig. 1. Aligning the points from the same classes can reconcile the optimization objective for classification and domain alignment tasks. Contrastive loss can pull positive points from the same class together and push negative points from different class away. So class invariant representations can be obtained naturally in domain alignment by using contrastive loss. By the way, LMMD [5] is indeed a method for learning invariant representations, however, its application is currently limited to domain adaptation and not domain generalization. Our contributions are as follows:

1. We are the first to derive three triplet contrastive losses as upper bounds of contrastive loss as far as we known. Then we select an effective triplet contrastive loss by comparing the range of three triplet contrastive losses and the rate of hard negative pairs. And experiments show the effectiveness of our theoretical analysis.
2. We find that the cosine similarity of points from the same classes can be bounded by the triplet contrastive loss even if the samples comes from different minibatches as a by-product of our theoretical analysis.
3. We prove that class invariant representations can be regarded as sufficient conditions for domain invariant representations.

2 Related Work

Contrastive Learning has achieved great success in recent years, both in theory and in applications. In an early application of contrastive learning, positive pairs were always derived from different views of the same images, whereas the negative pairs were uniformly sampled [6]. In supervised contrastive learning, negative pairs come from different classes and positive pairs come from the same classes [7]. And the contrastive loss in our paper refers to the loss in supervised contrastive learning. However, for contrastive loss, it is difficult to pull together points from the same classes [8] because of the high computational cost. Triplet loss [2] is defined by a triplet loss: anchors, positive points and negative points. Triplet loss would push anchors closer to positive points than negative points, but it suffers from sampling problems that may fail in optimization and convergence rates. Thereafter, proxy-based [9] loss as a form of triplet loss can be verified as a bound on triplet loss without computational burden and sampling issues. Then proxy-based loss [3] is used to domain generalization to learn class variant representations.

Domain Alignment has become one of the most popular domain generalization methods, aiming to minimize the differences in representations in order to achieve invariant representations across various source domains. The invariant representations trained from different source domains are also robust to the unknown target domain. There are many methods to minimize the difference between source domains including: minimizing the mean or variance [10], minimizing the contrastive loss [3], minimizing MMD [11], and domain adversarial learning [12].

3 Method

3.1 Problem Formulation and Notations

Domain alignment aims to achieve generalization to unseen target domains by training a model that depends only on different source domains. All K domains are defined as $\mathcal{D} = \{D_1, D_2, \dots, D_K\}$. In general, a domain D_t is chosen as the target domain from K domains, and the rest are chosen as the source domains. Considering image recognition tasks, the model $E_\theta: \mathcal{X} \rightarrow \mathcal{Y}$ is always divided into two parts: the encoder model $E_{\theta_1}: \mathcal{X} \rightarrow \mathcal{Z}$ and the classification model $E_{\theta_2}: \mathcal{Z} \rightarrow \mathcal{Y}$, where \mathcal{X} are input spaces from source domains, \mathcal{Z} are representation spaces extracted from source domains and \mathcal{Y} are label spaces containing C classes.

The general motivation of most domain alignment [11] is that representations that are invariant on different source domains are also robust on unseen domains. This motivation can be expressed as $P(Z|D) = P(Z)$, where $Z \in \mathcal{Z}$ and $D \in \mathcal{D}$. The representations from various source domains that are invariant are called domain invariant representations, which have been widely-used in domain alignment. However, minimizing the discrepancies from different domains would decrease the accuracy of classification models, as shown in the Fig. 1. To avoid such unexpected situations, we only align representations that have the same classes to obtain class invariant representations.

This motivation can be expressed as $P(Z|Y, D) = P(Z|Y)$, where $Y \in \mathcal{Y}$. We assume Y is independent with domains as $P(Y|D) = P(Y)$. From the perspective of sampling, this assumption means that the number of samples of any class in different domains is the same. Then we have

$$\begin{aligned} P(Z) &= \sum_{Y \in \mathcal{Y}} P(Z|Y)P(Y) = \sum_{Y \in \mathcal{Y}} P(Z|Y, D)P(Y|D) = \sum_{Y \in \mathcal{Y}} \frac{P(Y|Z, D)P(Z|D)}{P(Y|D)}P(Y|D) \\ &= \sum_{Y \in \mathcal{Y}} P(Y|Z, D)P(Z|D) = P(Z|D). \end{aligned} \quad (1)$$

This means that the class invariant representations is a sufficient condition of domain invariant representations. Contrastive loss aims to pull representations together from the same classes, while simultaneously push representations apart from different classes. This means that representations $Z \in \mathcal{Z}$ under the same condition of Y from different domains $\{D_1, D_2, \dots, D_K\}$ satisfies $(Z|Y, D_1) = P(Z|Y, D_2) = \dots = P(Z|Y, D_K)$, which means that $P(Z|Y, D)$ is invariant with respect to random variable D , namely $P(Z|Y, D) = P(Z|Y) \Rightarrow P(Z) = P(Z|D)$. Therefore, contrastive loss can be used in domain alignment tasks directly.

We define $z_i \in Z$ as a representation, a_p and a_n as positive pair and negative pair of z_i respectively. And normalized x are denoted as $\hat{x} = x/\|x\|_2$. Then the general form of contrastive loss is defined as

$$\mathcal{L}_{CL} = -\log \frac{e^{\hat{a}_p^T \hat{z}_i \cdot \tau}}{e^{\hat{a}_p^T \hat{z}_i \cdot \tau} + \sum_{q \in Q} e^{\hat{a}_q^T \hat{z}_i \cdot \tau}}, \quad (2)$$

where Q is the index of negative pair sets with \hat{z}_i and τ is a scaling factor. In practical applications, representations from different views [6] or the same classes [8] of z_i are always chosen as positive pairs of a_p .

In supervision contrastive learning settings, positive pairs z_p and z_i come from the same classes and negative pairs come from different classes. The contrastive loss is defined as

$$\mathcal{L}_{SCL} = -\log \frac{e^{\hat{z}_p^T \hat{z}_i \cdot \tau}}{e^{\hat{z}_p^T \hat{z}_i \cdot \tau} + \sum_{q \in Q} e^{\hat{z}_q^T \hat{z}_i \cdot \tau}}. \quad (3)$$

Obviously, it suffers from computational burden because we have to compute all combinations of positive pairs from the same classes [8]. Thereafter, triplet contrastive losses are proposed to reveal computational burden. But there still has not been a clear discussion on how to calculate negative distance. In the next subsection, we will discuss which scheme is better as triplet contrastive loss.

3.2 Triplet Contrastive Loss

In this section, we show the main theorem results of triplet contrastive loss. For more theorem details, please see the proofs in <https://github.com/lichangshihu/CLforDomainAlignment/tree/main>.

Theorem 1. Let z_i and z_j be the representations of two positive samples. Then, the cosine similarity of any two positive pairs satisfies

$$\hat{z}_i^T \hat{z}_j \geq \frac{4}{\tau} \log \frac{|S_-|}{(e^{\mathcal{L}_{CL}} - 1)} - 7, \quad (4)$$

where \mathcal{L}_{CL} is the triplet contrastive loss which has the form of Eq. (2).

According to Theorem 1, we can obtain a smaller angle between z_i and z_j by minimizing the triplet contrastive loss \mathcal{L}_{CL} . In other words, the angle between positive pairs would be very small regardless of batch size or sampling strategy.

Theorem 2. Let c be the number of classes, $\{w_1, w_2, \dots, w_c\}$ be a set of anchors, where w_p is the anchor of representations from the p -th class. Then the contrastive loss \mathcal{L}_{SCL} satisfies

$$\log \left(1 + \frac{|S_-|}{e^{2\tau}} \right) \leq \mathcal{L}_{SCL} \leq 2\tau \sqrt{4 - \frac{2}{\tau} \log \frac{|S_-|}{e^{\mathcal{L}_{W}} - 1}} + \mathcal{L}_{W}, \quad (5)$$

where the triplet contrastive loss \mathcal{L}_W can be any one of \mathcal{L}_w , \mathcal{L}_s and \mathcal{L}_{ws} . They are defined as $\mathcal{L}_w = -\log \frac{e^{\hat{w}_p^T \hat{z}_i \tau}}{e^{\hat{w}_p^T \hat{z}_i \tau} + \sum_{j \neq p} W_j e^{\hat{w}_j^T \hat{z}_i \tau}}$, $\mathcal{L}_s = -\log \frac{e^{\hat{w}_p^T \hat{z}_i \tau}}{e^{\hat{w}_p^T \hat{z}_i \tau} + \sum_{q \in Q} e^{\hat{z}_q^T \hat{z}_i \tau}}$ and $\mathcal{L}_{ws} = -\log \frac{e^{\hat{w}_p^T \hat{z}_i \tau}}{e^{\hat{w}_p^T \hat{z}_i \tau} + \sum_{j \neq p} e^{\hat{w}_j^T \hat{z}_i \tau} + \sum_{q \in Q} e^{\hat{z}_q^T \hat{z}_i \tau}}$ respectively, where W_j is the number of representations that share the same anchor w_j , \mathcal{L}_{ws} is the triplet contrastive loss employed in PCL [3].

According to Theorem 2, minimizing the contrastive loss \mathcal{L}_{SCL} can be achieved by optimizing the triplet contrastive loss \mathcal{L}_W . We will select one of triplet contrastive losses as our triplet contrastive loss according to their bounds and the influence of hard negative pairs. We first present the bounds of triplet contrastive losses. We denote the function $g(x) := 2\tau \sqrt{4 - \frac{2}{\tau} \log \frac{|S_-|}{e^{x-1}}} + x$, which is the upper bound in the right side of Eq. (5) by setting $x = \mathcal{L}_W$. Obviously, $g(\mathcal{L}_w)$ and $g(\mathcal{L}_s)$ can reach to the same lower bound $\log \left(1 + \frac{|S_-|}{e^{2\tau}} \right)$, which is the same with the lower bound of \mathcal{L}_{SCL} . The minimum value of $g(\mathcal{L}_{ws}) = \log \left(1 + \frac{|S_-| + c - 1}{e^{2\tau}} \right)$ is larger than the minimum value of \mathcal{L}_{SCL} by $\log \left(1 + \frac{c - 1}{e^{2\tau} + |S_-|} \right)$. So we exclude \mathcal{L}_{ws} as our triplet contrastive loss. As shown in Fig. 5, we find that \mathcal{L}_s is affected by hard negative pairs more. Finally, we choose \mathcal{L}_w as our triplet contrastive loss, which is effective on public domain generalization datasets.

3.3 Model Structure

Our model structure is shown in Fig. 2. The encoder can be any DNN backbone. In our experiments, we selected ResNet50 as our encoder due to the limitations of GPU memory.

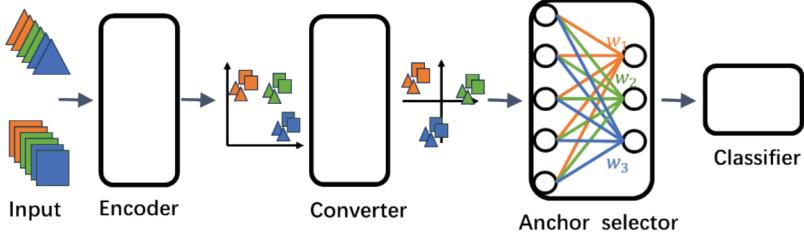


Fig. 2. Illustration of Model Structure. The model comprises four components: an encoder, converter, anchor selector, and classifier. The domains and classes of the input data are represented by different shapes and colors respectively.

The converter modifies the scales of the representations. Since the last layer activation function of ResNet is ReLU, which leads the dot of any two representations always ≥ 0 . This means that the maximum cosine similarity between negative representations $\hat{z}_i^T \hat{z}_j$ is 0. To push negative representations further away (to have a larger angle), we add a converter after the encoder, which can re-scale the range of representations to real number. And the converter is constructed using a feed-forward neural network that comprises a single layer. To keep more information from the representations, we add a regularization loss $L_r = -\log\left(1 + \sum_{j=1}^d e^{\hat{w}_j^T \hat{w}_i - 1}\right)$ on the converter matrix W_t to avoid linear correlation, where $\hat{w}_k = \frac{W_t[k,:]}{\|W_t[k,:]\|}$ for any $k \geq 0$.

The anchor selector is used to automatically select anchors inspired by PCL [3]. The i -th column of linear layer weights (the size is $d \times c$) is selected as a positive anchor for i -th class's representations, and the other weights are selected as negative anchors. Then anchors can be trained and tuned automatically as a part of the neural network. As shown in the anchor selector of Fig. 2, suppose the representations' dimension is 5, the number of class number is 3 (orange, green, blue). w_1, w_2 and w_3 are the linear layer weights. The class's representations and the positive anchor (weights of the linear layer) have the same color. The class's representations and the negative anchor have different colors. The positive anchor of the orange class's representations is w_1 , while the negative anchors of orange class's representations are w_2 and w_3 . The positive anchor of green class's representations and blue class's representations is w_2 and w_3 respectively. And the remaining weights are chosen as negative anchors for their own class.

The classifier is used to tackle image recognition, and we select a feed-forward neural network that comprises a single layer as the classifier. Finally, we have our loss \mathcal{L} by combining the cross entropy loss \mathcal{L}_{ce} from the classifier, the triplet contrastive loss from the anchor selector, and the regularization loss from the converter. $\mathcal{L} = \mathcal{L}_{ce} + \frac{\alpha}{B} \sum_{i=1}^B L_w + \frac{\beta}{d} \sum_{i=1}^d L_r$, where B is batch size, α and β are the trade-off for above losses.

4 Experimental Results

4.1 Datasets

We evaluate triplet contrastive loss on five publicly standard DG datasets. More information about those datasets is shown below. 1. **PACS** [13] contains 9991 images, 7 classes and 4 domains: clipart, painting, real and sketch. 2. **VLCS** [14] contains 10729 images, 5 classes and 4 domains: Caltech101, LabelMe, SUN09 and VOC2007. 3. **OfficeHome** [15] contains 15588 images, 65 classes and 4 domains: art, clipart, product and real. 4. **TerraInc** [16] contains 24788 images, 10 classes and 4 domains: 4 different geographical locations. 5. DomainNet [17] contains 586575 images, 345 classes and 6 domains: clipart, infograph, painting, quickdraw, real and sketch.

4.2 Implementation Details

Our algorithm is based on the open-source code of Domainbed [18]. For a fair comparison, the experimental settings are analogous to those used in Domainbed. For instance, we select one domain as the target domain and utilize the remaining domains as the training domains. Each training domain is further divided into two sections: 20% for validation and 80% for training. We opt for the model that achieves the highest accuracy on the validation sets, and so on. We have made our code publicly available on <https://github.com/lichangshihu/CLforDomainAlignment/tree/main>.

Table 1. We report accuracies and standard errors on five datasets. We highlight the best results in bold. Results with +, * are reported from [18] and [19] respectively. The other algorithms are reported from their own papers.

Algorithm	PACS	VLCS	OfficeHome	TerraInc	DomainNet	Avg
MMD ⁺ [11]	84.7 ± 0.5	77.5 ± 0.9	66.3 ± 0.1	42.2 ± 1.6	23.4 ± 9.5	58.8
Mixstyle [*] [20]	85.2 ± 0.3	77.9 ± 0.5	60.4 ± 0.3	44.0 ± 0.7	34.0 ± 0.1	60.3
DANN ⁺ [12]	83.6 ± 0.4	78.6 ± 0.4	65.9 ± 0.6	46.7 ± 0.5	38.3 ± 0.1	62.6
RSC ⁺ [21]	85.2 ± 0.9	77.1 ± 0.5	65.5 ± 0.9	46.6 ± 1.0	38.9 ± 0.5	62.7
MTL ⁺ [22]	84.6 ± 0.5	77.2 ± 0.4	66.4 ± 0.5	45.6 ± 1.2	40.6 ± 0.1	62.9
ERM [*]	84.2 ± 0.1	77.3 ± 0.1	67.6 ± 0.2	47.8 ± 0.6	44.0 ± 0.1	64.2
SagNet ⁺ [23]	86.3 ± 0.2	77.8 ± 0.5	68.1 ± 0.1	48.6 ± 1.0	40.3 ± 0.1	64.2
SelfReg [24]	85.6 ± 0.4	77.8 ± 0.9	67.9 ± 0.7	47.0 ± 0.3	42.8 ± 0.0	64.2
SWAD [*] [19]	88.1 ± 0.1	79.1 ± 0.1	70.6 ± 0.2	50.0 ± 0.3	46.5 ± 0.1	66.9
CORAL*	88.3 ± 0.1	78.9 ± 0.1	71.3 ± 0.1	51.0 ± 0.1	46.8 ± 0.0	67.3
MIRO [25]	88.4 ± 0.1	79.6 ± 0.2	72.4 ± 0.1	52.9 ± 0.2	47.0 ± 0.0	68.1
$\mathcal{L}_{\sqsupseteq}$ (Ours)	88.5 ± 0.3	80.1 ± 0.2	72.5 ± 0.1	52.6 ± 0.3	47.7 ± 0.1	68.3

4.3 Results and Discussion

We report out-of-domain accuracy for each domain and their average, as shown in Table 1. Our method achieves an average score on five benchmarks that is higher than those reported in other papers, resulting in an improvement of +0.2 percentage points (pp). In particular, our method beats the previous results on four benchmarks: +0.1pp in PACS, +0.5pp in VLCS, +0.1pp in OfficeHome, and + 0.7pp in DomainNet. Our method still achieves second accuracy that are lower than the best results of MIRO (-0.3pp in DomainNet). Our results did not achieve superior results compare to MIRO. So, we test training memory in 32 batches over 200 epochs. The training memory of our methods is 9036.81 MB, while MIRO uses 10819.32 MB. Therefore, the cost of our methods is much lower than MIRO’s. The above results demonstrate the effectiveness of our method on several public benchmarks.

Table 2. Analysis on different triplet contrastive loss.

Algorithm	PACS	VLCS	OfficeHome	TerraInc	DomainNet	Avg.
PCL [3]	88.2	78.4	71.6	52.1	47.7	67.4
\mathcal{L}_s	87.9	79.6	72.2	52.4	47.6	68.0
\mathcal{L}_{ws}	88.3	79.5	72.5	52.2	47.4	68.0
\mathcal{L}_w (Ours)	88.5	80.1	72.5	52.6	47.7	68.3

Analysis on Different Triplet Contrastive Loss. As shown in Table 2, the average accuracy of our model is 68.0% and higher than PCL (67.4%) on the same loss function \mathcal{L}_{ws} , verifying the effectiveness of our model structure. Although \mathcal{L}_s and \mathcal{L}_w have the same bounds in theory. The average accuracy of \mathcal{L}_s is lower than \mathcal{L}_w in experiments since the influence of hard negative pairs. This also verifies the effectiveness of our selection strategies.

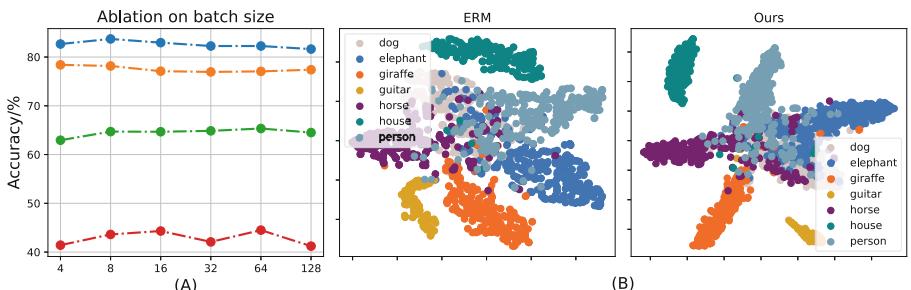


Fig. 3. (A) Ablations on batch size, where x-axis is batch size. (B) Visualizing the representations on the PACS dataset using t-SNE, we compare those learned by ERM and our methods.

Ablation Study on Batch Size. We evaluate the performance of batch size on ResNet18 which is shown in Fig. 3 (A). Experiments show that batch size does not affect our method much. The influence of batch size is less than 2.1% on PACS, less than 1.5% on VLCS, less than 2.4% on OfficeHome. Our method exhibits robust performance, even when the batch size is set to 4 on these datasets. Especially for PACS, the best accuracy is achieved when the batch size is 4. The robust performance also confirms the results of Theorem 1.

Ablation Study on Three Parameters. We examine the influence of three parameters α , β and τ on our models for different datasets, as shown in Fig. 4. Generally, we find that the accuracy of our methods is insensitive to these three parameters, except on TerraInc. The influence of the three parameters on the accuracy is less than 1.2% on PACS, less than 0.8% on VLCS, less than 0.8% on OfficeHome. Our method is not very sensitive on TerraInc and less than 2.5%. In particular, the accuracy is higher than the result reported in Table 1 and reaches 53.76% when $\beta = 0.01$ on TerraInc.

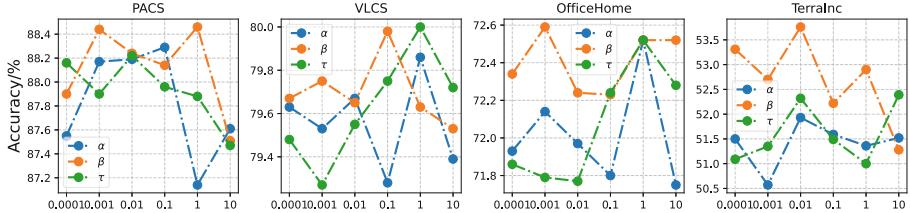


Fig. 4. Ablations on three parameters, where x-axis is the range of α , β and τ .

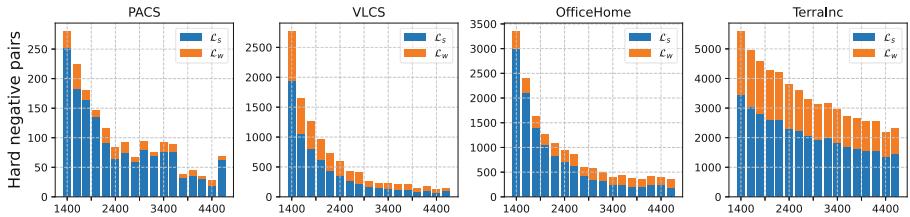


Fig. 5. Analysis on hard negative pairs about \mathcal{L}_s and \mathcal{L}_w , where x-axis is the number of iterations.

Analysis on Hard Negative Pairs. Hard negative pairs have a significant influence on contrastive loss [3], so it provide a standard for us to select the triplet contrastive loss between \mathcal{L}_w and \mathcal{L}_s . We define hard negative pairs as: the number of hard negative pairs is increased by one if the cosine similarity of positive pairs s_+ is greater than the cosine similarity of negative pairs s_- . Then we conduct the number of hard negative pairs in Fig. 5 after 1400 iterations on PACS, VLCS, OfficeHome and TerraInc calculated from the loss of \mathcal{L}_w and \mathcal{L}_s respectively. We find that the number of hard negative pairs decreases as the number of iterations increases on all datasets. At the beginning of training, the number of hard negative pairs in \mathcal{L}_s is higher than \mathcal{L}_w on all datasets. The number in \mathcal{L}_s is higher than \mathcal{L}_w on PACS, VLCS and TerraInc and is similar with \mathcal{L}_w on OfficeHome at

the end of training. These experiments show that \mathcal{L}_s is more affected by hard negative pairs than \mathcal{L}_w . Therefore, we prefer \mathcal{L}_w as our triplet contrastive loss.

Table 3. Accuracy without the converter.

	PACS	VLCS	OfficeHome	TerraInc	DomainNet	Avg
With converter	88.5	80.1	72.5	52.6	47.7	68.3
Without converter	8.7	25.5	1.6	4.0	0.3	8.2

Ablation Study on Converter. We report the results without the converter as shown in Table 3. Models without converter degraded strongly and failed to make classification. To explain this degradation, we report the number of hard negative pairs. Our experiments show that the converter can decrease hard negative pairs as shown in Fig. 6. The number of hard negative pairs is increased by one if any cosine similarity of negative pairs plus a margin of 0.35 is greater than the positive pair. We record hard negative pairs after 1400 iterations on four datasets, experiments show that the number of hard negative pairs without the converter is much higher than with the converter across all datasets. Therefore, the converter has a significant impact on the decrease of hard negative pairs.

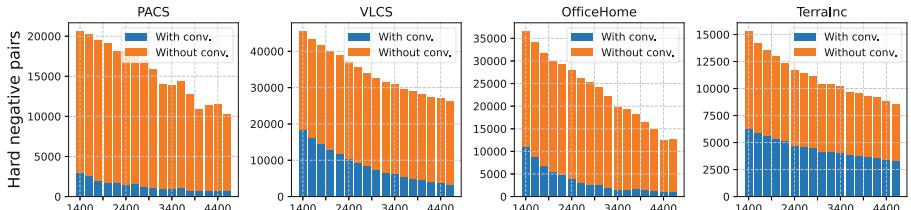


Fig. 6. Ablations on hard negative pairs about the converter (conv.) in model structure, where x-axis is the number of iterations.

Visualization on PACS. We train ResNet50 using ERM (Empirical risk minimizer) and our methods separately on clipart, real, and sketch domains. We report the representations on the painting domain, visualized using t-SNE, as shown in Fig. 3 (B). Our methods demonstrate the ability to effectively separate different classes, outperforming ERM, thereby validating their efficacy in generating class-invariant representations.

5 Conclusion

We have theoretically proven that class-invariant representations are a sufficient condition for domain-invariant representations. Therefore, in the future, other loss functions designed for domain-invariant representations can be directly applied to classes, potentially enhancing the ability of domain generalization. We introduce three triplet

contrastive losses as upper bounds of the standard contrastive loss. Notably, triplet contrastive loss \mathcal{L}_w offers a lower loss bound and reduced rate of hard negative loss pairs, enhancing its effectiveness. Therefore, in the future, we hope that some works utilizing triplet contrastive loss can be further improved by incorporating \mathcal{L}_w .

References

1. 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)
2. Schroff, F., Kalenichenko, D., Philbin, J.: FaceNet: a unified embedding for face recognition and clustering. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Los Alamitos, CA, USA, pp. 815–823. IEEE Computer Society (2015)
3. Yao, X., et al.: PCL: proxy-based contrastive learning for domain generalization. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Los Alamitos, CA, USA, pp. 7087–7097. IEEE Computer Society (2022)
4. Li, Y., et al.: Deep domain generalization via conditional invariant adversarial networks. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, Part XV, vol. 11219, pp. 647–663. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01267-0_38
5. Zhu, Y., et al.: Deep subdomain adaptation network for image classification. IEEE Trans. Neural Netw. Learn. Syst. **PP**, 1–10 (2020)
6. Bachman, P., Hjelm, R.D., Buchwalter, W.: Learning representations by maximizing mutual information across views. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems, Red Hook, NY, USA. Curran Associates Inc. (2019)
7. Khosla, P., et al.: Supervised contrastive learning. In: Advances in Neural Information Processing Systems, vol. 33, pp. 18661–18673 (2020)
8. Sohn, K.: Improved deep metric learning with multi-class n-pair loss objective. In: Advances in Neural Information Processing Systems, vol. 29 (2016)
9. Movshovitz-Attias, Y., Toshev, A., Leung, T.K., Ioffe, S., Singh, S.: No fuss distance metric learning using proxies. In: 2017 IEEE International Conference on Computer Vision (ICCV), Los Alamitos, CA, USA, pp. 360–368. IEEE Computer Society (2017)
10. Gong, M., Tian, X., Liu, T., Tao, D.: Domain generalization via conditional invariant representations. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
11. Li, H., Pan, S., Wang, S., Kot, A.C.: Domain generalization with adversarial feature learning. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Los Alamitos, CA, USA, pp. 5400–5409. IEEE Computer Society (2018)
12. Ganin, Y., et al.: Domain-adversarial training of neural networks. J. Mach. Learn. Res. **17**(1), 2096–2030 (2016)
13. Li, D., Yang, Y., Song, Y., Hospedales, T.M.: Deeper, broader and artier domain generalization. In: 2017 IEEE International Conference on Computer Vision (ICCV), Los Alamitos, CA, USA, pp. 5543–5551. IEEE Computer Society (2017)
14. Fang, C., Xu, Y., Rockmore, D.N.: Unbiased metric learning: on the utilization of multiple datasets and web images for softening bias. In: Proceedings of the 2013 IEEE International Conference on Computer Vision, ICCV 2013, USA, pp. 1657–1664. IEEE Computer Society (2013)
15. Venkateswara, H., Eusebio, J., Chakraborty, S., Panchanathan, S.: Deep hashing network for unsupervised domain adaptation. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Los Alamitos, CA, USA, pp. 5385–5394. IEEE Computer Society (2017)

16. Beery, S., Van Horn, G., Perona, P.: Recognition in terra incognita. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, Part XVI, vol. 11220, pp. 472–489. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01270-0_28
17. Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., Wang, B.: Moment matching for multi-source domain adaptation. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Los Alamitos, CA, USA, pp. 1406–1415. IEEE Computer Society (2019)
18. Gulrajani, I., Lopez-Paz, D.: In search of lost domain generalization. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, 3–7 May 2021. OpenReview.net (2021)
19. Cha, J., et al.: SWAD: domain generalization by seeking flat minima. In: Advances in Neural Information Processing Systems, vol. 34, pp. 22405–22418 (2021)
20. Zhou, K., Yang, Y., Qiao, Y., Xiang, T.: Domain generalization with mixstyle. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, 3–7 May 2021. OpenReview.net (2021)
21. 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, Part II, vol. 12347, pp. 124–140. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58536-5_8
22. Blanchard, G., Deshmukh, A.A., Dogan, U., Lee, G., Scott, C.: Domain generalization by marginal transfer learning. *J. Mach. Learn. Res.* **22**(1), 46–100
23. Nam, H., Lee, H., Park, J., Yoon, W., Yoo, D.: Reducing domain gap by reducing style bias. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Los Alamitos, CA, USA, pp. 8686–8695. IEEE Computer Society (2021)
24. Kim, D., Yoo, Y., Park, S., Kim, J., Lee, J.: SelfReg: self-supervised contrastive regularization for domain generalization. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Los Alamitos, CA, USA, pp. 9599–9608. IEEE Computer Society (2021)
25. Cha, J., Lee, K., Park, S., Chun, S.: Domain generalization by mutual-information regularization with pre-trained models. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) ECCV 2022. LNCS, Part XXIII, vol. 13683, pp. 440–457. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-20050-2_26



RCSnet——Flower Classification Network

Design Based on Transfer Learning and Channel Attention Mechanism

Zijun Mao¹, Tianyu Zhong², Mojieming Wei¹, Runjie Hu¹, and Jianzheng Liu¹(✉)

¹ College of Artificial Intelligence, Tianjin University of Science and Technology, Tianjin, China
jz_leo@126.com

² College of Food Science and Engineering, Tianjin University of Science and Technology,
Tianjin, China

Abstract. The current global landscape contains a vast array of flowering plant species. Given the limitations of small-scale datasets and the complexity of real-world photography, developing an efficient and accurate flower classification system is a significant challenge. This study introduces a novel flower classification network architecture, RCS-net, which uses transfer learning and channel attention mechanisms to improve the accuracy and efficiency of flower image classification. RCS-net is based on ResNet50, which was pre-trained on the ImageNet dataset. By incorporating channel attention mechanisms, it enhances the convolutional neural network's ability to capture key features within flower images, thereby improving classification performance. Experimental validation on a publicly available flower dataset on the Kaggle platform shows that RCS-net achieves an accuracy rate of 96.1% on a three-class flower dataset, outperforming currently popular flower classification models. In addition, ablation studies verify the effectiveness of combining channel attention mechanisms with convolutional neural networks, proving that RCS-net not only improves classification accuracy, but also optimises computational efficiency and model robustness. Future work will focus on extending the scope of the model, exploring optimisation strategies and improving generalisation capabilities to support a wider range of plant classification tasks, considering its potential application in resource-constrained environments.

Keywords: Flower classification · Transfer learning · Channel attention mechanism · Convolutional neural networks · Deep learning

1 Introduction

According to a 2016 report by the Royal Botanic Gardens, Kew, there are approximately 369,000 known species of flowering plants worldwide. This vast biodiversity makes the development of efficient and accurate flower classification systems both crucial and challenging [1]. Traditional classification methods rely on manual feature extraction and machine learning algorithms for image segmentation and classification. However, the diversity in morphology, colour distribution and lighting conditions among flowering

plants results in high species variability, making classification difficult. In addition, factors such as collection conditions, classification standards, category definitions and sample size also affect accuracy.

While manual classification remains feasible, it is hampered by human factors such as fatigue, cognitive bias and inefficiency, making it time consuming [2]. Therefore, accurate and automated flower classification technologies are essential in various fields, including content-based image retrieval and cataloguing [3], ecosystem monitoring and analysis [4], horticultural development [5], live plant identification [6], and educational materials [7]. These advances support botanical research, biodiversity conservation and environmental protection, and provide opportunities to improve classification using new computer technologies and data analysis methods [8–10].

With the rapid development of AI, especially transfer learning and attention mechanisms, significant progress has been made in flower recognition. Classical architectures such as ResNet, VGG, and EfficientNet, pre-trained on ImageNet, are widely used for transfer learning, achieving significant accuracy improvements over traditional machine learning methods [11]. Unlike traditional approaches that add parallel branches for feature extraction followed by manual techniques [12], deep learning models using transfer learning offer superior performance and computational efficiency.

This study presents RCS-net, a classification network that combines transfer learning with a channel attention mechanism. RCS-net starts with ResNet50, which is pre-trained on ImageNet and provides an optimised initial state for transfer learning. ResNet50 extracts basic flower features, while convolutional neural networks (CNNs) further refine the identification and classification of specific attributes. The integrated channel attention mechanism appropriately weights features across different channels, enhancing the network's ability to capture key information and suppress background noise. RCS-net improves classification accuracy, computational efficiency and robustness in various flower classification tasks.

2 Related Work

To achieve higher and more stable classification accuracy, the combination of transfer learning algorithms with deep neural networks has significantly advanced flower recognition. Wu et al. [13] proposed an efficient strategy using VGG-16, VGG-19, Inception-v3 and ResNet50 for network initialisation and transfer learning, achieving 96.57% accuracy on the OXFORD-102 dataset and 95.29% accuracy on the OXFORD-17 dataset. Kaya et al. [14] demonstrated that transfer learning improves the classification performance of plants, significantly improving poor performing models. Ghazi et al. [15] used GoogLeNet, AlexNet and VGGNet, fine-tuning pre-trained models and integrating classifiers to achieve 80% accuracy on the validation set, close to second place in the competition.

Attention mechanisms capture useful features according to the classification goals without changing the existing network architecture, assigning higher weights to critical information and improving performance. Fei et al. [16] optimised a ShuffleNet V2-based architecture with attention mechanisms and achieved 98.891% accuracy on an RGB flower dataset. Zhang et al. [17] embedded spatial and channel attention modules into

Xception and achieved 97.35% accuracy on the Oxford 102 flower dataset. Zhao et al. [18] introduced a Partial Priority Attention Mechanism (PPAM) with depth-separable convolution and label smoothing for accurate plant part identification.

Despite these advances, challenges remain. High model complexity and computational requirements limit deployment in resource-constrained environments. In addition, most research focuses on ideal conditions, with less attention paid to real-world variability. These issues highlight the need for more optimised and practical flower recognition models.

3 Methods

In the field of flower classification, Residual Networks (ResNet) have received widespread attention for their effective solution to the degradation problem encountered during the training of deep networks. However, despite ResNet's exemplary performance in various tasks, its potential for automatic learning and adjustment of feature weights remains underexplored. To this end, we introduce the Squeeze-and-Excitation (SE) module, an adaptive recalibration mechanism that focuses on channel relationships and is paired with convolutional neural networks to extract deeper features, further enhancing the network's representational capability.

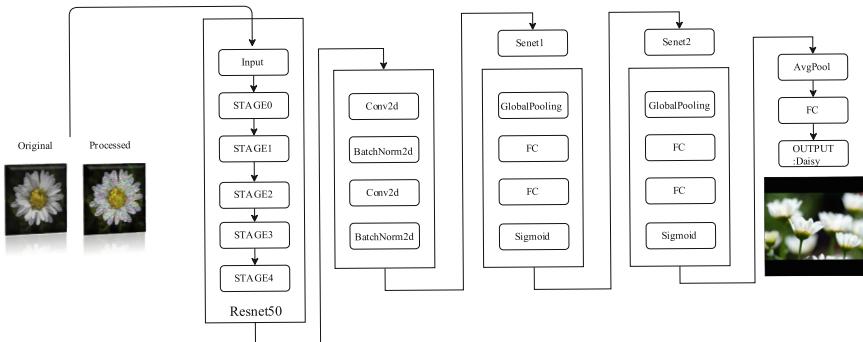


Fig. 1. RCS-net Network architecture.

Figure 1 depicts the network structure of RCS-net, a novel flower classification network that optimizes performance by incorporating neural network techniques. The architecture is based on a modified ResNet50 and employs channel attention mechanisms to enhance feature extraction.

Residual Networks: The architecture starts with a residual network based on the ResNet50 model, pre-trained on ImageNet, serving as the basic feature extractor. Residual connections mitigate the vanishing gradient problem and enable deeper network structures. These identity mappings within the residual blocks retain information from earlier layers, preserving critical features during propagation. The ResNet50 architecture progresses through stages STAGE0 to STAGE4, each comprising convolutional layers

and residual blocks. This hierarchical structure ensures feature extraction and robust information propagation, forming a strong foundation for accurate classification.

CNN and Channel Attention Module: To refine feature extraction, the model incorporates a Squeeze and Excitation (SE) layer for channel-wise attention. This layer includes an adaptive mean pooling layer that reduces the spatial dimensions of the input feature map to a single value per channel, capturing global information. The pooled features pass through a fully connected layer for dimensionality reduction, followed by ReLU activation and another fully connected layer. The final output is a sigmoid-activated attention map distributed across the input channels, modulating their responses.

Final Classification: The extracted features, after passing through the SE layer, are pooled again via an adaptive average pooling layer, reducing them to a fixed size 1×1 feature map. These features are then flattened and fed into a fully connected layer to produce the final classification logits for each class.

3.1 Residual Networks

The core idea of ResNet is the introduction of residual blocks, which allow the network to learn the residual (i.e. the difference) between inputs and outputs. Ideally, if a deep network can learn an identity mapping directly from the input data, then theoretically adding more layers to the network would not degrade its performance. The introduction of the residual module allows the network not to learn the target prediction directly, but to learn the difference between the target and the input, thus overcoming the difficulty of training deep networks. A residual module can be represented by the following formula:

$$y = F(x, \{W_i\}) + x \quad (1)$$

where x represents the input features, $F(x, \{W_i\})$ represents the residual function learned by a multi-layer convolutional network with weights $\{W_i\}$, and y is the output of the module. Here, $F(x, \{W_i\}) + x$ means that the goal of residual learning is to learn the difference or residual, $F(x, \{W_i\})$, between input x and output y , and then add this residual back to input x to form the final output y .

In certain cases, adding the input directly to the output can lead to a mismatch in dimensions. To solve this problem, ResNet introduced a shortcut connection, i.e. if the dimensions of the input and output do not match, a convolutional layer is used to adjust the dimension of the input:

$$y = F(x, \{W_i\}) + W_s x \quad (2)$$

Here, $W_s x$ represents the weight of the convolutional layer used to adjust the dimension.

By learning the residual between the input and output, rather than directly learning the mapping itself, the degradation problem encountered in training deep networks is successfully solved, thus achieving a breakthrough in deeper network configurations in the field of deep learning.

3.2 SENet

We introduced SE blocks to enhance certain features of flowers while ignoring other less important features. By concatenating and applying convolutional transformations

through the convolutional neural network, the SE blocks adaptively optimise the channels for flower classification. The design of the SE block is based on three core operations: Squeeze, Excitation and Scale, which enhance the network's ability to perceive important features.

The squeeze operation uses global average pooling (GAP) to capture the global contextual information of each channel in the feature map. For a given feature map $F \in R^{H \times W \times C}$ where H , W , and C represent the height, width and number of channels, respectively, resulting from the global average pooling, $z \in R^C$ is calculated as follows:

$$z_c = \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W F_{ijc} \quad (3)$$

The excitation operation is achieved by two fully connected (FC) layers. The first layer reduces the dimension to reduce the number of parameters and computational complexity, and then increases the dimension back to the original channel dimension. Let the initial dimension of the first FC layer be $\frac{C}{r}$, where r is the reduction ratio used to control model complexity and capacity. Let W_1 and W_2 are the weight matrices of the two FC layers, then the excitation function s can be defined as:

$$s = \sigma(W_2 \delta(W_1 z)) \quad (4)$$

where σ is the sigmoid activation function, δ is the ReLU activation function, and z is the output of the squeeze operation. The Sigmoid function compresses the output to between 0 and 1, which generates the importance weights for each channel.

Finally, the Scale operation applies the weights s from the Excitation operation to the original feature map, achieving adaptive recalibration of the feature channels. Specifically, given the original feature map F and the weights s from the excitation operation, the scaled feature map F'_c is calculated as follows

$$F'_c = s_c \cdot F_c \quad (5)$$

where c is the channel index, and \cdot denotes the element-wise multiplication.

Through this series of operations, SE blocks effectively enhance the model's focus on important features and suppress irrelevant information, thereby improving the accuracy of flower image classification. Furthermore, due to their lightweight design, SE blocks can be easily embedded into existing convolutional neural network architectures without imposing a significant computational burden.

3.3 Transfer Learning

Transfer learning aims to improve a target learner's learning in the target domain by transferring knowledge contained in different but related source domains. Thus, the construction of the target learning object can reduce the dependence on a large amount of target domain data. Due to its broad application prospects, transfer learning has become a hot and promising research direction in the field of machine learning.

A domain D is defined by a feature space X and a marginal probability distribution $P(X)$:

$$D = \{X, P(X)\} \quad (6)$$

where $X = \{x_1, x_2, x_3, \dots, x_n\}$ are instances in the feature space, each x_i is a n-dimensional feature vector, and $P(X)$ is the marginal probability distribution over X .

Task definition:

A task T is defined by a label space Y and a predictive function $f(\cdot)$:

$$T = \{Y, f(\cdot)\} \quad (7)$$

where Y is the set of all possible labels, and $f : X \rightarrow Y$ is the function learned by the learning algorithm from the training data, with the goal of mapping feature vectors to their corresponding labels.

Source domain and target domain data:

Source domain data D_S and target domain data D_T are each defined as

$$D_S = \{(x_{S1}, y_{S1}), (x_{S2}, y_{S2}), \dots, (x_{Sn}, y_{Sn})\} \quad (8)$$

$$D_T = \{(x_{T1}, y_{T1}), (x_{T2}, y_{T2}), \dots, (x_{Tn}, y_{Tn})\} \quad (9)$$

where x_{Si} and x_{Ti} are feature vectors in the source and target domains respectively, and y_{Si} and y_{Ti} are the corresponding labels.

Definition of transfer learning:

Transfer learning aims to improve the target prediction function $f_T(\cdot)$, with source domain data D_S and task T_S where $D_S \neq D_T$ or $T_S \neq T_T$. This means that although the source domain/task is different from the target domain/task, we still try to use the knowledge learned from the source domain/task to help the learning process in the target domain/task.

Homogeneous and heterogeneous transfer learning:

Homogeneous transfer learning, where $X_S \neq X_T$ focuses on knowledge transfer across domains under the same feature space and label space, whereas heterogeneous transfer learning, where $X_S = X_T$, faces the challenge of differences in feature space or label space, and explores how knowledge can be effectively transferred and adapted in these differences. In homogeneous transfer learning, the source and target domains share the same feature space.

The above formulas and concepts together form the mathematical framework of transfer learning, which allows algorithms to identify and use knowledge learned in one domain (source domain) to improve learning and prediction in another domain (target domain). This approach is particularly useful in situations where labelled data in the target domain is scarce or expensive to collect, enabling models to improve performance by using data and knowledge from different but related domains.

Algorithm 1 The RCSnet training algorithm

Require:

pretrained ResNet50 model M_{pre} , Learning rage α , Training dataset r_{train} , Test dataset r_{test} , Validation dataset r_{val} , Cross-Entropy Loss function L , Adam Optimizer O

Initialize:

- 1: Initialize the model M as RCSnet with weights from M_{pre} .
 - 2: For each training epoch e from 1 to E, perform the following:
 - a: Set M to training mode.
 - b: For each batch b of data (X_b, Y_b) , perform the following:
 - i: Compute the forward pass of M , $M(X_b)$.
 - ii: Calculate the loss $L(M(X_b), Y_b)$.
 - iii: Perform backpropagation, $\nabla_\theta L(M(X_b), Y_b)$.
 - iv: Update the weights θ of M using the optimizer O .
 - c: Compute and record the training loss L_{train} .
 - d: Evaluate M on the validation set and calculate the validation loss L_{val} .
 - e: Output the training loss and validation loss for epoch e.
 - f: Plot the graph of training and validation losses.
 - 3: Evaluate M on the test set and calculate performance metrics $P = \{\text{Accuracy}, \text{Precision}, \text{Recall}, \text{F1 Score}\}$.
 - 4: Output the performance metrics P on the test set.
-

This Algorithm 1 implements an image classification workflow utilizing a modified ResNet50 architecture with Squeeze-and-Excitation (SE) layers. The model M , initialized with pretrained weights from M_{pre} , undergoes training and validation over epochs E on dataset r. The dataset is partitioned into batches of images X_b and labels Y_b . Predictions are generated and compared with labels using a loss function L , triggering backpropagation to compute gradients $\nabla_\theta L$ and adjust weights θ via an optimizer O. Training loss L_{train} and validation loss L_{val} are tracked and reported. After training, M is evaluated on a test dataset, yielding metrics P including Accuracy, Precision, Recall and F1 Score. Finally, the model was used for flower classification.

4 Experiment

4.1 Baseline

The effectiveness of the proposed network is evaluated against several state-of-the-art methods, including classical recommendation models such as MobileNets, which achieved an accuracy rate of 95.5%, and DenseNet201-MLP with an accuracy rate of 94.47%. In addition, deep learning models such as FlowerConvNe and MobileNet CNN are evaluated, achieving accuracies of 95% and 95.93% respectively. However, existing methods in the field of flower recognition have problems such as weak generalisation ability, low computational efficiency and lack of adaptability. Therefore, our method aims to maintain high accuracy while improving model robustness and generalisation ability through further innovation and optimisation. Our approach deeply analyses the characteristics of flower image data and combines channel attention mechanisms and algorithm optimisation to achieve higher scientific precision and accuracy in flower classification tasks.

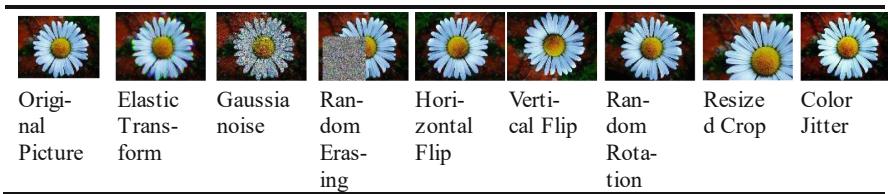
4.2 Datasets

The dataset, available on Kaggle, comprises 4242 flower images from sources like Flickr, Google Images, and Yandex Images, spanning five categories: daisies, tulips, roses, sunflowers, and dandelions, each with roughly 800 photos [19]. These images, which have a resolution of about 320×240 pixels and vary in aspect ratio, enhance the dataset's diversity. For classification purposes, the categories selected are daisy, rose, and tulip, with the dataset divided into training, validation, and test sets in a 7:2:1 ratio.

4.3 Data Preprocessing

As shown in Table 1, to simulate the randomness of real-world image capture and to effectively augment a small dataset, data augmentation strategies such as elastic transformations, Gaussian noise, random erasure, horizontal flipping, vertical flipping, random rotation, resized cropping, and colour jitter were applied to the images of the three flower categories in the dataset. Each flower category received an additional 80 enhanced images. These enhancement strategies aim to increase the diversity of the data, thereby improving the model's tolerance to image variation, distortion and noise, further enhancing the model's generalisation ability and performance stability in complex environments.

Table 1. Data augmentation strategies.



The study uses data enrichment strategies such as elastic transformations, Gaussian noise and various flipping techniques to address two main challenges: enriching small datasets by increasing sample size, and increasing model robustness by simulating real-world conditions. These methods are crucial for preventing overfitting in flower recognition and classification by enriching the diversity and volume of the data, thus improving the model's performance on novel or different images. In addition, this approach helps to effectively manage small datasets, increasing the applicability and stability of deep learning models in complex scenarios.

4.4 Evaluation Indicators

To assess the overall performance of the RCS-net model in detecting anomalies, we used accuracy, precision, recall, and F1 score as evaluation metrics:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FN + FP} \quad (10)$$

For flower classification tasks, accuracy provides a holistic measure of the overall performance of the model, reflecting its ability to correctly classify both anomalous and normal samples across different categories. This metric is particularly useful for assessing the overall effectiveness of a classification model.

$$\text{Precision} = \frac{TP}{TP + FP} \quad (11)$$

Precision is crucial in contexts where minimising false positives is important, and reflects the model's ability to accurately identify the correct flower category from those labelled as such. This metric is particularly useful in flower classification tasks, where misclassifying one type of flower as another (e.g. mistaking a tulip for a rose) can significantly reduce the effectiveness of the model in practical applications.

$$\text{Recall} = \frac{TP}{TP + FN} \quad (12)$$

Recall is particularly relevant when minimising false negatives is a priority, ensuring that the model captures as many true positives as possible. In flower classification, this metric ensures that the model effectively identifies a significant proportion of each flower type, avoiding under-representation in the classification process.

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (13)$$

This score reflects the model's ability to balance between minimising false positives and false negatives, making it an essential metric for evaluating overall performance. In the context of flower classification, the F1 score ensures that the model effectively balances between correctly classifying different flower types while minimising both false positives and false negatives.

Here TP (True Positives) indicates the number of correctly classified flower images for each category, TN (True Negatives) represents the correctly classified normal or non-anomalous samples, FP (False Positives) indicates images incorrectly classified as belonging to a flower category, and FN (False Negatives) indicates images belonging to a category but not correctly classified.

4.5 Experimental Results

In Table 2, we have compared the performance of the RCS-net model on the flower dataset with relevant research results from recent years. The results show that RCS-net outperforms the models proposed in recent years, even when the dataset is artificially noisy. Specifically, RCS-net integrates the channel attention mechanism with the convolutional neural network, and deeply extracts and optimises features from Resnet50. This method effectively captures key features in flower images, such as colour and shape, and significantly improves the model's recognition accuracy. With training over 8 epochs, RCS-net achieved a high accuracy of 96.1% on a three-class flower dataset. Furthermore, the comparative analysis highlights RCS-net's robustness and learning efficiency when dealing with noisy data, and emphasises the importance of combining channel attention

Table 2. Performance of different models.

Method	Accuracy (%)
MobileNet CNN [20]	89.84
FlowerConvNet [21]	95.00
DenseNet201-MLP [22]	94.47
MobileNets & Dynamic ensemble selection [23]	95.93
Ours	96.10

mechanisms with deep convolutional networks, providing an effective method for image classification on complex datasets in the future.

To evaluate the impact of integrating SENet with CNN structures on model performance, this study designed an ablation experiment where only the pre-trained ResNet50 model was retained as a baseline for comparison. The introduction of SENet and CNN into the model resulted in significant improvements across several performance metrics. As shown in Table 3, the incorporation of SENet and CNN resulted in an increase in accuracy from 94.81% to 96.10%, precision from 94.71% to 96.12%, recall from 94.67% to 96.27% and F1 score from 94.69% to 96.16% compared to the baseline model. These results clearly demonstrate that the combination of SENet and CNN can significantly improve the performance of the model in flower recognition tasks, especially in the accurate identification of fine features, thereby improving the overall performance of the model.

Table 3. Ablation experiment

Method	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
Baseline	94.81	94.71	94.67	94.69
+ SENet +CNN	96.10	96.12	96.27	96.16

In this study, the RCS-net model achieved an accuracy of 96.1% after eight training epochs, demonstrating its ability to learn key features from the flower dataset, such as colour and shape. However, as training progresses, the model may begin to show signs of overfitting, characterised by continued improvement in performance on the training set, while performance on the validation or test set stagnates or even declines. The risk of overfitting lies in the model's potential reliance on specific details in the training set that may not represent the broader data distribution. Implementing strategies such as early stopping and regularisation can significantly mitigate this effect and ensure that the model retains robust generalisation capabilities on new data. Future research will focus on exploring ways to balance model complexity and generalisation capacity, which will be key to improving performance in image classification tasks.

5 Summary

This research proposes a novel flower classification network architecture, RCS-net, which combines transfer learning with channel attention mechanisms. By integrating the ResNet50 model pre-trained on the ImageNet dataset, RCS-net optimises the initial state and training basis, and significantly enhances the network's ability to capture critical flower information through an integrated channel attention mechanism module. This approach effectively improves classification accuracy and suppresses background noise and irrelevant information. Experimental results show that RCS-net achieves superior classification performance on complex flower datasets augmented with data augmentation techniques, achieving an accuracy of 96.1%, outperforming existing flower classification models. Furthermore, ablation experiments confirm the effectiveness of combining channel attention mechanisms with convolutional neural networks, highlighting the potential application of deep learning technology in accurate flower recognition tasks.

Future research will aim to broaden the application scope of RCS-net, extending it to the classification and recognition of a wider variety of plants to support biodiversity research and environmental conservation efforts. Given computational resource constraints, future work will explore more efficient model optimisation strategies and lightweight network designs for rapid deployment in resource-constrained environments. In addition, investigating the model's adaptability and generalisation capabilities in uncontrolled environments, and effectively addressing challenges posed by image acquisition conditions, classification standards and category definitions will be critical directions for future research.

References

1. Bae, K.I., Park, J., Lee, J., et al.: Flower classification with modified multimodal convolutional neural networks. *Expert Syst. Appl.* **159**, 113455 (2020)
2. Hiary, H., Saadeh, H., Saadeh, M., et al.: Flower classification using deep convolutional neural networks. *IET Comput. Vis.* **12**(6), 855–862 (2018)
3. Deepa, S.N., Rasi, D.: FHGSO: flower Henry gas solubility optimization integrated deep convolutional neural network for image classification. *Appl. Intell.* **53**(6), 7278–7297 (2023)
4. Kolarik, N.E., Roopsind, A., Pickens, A., et al.: A satellite-based monitoring system for quantifying surface water and mesic vegetation dynamics in a semi-arid region. *Ecol. Ind.* **147**, 109965 (2023)
5. Kumar, M., Chaudhary, V., Sirohi, U., et al.: Economically viable flower drying techniques to sustain flower industry amid COVID-19 pandemic. *Environ. Dev. Sustain.*, 1–46 (2023)
6. Liu, S., Huang, Y., Duan, Y., et al.: Volatile/semi-volatile metabolites profiling in living vegetables via a novel covalent triazine framework based solid-phase microextraction fiber coupled with GC-QTOF-MS. *Food Chem.* **430**, 137064 (2024)
7. Alfano, P.D., Pastore, V.P., Rosasco, L., et al.: Top-tuning: a study on transfer learning for an efficient alternative to fine-tuning for image classification with fast kernel methods. *Image Vis. Comput.* **142**, 104894 (2024)
8. Jiang, Y., Li, C.: Convolutional neural networks for image-based high-throughput plant phenotyping: a review. *Plant Phenomics* (2020)
9. Tyllianakis, E., Martin-Ortega, J.: Agri-environmental schemes for biodiversity and environmental protection: how we are not yet “hitting the right keys.” *Land Use Policy* **109**, 105620 (2021)

10. Corlett, R.T.: Safeguarding our future by protecting biodiversity. *Plant Divers.* **42**(4), 221–228 (2020)
11. Talukder, M.A., Layek, M.A., Kazi, M., et al.: Empowering covid-19 detection: optimizing performance through fine-tuned efficientnet deep learning architecture. *Comput. Biol. Med.* **168**, 107789 (2024)
12. Kumar, A., Sachar, S.: Deep learning techniques in leaf image segmentation and leaf species classification: a survey. *Wirel. Pers. Commun.*, 1–32 (2024)
13. Wu, Y., Qin, X., Pan, Y., et al.: Convolution neural network based transfer learning for classification of flowers. In: 2018 IEEE 3rd International Conference on Signal and Image Processing (ICSIP), pp. 562–566. IEEE (2018)
14. Kaya, A., Keceli, A.S., Catal, C., et al.: Analysis of transfer learning for deep neural network based plant classification models. *Comput. Electron. Agric.* **158**, 20–29 (2019)
15. Ghazi, M.M., Yanikoglu, B., Aptoula, E.: Plant identification using deep neural networks via optimization of transfer learning parameters. *Neurocomputing* **235**, 228–235 (2017)
16. Fei, Y., Li, Z., Zhu, T., et al.: A lightweight attention-based convolutional neural networks for fresh-cut flower classification. *IEEE Access* **11**, 17283–17293 (2023)
17. Zhang, M., Su, H., Wen, J.: Classification of flower image based on attention mechanism and multi-loss attention network. *Comput. Commun.* **179**, 307–317 (2021)
18. Zhao, Y., Sun, Z., Tian, E., et al.: A CNN model for herb identification based on part priority attention mechanism. In: 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 2565–2571. IEEE (2020)
19. Flowers Dataset. <https://www.kaggle.com/datasets/imsparsiflowers-dataset/>. Accessed 21 Aug 2023
20. Venkatesh, V.: Fine-tuned MobileNet classifier for classification of strawberry and cherry fruit types. *J. Comput. Sci.* **17**(1), 44–54 (2021)
21. Rabbi, M.F.: An ensemble-based deep learning model for multi-class flower recognition. In: 2023 International Conference on Next-Generation Computing, IoT and Machine Learning (NCIM), pp. 1–6. IEEE (2023)
22. Shee, J.X., Lim, K.M.: Flower species recognition using DenseNet201 and multilayer perceptron. In: 2023 11th International Conference on Information and Communication Technology (ICoICT), pp. 307–312. IEEE (2023)
23. Wang, Z.: Dynamic ensemble selection of convolutional neural networks and its application in flower classification. *Int. J. Agric. Biol. Eng.* **15**(1), 216–223 (2022)



MDGCL: Message Dropout Graph Contrastive Learning for Recommendation

Qijia Xu, Wei Li^(✉), and Jingxin Chen

School of Software Engineering, Southeast University, Nanjing, China
liwei@seu.edu.cn

Abstract. In the realm of recommendation systems, the representation learning of users and items plays a pivotal role. The advent of Graph Neural Networks (GNN) has propelled Graph Collaborative Filtering (GCF) to new heights by effectively capturing high-order connectivities. However, the persistent challenge of matrix sparsity in collaborative filtering continues to impede optimal performance enhancement. While Graph Contrastive Learning (GCL) has emerged as a solution to mitigate data sparsity by extracting general features from raw data, its conventional methods, involving node and edge dropout for graph augmentation, often result in the loss of crucial graph information, thereby diminishing model performance. Addressing this, we introduce a novel technique termed Message-Dropout, implemented during the message passing phase. This approach offers a more refined granularity and ensures the retention of key information within the graph. Additionally, our Graph Contrastive Learning (GCL) method uniquely eliminates the need for pre-graph augmentation and avoids the requirement of two additional forward trainings in each mini-batch. We name our proposed method MDGCL. Experiments on three public datasets have proven the effectiveness of our proposed model. The code is released at <https://github.com/anorepo/MDGCL>.

Keywords: Graph Collaborative Filtering · Contrastive Learning · MessageDropout

1 Introduction

To address the issue of information overload on the internet, recommendation systems are widely used for personalized information filtering [1]. Collaborative filtering [2–4] is a fundamental paradigm in recommendation systems, generating personalized recommendations through implicit feedback from users and items. Recently, graph collaborative filtering (GCF) [5–7], by aggregating neighborhood information and utilizing high-order links in user-item interactions, has achieved state-of-the-art recommendation performance. Nonetheless, it is frequently observed that bipartite graphs delineating user-item interactions are characteristically marked by significant sparsity [9, 10], which consequently precipitates a diminution in the efficacy of recommendations. Furthermore, the interactions that are recorded often embody a degree of noise. For instance, a user might inadvertently engage with an item due to misleading cues only to discover its lack

of appeal upon utilization [12]. The mechanism of neighborhood aggregation employed by Graph Convolutional Networks (GCNs) serves to accentuate the impact of such interactions on the process of representation learning, thereby rendering it increasingly vulnerable to the perturbations introduced by noisy interactions.

Due to its ability to extract general features from a large amount of unlabeled data and to standardize representations in a self-supervised manner, Contrastive Learning (CL) has made significant progress in multiple research fields [12–16]. As CL does not require prior labeling of data, it effectively addresses the data sparsity problem and noise in interactions within recommendation systems. The role of CL tasks as auxiliary tasks in recommendation systems has been proven effective by multiple studies [9, 17–19]. A conventional approach [17] to incorporating Contrastive Learning (CL) within recommendation systems involves initially augmenting the user-item bipartite graph via structural modifications. Subsequently, the objective is to optimize the consistency between representations generated by graph encoders when exposed to disparate views. In this way, the CL task acts as an auxiliary task and is jointly optimized with the recommendation task. Although effective, the commonly used current graph augmentation strategies in graph Contrastive Learning (GCL) may hinder further performance improvements. For example, the notable method SGL, which integrates Contrastive Learning (CL) with Graph Convolutional Filtering (GCF), necessitates alterations in the graph’s architecture to produce CL sample pairs. This process can result in the inadvertent omission of critical edge data, consequently diminishing the representational efficacy of the generated embeddings.

In the MP-GNN [20] paradigm, the message propagation process implicitly forms a message propagation matrix. We attempt to apply dropout to this, termed as MessageDropout. Previous work [17] extensively used NodeDropout and EdgeDropout, both of which can lead to the loss of important structural information in the graph, affecting the final performance of the model. Upon careful analysis, we propose MessageDropout, which reduces the granularity of the masked information. It does not discard the information of an entire edge during the message propagation process, thereby significantly mitigating the impact of dropping important nodes and edges on performance. Moreover, the current mainstream GCL architectures [18] are somewhat cumbersome, requiring two additional forward propagations for contrastive learning within each mini-batch and involving all nodes in the graph, thus significantly increasing training time. We employ MessageDropout for graph data augmentation in graph contrastive learning, constructing the InfoNCE [21] loss using layer embedding and final embedding during the forward propagation process. This approach avoids the need for two additional forward trainings in each mini-batch, reducing a considerable amount of computational cost.

It is pertinent to highlight that the enhancements we propose can be universally applied across all models that operate within the framework of Message Passing Graph Neural Networks (MP-GNN). In this context, we have applied our improvements to the straightforward yet potent model, LightGCN [6]. Empirical investigations conducted across three benchmark datasets furnish evidence of the effectiveness of these two enhancements. The main contributions of this paper are as follows:

1. We proposed MDGCL, a CL-based GCF model that uses MessageDropout for graph augmentation, substantially mitigating the impact of dropping important information on model performance.
2. Our proposed method constructs a self-supervised task by contrasting layer embedding and final embedding. This approach avoids the need for two additional forward trainings, avoiding two additional forward computations within each mini-batch.
3. Comprehensive experiments conducted on three datasets reveal that our method consistently outperforms numerous competitive baselines, exhibiting a faster convergence rate.

2 Preliminaries

2.1 Graph Collaborative Filtering

In the domain of Graph Neural Network (GNN)-based Collaborative Filtering (CF), the user-item interaction matrix is structured as a bipartite graph, symbolized by $G = (V, E)$. Here, $V = \{U \cup I\}$ represents the set of nodes, encompassing all users and items, while $E = \{(u, i) \mid u \in U, i \in I, Y_{u,i} = 1\}$ constitutes the set of edges, indicating the interactions between users and items. The foundational element of Graph Collaborative Filtering (GCF) is the Graph Convolutional Network (GCN) [22], which is primarily concerned with integrating higher-order connectivity information into the representations of target nodes. This is achieved through multiple iterations of graph convolution operations. We focus on the graph convolution approach as employed in the backbone model LightGCN [6], a predominant framework in GCF research. As an illustration, consider a user u . The essential computation in LightGCN is outlined as follows:

$$\mathbf{z}_u^{(l)} = f_{\text{combine}}\left(\mathbf{z}_u^{(l-1)}, f_{\text{aggregate}}\left(\left\{\mathbf{z}_i^{(l-1)} \mid i \in \mathcal{N}_u\right\}\right)\right), \quad (1)$$

where $\mathbf{z}_u^{(l)}$ represents the embedding of user u at the l -th layer, and $\mathbf{z}_u^{(l-1)}$ signifies the embedding of the same user at the preceding layer, specifically layer $l-1$. The term $\mathbf{z}_i^{(l-1)}$ refers to the embedding of the $l-1$ th order neighbor of user u . The initial representation of the user, or the self-node representation, is denoted by $\mathbf{z}_u^{(0)}$. The function $f_{\text{aggregate}}(\cdot)$ is employed as the aggregation function, tasked with processing the embeddings of the $l-1$ th order neighbors of user u . Additionally, the function $f_{\text{combine}}(\cdot)$ serves as the combine function, which integrates the $l-1$ th order embeddings with their neighboring information to yield the embedding of the l layer. Following the acquisition of l layer embeddings, which encompass information across different orders, these embeddings are collectively aggregated to formulate the final embedding:

$$\mathbf{z}_u = f_{\text{readout}}\left(\mathbf{z}_u^{(0)}, \mathbf{z}_u^{(1)}, \dots, \mathbf{z}_u^{(l)}\right), \quad (2)$$

represents the readout function used to compute the final embedding, typically implemented through a weighted sum approach. The probability of user u interacting with item i is calculated based on the inner product of their respective embeddings:

$$\hat{y}_{u,i} = \mathbf{z}_u^T \mathbf{z}_i, \quad (3)$$

Conventional techniques for optimizing model parameters typically approach the task as a form of supervised learning. Here, the supervision signal is derived from observed interactions, which also constitute the edges of the graph G . For example, these strategies aim to reduce the variance between the estimated value \hat{y}_{ui} and the actual observed value y_{ui} , frequently by incorporating negative samples drawn from data that has not been observed. Beyond this point-wise learning paradigm, another widely adopted approach is the pairwise Bayesian Personalized Ranking (BPR) loss. This method prioritizes ensuring that the model's prediction for an observed interaction is rated higher than that for its unobserved counterparts:

$$\mathcal{L}_{bpr} = \sum_{u,i,j \in Y} -\log \left(\sigma \left(\hat{y}_{u,i} - \hat{y}_{u,j} \right) \right), \quad (4)$$

where $Y = \{(u, i, j) | (u, i) \in E^+, (u, j) \in E^- \}$ is the training data, and $E^- = \mathcal{U} \times I \setminus E^+$ is the unobserved interactions.

2.2 Graph Contrastive Learning

Graph Contrastive Learning (GCL) typically utilizes data enhancement techniques to create contrasting perspectives for each sample. In this approach, GCL treats different views of the same node as positive pairs and views from different nodes as negative pairs. The auxiliary supervision from positive pairs enhances consistency between different views of the same node for prediction purposes, while the supervision from negative pairs helps distinguish among different nodes. Formally, the contrastive loss, specifically InfoNCE [21], is utilized to enhance the concordance of positive pairs while reducing the similarity of negative pairs:

$$\mathcal{L}_{ssl}^{user} = \sum_{u \in \mathcal{U}} -\log \frac{\exp \left(\frac{s(\mathbf{z}'_u, \mathbf{z}''_u)}{\tau} \right)}{\sum_{v \in \mathcal{U}} \exp \left(\frac{s(\mathbf{z}'_u, \mathbf{z}''_v)}{\tau} \right)}, \quad (5)$$

where $s(\cdot)$ represents the cosine similarity function between two vectors, and τ is a hyperparameter known as the temperature in softmax. Similarly, the contrastive loss for the user side, \mathcal{L}_{ssl}^{item} can be obtained. By combining these two losses, we derive the objective function for the self-supervised task as:

$$\mathcal{L}_{ssl} = \mathcal{L}_{ssl}^{user} + \mathcal{L}_{ssl}^{item}. \quad (6)$$

3 Methodology

In this section, we build upon the existing graph-augmentation-based SGL [17] paradigm, proposing an enhanced message-passing method termed MessageDropout. This can be understood as a more granular form of graph augmentation. Moreover, inspired by XSimGCL [9], we abandon the commonly used framework of contrastive

learning. Instead, we use message dropout for graph data augmentation in graph contrastive learning. During the forward propagation process, we construct the InfoNCE [21] loss using both layer embeddings and final embeddings, thereby avoiding the need for two additional forward trainings in each mini-batch and significantly reducing computational costs. We provide a detailed analysis of these two improvements below.

3.1 Analysis of MessageDropout

Dropout on graph structure has been shown to improve the performance of GNNs [23]. However, we argue that the conventional dropout methods, such as node dropout and edge dropout, are relatively coarse. NodeDropout may lead to the loss of a node and several edges, while EdgeDropout can cause the loss of critical edges, leading to performance degradation. Figure 1 illustrates this phenomenon, showing that node dropout, in particular, results in the loss of multiple edges, thereby significantly increasing the risk of missing crucial information. This observation leads us to conclude that the granularity of these dropout methods is excessively coarse, resulting in suboptimal performance.

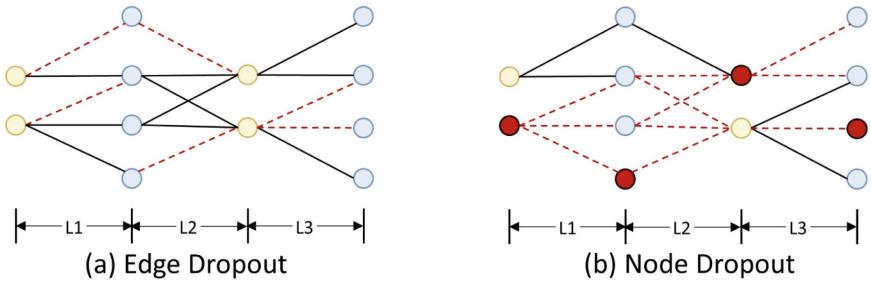


Fig. 1. A toy example of higher-order connectivity in a three-layer GCN model with Edge Dropout (left) and Node Dropout (right). The red nodes/edges represent potential critical nodes/edges, and both NodeDropout and EdgeDropout may lead to this situation. (Color figure online)

To address these issues, we introduce a method of MessageDropout. Most GNNs implementations follow the MP-GNN paradigm [20]. For a node u , the message passed to it by a neighboring node v at layer t can be represented as:

$$\mathbf{M}_{(u,v)}^{(t)} = \phi^{(t)}\left(h_u^{(t)}, h_v^{(t)}, e_{u,i}^{(t)}\right), \quad (7)$$

where $\phi^{(t)}$ represents the message functions, and $h_u^{(t)}$ denotes the representation of node u at layer t . The message passing matrix for node u is given by:

$$\mathbf{M}_u^{(t)} = (m_1, m_2, \dots, m_{\mathcal{N}(u)}), \quad (8)$$

where m_i represents receiving messages from its neighboring nodes. MessageDropout is equivalent to dropping elements from $\mathbf{M}_{i,j}$ with a probability of δ . The message matrix after dropout is given by:

$$M_{i,j}' = P_{i,j} M_{i,j}, \quad (9)$$

where \mathbf{P} (with $\mathbf{P} \in \mathbb{R}^{\mathcal{N}(u) \times k}$, k is the dimension of message features) is the mask matrix. The elements of $P_{i,j}$ follow a Bernoulli distribution ($P_{i,j} \sim \text{Bernoulli}(1 - \delta)$).

Figure 2 shows the differences between MessageDropout, NodeDropout, and EdgeDropout. MessageDropout offers the highest flexibility and finest granularity. Notably, NodeDropout affects multiple message matrices as it causes the loss of multiple edges, indicating its coarser granularity. From a distributional perspective, dropout can be interpreted as a sampling process with $M_{i,j}$ as the fundamental unit. The reason why DropMessage exhibits the smallest sample variance lies in its being the most fine-grained stochastic dropping approach within the GNN (Graph Neural Network) models. When applying DropMessage, each element, $M_{i,j}$ is independently evaluated to determine whether it should be masked. SimGCL [18] demonstrates that uniform distribution is crucial for performance enhancement, making MessageDropout more suitable for graph augmentation.

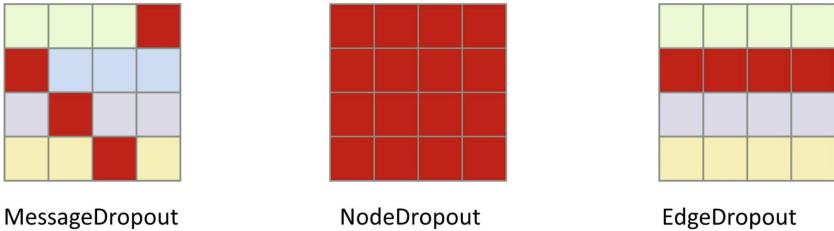


Fig. 2. Overview of a node’s message matrix. This represents the message matrix of a certain node. The red boxes indicate the vectors that are dropped, intuitively showing the effects of MessageDropout, NodeDropout, and EdgeDropout on the message propagation matrix. MessageDropout leads to sparse and uniformly distributed drops. EdgeDropout results in entire rows missing from the message matrix, while NodeDropout causes the greatest loss of information. (Color figure online)

In practice, we do not need to explicitly construct a complete message matrix. As can be seen from Fig. 2, each row in the message matrix represents an edge in the user-item graph, and MessageDropout can be independently applied to each edge, allowing for parallel execution. This property ensures that it does not affect the performance of the model.

3.2 A Simplified Architecture for GCL

To solve the issue in SGL of a mini-batch undergoing two additional forward training processes [17, 18]. In Fig. 3, we demonstrate the original graph contrastive learning architecture, clearly showing the phenomenon that requires two additional forward training processes.

Inspired by XSimGCL [9], we combine MessageDropout with GCL learning. This is achieved by simplifying the GCL architecture through contrasting layer embedding and

final embedding. This approach avoids the need for two additional forward trainings, as illustrated in Fig. 4.

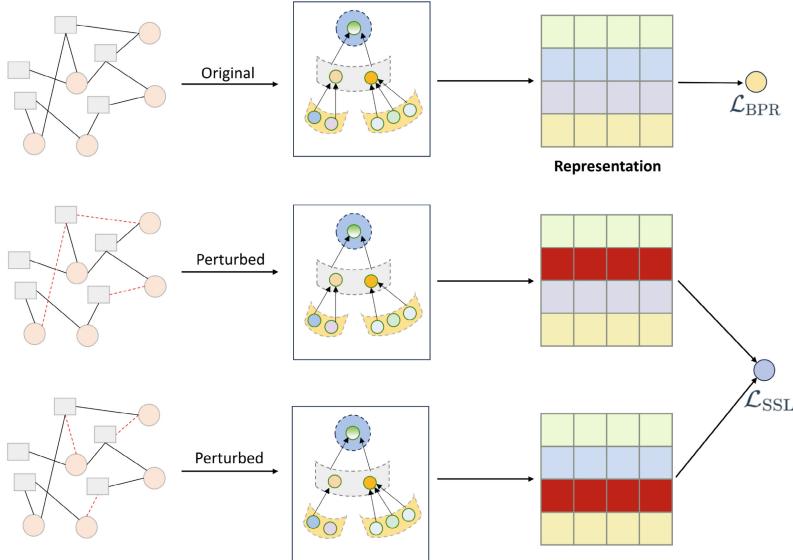


Fig. 3. The overall system framework for graph contrastive learning (illustrated with edge dropout, where red indicates the edges that are dropped). The upper layer displays the workflow of the main supervised learning tasks, while the lower layer illustrates the workflow of the SSL (Self-Supervised Learning) tasks, with an enhancement of the graph structure. (Color figure online)

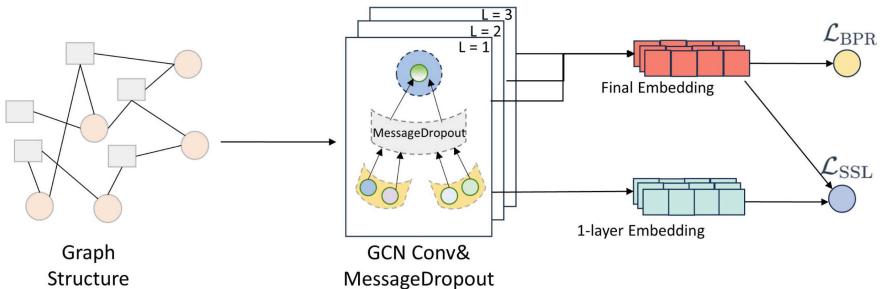


Fig. 4. The architecture of MDGCL avoids the need for two additional forward propagations.

The embeddings of different layers share some common information, and since each MessageDropout is random, the differences between these embeddings are often greater than those obtained from traditional twice graph augmentation. This is in line with the perspective presented in [24] that the mutual information between correlated views should be neither too high nor too low. We employ a multi-task training strategy

to jointly optimize the recommendation task (Eq. 4) and the self-supervised task (Eq. 6):

$$L = \mathcal{L}_{bpr} + \lambda_1 \mathcal{L}_{ssl} + \lambda_2 \|\Theta\|_2^2, \quad (10)$$

where Θ denotes the parameter set of the model, and λ_1 and λ_2 are hyperparameters that control the strengths of SSL and L2 regularization, respectively.

4 Experiments

4.1 Experimental Settings

Datasets and Evaluation Protocols

Consistent with NGCF [5], our experiments are conducted on three widely recognized benchmark datasets for recommendation:

- Yelp2018, Comprising 31,669 users, 38,049 items, and 1,885,553 interactions, this dataset is amassed from the rating interactions on the Yelp platform.
- Amazon-book, Encompassing 52,644 users, 91,600 items, and 3,587,486 interactions, it consists of users' ratings on books gathered from Amazon.
- Gowalla, With 29,859 users, 40,982 items, and 1,244,612 interactions, this dataset incorporates users' check-in records collected from the Gowalla platform.

We divided each dataset into training, validation, and testing subsets following an 8:1:1 ratio, respectively. For users included in the testing subset, we adhered to the all-ranking protocol [5] to assess the performance of top-K recommendations, with K being set at 20. The evaluation metrics reported include the average Recall@ K and NDCG@ K.

Baseline Methods

We benchmark our proposed MDGCL against the following CF models:

- LightGCN [6], which simplifies GCN based on NGCF [5] by omitting nonlinear activation and transformation, thereby enhancing performance.
- MultVAE [25], an item-based collaborative filtering method that employs a variational autoencoder (VAE). It is optimized using an additional reconstruction objective and can be seen as a special case of self-supervised learning (SSL).
- DirectAU [26], integrating self-supervised contrastive learning (CL) into LightGCN and introduces three enhancement methods for generating comparative data.
- SGL [17], integrating self-supervised contrastive learning based on LightGCN and proposing three enhancement operators for generating comparative data.
- SimGCL [18] does not use a graph enhancement techniques. Instead, it introduces uniform noise into the latent space to generate augmented view pairs.

Moreover, to demonstrate the effectiveness of our proposed MessageDrop (MDGCL) approach in comparison to EdgeDropout and NodeDropout, we compare MDGCL with EDGCL (based on EdgeDropout) and NDGCL (based on NodeDropout).

Implementation Details

For a fair comparison, all models are optimized using the Adam optimizer with a learning rate of $1e-3$, and the training batch size is set to 2048. The embedding size or hidden dimensions for all models are fixed at 64, utilizing the Xavier distribution. For CF models based on LightGCN, we perform 3-layer propagation. Specifically, the weight of L2 regularization is optimized within the range $\{1e^{-3}, 1e^{-4}, 1e^{-5}\}$, the weight of τ is optimized within the range $\{0.1, 0.2, \dots, 0.9, 1.0\}$, and dropout ratios were selected from $\{0.1, 0.2, \dots, 0.9, 1.0\}$. All experiments and evaluations are uniformly conducted under the open-source framework Recbole [27].

Table 1. Performance comparison of different methods on three datasets. Improv (Baseline) indicates performance improvement relative to the baseline model. Improv (Variants) indicates performance improvement relative to the two variants.

Dataset	Yelp2018		Amazon-Book		Gowalla	
Method	Recall	NGCG	Recall	NGCG	Recall	NGCG
LightGCN	0.0629	0.0516	0.0396	0.0308	0.1471	0.1271
MultVAE	0.0612	0.0498	0.0400	0.0306	0.154	0.1232
DirectAU	<u>0.067</u>	<u>0.0555</u>	0.0397	0.0318	0.1614	0.1379
SGL	0.065	0.0531	0.0467	0.0369	0.161	<u>0.1381</u>
SimGCL	0.0638	0.0523	0.0491	0.0391	0.1558	0.1341
MDGCL	0.0701	0.0579	0.052	0.0417	0.1708	0.1393
EDGCL	0.0611	0.0505	0.0429	0.0341	0.1507	0.1274
NDGCL	0.0568	0.0469	<u>0.0502</u>	<u>0.0404</u>	<u>0.1678</u>	0.1371
Improved (Baseline)	4.63%	4.32%	5.91%	6.65%	5.82%	0.87%
Improved (Variants)	14.73%	14.65%	3.59%	3.22%	1.79%	1.60%

4.2 Experimental Results

Performance Comparison

Table 1 compares the performance of our proposed MDGCL with baseline models on three datasets. The bold and underlined values indicate the optimal and suboptimal performances, respectively. Analysis of the experimental results reveals several observations:

- LightGCN consistently outperforms MultVAE, indicating that GNNs have stronger modeling capabilities for high-order connections.
- The contrastive learning methods SGL and SimGCL, both employing LightGCN as their backbone, have shown to consistently outshine LightGCN itself. This underlines the contribution of Self-Supervised Learning (SSL) in boosting the efficacy of recommendation systems.

- DirectAU achieves sub-optimal performance on some datasets, indicating that directly optimizing alignment and uniformity is effective. This can potentially serve as a direction for future work.
- In the Gowalla dataset, our model shows little improvement in NDCG but a significant increase in Recall. We speculate that this is because MDGCL uncovers some long-tail items, which are not ranked high, resulting in a modest increase in NDCG.
- Finally, we observe that MDGCL consistently outperforms all baselines as well as the EDGCF and NDGCF variants. This finding further demonstrates the effectiveness of MessageDropout from an experimental perspective. We attribute this success to its refined ability to enrich graph representations with finer granularity, which crucially minimizes the loss of vital information.

Comparison of Training Efficiency

Figure 5 illustrates the training speed of the compared methods utilizing three layers. The data presented in the graph were collected from a device equipped with a GeForce RTX 4070TI GPU. Based on Fig. 5, we have the following observations:

- Due to the simplicity of LightGCN’s structure, it has the shortest training time per epoch. MDGCL requires only one forward pass per mini-batch and has an average training time 1.5 times that of LightGCN across three datasets. SimGCL demonstrates higher efficiency compared to SGL, as it constructs contrastive information by simply adding uniform noise. SGL requires the longest training time, needing two graph augmentations, and most of this computation is done on the CPU, making it even 3.1 times slower than LightGCN on the Amazon-Book dataset.
- MDGCL reached its peak performance on the Yelp2018, Amazon-book, and Gowalla datasets respectively at the 35th, 49th, and 30th epochs, surpassing the convergence speeds of SGL and SimGCL. We attribute this to the cross-layer contrastive architecture used. All CL-based methods demonstrated an advantage over LightGCN in terms of convergence speed. [17] speculated that the multiple negatives in the CL loss may contribute to the fast convergence. While the other three methods began to overfit, LightGCN was still hundreds of epochs away from convergence.

The observations demonstrate that our model exhibits significantly faster performance in single epoch training times, compared to models that require pre-graph enhancement. Additionally, compared to other Graph Contrastive Learning (GCL) based models, our model displays a notably quicker convergence rate.

Impact of the MessageDropout Ratio

In our model, the dropout ratio is an important parameter. With the layer set to 3 and temperature set to 0.2, on the Yelp dataset and amazon-book dataset, we adjusted the parameter within the range of $[0.1, 0.2, \dots, 1.0]$ to examine the effect of the dropout ratio on performance. The findings are presented in Fig. 6. A MessageDropout ratio in the range of $[0.3-0.5]$ demonstrates good performance. This aligns with our expectations, as the granularity of MessageDropout is relatively small. Even when the MessageDropout ratio reaches 0.5, the model still performs well on these two datasets. Only when the majority of the messages are dropped does the model’s performance deteriorate.

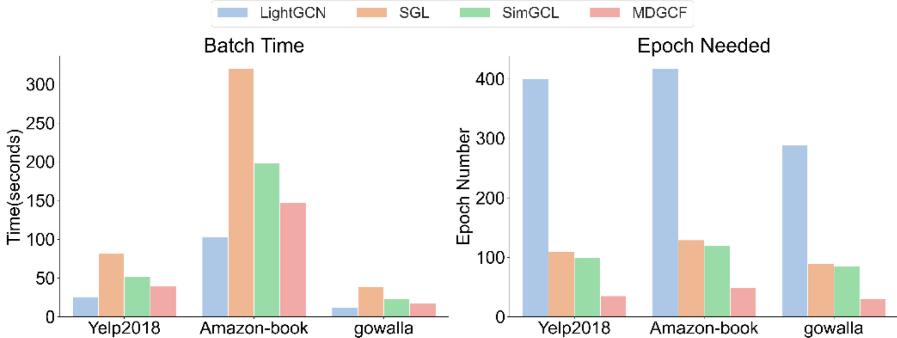


Fig. 5. The training speed of compared methods.

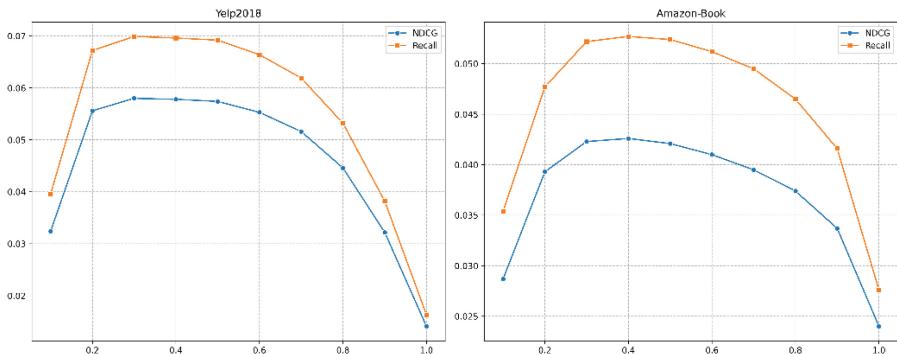


Fig. 6. Comparison of performance of different MessageDropout ratio on the Yelp 2018 and Amazon Book datasets.

5 Conclusion

In this paper, we propose a novel GCL model based on MP-GNN, utilizing MessageDropout, named MDGCL. We analyze its advantages over NodeDropout and EdgeDropout, highlighting its finer granularity and greater applicability. Unlike traditional CL methods, we contrast layer embedding and final embedding. This approach avoids the need for two additional forward trainings, thus avoiding the need for a mini-batch to undergo two additional forward training processes, effectively saving training time. Moreover, compared to other contrastive learning methods, MDGCL exhibits a faster convergence rate. Extensive experiments conducted on three highly sparse datasets demonstrate the effectiveness of MDGCL.

References

1. Covington, P., et al.: Deep neural networks for Youtube recommendations. In: RecSys (2016)
2. Ebisu, T., et al.: Collaborative memory network for recommendation systems. In: SIGIR (2018)
3. He, X., et al.: Fast matrix factorization with nonuniform weights on missing data. In: TNNLS, vol. 31 (2018)
4. He, X., et al.: Neural collaborative filtering. In: WWW (2017)
5. Wang, X., et al.: Neural graph collaborative filtering. In: SIGIR (2019)
6. He, X., et al.: LightGCN: simplifying and powering graph convolution network for recommendation. In: SIGIR (2020)
7. Ying, R., et al.: Graph convolutional neural networks for web-scale recommender systems. In: KDD, pp. 974–983 (2018)
8. Su, C., et al.: Graph convolutional matrix completion via relation reconstruction. In: ICSA (2021)
9. Yu, J., et al.: XSimGCL: towards extremely simple graph contrastive learning for recommendation (2023)
10. Lin, X., et al.: Task-adaptive neural process for user cold-start recommendation. In: WWW (2021)
11. Wang, W., et al.: Denoising implicit feedback for recommendation. In: WSDM 2021 (2021)
12. Park, S., et al.: Fair contrastive learning for facial attribute classification. In: CVPR (2022)
13. Khosla, P., et al.: Supervised contrastive learning (2021)
14. Gao, T., et al.: SimCSE: simple contrastive learning of sentence embeddings (2022)
15. Chen, T., et al.: A simple framework for contrastive learning of visual representations. In: ICML (2020)
16. Giorgi, J., et al.: DeCLUTR: deep contrastive learning for unsupervised textual representations. arXiv, vol. abs/2006.03659 (2020)
17. Wu, J., et al.: Self-supervised graph learning for recommendation. In: SIGIR (2021)
18. Yu, J., et al.: Are graph augmentations necessary? Simple graph contrastive learning for recommendation. In: SIGIR (2022)
19. Lin, Z., et al.: Improving graph collaborative filtering with neighborhood-enriched contrastive learning. In: WWW (2022)
20. Gilmer, J., et al.: Neural message passing for quantum chemistry. In: ICML (2017)
21. van den Oord, A., et al.: Representation learning with contrastive predictive coding. arXiv, vol. abs/1807.03748 (2018)
22. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: ICLR (2017)
23. Shu, J., et al.: Understanding dropout for graph neural networks. In: WWW (2022)
24. Tian, Y., et al.: What makes for good views for contrastive learning? In: NIPS (2020)
25. Liang, D., et al.: Variational autoencoders for collaborative filtering. In: WWW (2018)
26. Wang, C., et al.: Towards representation alignment and uniformity in collaborative filtering. In: KDD (2022)
27. Zhao, W.X., et al.: RecBole: towards a unified, comprehensive and efficient framework for recommendation algorithms. In: CIKM (2021)



Improved CNN Model Using Innovative Adaptive-DropMessage for Gomoku Game

Kangjie Cao^{1,2}, Xiali Li^{1,2(✉)}, Jinyao Wu^{1,2}, Hu Yuan², Wentao Li², Jiayun Li², He Huang², Jueqiao Huang^{1,2}, and Weijun Cheng^{1,2}

¹ Key Laboratory of Ethnic Language Intelligent Analysis and Security Governance of MOE,
Minzu University of China, Beijing 100081, China
xiaer_li@163.com

² School of Information Engineering, Minzu University of China, Beijing 100081, China

Abstract. Convolutional Neural Networks (CNNs) has been used in many Gomoku AI systems. This paper introduces the innovative Adaptive-Drop Message method within the CNN framework. This method, combined with techniques including attention mechanisms, residual dense blocks, and depth-wise separable convolutions. The Adaptive-Drop Message method dynamically discards portions of feature map nodes based on loss values, finely controlling the activity of neurons. The model does not overly rely on any single feature set while learning critical features on the game board. Experiments were conducted on the A100 GPU platform. Compared to other outstanding Gomoku AI models with equivalent parameter scales and resource consumption, the model proposed performs excellent in terms of winning rates, average winning moves, and average winning time. it is validated that the proposed methods can improve rapid and efficient decision-making.

Keywords: Gomoku · convolutional neural network · Deep Reinforcement Learning · Adaptive-DropMessage · Attention Mechanism · Residual Dense Blocks · Depth-wise Separable Convolution

1 Introduction

The exceptional capability of AlphaZero in mastering complex board games has garnered widespread attention in the field of artificial intelligence [1]. Researchers have extended the AlphaZero method to the game of Gomoku [2] achieving impressive results. Further advancements in Gomoku algorithms have been made through various approaches such as rapid algorithm convergence [3, 4], integrating PCN heuristic search with the AlphaZero method [5], and employing dynamic sampling tree strategies for effective allocation of limited computational resources [6].

However, the research in Gomoku algorithms faces challenges: firstly, the majority of AI models trained using the AlphaZero framework adopt conventional random dropout

This paper was supported by the National Natural Science Foundation of China, project number is 62276285, 52374169 and 62236011.

methods like dropout [7], which are outdated compared to Adaptive-Drop Message. Secondly, widely-used standard deep convolutional neural networks (CNNs), although capable of recognizing specific features of Gomoku and selecting the most probable next moves, exhibit limitations like a small receptive field, insensitivity to global features, and poor generalization.

The stellar performance of attention mechanisms in fields like computer vision [8], and engineering tasks [9] has offered new perspectives for improvements in CNNs. A global attention mechanism for convolutional spatial attention submodules was proposed [10], consistently outperforming several attention mechanisms using ResNet and lightweight MobileNet. Residual dense blocks [11] and their specific applications in the imaging field [12] have proven to aid CNNs in deeper feature extraction and mitigating gradient vanishing issues. The SE module proposed [13] brings significant performance enhancements to CNNs with minimal additional computational cost, as thoroughly validated on multiple image datasets [14]. Dilated convolutions and depthwise separable convolutions [15] have been proven to effectively improve the receptive field and feature extraction capabilities of CNN models.

However, no researchers have yet integrated these modules and mechanisms into CNNs to construct an Alpha Zero framework for Gomoku AI models. Therefore, this paper, based on the AlphaGo Zero framework, designs a Gomoku AI model by integrating a CNN model with Adaptive-Dropmessage, depthwise separable convolutions, attention mechanisms, and residual dense blocks combined with MCTS reinforcement learning. By introducing dilated depthwise separable convolutions into the CNN model, the approach reduces parameter count, lowers computational costs, expands the receptive field, and enhances feature extraction. The introduction of residual dense blocks, incorporating residual connections between each convolutional layer, enables the model to better capture inter-feature dependencies and effectively address gradient vanishing and information loss in deep network training.

The main contributions of this paper are as follows:

- Pioneering the use of the Adaptive-DropMessage novel random dropout strategy in convolutional neural networks. This strategy influences neurons in the neural network with finer granularity during the message passing process. As a result, the model can learn critical features on the chessboard without overly relying on any single set of features, thereby enhancing its generalization capability.
- Integrating Adaptive-DropMessage, depthwise separable convolution, attention mechanisms, and residual dense blocks into the convolutional neural network for training the AlphaZero framework on the game of Gomoku (Five in a Row). This integration serves to confirm the superiority of the model in this task.

2 Related Work

The extension of the AlphaZero method to Gomoku has led to significant advancements in autonomous learning and decision-making in Gomoku AI. Reference [4] combines DNN with the AlphaZero method to propose an improved method for accelerating MCTS search, enhancing playing strength within limited hardware resources and shorter training times. Reference [3], inspired by AlphaGo Zero and integrating Monte Carlo tree

search, deep neural networks, and reinforcement learning techniques, demonstrates the feasibility of Gomoku AI evolving independently without human knowledge.

Residual dense blocks [11] combine lightweight residual dense connection blocks with CNNs, reducing the cost of training and inference processes without the need for special software or hardware, simply by reducing the number of parameters and computational operations, while achieving feasible accuracy. Reference [20] uses a CNN model combining the advantages of dense and residual blocks, enhancing information flow, gradient propagation, and feature reuse through the benefits of residual and dense connections, thereby improving model performance. Reference [21] introduces DMSeqNet-mBART, incorporating Adaptive-DropMessage technology, a novel approach that intelligently discards or retains information based on the output of the attention mechanism.

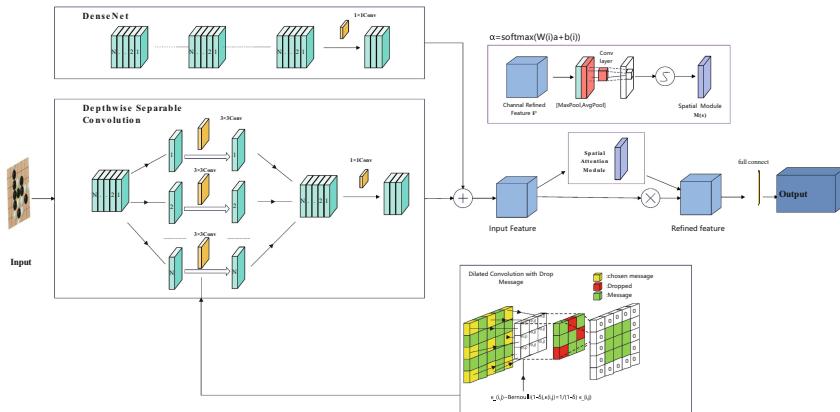


Fig. 1. Schematic diagram of the improvement of the CNN model proposed in the paper

3 Methodology

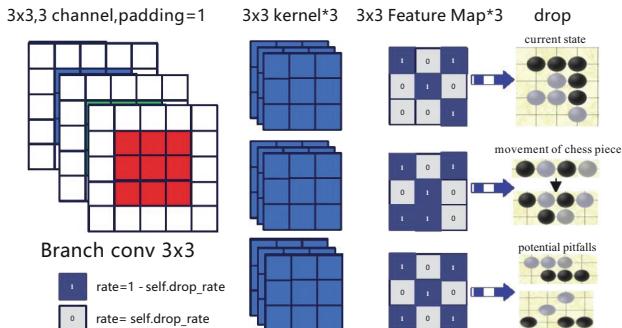


Fig. 2. Schematic diagram of Adaptive-DropMessage method

The improvement of the CNN model is shown in Fig. 1. We have introduced DropMessage [22] from graph neural networks into convolutional neural networks for the first time, pioneering the Adaptive-DropMessage technique, which operates during the message passing process, affecting neurons with finer granularity. Combined with dilated depthwise separable convolutions, residual dense blocks, SENet modules, attention modules, and a specific output layer for the game of Gomoku, we have constructed a model for the AlphaGo Zero framework.

3.1 Adaptive-DropMessage Method

As shown in Fig. 2, we proposed an Adaptive-DropMessage method to reduce sample variance, stabilize the training process, and maintain diverse information. Adaptive-DropMessage can be regarded as a sampling process. The perturbed message matrix m_f is obtained by formula (1), where $\frac{1}{1-\delta}$ is the scaling factor, $\epsilon_{i,j}$ is the independent mask corresponding to element $m_{i,j}$ in the matrix, and obeys the Bernoulli distribution $\epsilon_{i,j} \sim \text{Bernoulli}(1 - \delta)$, indicates the probability of keeping or discarding.

$$M_{f_{i,j}} = \frac{1}{1 - \delta} \epsilon_{i,j} M_{i,j} \quad (1)$$

The “Adaptive-DropMessage” technique adjusts the dropout rate adaptively based on the training loss and is applied to both dilated depthwise separable convolutional layers and standard depthwise separable convolutional layers. It is placed after the depthwise convolution and pointwise convolution operations to randomly drop some nodes in the feature maps.

In training mode, the process starts by initializing the `drop_rate`, which determines the proportion of feature maps to randomly set to zero. A mask is generated using a Bernoulli distribution, randomly creating a mask with the same shape as the input tensor x . Each element in the mask has a probability of being set to 1 with a probability of $(1 - \text{self.drop_rate})$ and a probability of 0 with a probability of (self.drop_rate) . The generated mask is then multiplied element-wise with the input tensor x achieving random dropout. To compensate for the reduced activation values caused by dropping feature maps, the output is divided by $(1 - \text{self.drop_rate})$ to maintain its scale. In testing mode, DropMessage does not perform any operations and directly returns the input x .

3.2 Dilated Depthwise Separable Convolution

Dilated depthwise separable convolution introduces dilation into the convolutional kernel, allowing the kernel to sample the input with a fixed interval. The calculation for dilated convolution is represented by Eq. (2).

$$\text{Output}(x, y) = \sum_{i=0}^{k-1} \sum_{j=0}^{k-1} \text{Input}(x + i \cdot \text{dilation}, y + j \cdot \text{dilation}) \cdot \text{Kernel}(i, j) \quad (2)$$

In this context, $\text{Output}(x, y)$ represents the value at a specific position in the output feature map, $\text{Input}(x, y)$ corresponds to the value at the respective position in the input

feature map, $\text{Kernel}(i, j)$ denotes the weight of the convolutional kernel, and dilation indicates the sampling interval of the convolutional kernel on the input.

Depthwise separable convolution decomposes traditional convolution into two steps: depthwise convolution and pointwise convolution. Firstly, it performs independent depthwise convolutions for each input channel, and then it utilizes a 1x1 convolutional kernel for pointwise convolution to integrate information across channels. This approach significantly reduces the number of parameters and enhances the computational efficiency of the model. The calculation method of depth convolution is as formula (3):

$$(I * K_d)(x, y, c) = \sum_{i=0}^{k-1} \sum_{j=0}^{k-1} I(x + i, y + j, c) \cdot K_d(i, j, c) \quad (3)$$

I is the input tensor, K_d is the depth convolution kernel. Depthwise convolution performs a convolution operation on each input channel channel by channel. In the formula (3), x and y represent the position of the output feature map, and c represents the channel. $I * K_d$ represents the result of depth convolution, calculated by applying the corresponding depth convolution kernel K_d to each channel of the input tensor I . The calculation method of pointwise convolution is given by Eq. (4):

$$(I * K_p)(x, y, f) = \sum_{c=1}^C I(x, y, c) \cdot K_p(c, f) \quad (4)$$

I is the input tensor, and K_p is the point-wise convolution kernel. Pointwise convolution uses a 1×1 convolution kernel to integrate information from each channel to produce the final output. $I * K_p$ represents the result of pointwise convolution, calculated by applying the corresponding pointwise convolution kernel K_p to each channel of the input tensor I .

In our model, the depthwise convolution employs dilated convolution, while the pointwise convolution is a standard 1x1 convolution. Each component consists of convolution, batch normalization, and the Mish activation function. By introducing dilated depthwise separable convolution, the network enlarges its receptive field, enabling it to capture a broader context. This contributes to a better understanding of the relationships between the chessboard, local board configurations, and the overall game state.

3.3 Residual Dense Block and SENet Module

The Residual Dense Block (RDB) [11] plays a significant role in the training of Gomoku, especially in feature extraction and deep network learning. RDB uses depthwise separable convolution for efficient feature extraction in complex strategy games like Gomoku. It maintains effective feature extraction while reducing parameters. Dense connections and local residual learning prevent gradient vanishing, enabling deep feature utilization. Channel adjustment and downsampling integrate feature maps and focus on key information, capturing essential patterns and threats in Gomoku.

The Squeeze-and-Excitation Network (SENet) module is a mechanism used to enhance the feature representation capabilities of convolutional neural networks [19].

The SENet module introduces a channel attention mechanism, automatically learning the importance of different feature channels. In Gomoku, different feature channels represent various patterns and strategies on the board. The SENet module enhances the focus on key channels while suppressing less important ones, thereby improving the model's judgment capability.

By combining RDB and SENet modules, the constructed reinforced CNN neural network can more deeply understand and learn the complex features of Gomoku, including patterns, strategies, and situations.

3.4 Spatial Attention Module

In our model, the Spatial Attention Module is applied to the neural network training for the game of Gomoku. Its purpose is to increase the network's attention to specific regions on the game board, thereby improving its understanding and prediction of effective moves and game outcomes. The module architecture is illustrated in Fig. 3.

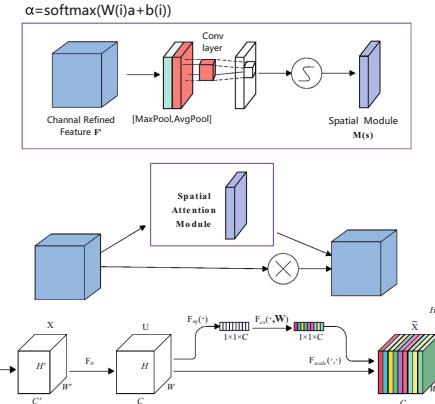


Fig. 3. Schematic diagram of Spatial Attention module

The Spatial Attention Module is instantiated twice in the EnhancedCNN class (named as sa1 and sa2) and embedded at different levels of the network. It operates through the following steps:

Calculation of Average and Maximum Feature Maps: For the input feature map x , the module first calculates the average average_out and maximum max_out across the channel dimension. These two operations capture different information: the average feature map represents the overall features, while the maximum feature map highlights the most prominent features. In Gomoku training, this allows the network to focus more on key areas of the board, such as potential sequences for forming chains or critical defense areas. This weighted feature map is further processed through subsequent layers, helping the model to more accurately predict the next move and assess the current game situation.

3.5 Gomoku-Specific Output Layer

We have customized a Gomoku-specific output layer for the model, comprising two parts: the policy head and the value head.

- The Policy Head’s primary function is to generate a probability distribution for the next move. It first reduces the number of input channels with a 1×1 convolutional kernel, followed by batch normalization for enhanced stability. It then introduces non-linear transformations using a rectified linear activation function, flattens the multi-dimensional output into a one-dimensional vector, and maps it to the final probability distribution through a fully connected layer for the model’s decision-making.
- The Value Head’s main task is to assess the win probability of the current situation. It maps the input features to an intermediate representation space, introduces non-linear transformations, and maps the intermediate representation to a scalar value representing the win probability.

This design integrates strategy and value information, enhancing the model’s situational understanding and decision-making accuracy. The loss functions of both heads are combined during training to optimize the model comprehensively.

4 Experiments and Analysis

4.1 Experimental Setup

The training software and hardware environment configuration, along with specific parameters, are detailed in Tables 1, 2, and 3:

Table 1. Hardware configuration parameter.

Hardware	Description
GPU	2xNVIDIA A100 80 GB PCIe
CPU	2xIntel Xeon Platinum 8358 @ 2.60 GHz
Motherboard	Supermicro X12DPi-N6
Memory	64 GB DDR4 RDIMM Server Memory *16

Under the condition that the key experimental parameters are the same, under the AlphaZero framework, we compared the training results of this model with four neural network models that have been proven to have excellent performance in Gomoku artificial intelligence, namely CNN (traditional, with Dropout), Liang, 2023 [16]), deep convolutional neural network (Yan, 2018 [4]), GomokuNet (Y. Gao, 2021 [17]), GomokuPro (Fu, 2022 [18]).

Table 2. Software configuration parameter.

Software Parameter	Description
Operating System	Linux
CUDA	CUDA Toolkit 12.1 Update 1
PyTorch	torch-2.0.1
Python	3.9

Table 3. Training Key Parameter Settings.

Parameter	Value
Initial Learning Rate	3e-4
Number of Simulations per Move Location	1200
Initial Exploration Constant in MCTS Algorithm	7
Mini-batch Data Size for Each Training	1024
Number of Games Played in Each Self-Play	3
Training Iterations per Self-Play Dataset	20
Total Batches of Self-Play Training	10000
Initial Simulations by Pure MCTS Player	1200

4.2 Game Experiment Results and Analysis

Game Experiment

The game rules are as follows: Record wins and draws: Record the number of wins and draws for each model to ultimately determine which model performs best (Fig. 4).

In games against the Deep Convolutional Neural Network (Deep-CNN), our proposed model won 266 games and lost only 34. When playing against the traditional CNN model, our proposed model won 259 games and lost 41. Facing GomokuNet, our proposed model won 247 games and lost 53. Most notably, in games against GomokuPro, our proposed model significantly led with 280 wins and only 20 losses. The gameplay results fully demonstrate the superior performance and exceptional decision-making abilities of our Gomoku AI model based on Adaptive-DropMessage in various scenarios. It not only maintains a consistently high win rate against a variety of opponents but also shows its strong adaptability and strategic flexibility in diverse game situations.

Average Winning Moves and Average Winning Time

Table 4 shows that the model proposed in this paper also performs well in terms of efficiency and speed. Our model takes an average of 19 moves to win in the Gomoku game, outperforming other models. This shows that our proposed model can find key winning moves faster and more accurately during the game, which not only reflects the depth of its game analysis and accuracy, also reflects the efficiency of its decision-making.

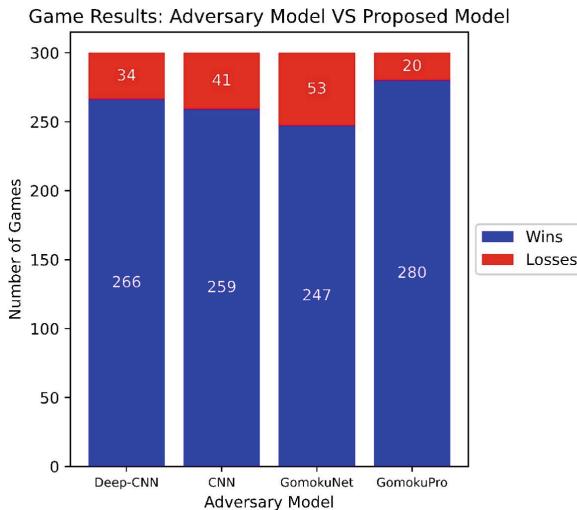


Fig. 4. Comparison between Proposed Model VS Other Model

Table 4. Comparison of average winning moves and average winning time of different models.

Model	Pro-posed	Deep-CNN	Gomo kuNet	Gomo kuPro	CNN
Average Winning Moves	19	23	23	25	27
Average Winning Time (s)	23.8	24.9	25.6	26.2	28.3

Our model also shows superiority in terms of average winning time, with an average time of 23.8 s, lower than all other models. Therefore, we can conclude that the Adaptive-DropMessage-based Gomoku game model proposed in this article has significant advantages in decision-making efficiency and processing speed, making it highly valuable in the field of high-level Gomoku AI.

Table 5. Average compute resource usage for model training and training time to achieve optimal performance.

Model	Average Resource Usage	Training Time
Proposed	21.13%	583 h
Deep-CNN	20.92%	600 h
CNN	19.47	620 h
GomokuNet	22.75%	600 h
GomokuPro	23.75	592 h

Average Computing Resource Usage and Training Time to Achieve Optimal Performance

As can be seen from Table 5, the average computing resource occupancy of our Proposed model is 21.13%, Moreover, our model requires only 583 h of training to reach optimal performance, which is shorter than all other models.

Overall, our proposed model achieves shorter training time and higher performance while maintaining reasonable computing resource usage.

Model Ablation Experiment

To evaluate the impact of various model components on performance, we systematically removed or modified different components. To accelerate the experiments, we increased the batch size from 1024 to 2048, reduced the number of self-play games per iteration from 3 to 2, decreased the total self-play training batches from 10000 to 3000, and lowered the number of training iterations per dataset from 20 to 8. When ablation involved Adaptive-DropMessage, we re-placed it with dropout to clearly demonstrate its advantages. Both the proposed model and five ablation models were trained to convergence and evaluated over 200 games.

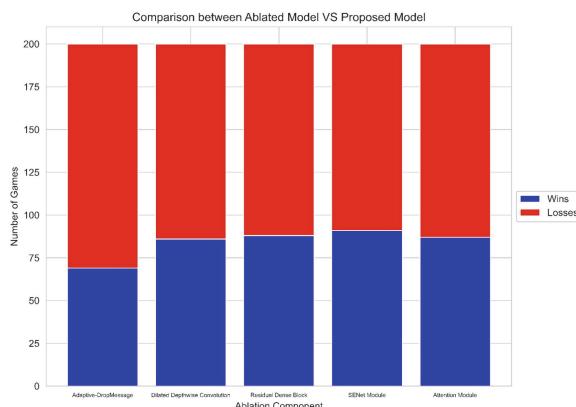


Fig. 5. Comparison between Ablated Model VS Proposed Model

Figure 5 showcasing the results of the ablation study, the analysis is as follows:

- Adaptive-DropMessage: After the removal of the Adaptive-DropMessage component, the model's win rate decreased to 69 wins and 131 losses. This significant performance drop indicates the pivotal role of Adaptive-DropMessage in the model.
- Dilated Depthwise Separable Convolution: When the dilated depthwise separable convolution is removed, the model's win rate is 86 wins to 114 losses. This suggests that dilated depthwise separable convolution is also crucial for enhancing model performance.
- Residual Dense Block: Removing the residual dense block resulted in the model performing at 88 wins and 112 losses. Its impact on model performance is relatively smaller but still significant.

- SENet Module: With the SENet module eliminated, the model's win rate was 91 wins to 109 losses. Although its impact on performance is less pronounced compared to the former components, its presence still positively affects the model.
- Attention Module: After the removal of the attention module, the model's win rate was 87 wins to 113 losses. This result indicates that while the contribution of the attention module is not the most critical, it still significantly enhances the overall performance of the model.

Overall, these experiments demonstrate that our proposed model is the result of the collaborative effect of multiple key components, each contributing to the model's performance to varying degrees.

5 Conclusion

The Gomoku game model based on Adaptive-DropMessage presented in this paper shows significant innovation and advantages in the field of Gomoku AI. By integrating deep learning and reinforcement learning techniques and combining various advanced neural network structures and strategies such as Adaptive-DropMessage, attention mechanisms, residual dense blocks, and depthwise separable convolution, experimental results indicate that our model has achieved substantial improvements in performance in Gomoku gameplay.

In summary, this research not only presents a Gomoku AI model that excels in multiple aspects but also validates the applicative value of various advanced technologies in AI for board games. In the future, we hope this research will inspire further advancements in the field of Gomoku AI and related areas.

References

1. Zhang, H., Yu, T.: AlphaZero. In: Deep Reinforcement Learning: Fundamentals, Research and Applications, pp. 391–415 (2020). Author, F., Author, S.: Title of a proceedings paper. In: Editor, F., Editor, S. (eds.) CONFERENCE 2016, LNCS, vol. 9999, pp. 1–13. Springer, Heidelberg (2016)
2. Liang, W., Yu, C., Whiteaker, B., Huh, I., Shao, H., Liang, Y.: Mastering Gomoku with AlphaZero: a study in advanced AI game strategy. Sage Sci. Rev. Appl. Mach. Learn. **6**(11), 32–43 (2023)
3. Wang, Y.: Mastering the game of Gomoku without human knowledge (2018). <https://doi.org/10.15368/theses.2018.47>
4. Yan, P., Feng, Y.: A hybrid Gomoku deep learning artificial intelligence. In: Proceedings of the 2018 Artificial Intelligence and Cloud Computing Conference, December 2018, pp. 48–52 (2018)
5. Wu, T.R., et al.: AlphaZero-based proof cost network to aid game solving. In: Proceedings of the 10th International Conference on Learning Representations, ICLR (2022)
6. Zhang, G., Peng, Y., Xu, Y.: An efficient dynamic sampling policy for Monte Carlo tree search. In: Proceedings of the 2022 Winter Simulation Conference (WSC), December 2022, pp. 2760–2771. IEEE (2022)
7. Wu, L., Li, J., Wang, Y., Meng, Q., Qin, T., Chen, W., et al.: R-drop: regularized dropout for neural networks. In: Advances in Neural Information Processing Systems, vol. 34, pp. 10890–10905 (2021)

8. Cheng, G., Wang, G., Han, J.: ISNet: towards improving separability for remote sensing image change detection. *IEEE Trans. Geosci. Remote Sens.* **60**, 1–11 (2022)
9. Liu, L., Song, X., Zhou, Z.: Aircraft engine remaining useful life estimation via a double attention-based data-driven architecture. *Reliab. Eng. Syst. Saf.* **221**, 108330 (2022)
10. Liu, Y., Shao, Z., Hoffmann, N.: Global attention mechanism: retain information to enhance channel-spatial interactions. arXiv preprint [arXiv:2112.05561](https://arxiv.org/abs/2112.05561) (2021)
11. Zhang, Y., Tian, Y., Kong, Y., Zhong, B., Fu, Y.: Residual dense network for image restoration. *IEEE Trans. Pattern Anal. Mach. Intell.* **43**(7), 2480–2495 (2020)
12. Song, D., Xu, C., Jia, X., Chen, Y., Xu, C., Wang, Y.: Efficient residual dense block search for image super-resolution. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 07, pp. 12007–12014, April 2020
13. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: Proceedings of the IEEE Conference on Computer Vision Pattern Recognition, pp. 7132–7141 (2018)
14. Jin, X., Xie, Y., Wei, X.S., Zhao, B.R., Chen, Z.M., Tan, X.: Delving deep into spatial pooling for squeeze-and-excitation networks. *Pattern Recognit.* **121**, 108159 (2022)
15. Drossos, K., Mimalakis, S.I., Gharib, S., Li, Y., Virtanen, T.: Sound event detection with depthwise separable and dilated convolutions. In: Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1–7. IEEE, July 2020
16. Liang, W., Yu, C., Whiteaker, B., Huh, I., Shao, H., Liang, Y.: AlphaZero Gomoku. arXiv preprint [arXiv:2309.01294](https://arxiv.org/abs/2309.01294) (2023)
17. Gao, Y., Wu, L., Li, H.: GomokuNet: a novel UNet-style network for Gomoku zero learning via exploiting positional information and multiscale features. In: Proceedings of the 2021 IEEE Conference on Games (CoG), August 2021, pp. 1–4. IEEE (2021)
18. Fu, X.: GomokuPro: an implementation of enhanced machine learning algorithm utilizing convolutional neural network in Gomoku strategy and predictions model. In: Proceedings of the 2022 7th International Conference on Intelligent Computing and Signal Processing (ICSP), April 2022, pp. 1671–1677. IEEE (2022)
19. Fooladgar, F., Kasaei, S.: Lightweight residual densely connected convolutional neural network. *Multimed. Tools Appl.* **79**(35–36), 25571–25588 (2020)
20. Ahmed, A.E., Abbas, Q., Daadaa, Y., Qureshi, I., Perumal, G., Ibrahim, M.E.: A residual-dense-based convolutional neural network architecture for recognition of cardiac health based on ECG signals. *Sensors* **23**(16), 7204 (2023)
21. Cao, K., Hao, Y., Huang, J., et al.: DMSeqNet-mBART: a state-of-the-art adaptive-DropMessage enhanced mBART architecture for superior Chinese short news text summarization. *Authorea Preprints* (2024)
22. Fang, T., Xiao, Z., Wang, C., Xu, J., Yang, X., Yang, Y.: Dropmessage: unifying random dropping for graph neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, no. 4, pp. 4267–4275, June 2023



A Survey: Feature Fusion Method for Object Detection Field

Zhe Lian, Yanjun Yin^(✉), Jingfang Lu, Qiaozhi Xu, Min Zhi, Wei Hu,
and Wentao Duan

School of Computer Science and Technology, Inner Mongolia Normal University,
Hohhot 010022, China
ciecyj@163.com

Abstract. Feature fusion techniques represent a critical research topic within the field of computer vision, playing an extensive role in downstream tasks that necessitate rich object representations, such as image classification, semantic segmentation, and object detection. The Feature Pyramid Network (FPN) was proposed in 2017 for object detection, emerged as a groundbreaking work in the area of feature fusion methods. Subsequently, a variety of powerful and innovative feature fusion architectures have been successively proposed, bringing significant performance enhancements in object detection tasks. This paper takes a depth look into feature fusion technologies starting from the domain of object detection. Systematically categorizes existing methods into two main classes based on the intrinsic properties of different feature fusion structures: simple topological fusion structures and complex topological fusion structures. The paper conducts analyses and summarizes the mechanisms of these two types of structures, introduced the evaluation dataset and evaluation indicators, accompanied by a collation of open-source codes for mainstream feature fusion architectures. Ultimately, through systematic review, the paper summarizes the challenges faced by feature fusion methodologies and provides an outlook on future development trends in this area.

Keywords: Deep learning · Computer vision · Object detection · Feature fusion

1 Introduction

Since AlexNet triumph at the 2012 ImageNet Large Scale Visual Recognition Challenge [1], there has been a surge in scholarly engagement with Convolutional Neural Networks (CNN). Due to its powerful feature learning ability, CNN have progressively supplanted traditional handcrafted feature design approaches, thereby becoming the prevailing network architecture for feature extraction in computer vision. In the task of object detection, the detection results greatly depend on the low-level detail features and high-level semantic features in the feature extraction process. However, most CNN difficulty in effectively balancing these two types of features. The principal reason lies in the pronounced semantic information gap between features at different levels. In order to address the issue of semantic gaps, researchers have focused on the study of different feature fusion strategies.

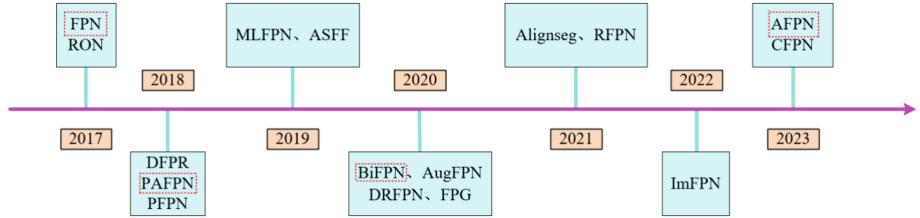


Fig. 1. Classic feature fusion structure timeline (the methods within the red dashed box are pioneering achievements). (Color figure online)

The initial feature fusion method was to connect features from different layers through addition or concatenation operations before the prediction task, such as Hypercolumns and ION [2, 3], which concatenated or summed features across various depths. On the other hand, algorithms like SPPnet and Inception employed multiple scale convolutional kernels to extract features with diverse receptive fields and then merged these extracted features to achieve feature fusion [4, 5]. While these methods proved successful in early computer vision tasks, they became less effective as data complexity and model sophistication increased.

In 2017, Facebook AI Research proposed the Feature Pyramid Network (FPN) for object detection [6], which ingeniously fused features from different levels of the feature extraction network through a pyramidal structure. This approach effectively propagated features, bridged the semantic gap, and provided richer and more accurate data representations. The birth of FPN marked a milestone achievement in feature fusion technology, subsequently serving as the foundational inspiration for numerous subsequent outstanding works ranging from [7–21]. The development timeline is shown in Fig. 1.

Despite substantial advancements in feature fusion techniques over the years, and develop fusion structures tailored to specific needs in various fields [22, 23], but only a limited number of these approaches exhibit broad applicability. To this end, we have conducted an extensive literature review revealing that since 2017, several feature fusion methods possessing universal properties for object detection have been published in premier conferences such as ECCV, CVPR, ICCV. As of the present date, the cumulative citation count for these works has surpassed fifty thousand, sufficiently demonstrating the current popularity trend of general-purpose object detection feature fusion techniques within the field of computer vision.

This paper systematically summarizes the research progress of feature fusion technology in the field of object detection since the proposal of FPN. In the Sect. 2, based on the inherent characteristics of different feature fusion technologies, the current fusion methods are divided into two categories: simple topology fusion structures and complex topology fusion structures. The mechanism of each method is summarized, and the open source code links of different technologies are organized. The Sect. 3 lists commonly used evaluation datasets and evaluation indicators. The Sect. 4 summarizes the future trends and challenges of feature fusion structures, and provides prospects for the development of this field. The Sect. 5 summarizes the entire text.

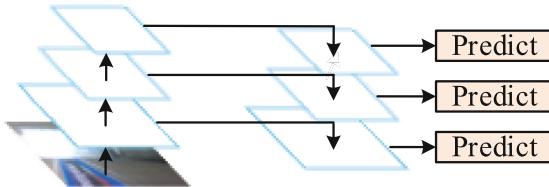


Fig. 2. Schematic diagram of FPN network structure.

2 Feature Fusion Methods

In the realm of computer vision, recognizing objects at varying scales is a fundamental challenge. The concept of a pyramid has been in existence since 1984 [24], the initial image pyramid preserving scale invariance, revolutionizing the era of handcrafted feature design. Subsequently, the advent of CNN led researchers to discern that the downsampling operations inherent to CNN could produce a fixed multi-scale pyramid-like structure of features. SSD was the first to employ the intrinsic pyramid structure of CNN for feature representation [25], but it merely leveraged a hierarchical structure without addressing the interplay of information across different levels of features.

The emergence of FPN became the beginning of modern topological feature fusion structures, demonstrating strong competitive performance in tackling early-stage object detection challenges. With the escalation in dataset size and complexity, however, simplistic topologies proved insufficient to meet the demands of these tasks. Consequently, researchers are attempting to add more information propagation paths to make the topology more complex.

2.1 Simple Topology Fusion Structure

Simple topological fusion structures build upon the foundation laid by FPN [6]. FPN is a network that exploits the inherent multi-scale pyramid hierarchy of convolutional networks to tackle the computationally expensive pyramid representation problem in object detection models, as depicted in Fig. 2. FPN introduces a lateral connection-based feature fusion architecture that ensures high-level semantics across all scales. Through top-down information propagation paths, the semantic gap between feature maps at different levels is notably reduced. Researchers conducted further research on the pyramid defect problem and multi scale localization problem of FPN.

Pyramid Defect Problem. Guo *et al.* summarized the inherent drawbacks in pyramid-style feature fusion structures as substantial semantic gaps between different levels of features the loss of information in the highest-level feature maps and the inadequacy of heuristic allocation strategies for Regions of Interest (RoIs). To tackle these issues, they propose the Augmentation Feature Pyramid Network (AugFPN) [7], incorporating three distinct components to individually address these problems. Firstly, they introduce Consistent Supervision (CS), which applies the same supervisory signals onto the feature maps following lateral connections, thus ensuring that the feature maps carry similar semantic information. Secondly, Ratio-Invariant Adaptive Pooling (RIAP) is utilized

to extract diverse context information, implemented residual to mitigate the loss of information in the highest level of the feature pyramid. Thirdly, they incorporate Soft RoI Selection, aiming to better leverage RoI features from different pyramid levels and consequently generating improved RoI.

Ma *et al.* argue that the shortcomings of pyramidal feature fusion architectures lie in inaccurate spatial sampling and imprecise channel fusion. Therefore, they propose Dual Refinement Feature Pyramid Networks (DRFPN) [8], which embeds Spatial Refinement Blocks (SRB) into the FPN. The SRB adaptively learn both the positions and content of sampling points based on information from adjacent layers, thereby rectifying spatial inaccuracies. Concurrently, DRFPN incorporates Channel Refinement Blocks (CRB) that utilize attention mechanisms to learn the importance of channels during the fusion process.

However, the aforementioned feature fusion methods simply perform element-wise addition or concatenation along the channel dimension, neglecting the fact that due to the semantic disparities and similarities between features, different layers require varying weights for effective fusion. Consequently, Zhu *et al.* propose the Improved Feature Pyramid Network (ImFPN) [9], which enhances the conventional pyramid network by appending a Split-Attention Module (SAM) from ResNeXt to the top-level features [27]. This module divides the features into K groups, with each group further partitioned into R basis sets. Within each basis set, global context is condensed, and channel-wise attentions are employed to integrate the basis representations, thereby mitigating the occurrence of information loss.

Multi Scale Localization Problem. Diverging from the focal points of FPN, Kong *et al.* focus on the localization problem of multi-scale objects, think that targets of various scales may appear at any position in the image. Thus, they propose the Reverse Connection with Objectness Prior Networks (RON) [10]. Inspired by the success of residual connections in ResNet [28], RON introduces reverse connections onto the CNN architecture, thereby endowing shallow features with richer semantic information. Given that these reverse connections yield multiple feature maps at differing scales, RON is designed to generate a distribution of multiple candidate bounding boxes as objectness priors, allowing it to learn specific locations on the feature maps that respond to the particular scale of a target.

Although RON and FPN methods have enhanced shallow feature detection capabilities, their linear combination strategies for features are overly simplistic and inadequate for effectively capturing highly nonlinear patterns under more complex circumstances. Hence, Kong *et al.* propose the Deep Feature Pyramid Reconfiguration Network (DFPRN) [11], which employs global attention to emphasize global information throughout the entire image and subsequently performs local reconfiguration to model local blocks within the receptive field. This results in a pyramid representation capable of propagating strong semantics across all scales. In contrast to previous studies, the global-local reconfiguration of DFPRN involves non-linear transformations, thereby rendering the feature representation more expressive. The pyramid fusion across all scales in DFPRN occurs simultaneously, proving to be more effective than sequential or layer-by-layer translation.

Huang *et al.* astutely observed that most popular feature fusion architectures tend to overlook the issue of feature misalignments arising from progressive downsampling operations and indiscriminate context information integration. To tackle this problem, they propose the Feature-Aligned Segmentation Network (AlignSeg) [12], composed of an Aligned Feature Aggregation module (AlignFA) and an Aligned Context Modeling module (AlignCM). AlignFA adopts a simple yet learnable interpolation strategy to learn transformation offsets for pixels, effectively mitigating feature misalignment issues caused by multi-resolution feature aggregation. Meanwhile, AlignCM enables each pixel to adaptively select its own customized contextual information, thus aligning context embeddings more accurately.

2.2 Complex Topology Fusion Structure

Simple topology fusion structure performs excellently in handling early simple tasks, but as complex visual tasks emerge, these simple architectures struggle to effectively extract meaningful features from vast amounts of data. To overcome these limitations, researchers have explored more sophisticated feature fusion methods from a structural perspective, primarily dividing them into two categories: single bidirectional pathways and multiple bidirectional pathways.

Single Bidirectional Pathways. Liu *et al.* aimed to shorten information propagation paths while enhancing feature pyramids with precise positional information contained in lower-level features. Building upon the foundational idea of augmenting the top-down pathway in pyramidal feature fusion, they developed the Path Aggregation Feature Pyramid Network (PAFPN) [13]. This network extends the upward path in FPN thereby creating dual information transmission routes, as illustrated in Fig. 3. The green dashed lines depict shortcuts that bypass over a hundred convolutional layers compared to the direct route from lower to higher levels in the CNN backbone within FPN (represented by red dashed lines), resulting in significant time savings when fewer than ten convolutional layers are involved.

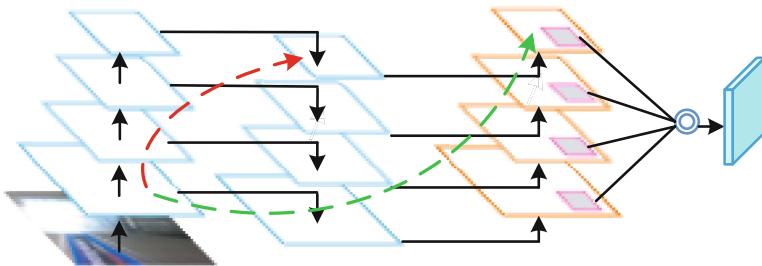


Fig. 3. Schematic diagram of PAFPN network structure.

Kim *et al.* argue that improvement strategies for simple fusion structures overlook the limitations imposed by different levels of abstraction on the detection of small objects. Consequently, they propose the Parallel Feature Pyramid Network (PFPN) [14], which

constructs its fusion architecture by widening rather than deepening the network. In PFPN, Spatial Pyramid Pooling (SPP) [4] is employed alongside additional parallel feature transformations to generate feature maps of diverse sizes. The advantage of parallel processing lies in the production of cross-scale feature maps with comparable levels of semantic abstraction.

The aforementioned fusion methods predominantly focus on inter-layer feature interactions while neglecting the modulation within individual layers. Therefore, Quan *et al.* propose the Centralized Feature Pyramid Network (CFPN) [15]. CFPN utilizes a spatially explicit visual center approach, incorporating lightweight Multilayer Perceptrons (MLPs) to capture long-range dependencies throughout the entire image and parallel learnable visual center mechanisms to encapsulate local corner regions of the input image. Based on this, CFPN introduces a top-down, globally centralized modulation method for conventional feature fusion structures. Compare with existing feature fusion architectures, CFPN not only captures global long-range dependencies effectively but also yields comprehensive yet discriminative feature representations.

Multiple Bidirectional Pathways. Zhao *et al.* contend that pyramid-type feature fusion methods are limited in their reliance on simply utilizing the inherent multi-scale pyramid structure of the backbone network to construct a feature pyramid, sometimes even being confined to the construction a single layer of the backbone, which restricts their effectiveness for object detection tasks. Consequently, they propose the Multi- Level Feature Pyramid Network (MLFPN) [16]. MLFPN integrates multi-level features extracted from the backbone network as base features, feeding these base features into alternating Thinned U-Shape Modules (TUMs) and Feature Fusion Modules (FFMs). Each U-shape module decoder layers serve as detection-specific feature representations. Ultimately, the decoder layers of equivalent scale dimensions are collected to construct a feature fusion pyramid dedicated to object detection.

On the other hand, Liu *et al.* discovered inconsistencies among different features within pyramid-style feature fusion representations and proposed Adaptively Spatial Feature Fusion (ASFF) [17]. ASFF directly learns how to perform spatial filtering on features from other levels such that only pertinent information is retained for combination. For a given level feature, it first aggregates and resizes features from other levels to match the same resolution, subsequently training to identify the optimal fusion. At each spatial location, features from different levels are multiplied by learned weight parameters to achieve adaptive fusion, effectively filtering out some features.

To encompass a larger search space, Chen *et al.* introduced the Feature Pyramid Grids (FPG) [18], FPG is a deep multi-path feature pyramid network that represents the feature scale space as a regular grid of parallel paths, where multiple directional lateral connections fuse information among the paths. This arrangement enriches the hierarchical feature representation built within the backbone network through multiple parallel pyramid paths. At a higher level, FPG is an extensive extension of FPN under a dense lateral connection structure, transitioning from a single path to N paths.

Tan *et al.* think that FPN and PAFPN, among others, essentially employ a straightforward additive blending of multi-scale features, wherein these methods do not intrinsically differentiate among the contributions from various levels. Recognizing that features from different levels possess unique resolutions and contribute differently to the output

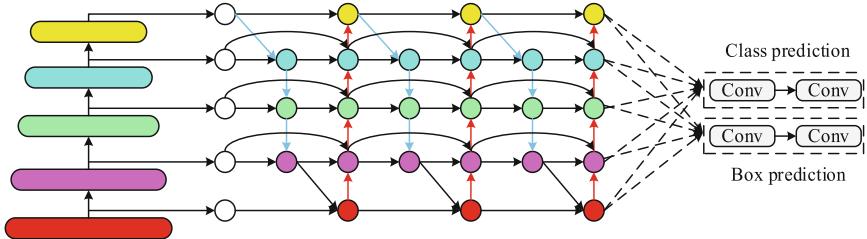


Fig. 4. Schematic diagram of BiFPN network structure.

features, they propose the Bi-directional Feature Pyramid Network (BiFPN) [19]. This architecture introduces learnable weights to determine the significance of different input features and iteratively applies both top-down and bottom-up multi-scale feature fusion processes, as illustrated in Fig. 4. This strategy culminates in an efficient and streamlined bi-directional feature fusion structure. This is the first time that feature fusion structures consider efficiency issues.

Advanced methods exhibit superior performance by engaging in multiple rounds of information exchange during feature fusion, a mechanism akin to ‘look before you leap’. This concept has been applied to backbone networks by Qiao *et al.* who propose a Recursive Feature Pyramid Network (RFPN) [20], integrating additional feedback connections from FPN layers into the bottom-up primary stages of the backbone. The recursive structure unfolds sequentially, as depicted in Fig. 5.

The RFPN with its ‘look before you leap’ mechanism, still adheres to the cross-scale fusion pattern of the original FPN, however, RFPN approaches overlook the issue of information loss or degradation in fusing features across non-adjacent levels. To address this, Yang *et al.* propose the Asymptotic Feature Pyramid Network (AFPN) [21]. AFPN initiates the feature fusion process by merging two adjacent lower-level features and progressively incorporates higher-level features into the fusion process, thereby circumventing substantial semantic gaps between non-adjacent levels. Moreover, recognizing potential conflicts in multi-object information during the feature fusion at each spatial location, AFPN further employs adaptive spatial fusion operations to mitigate these inconsistencies.

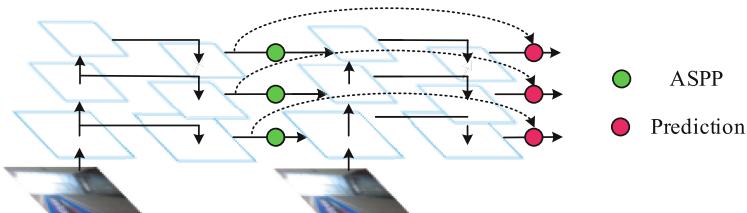


Fig. 5. Schematic diagram of RFPN network structure.

2.3 Analysis and Summary

Simple topological fusion structures, rooted in the design of the FPN, fundamentally rely on a single, straightforward information flow path. These methods demonstrate strong competitiveness in tackling early-stage object detection challenges. However, as the complexity of datasets grows, these methods are no longer applicable.

Complex topological fusion structures evolved from the introduction of PAFPN, which pioneered the alteration of the FPN base structure by adding an expanded bottom-up path. Following this innovation, there has been a proliferation of complex fusion architectures, such as BiFPN and FPG, effectively reduce disparities between feature maps by introducing new propagation pathways.

A summary of the mechanisms of fusion structures is compiled in Table 1.

Table 1. Fusion Structure Mechanism and Access Links for Fusion Structures.

Type	Method	Mechanism and Access Links
Simple Topology Fusion Structure	FPN [6]	Single Information Path, Multi-Scale Features, and Lateral Connections Access Links: https://github.com/facebookresearch/detectron
	AugFPN [7]	Consistent Supervision, Ratio-Invariant Adaptive Pooling, Soft RoI Access Links: https://github.com/Gus-Guo/AugFPN
	DRFPN [8]	Channel Refinement, Spatial Refinement
	ImFPN [9]	Split Attention, Compressing Global Context
	RON [10]	Backward Connections, Multi-Objectness Prior Access Links: https://github.com/taokong/RON
	DFRPN [11]	Pyramid Local Reconstruction, Global Attention Information Emphasis
Complex Topology Fusion Structure	AlignSeg [12]	Aligned Feature Aggregation, Aligned Context Modeling Access Links: https://github.com/speedinghzl/AlignSeg
	PAFPN [13]	Bidirectional Information Propagation Paths, Adaptive Feature Pooling Access Links: https://github.com/ShuLiu1993/PANet
	PFPN [14]	Parallel Feature Pyramids, Spatial Pyramid Pooling Access Links: https://github.com/chosj95/PFPNet.pytorch
	CFPN [15]	Learnable Visual Center Mechanism, Global Centralized Modulation Access Links: https://github.com/QY1994-0919/CFPNet
	MLFPN [16]	Multi-Level Feature Pyramids, Jointly Thinned U-Shape Modules Access Links: https://github.com/qijiezhao/M2Det

(continued)

Table 1. (*continued*)

Type	Method	Mechanism and Access Links
	ASFF [17]	Adaptive Spatial Feature Fusion, Scale Invariance Access Links: https://github.com/GOATmessi7/ASFF
	FPG [18]	Deep Multi-Path Feature Pyramid, Large Search Space
	BiFPN[19]	Weighted Repeated Bidirectional Fusion, Cross-Scale Optimization Access Links: https://github.com/google/automl/tree/master/efficientdet
	RFPN [20]	Feedback Connections, Switchable Atrous Convolution Access Links: https://github.com/joe-siyuan-qiao/DetectoRS
	AFPN [21]	Progressive Feature Pyramid, Adaptive Spatial Fusion Access Links: https://github.com/gyyang23/AFPN

Table 2. MS-COCO and Pascal VOC features.

Dataset	Years	Category	Training Set	Testing Set	Validation Set
MS-COCO [29]	2014	80	118k	40.6k	5.0k
Pascal VOC [30]	2010	20	5.0k	4.9k	5.7k

3 Datasets and Evaluation Indicators

3.1 Datasets

The general approach to verify the effectiveness of new feature fusion structures is to embed the feature fusion structure into the existing backbone, comparative experiments were conducted on MS-COCO and Pascal VOC object detection datasets [29, 30]. Because such datasets contain a large amount of data and are difficult to collect, there are currently no new large-scale object detection datasets open source. A summary of the characteristics of MS-COCO and Pascal VOC is presented in Table 2.

3.2 Evaluation Indicators

In order to objectively evaluate the detection performance of various methods, the main assessment metrics currently employed is Average Precision (AP) per class. Typically, the measure AP is obtained by computing the curve derived from $P(\text{Precision})$ and $R(\text{Recall})$, and then calculating the area under this curve. The calculation formula for AP is expressed as Eq. (1):

$$AP = \int P \times \frac{R}{1+R} dR \quad (1)$$

More specifically, the performance of the same method needs to be evaluated under different preset category labels. Common labels include AP_S , AP_M , AP_L , AP_{50} , AP_{75} , where S represents the category of smaller-sized objects in the dataset, M represents the category of medium-sized objects, and L represents the category of larger-sized objects. Thresholds of IoU at 50% and 75% are commonly set, where a prediction is considered correct only if the IoU value exceeds either 50% or 75%. The calculation method for IoU is shown in Eq. (2):

$$IoU = \frac{DR \cap GT}{DR \cup GT} \quad (2)$$

where DR represents the predicted bounding boxes, while GT denotes the ground truth bounding boxes.

For a clear illustration of the performance of various feature fusion architectures, we summarize part of the detection results using Faster R-CNN as the object detection model on the MS-COCO dataset [31], with all experiments employing ResNet50 as the backbone network. The outcomes are presented in Table 3.

Table 3. Quantitative detection results of some classic methods on MS COCO datasets.

Method	Size	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Params	GFLOPs
FPN [6]	640 × 640	37.4	57.3	40.3	18.4	41.7	52.7	41.5	91.4
AugFPN [7]	800 × 1333	38.7	61.2	41.9	24.1	42.5	49.5	—	—
ImFPN [9]	800 × 1333	39.2	59.9	42.6	22.2	42.8	52.1	—	—
DRFPN [11]	800 × 1333	39.1	60.3	42.5	22.9	43.1	50.7	48.4	263.0
PAFPN [13]	640 × 640	38.1	58.1	41.3	19.1	42.5	54.0	45.1	101.3
RFPN [20]	800 × 1000	44.4	67.7	48.3	25.6	47.5	58.3	—	—
AFPN [21]	640 × 640	39.0	57.6	42.1	19.4	43.0	55.0	50.2	90.0
AFPN [21]	800 × 1000	41.0	60.3	44.2	23.7	44.8	53.0	50.2	165.6

4 Trends and Challenges

Lightweight Fusion. While some complex feature fusion methods have indeed proven instrumental in boosting detection performance, an inevitable trade-off comes in the

form of decreased inference speed, which is counterproductive for real-time applications. With the widespread adoption of intelligent edge devices, there is a growing demand for portability and lightweight solutions in real-time tasks like traffic sign recognition, face identification, and natural scene text detection [32, 33]. It is anticipated that future research will increasingly prioritize lightweight strategies.

Evaluation Dataset Generation. Object detection datasets serve as a ubiquitous means to evaluate feature fusion techniques. The larger benchmark MS-COCO used for pre-training hosts a massive collection of 163.6k images. Amidst the prevailing trend of pre-training with large-scale datasets, open-source, massive datasets have become a scarce resource. In 2021, Zhang *et al.* proposed Datasetgan, successfully generating small-scale pixel-level labeled datasets for seven image segmentation tasks [34]. Given the recent surge of articles on generative models in top-tier conferences and journals, it is expected that more large-scale, pixel-level annotated datasets will emerge in the future, providing ample data support for the development of object detection technologies.

5 Conclusion

In this paper, we provide a detailed introduction to feature fusion methods in the field of object detection. Firstly, based on the characteristics of different fusion structures, typical feature fusion methods and their motivations proposed by FPN since 2017 have been elaborated in detail. They have been classified and summarized, and the mechanisms of each method have been condensed. Besides, we summarized the performance of various methods and we also provided Open source code links. Finally, the future development trends and challenges in this field have been summarized.

Acknowledgments. This study was funded by Inner Mongolia Autonomous Region Natural Science Foundation Project (2021HMS06009, 2023MS06009), Science and Technology Program of Inner Mongolia Autonomous Region (2022YFSH0010) and Inner Mongolia Normal University Graduate Research Innovation Fund Project (TY20240031).

References

1. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NeurIPS, p. 25 (2012)
2. Hariharan, B., Arbeláez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization. In: CVPR, pp. 447–456 (2015)
3. Bell, S., Zitnick, C.L., Bala, K., Girshick, R.: Inside-outside net: detecting objects in context with skip pooling and recurrent neural networks. In: CVPR, pp. 2874–2883 (2016)
4. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Trans. Pattern Anal. Mach. Intell. **37**(7), 1904–1916 (2015)
5. Szegedy, C., et al.: Going deeper with convolutions. In: CVPR, pp. 1–9 (2015)
6. 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)
7. Guo, C., Fan, B., Zhang, Q., Xiang, S., Pan, C.: Augfpn: Improving multi-scale feature learning for object detection. In: CVPR, pp. 12595–12604 (2020)

8. Ma, J., Chen, B.: Dual refinement feature pyramid networks for object detection. Arxiv. **2012.01733** (2020). <https://doi.org/10.48550/arXiv.2012.01733>
9. Zhu, L., Lee, F., Cai, J., Yu, H., Chen, Q.: An improved feature pyramid network for object detection. Neurocomputing **483**, 127–139 (2022)
10. Kong, T., Sun, F., Yao, A., Liu, H., Lu, M., Chen, Y.: Ron: reverse connection with objectness prior networks for object detection. In: CVPR, pp. 5936–5944 (2017)
11. Kong, T., Sun, F., Huang, W., Liu, H.: Deep feature pyramid reconfiguration for object detection. In: ECCV, pp. 169–185 (2018)
12. Huang, Z., Wei, Y., Wang, X., Liu, W., Huang, T.H., Shi, H.: Alignseg: feature-aligned segmentation networks. IEEE Trans. Pattern Anal. Mach. Intell. **44**(1), 550–557 (2021)
13. Liu, S., Qi, L., Qin, H., Shi, J., Jia, J.: Path aggregation network for instance segmentation. In: CVPR, pp. 8759–8768 (2018)
14. Kim, S.W., Kook, H.K., Sun, J.Y., Kang, M.C., Ko, S.J.: Parallel feature pyramid network for object detection. In: ECCV, pp. 234–250 (2018)
15. Quan, Y., Zhang, D., Zhang, L., Tang, J.: Centralized feature pyramid for object detection. IEEE Trans. Image Process. **32**, 4341–4354 (2023)
16. Zhao, Q., et al.: M2det: a single-shot object detector based on multi-level feature pyramid network. In: AAAI, pp. 9259–9266 (2019)
17. Liu, S., Huang, D., Wang, Y.: Learning spatial fusion for single-shot object detection. Arxiv. **1911.09516** (2019). <https://doi.org/10.48550/arXiv.1911.09516>
18. Chen, K., Cao, Y., Loy, C.C., Lin, D., Feichtenhofer, C.: Feature pyramid grids. Arxiv. **2004.03580** (2020) <https://doi.org/10.48550/arXiv.2004.03580>
19. Tan, M., Pang, R., Le, Q.V.: Efficientdet: scalable and efficient object detection. In: CVPR, pp. 10781–10790 (2020)
20. Qiao, S., Chen, L.C., Yuille, A.: Detectors: detecting objects with recursive feature pyramid and switchable atrous convolution. In: CVPR, pp. 10213–10224 (2021)
21. Yang, G., Lei, J., Zhu, Z., Cheng, S., Feng, Z., Liang, R.: AFPN: asymptotic feature pyramid network for object detection. In: SMC, pp. 2184–2189 (2023)
22. Fan, C., Li, Y., Wu, Y.: Multi feature fusion paper classification model based on attention mechanism. In: ICNLP, pp. 308–312 (2023)
23. He, S., Ye, X., Dou, L., Sakurai, T.: FIAMol-AB: feature fusion and attention-based deep learning method for enhanced antibiotic discovery. Comput. Biol. Med. **168**, 107762 (2024)
24. Adelson, E.H., Anderson, C.H., Bergen, J.R., Burt, P.J.: Pyramid methods in image processing. RCA Eng. **29**(6), 33–41 (1984)
25. Liu, W., et al.: Ssd: Single shot multibox detector. In: ECCV, pp. 21–37 (2016)
26. Lu, X., Wang, W., Shen, J., Crandall, D., Luo, J.: Zero-shot video object segmentation with co-attention siamese networks. IEEE Trans. Pattern Anal. Mach. Intell. **44**(4), 2228–2242 (2020)
27. Zhang, H., et al.: Resnest: split-attention networks. In: CVPR, pp. 2736–2746 (2022)
28. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR, pp. 770–778 (2016)
29. Lin, T.Y., et al.: Microsoft coco: common objects in context. In: ECCV, pp. 740–755 (2014)
30. Everingham, M., Van, G.L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. In: IJCV, pp. 303–338 (2010)
31. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: towards real-time object detection with region proposal networks. In: NeurIPS, p. 28 (2015)
32. Meng, Z., Xia, X., Xu, R., Liu, W., Ma, J.: HYDRO-3D: hybrid object detection and tracking for cooperative perception using 3D LiDAR. IEEE Trans. Intell. Veh. **8**(8), 4069–4080 (2023)
33. Lian, Z., Yin, Y., Zhi, M., Xu, Q.: PCBSNet: a pure convolutional bilateral segmentation network for real-time natural scene text detection. Electronics **12**(14), 3055 (2023)
34. Zhang, Y., et al.: Datasetgan: efficient labeled data factory with minimal human effort. In: CVPR, pp. 10145–10155 (2021)



Dual-Branch Collaborative Learning for Visual Question Answering

Weidong Tian^{1,2}, Junxiang Zhao¹, Wenzheng Xu¹, and Zhongqiu Zhao^{1,2,3(✉)}

¹ School of Computer Science and Information Engineering, Hefei University of Technology, Hefei, China

2021171161@mail.hfut.edu.cn

² Intelligent Manufacturing Institute of HFUT, Hefei, China

³ Guangxi Academy of Sciences, Guangxi, China

Abstract. Good visual question answering models can reason about the underlying relationships in the context of images and questions. Recently, some works have used graph-based methods for visual reasoning, but graph-based methods cannot perform better reasoning when the connection between the question statement and the visual object is unclear. In this paper, we design a dual-branch network based on collaborative learning that can simultaneously focus on relational reasoning and attention-based deep alignment between images and questions. The question-aware enhancement module we designed can better utilize question information, and the joint prediction module we designed can fully integrate the performance of the two branches. Extensive experimental results demonstrate that our proposed method outperforms the current state-of-the-art methods in terms of performance.

Keywords: VQA · Relational Reasoning · Attention · Collaborative learning

1 Introduction

Visual question answering (VQA) [1] is a task that involves multiple modalities of information, and has received extensive research attention in it in recent years, the goal of it is to answer questions related to the content of images correctly. Improving reasoning skills is a core step towards building a reliable VQA system. Graph-based reasoning methods [2–4] utilize context-aware graph networks to simulate the semantic relationships between visual objects, enabling effective execution of relational reasoning tasks. Attention-based methods [5–7] have been popular since it [8] was proposed in VQA. With the popularity of Transformer, an increasing number of methods make use of multiple Transformer-based attention modules [7, 9] which enable the model to more accurately focus on image areas and textual information relevant to the question, thereby enhancing the neural network's understanding and reasoning capabilities.

Although graph-based methods achieve good results in visual reasoning, we find that there are still some problems in them. When the objects included in the question are prominently present in the image, the model can better perform visual relationship

reasoning. As shown in Fig. 1(a), the bowl and corn appearing in the question are both prominently present in the image. At this time, the model can first focus on the corn, then the bowl on the right (not the bowl on the left), and finally the objects in the bowl to answer the question correctly. But when the question does not contain objects that are significantly present in the picture, as shown in Fig. 1(b), the graph-based methods can focus on various foreground objects in the picture, but the question asks “How is the weather?” At this time, the model needs to focus on the background in the image (Marked by a red frame in the picture) to answer the question correctly. There are certain difficulties in dealing with such problems for graph-based methods, while global attention-based methods can focus on the background to answer the question correctly.



Fig. 1. Samples in the GQA dataset.

To help the model solve the above problems and act like a human who can focus on the right areas when answering questions. We design a two-branch network named Relational Reasoning and Attention Collaborative Learning Network (RACN). One branch focuses on relational reasoning between images and questions, and the other branch uses a new Transformer-base architecture to achieve deep alignment of features between images and questions. The two branches serve as mutual supervision signals and learn from each other. In addition, by observing the dataset, we found that when multiple objects appear in the question and the orientation relationship between the objects needs to be understood, the graph-based method can significantly pay attention to these objects in the graph and answer the question correctly, so we designed a joint prediction module related to the number of question nouns to fully integrate the information of the two branches.

To sum up, the main contributions we have made can be summarized as follows:

- RACN’s two-branch model can fully learn the relationship reasoning and attention between images and questions.
- We propose the question-aware reinforcement module in the attention branch to deeply align image and question features.
- We proposed the joint predict module in order to fully integrate the performance of the two branches.

2 Related Work

2.1 Visual Question Answering

As a vital yet challenging multimodal task, VQA [1] recently has drawn more and more attention. In most VQA methods, the fusion and alignment of vision and language are fundamental techniques (MUTAN [10], MCB [11], MLB [12]). Attention mechanisms [8] are currently widely used in the field of VQA since they allow the model to focus on specific areas of the image or keywords in the question to better answer the question. Transformer-based models utilized intra-modal and inter-modal co-attention mechanisms for the purpose of selectively combining features from specific image regions and corresponding words. MCAN [7] designs a deep collaborative attention network to combine keywords in questions and key targets in images to refine the understanding of visual content and textual content.

2.2 Graph-Based Visual Relationship Reasoning

Graph networks, as described in [13], are robust frameworks that facilitate relational reasoning through message-passing techniques [14, 15]. The basic concept is to achieve communication between image regions, establishing rich contextual representations for these regions. A notable contribution to incorporating relational reasoning in VQA is [16], introducing Relational Networks that model relationships between all pairs of objects based on a given question [17] marks the inaugural effort in leveraging graph networks for VQA, merging dependency parses of questions with scene graph representations of abstract scenes. LCGN [2] constructs contextualized representations for objects within a visual scene, supporting relational reasoning. Meanwhile, [3] introduces a reinforced path routing approach that represents an input image through a structured visual graph. This method employs a reinforcement learning-based model to explore paths (sequences of nodes) over the graph, guided by an input sentence to infer reasoning results.

2.3 Collaborative Learning

Knowledge distillation [18] allows the student model to learn the knowledge taught by the teacher, but it is a two-stage process and the knowledge is transferred in one direction. There is no distinction between teachers and students in collaborative learning [19]. Multiple sub-networks perform collaborative training together. The method proposed in this article does not require a large model to be trained in advance as a teacher. Two models of the same scale can be trained simultaneously and learn from each other to improve the overall performance of the model.

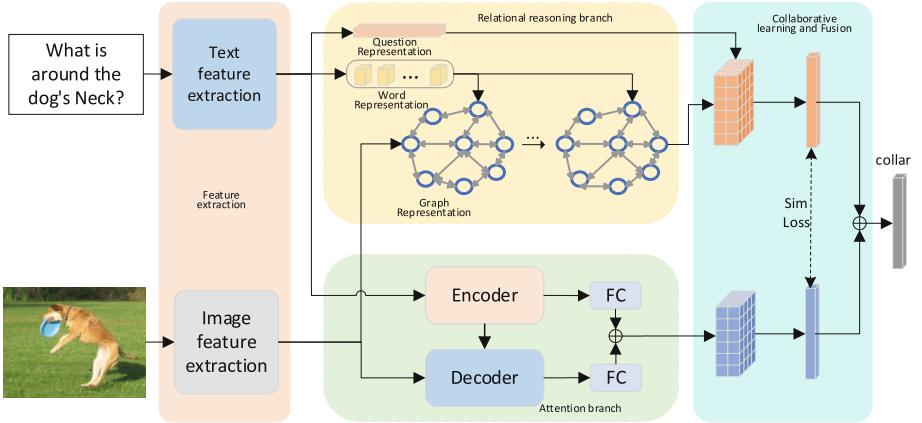


Fig. 2. The overall framework of the proposed method RACN.

3 Method

3.1 Overview

The purpose of VQA is to obtain the answer $a \in A$ based on the input image I and question Q , where A represents the set of candidate answers. In this section, the proposed RACN method will be discussed in detail, and the overall framework is shown in Fig. 2. The overall framework of **the proposed** method RACN. The feature extractor module will extract the feature representation of the image and the question, and send them to the relational reasoning-based branch(r-branch) and the attention-based branch(a-branch) respectively. The feature vectors output by the two branches are subjected to softmax to obtain the prediction vector, and the prediction vectors learn from each other. Finally, the joint prediction is used to obtain the predicted answer.

3.2 Question Representation and Image Representation

We utilize Faster R-CNN to extract regional features from images and use Resnet-101 [20] to extract grid features from images following [21]. Then we use image region features to construct a visual graph $G = \{V; E\}$ as [2] for r-branch, we denote a node v_i with local features $l_i \in \mathbb{R}^{d_l}$, representing appearance details, and spatial features $b_i \in \mathbb{R}^{d_b}$, representing its position and size. These features are concatenated and projected into a common space \mathbb{R}^d via linear mapping, expressed as $v_i = W_v[l_i; b_i]$, where $W_v \in \mathbb{R}^{d \times (d_l+d_b)}$. The visual graph is fully connected, implying an undirected edge e_{ij} exists between every pair of nodes v_i and v_j . In addition, we employ glove to extract question semantic features, and we use LSTM to encode question sequence features as q_a for a-branch and use BiLSTM to obtain a sequence $\{h_s\}_{s=1}^S$ and a summary vector q_r for r-branch.

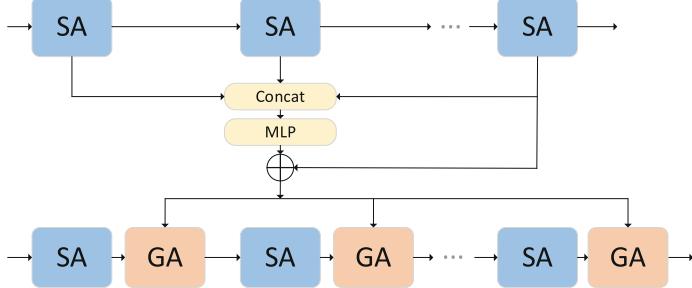


Fig. 3. The Module of Question-Aware Reinforcement

3.3 Relational Reasoning Branch

We employ the language-guided graph convolution [2] to learn context-aware visual representations for each question $q \in Q^f$. The branch utilizes iterative message passing, conditioned on the input text, to ultimately generate a context-aware representation v_i^{out} for each node i . In this process, each node i is characterized by both a local feature v_i^{loc} and a context feature $v_{i,t}^{ctx}$. For each node v_i , the local features of each node are represented as $v_i^{loc} = v_i$ and the Initialized context features are represented as $v_{i,0}^{ctx} \in R^{d \times 1}$ from a learned parameter as [2].

During each iteration, a message vector $m_{j,i}^{(t)}$ is transmitted from each node j to every other node i .

$$m_{j,i}^{(t)} = M(v_j^{loc}, v_{j,t-1}^{ctx}, v_i^{loc}, v_{i,t-1}^{ctx}, c_t) \quad (1)$$

where $M(\cdot)$ is message passing function and the c_t is the textual command.

Then node i gathers the message vectors to refresh its context feature, denoted as $v_{i,t}^{ctx}$.

$$v_{i,t}^{ctx} = Q(v_{i,t-1}^{ctx}, \Sigma_j m_{j,i}^{(t)}) \quad (2)$$

where $Q(\cdot)$ is context update function.

To determine the number of iterations T_q , we utilize the Spacy tool [22] to perform Part-of-speech (POS) tagging on the question and calculate the total number of nouns present within it. The local feature v_i^{loc} and the final context feature $v_{i,T}^{ctx}$ are combined into a joint context-aware feature v_i^{out} . Then v_i^{out} and vector q_r are used to aggregate visual information to obtain the predicted answer distribution.

$$y = P(v_i^{out}, q_r) \quad (3)$$

where $P(\cdot)$ is feature fusion function.

3.4 Attention Branch

In recent years, transformer-based VQA models have shown superior effectiveness compared to other models. In this branch, we adopt a Transformer-based method [7] that uses self-attention and guided attention.

However, since the questions that require reasoning contain more spatial and other reasoning information, simply outputting the final layer of the encoder in the transformer to the decoder will ignore the low-level semantic information. Low-level semantic information refers to information that represents a simple abstract representation and understanding of a question, while high-level semantic information represents information that provides a deep understanding and reasoning of a question. To deal with this problem, [23] and [24] proposed meshed cross-attention and scale-aware reinforcement respectively. we further proposed an effective and simple question-aware reinforcement (QAR) module to address the above limitations by incorporating all features of each question from each encoding layer into the top features. The structure of the QAR is shown in the Fig. 3.

For simplicity, given the question output features (T_1, T_2, \dots, T_n) of each encoder layer, we first concatenate them all together.

$$T_L = (T_1, T_2, \dots, T_n) \quad (4)$$

Then, to combine both low and high-level semantic information, we employ a Multi-Layer Perceptron (MLP) that has the capability to adjust feature weights dynamically in each of its layers. Since the output of the top encoder layer contains more semantic information, we add features T_n to T'_L to obtain the final semantic augmented encoded features T' .

$$T'_L = \text{MLP}(T_L) \quad (5)$$

$$T' = T_n + T'_L \quad (6)$$

3.5 Dual-Branch Collaborative Learning

The r-branch focuses on the relationship reasoning between objects, and the a-branch focuses on feature alignment between questions and images. The output results of the two branches can serve as supervision signals for each other. The loss function is as follows:

$$L = L_{CE}^r + L_{CE}^a + L_{sim} \quad (7)$$

$$L_{sim} = \text{KL}(P_r || P_a) + \text{KL}(P_a || P_r) \quad (8)$$

where L_{CE}^a and L_{CE}^r represent the cross-entropy losses for the two distinct branches. Additionally, L_{sim} expresses the collaborative learning loss, which employs the Kullback-Leibler divergence denoted by $\text{KL}(\cdot || \cdot)$.

3.6 Joint Predict

Since the more information the question contains, the more iterations the model requires to obtain relational reasoning results, we design a joint prediction (JP) module that

is related to the number of iterations T_q . The final prediction is a combination of the predictions from both branches:

$$P_f = (1 - \lambda)P_r + \lambda P_a \quad (9)$$

$$\lambda = \frac{1}{T_q + \alpha} \quad (10)$$

where P_r and P_a represent the prediction results of r-branch and a-branch respectively, $T_q (T_q \geq 1)$ is the number of iterations, and α is a hyper-parameter.

4 Experiments

4.1 Datasets and Implementation Detail

Datasets: We validate our model using the GQA dataset [25], which contains a large number of visual reasoning problems. It has over 110k real-world images and 2.2 million questions. The dataset is appropriately segmented into training, validation, and test subsets. Furthermore, we also use the GQA-OOD dataset [26] to test the robustness of the model. The GQA-OOD dataset changes the data distribution of the GQA validation set and test set, and it can better detect the model’s reasoning ability.

Implementation Details: We conduct experiments on two Nvidia RTX 3080Ti GPU. For the visual features, The local features d_l is assigned a dimensionality of 2048, while the context features d_b and the common space d are set to 96 and 512 dimensions respectively. We retain the 48 bounding boxes with the highest confidence scores as [27]. And for the dimensionality d_x of grid features is set to 2048. We set the dimensions of question feature d_y and answer feature d_z are all 512. The multi-head attention mechanism has a dimensionality d of 512 and utilizes 8 heads. The network employs a batch size of 128 and a learning rate of 0.001, leveraging the Adam optimizer for training.

4.2 Ablation Experiment

We conduct extensive experiments to evaluate the effectiveness of the proposed RACN method. We use LCGN and MCAN as baselines in the two branches of RACN respectively. A question-aware reinforcement module that is more consistent with relational reasoning is designed in the a-branch, and a joint prediction module related to the question is designed in the final joint prediction. To understand the distinct contributions of each module method towards the final result, we devised a range of ablation experiments using the GQA dataset. The experimental results can be found in Table 1 and Table 2.

As can be seen from the first four rows of Table 1, the two branches trained using the collaborative learning method have greatly improved compared with their baselines, indicating that the collaborative learning training method is effective in improving model performance. As can be seen from the fifth and sixth rows, the question-aware reinforcement module we designed can make full use of the low-level and high-level semantic information of the question to improve model performance. From the sixth and seventh

Table 1. Ablation study on GQA test-dev with different strategies. RB represents r-branch only, AB represents a-branch only, QAR represents question-aware reinforcement module, and JP represents joint prediction module.

Method	RB	AB	QAR	JP	Acc
LCGN(baseline1)					57.07
MCAN(baseline2)					57.40
RACN	r-branch	✓			57.93
	a-branch		✓		57.82
	w/o QA	✓	✓		58.08
	w/o JP	✓	✓	✓	58.11
	joint	✓	✓	✓	58.18

Table 2. Performance on GQA validation split.

Model	α	Accuracy
RACN	0.3	64.73
RACN	0.5	64.90
RACN	0.6	64.94
RACN	0.7	64.86
RACN	1.0	64.51

lines, we can find that adding the joint prediction module can improve the performance again, indicating that the model prediction is related to the specific problem and the output vectors cannot be predicted simply by adding them.

Furthermore, we adjust the parameter α to control the proportion of each branch in the joint prediction of RACN. By observing the dataset, we found that when the question contains more objects with spatial relationships, the model requires more reasoning about the relationships between objects. By controlling the parameter α , the joint prediction proportion of the r-branch and the a-branch can be controlled. Table 2 shows the experimental results related to various α values in the RACN network, and extensive experiments show that the model achieves the best performance when α is set to 0.6.

4.3 Comparison with SOTAs

We assess our model on the GQA dataset, comparing it with state-of-the-art models such as attention-based methods (BUTD, MAC, and BAN), and graph-based methods (LCGN, GRNs, and GMA). Table 3 shows the experimental results on the GQA test dataset. As demonstrated in Table 3. RACN has significantly improved its Binary, Open and Accuracy indicators compared with other models. Compared with the most advanced graph-based method GMA model, RACN has improved by 2.17, 1.56 and 0.92 points respectively, this shows that RACN has better reasoning capabilities.

Table 3. Comparison of our RACN model with other VQA methods on GQA test-dev dataset.

Method	Binary	Open	Vailed	Plausibility	Consistency	Distribution	Accuracy
BUTD [5]	66.64	34.73	96.18	84.57	78.71	5.98	49.74
MAC [28]	71.23	38.91	96.16	84.48	81.59	5.34	54.06
BAN [29]	76.00	40.41	91.70	85.58	96.16	10.52	57.1
LCGN [2]	73.77	42.33	84.68	84.81	96.48	4.70	57.07
GRNs [30]	74.93	41.24	96.14	84.68	87.41	5.76	57.04
GMA [31]	73.20	42.30	96.41	85.05	83.95	5.56	57.26
RACN(ours)	75.37	43.86	96.11	84.80	95.40	5.65	58.18

Table 4. Comparison with other methods on the GQA-OOD test dataset

Model	Acc-all	Acc-tail	Acc-head
BUTD [5]	46.4	42.1	49.1
BAN [29]	50.2	47.2	51.9
MCAN [7]	50.8	46.5	53.4
LCGN [2]	52.8	47.2	56.1
MMN [27]	52.7	48.0	55.5
RACN(ours)	53.3	48.2	56.0

Then we also explored the performance of RACN on the GQA-OOD dataset. Table 4 shows the experimental results on the GQA-OOD test dataset. Compared with other models, RACN achieves the best performance on Acc-all and Acc-tail indicators, RACN also achieved competitive performance on the ACC-head indicator, proving the effectiveness of RACN in terms of robustness and reasoning capabilities.

4.4 Qualitative Analysis

As shown in Fig. 4, the method we proposed can answer the question more accurately than other methods. It can be found from pictures (1)–(4), when the semantic information contained in the question is not obviously closely related to the objects in the image, the LCGN method performs poorly. The RACN method deeply understands the semantic information of the question and the visual information of the image, while taking into account the relationship between objects and the global attention of the entire image, and obtains the correct answer.

It can also be found from pictures (5)–(8) that when the question is more complex, when the question contains multiple objects and the relationship between the objects needs to be understood, the RACN method can correctly understand the complex relationship between the objects and give the correct answer to the question.



Fig. 4. Performance presentation for RACN, LCGN and MCAN.

5 Conclusion

In this paper, we introduce a new RACN model that utilizes collaborative learning to effectively utilize relational reasoning and attention information between images and questions. The two branches combine views from different perspectives to improve the model's generalization ability. To make full use of low-level and high-level semantic information, we propose the QAR module. To better fuse the prediction results of the two branches, we propose the JP module, which can dynamically fuse the results of the two branches according to different problems. Extensive experiments show that the RACN method can achieve competitive results on GQA and GQA-OOD datasets, and ablation experiments also show that our proposed modules are effective.

Although RACN performs well on the visual question answering task, the dual-branch network structure undoubtedly increases the number of parameters and computational complexity during inference. Future research can consider selecting the most suitable branch for inference prediction based on different question types after the network training is completed.

Acknowledgment. This work was supported by the National Natural Science Foundation of China under Grant 61976079, the Guangxi Key Research and Development Program under Grant AB22035022.

References

1. Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C.L.: Vqa: visual question answering. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2425–2433 (2015)
2. Hu, R., Rohrbach, A., Darrell, T., Saenko, K.: Language conditioned graph networks for relational reasoning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 10294–10303 (2019)

3. Jing, C., Jia, Y., Wu, Y., Li, C., Wu, Q.: Learning the dynamics of visual relational reasoning via reinforced path routing. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, pp. 1122–1130, 2022
4. Jing, C., Jia, Y., Wu, Y., Liu, X., Wu, Q.: Maintaining reasoning consistency in compositional visual question answering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5099–5108 (2022)
5. Anderson, P., et al.: Bottom-up and top-down attention for image captioning and visual question answering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6077–6086 (2018)
6. Yang, Z., He, X., Gao, J., Deng, L., Smola, A.: Stacked attention networks for image question answering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 21–29 (2016)
7. Yu, Z., Yu, J., Cui, Y., Tao, D., Tian, Q.: Deep modular coattention networks for visual question answering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6281–6290 (2019)
8. Shih, K.J., Singh, S., Hoiem, D.: Where to look: focus regions for visual question answering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4613–4621 (2016)
9. Tan, H., Bansal, M.: Lxmert: Learning cross-modality encoder representations from transformers. arXiv preprint [arXiv:1908.07490](https://arxiv.org/abs/1908.07490) (2019)
10. Ben-Younes, H., Cadene, R., Cord, M., Thome, N.: Mutan: multimodal tucker fusion for visual question answering. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2612–2620 (2017)
11. Fukui, A., Park, D.H., Yang, D., Rohrbach, A., Darrell, T., Rohrbach, M.: Multimodal compact bilinear pooling for visual question answering and visual grounding. arXiv preprint [arXiv:1606.01847](https://arxiv.org/abs/1606.01847) (2016)
12. Kim, J.H., On, K.W., Lim, W., Kim, J., Ha, J.W., Zhang, B.T.: Hadamard product for low-rank bilinear pooling. arXiv preprint [arXiv:1610.04325](https://arxiv.org/abs/1610.04325) (2016)
13. 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)
14. Battaglia, P.W., et al.: Relational inductive biases, deep learning, and graph networks. arXiv preprint [arXiv:1806.01261](https://arxiv.org/abs/1806.01261) (2018)
15. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint [arXiv:1609.02907](https://arxiv.org/abs/1609.02907) (2016)
16. Santoro, A., et al.: A simple neural network module for relational reasoning. Adv. Neural Inf. Process. Syst. **30** (2017)
17. Teney, D., Liu, L., van Den Hengel, A.: Graph-structured representations for visual question answering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2017)
18. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint [arXiv:1503.02531](https://arxiv.org/abs/1503.02531) (2015)
19. Song, G., Chai, W.: Collaborative learning for deep neural networks. Adv. Neural Inf. Process. Syst. **31** (2018)
20. 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)
21. Jiang, H., Misra, I., Rohrbach, M., Learned-Miller, E., Chen, X.: In defense of grid features for visual question answering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10267–10276 (2020)
22. Honnibal, M., Montani, I.: Spacy 2: natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. To Appear **7**(1), 411–420 (2017)

23. Cornia, M., Stefanini, M., Baraldi, L., Cucchiara, R.: Meshedmemory transformer for image captioning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10578–10587 (2020)
24. Zeng, P., Zhang, H., Song, J., Gao, L.: S2 transformer for image captioning. In: Proceedings of the International Joint Conferences on Artificial Intelligence, vol. 5 (2022)
25. Hudson, D.A., Manning, C.D.: Gqa: a new dataset for real-world visual reasoning and compositional question answering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6700–6709 (2019)
26. Kervadec, C., Antipov, G., Baccouche, M., Wolf, C.: Roses are red, violets are blue... but should vqa expect them to? In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2776–2785 (2021)
27. Chen, W., Gan, Z., Li, L., Cheng, Y., Wang, W., Liu, J.: Meta module network for compositional visual reasoning. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 655–664 (2021)
28. Hudson, D.A., Manning, C.D.: Compositional attention networks for machine reasoning. arXiv preprint [arXiv:1803.03067](https://arxiv.org/abs/1803.03067) (2018)
29. Kim, J.H., Jun, J., Zhang, B.T.: Bilinear attention networks. Adv. Neural Inf. Process. Syst. **31** (2018)
30. Guo, D., Xu, C., Tao, D.: Bilinear graph networks for visual question answering. IEEE Trans. Neural Netw. Learn. Syst. (2021)
31. Cao, J., Qin, X., Zhao, S., Shen, J.: Bilateral crossmodality graph matching attention for feature fusion in visual question answering. IEEE Trans. Neural Netw. Learn. Syst. (2022)



SCAI: A Spectral Data Classification Framework with Adaptive Inference for Rapid and Portable Identification of Chinese Liquors

Yundong Sun¹, Yansong Wang², Xuguang Xu³, Dongjie Zhu²(✉), and Zhaoshuo Tian¹

¹ Department of Electronic Science and Technology, Harbin Institute of Technology, Harbin, China

² School of Computer Science and Technology, Harbin Institute of Technology, Weihai, China
zhudongjie@hit.edu.cn

³ Media Convergence Center of Huancui District, Weihai, China

Abstract. To achieve accurate, rapid, and portable identification of Chinese liquors based on spectral technology, this paper leverages the adaptive inference technique to solve the computational efficiency problems of existing deep learning models. We propose a Spectral data Classification framework with Adaptive Inference (SCAI), which leverages the Early-exit paradigm and can allocate appropriate computations for different samples on different portable devices. It can adjust the computation layers in each neural network block for different spectral curve positions of liquors so that it can focus more on important information. To our knowledge, this paper is the first attempt to leverage adaptive inference for liquor identification. The experimental results show that our method can achieve higher identification performance (+6% – +13% under the same budget) with less computational budget (1/6 for the same performance) than existing methods.

Keywords: Adaptive Inference · Early Exit Neural Networks · Liquor Identification · Spectral Detection

1 Introduction

Currently, the low costs, high profits, and challenges in identification associated with liquor counterfeiting have made it a concentrated and high-incidence area of food safety crimes. Therefore, how to achieve accurate, portable, and rapid liquor identification has emerged as a critical issue requiring immediate attention [1].

In recent years, deep learning models combined with spectroscopy technology have become a new powerful weapon for liquor identification [2, 3]. Compared to traditional methods, deep learning models can reduce the dependence on feature engineering and extract high-level features directly from the raw data [4]. Therefore, while improving the identification performance, the difficulty of data preprocessing is also greatly reduced, facilitating rapid identification to some extent.

However, many critical issues of deep learning identification algorithms still need to be solved urgently before applying them to portable devices, becoming obstacles to

realizing rapid and portable liquor identification. The following are the two most serious problems:

- High computational budget

More powerful deep learning models have been continuously proposed in recent years, and their identification performance has been constantly promoted. However, the powerful performance of these models comes at the cost of a deeper and more complex structure [5]. Moreover, the inference of most models is performed statically, i.e., once the training is completed, the structure and parameters are fixed, and its computational load cannot be dynamically adjusted according to the computational capacity of the device. This severely limits its application, making it impossible for portable devices, which have low performance or require strict control of power consumption.

- Low computational efficiency

Not all samples need to allocate all computations for confident identification [5]. Some liquor samples are easily identifiable (called “easy” samples), while others are difficult (called “hard” samples). Similarly, just as informative features in computer vision may only exist in certain regions of an image [5], for the spectral curve (Fig. 1), the informative features like Raman and fluorescence peaks are only located in some regions, while other parts are less class-discriminative and may contain redundant features that degrade identification performance. Current static deep learning methods fail to discern informative features from uninformative ones and allocate suitable computing resources, leading to poor efficiency and performance degradation.

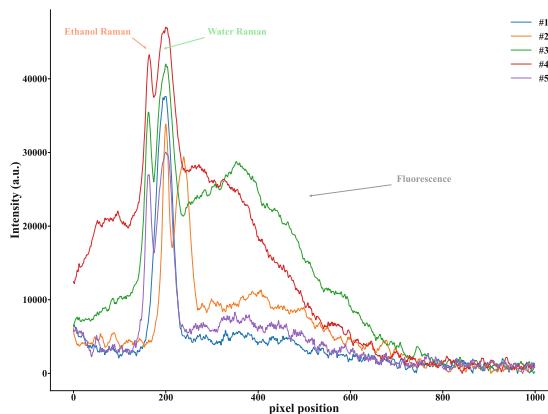


Fig. 1. Some representative spectral curves of Chinese liquors.

Considering the fact that not all inputs need to allocate the same computation to yield a confident prediction, the adaptive inference technique is becoming a promising strategy for efficient deployment under limited computing resources. Among them, the Early-exit strategy is a simple yet surprisingly effective paradigm, which allows “simple” samples to output results at shallow exits without continuing to perform subsequent computations

[5, 6]. However, there is no relevant research has been devoted to the portable and rapid liquor identification scenario.

Aiming at the computational efficiency problems of deep learning models in the face of portable and rapid liquor identification, this paper first attempts to leverage adaptive inference for liquor identification. We leverage Early-exit architecture, place intermediate classifiers at different depths of the architecture, and the model stops inference and outputs the results when the prediction confidence at the current classifier position reaches a preset threshold. Additionally, we also propose a Position-Adaptive residual network (PA-ResNet). It can adjust the number of computation layers in each block at different positions of the curve, so it can not only pay more attention to the information of important positions of the curve (e.g. Raman peak) but also can accurately allocate the appropriate amount of computational budget based on identification performance and computing resources.

2 Methodology

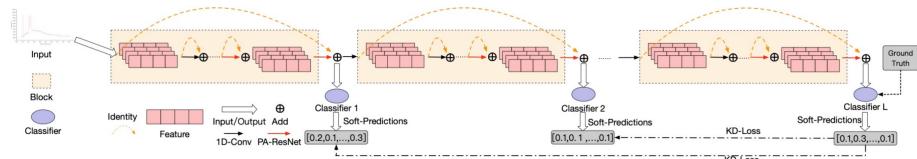


Fig. 2. The schematic diagram of the overall architecture of SCAI.

2.1 SCAI: Early-Exit with Self-distillation

Architecture. As shown in Fig. 2, the overall architecture we proposed consists of several blocks, and an intermediate classifier is inserted between every two blocks. In this paradigm, the early classifier will enforce the feature map in the early stage to be as beneficial as possible to the prediction results of the current classifier during the training phase, which is unbeneficial to the subsequent classifiers. Therefore, it will deteriorate the final performance of the overall architecture. To alleviate this problem, we propose a scheme of hierarchical ResNet. Specifically, we first adopt the ResNet structure inside each block, so that the block’s original input information can be retained while learning higher-order features during the layer-by-layer forward process. Similarly, we also leverage the ResNet structure between different blocks, so that each classifier can obtain the shallow information at different distances or even the raw information while obtaining the high-level information learned by the previous blocks.

We denote the feature map output by the s -th layer network in the l -th block as x_s^l , and the feature map output by the last layer network in the l -th block as x^l , which is also the input of the l -th classifier. The original spectral input features are represented as x_0^0 , so:

$$x_s^l = F_s^l(x_{s-1}^l) + x_{s-1}^l, \quad x^l = F^l(x^{l-1}) + x^{l-1} \quad (1)$$

where $F_s^l(\cdot)$ represents the function of the s -th layer network in the l -th block, $F^l(\cdot)$ is the function between the l -th block and the $(l + 1)$ -th block. In addition, to make the feature dimension and channel design between blocks more flexible, we add a feature transfer layer between every two blocks:

$$x_0^{l+1} = \text{trans}(x^l) = \text{Conv1d}(x^l) \quad (2)$$

where x_0^{l+1} represents the input of the $(l + 1)$ -th block. For intermediate classifiers and final classifiers, we employ the simplest but most effective SoftMax classifier:

$$\hat{y}_l = \text{softmax}(x^l) \quad (3)$$

Training Strategy. Ideally, classifiers located in the deep layers will perform better than those located in the shallower layers. Without the knowledge guidance of the later classifier during the training phase, the early classifier cannot achieve high accuracy as soon as possible. However, we prefer to get as higher performance as possible with a small computational budget. In this section, we propose a training paradigm of self-distillation learning, through which the output of the deepest classifier performs soft supervision on the previous classifiers to maximize the performance and training speed of the whole architecture, especially the shallow classifier.

Specifically, there are two schemes: One scheme is the step-by-step distillation, that is, leveraging the soft prediction of the next classifier \hat{y}_{s+1} to guide the current output \hat{y}_s . Another scheme is more intuitive, which directly guides the output of each intermediate classifier with the help of the output of the last classifier. After experimental verification, we find that the performance of the latter is better. Therefore, we adopt the latter scheme:

$$L_{\text{task}}^{(l)} = \begin{cases} KL(\hat{y}_l, \hat{y}_L) & , l < L \\ \sum_{i=1}^C y_i \log(\hat{y}_l) & , l = L \end{cases} \quad (4)$$

where $L_{\text{task}}^{(l)}$ is the classification task loss of the l -th classifier, C represents the number of node labels, y_i is the probability that the true label of the node is i , y_i is 1 or 0, KL is the Kullback-Leibler Divergence, and L is the number of blocks.

2.2 SCAI+: SCAI with PA-ResNet

As the previous analysis of Fig. 1, informative features may only exist in certain positions of the spectral curve. Based on this analysis, in this section, we propose a Position-Adaptive Residual Network (PA-ResNet). It can adjust the number of computation layers in each block at different positions of the curve, so it can not only pay more attention to the information of important positions of the curve (e.g.: Raman peak), but also can accurately allocate the appropriate amount of computational budget based on task performance and computing resources, greatly improving the efficiency and alleviating the sensitivity of intermediate classifiers' settings. In this paper, we denote the enhanced SCAI with PA-ResNet as SCAI+.

Figure 3 shows the schematic diagram of the PA-ResNet model structure within a block. Inspired by ACT [7], we add the prediction branch of the halting score based

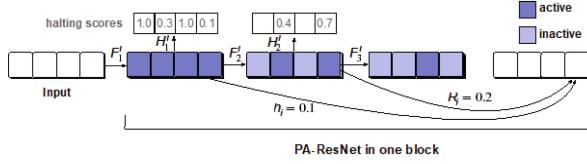


Fig. 3. The schematic diagram of the PA-ResNet model structure within a block.

on the Residual unit. But our purpose is different from that of ACT. ACT wants to control where the inference of the model should stop through the halting score. It is oriented to the sample sequence, and the calculation of the halting score is oriented to each sample. In this paper, our purpose is to automatically choose to allocate different amounts of computation in a block at different positions of the curve, expecting to allocate more computation in more “class-discriminative” positions, while avoiding too much computation in “class-indiscriminative” positions.

The Residual unit structure of PA-ResNet is shown in Fig. 4 (a). We define the position of the feature map of the spectral curve as active or inactive in each layer of the network. The halting score of the feature map in the active position does not reach the preset threshold ε , which means that the feature in this position needs to be used to extract higher-order features in the next Residual unit; the halting score of the feature in the inactive position reaches the preset threshold ε , which means that the feature in this position is sufficient to meet the classification requirements of the intermediate classifier in this block. If the extraction of high-order features is continued, the classification performance will be weakened, the computational performance will be wasted, and the inference latency will be increased. Therefore, for inactive information, only a copy operation is required.

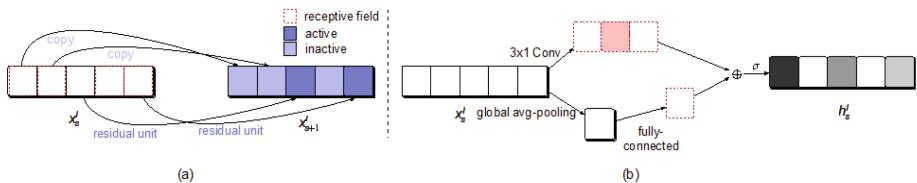


Fig. 4. (a) is the Residual unit structure of PA-ResNet. (b) is the calculation of the halting score, which is obtained through the fusion of local information and global information.

The calculation of the halting score is shown in Fig. 4 (b). We believe that whether the feature information of the current position of the curve needs to be stopped depends not only on the current position and its local information but also on the global information of the whole curve. Therefore, we design a convolutional neural network to capture the current position and its local information, while leveraging global avg-pooling and fully connected layers to capture the global information of the whole curve. Finally, we use two kinds of information to calculate the halting score. Specifically, it can be expressed

by the following equation:

$$H_s^l(x) = \sigma \left(W_s^l * x + \tilde{W}_s^l \text{pool}(x) + b_s^l \right) \quad (5)$$

where $*$ represents a 3×1 one-dimensional convolutional neural network with 1 channel, pool is global avg-pooling, W_s^l , \tilde{W}_s^l and b_s^l are all learnable parameter matrices of the s -th layer network in the l -th block, $\sigma(x) = \frac{1}{1+\exp(-x)}$. In particular, we force the halting score of the last layer in each block to 1:

$$h_s^* = H_s^*(x_s^*), s = 1 \dots (S - 1) \quad (6)$$

where h_s^* represents the halting score of the s -th layer network in any block, and S is the number of layers in the current block, $h_S^* = 1$.

For the information of a certain position in the active state, the number of computational layers (N) in the inference process can be obtained by the following equation:

$$N = \min\{n \in \{1 \dots S\} : \sum_{s=1}^n h_s \geq 1 - \varepsilon\} \quad (7)$$

We also define retainer R :

$$R = 1 - \sum_{s=1}^{N-1} h_s \quad (8)$$

Further, the distribution of the halting score is given by:

$$p_s = \begin{cases} h_s & s < N \\ R & s = N \\ 0 & s > N \end{cases} \quad (9)$$

Finally, the final output feature of each block can be given by the weighted summation of the halting score:

$$\text{output} = \sum_{s=1}^S p_s x_s \quad (10)$$

At the same time, we also expect that while maintaining the performance of the model, the amount of computation is as small as possible, that is, the smaller the ponder cost, the better. Therefore, we define the loss function of the model's computational cost L_{cost} and add it to the model training loss:

$$L = L_{task} + \gamma L_{cost} = \sum_{l=1}^L L_{task}^{(l)} + \gamma \sum_{l=1}^L \rho^l \quad (11)$$

where $\gamma \geq 0$ is a regularization coefficient that controls the trade-off between optimizing the original loss function and the ponder cost. ρ^l is the ponder cost of the l -th block.

3 Experiment

3.1 Experiment Instrument

The physical diagram of the experimental system is shown in Fig. 5(a). The system includes a laser spectrometer and a smart terminal. The size of our self-developed micro-spectrometer is 103 mm × 58 mm × 25 mm. The laser diode has a wavelength of 405 nm and can be set to continuous or pulse output mode as required. Fig. 5(b) shows the internal structure of the experimental system. When conducting experiments, we put liquor samples in a clear glass bottle, which were placed into the shading box for testing. The spectral data is transmitted to the smart terminal through the wireless router and our model is deployed on smart terminals.

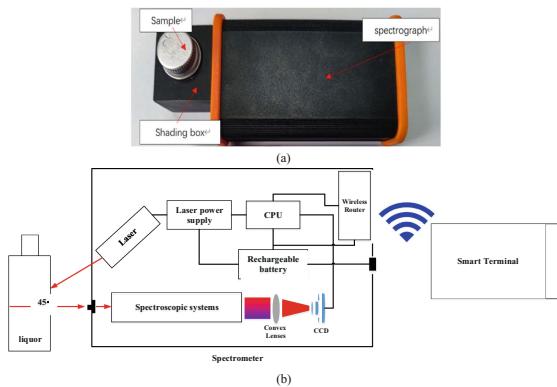


Fig. 5. (a) The physical diagram and (b) the internal structure of the experimental system.

3.2 Datasets

The tested liquors were selected from 12 brands in the Chinese market, including 3 Maotai-Flavor liquors, 3 Luzhou-Flavor liquors, and 6 Fen-Flavor liquors. The names and numbers are Beijing Erguotou (# 1), Weihai Wei Shao Guo (# 2), Clove Love (# 3), Raw Pulp (# 4), Beidacang (# 5), Bitter Mustard (# 6); Jing Zhi Bai Gan (# 7), Hengshui Lao Bai Gan (# 8), Lao Jiu Hu (# 9), Niu Lan Shan Aged (# 10), Du Er Jiu (# 11) and Xiao Lang Jiu (# 12). 100 data were taken at different times from different samples for each liquor. To ensure the accuracy and consistency of the measurement results, we must first calibrate the laser intensity of the instrument, and the sample used for calibration is pure water. All obtained data are normalized and then input into different models for experiments.

3.3 Baselines

We compare our method with commonly used methods in this field, including: (1) classic machine learning methods, such as Random Forest (RF) [8], Support Vector Machine

(SVM) [9], and (2) neural network methods, such as Backpropagation neural network (BP) [10], 1D-CNN [3], 1D-ResNet [11], DRSN-CW [12] and DRSN-CS [12]. We refer to the original papers to set up the above methods, and the grid search algorithm is leveraged to determine the hyperparameters. We removed the self-distillation training paradigm based on SCAI + and obtained SCAI+_{w/oKD}. Comparing it with SCAI +, we can see the superiority of the self-distillation training paradigm.

3.4 Common Performance Result

We first verify the performance of different methods in ideal scenarios (without limitation of computational budget). Specifically, we split the data according to train: validation: test = 2: 4: 4, and the experimental results are shown in Table 1.

Table 1. Comparison results of prediction experiment(%) with 20% training samples.

Methods	Macro-F1	Micro-F1	AUC
RF	52.52 ± 3.06	55.56 ± 3.05	75.93 ± 1.42
1D-CNN	73.25 ± 1.49	78.21 ± 1.16	97.74 ± 0.11
SVM	87.35 ± 0.85	90.40 ± 0.81	76.39 ± 0.39
BP	88.14 ± 3.79	90.83 ± 3.17	99.11 ± 0.51
DRSN-CW	89.63 ± 3.63	90.71 ± 2.83	94.97 ± 1.53
DRSN-CS	90.12 ± 4.01	91.15 ± 3.58	95.21 ± 1.94
1D-ResNet	93.86 ± 5.65	95.25 ± 3.76	99.00 ± 1.47
SCAI	95.27 ± 3.67	95.65 ± 2.84	99.40 ± 0.96
SCAI+ _{w/oKD}	94.97 ± 3.00	95.33 ± 2.41	99.71 ± 0.23
SCAI +	96.42 ± 2.25	96.48 ± 2.16	99.79 ± 0.45

From Table 1, we can find that our methods SCAI+_{w/oKD}, SCAI, and SCAI + have achieved the best results on all metrics, which reflects the superiority of our proposed Early-exit computing architecture. We believe that the noise reduction effect of the PA-Resnet structure on redundant features makes SCAI + focus on “class-discriminative” features and further improves the model performance compared to SCAI. The self-distillation module also contributed to the competitive advantage of SCAI + in comparison with SCAI+_{w/oKD}.

The performance of neural network methods is generally better than that of traditional machine learning methods. The powerful feature extraction capabilities of neural networks have greatly improved their performance. 1D-CNN makes the feature degenerate after the transfer of the multi-layer networks [11], and the 1D-ResNet with the residual structure can alleviate this problem. This is why all blocks of our SCAI model in this paper are based on 1D-ResNet, and we also insert the residual structures between different blocks. With 20% training data, the threshold units proposed by DRSNs (DRSN-CW and DRSN-CS) cannot learn reasonable thresholds, so their performance is not satisfactory compared with 1D-ResNet.

3.5 Application Performance Result

We also compare the proposed method with some representative neural network methods in two application scenarios, anytime prediction and budgeted batch prediction, respectively. To avoid the phenomenon that baseline methods are inferior to our methods due to insufficient training data (as shown in Table 1). In this experiment, we split the data according to train: validation: test = 8:1:1. We obtain the performance result with different computational budgets as shown in Fig. 6.

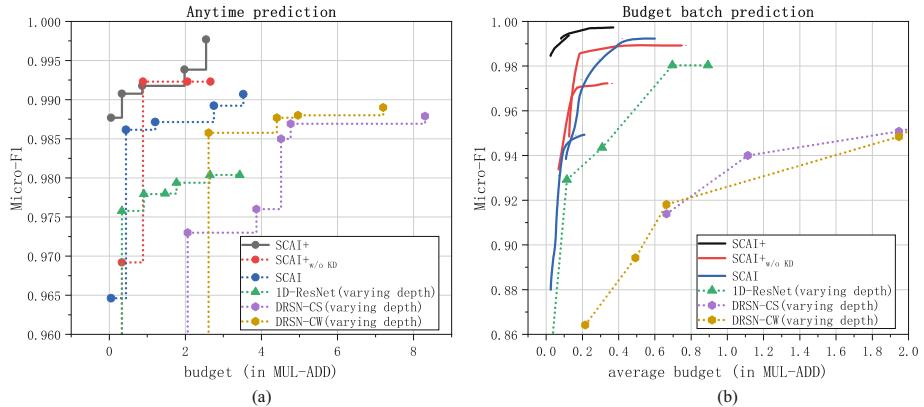


Fig. 6. The performance result of different methods with different computational budgets in the (a) anytime prediction and (b) budgeted batch prediction scenarios.

Anytime Prediction. In this scenario, the neural network model needs to give the prediction result of a single spectral sample with minimal latency. This procedure is usually done by different portable devices. The adaptive inference computing framework proposed in this paper can be deployed to different devices after one-time training and maximize performance. Fig. 6 (a) shows the performance result of different methods with different computational budgets in the anytime prediction scenario. It can be seen obviously that SCAI, SCAI +, and SCAI+_{w/o KD} outperform other baseline methods by a large margin. This shows that the Early-exit computing architecture proposed in this paper has a significant performance improvement compared to other methods. Let us focus on the left half of Fig. 6 (a). We can find that SCAI + and SCAI have more significant advantages than other methods. This phenomenon reflects that the training paradigm of self-distillation can significantly improve the performance of the classifier in the shallow layer of the architecture so that the architecture can achieve higher performance with as fewer computational budget as possible.

As expected, SCAI + achieves the best performance and makes further performance improvements over other methods under the condition of few computational budgets. This fully verifies the previous analysis of Fig. 1: informative features may only exist in certain regions of the spectral curve. The reason why SCAI + achieves a significant performance improvement over other methods under the condition of fewer computational budgets is that the PA-ResNet captures these "class-indiscriminative" features and

reduces the computational allocation in these positions. Moreover, the introduction of the self-distillation paradigm improves the performance of the classifier in the shallow layers of the architecture, which also promotes this phenomenon. More interestingly, with the increase of computational budget, the performance of the SCAI + model reaches a peak that cannot be touched by other models. We believe that the noise reduction effect of the PA-ResNet structure on redundant features makes it focus on “class-discriminative” features and further improves the model performance. The above two points consistently show that SCAI + can be applied to scenarios with both sufficient and insufficient computational budgets at the same time.

Budgeted Batch Prediction. In the budgeted batch prediction setting, the model accepts a batch of M samples and classifies these samples without exceeding the total computational budget of B . Referring to the experimental scheme of MSDNet [13], we also adopt dynamic evaluation, that is, let “easy” samples be output in the early classifier, and let “hard” samples be output in the latter classifier. To cover a wider range of computational budgets, we train models of SCAI, SCAI +, and SCAI+_{w/oKD} with different scales. The experimental results of this section are reported in Fig. 6 (b), it can be seen that the experimental results are similar to the results in anytime prediction., SCAI, SCAI +, and SCAI+_{w/oKD} consistently achieve the best performance. For example, SCAI + achieves approximately 6% or 13% accuracy improvement over 1D-ResNet and DRSNs (DRSN-CW and DRSN-CS) at an average budget of 0.2×10^6 Flops, respectively. To maintain the same accuracy (e.g., 98%), 1D-ResNet requires at least 6 times more computational budget than SCAI, SCAI + and SCAI+_{w/oKD}. This shows that the method proposed in this paper can reasonably allocate computing resources under the premise of a limited total budget, give more computing budget to complex samples, and at the same time, avoid the “over-computation” of simple samples, which is significant for improving the overall performance of the identification platform.

3.6 Self-distillation Training Analysis

Another core module of this paper is the self-distillation training paradigm. We recorded the training data of the classifiers within 4 blocks in SCAI + and SCAI+_{w/oKD}, and plotted the train and valid performance curves as shown in Fig. 7.

By comparing the performance of SCAI+_{w/oKD} and SCAI + in each figure, we can intuitively see that the two last classifiers achieve roughly the same performance (Fig. 7 (d)), but other intermediate classifiers of SCAI + (Fig. 7 (a-c)) perform significantly better than that of SCAI+_{w/oKD}. More importantly, this advantage is more obvious in the shallower classifier, which converges faster and is more stable after convergence. Taking Fig. 7 (a) as an example, SCAI + converges at least 50 epochs earlier than SCAI+_{w/oKD}, and the performance of SCAI + after convergence is -18% higher than that of SCAI+_{w/oKD}. This fully verifies the knowledge guiding role of the last classifier on the previous classifiers in the self-distillation mode, thereby greatly improving the performance of the model, especially under an extremely small computational budget. This can also well explain the phenomenon that the performance advantage of our method in Sect. 3.5 is more obvious under the condition of less computation allocation.

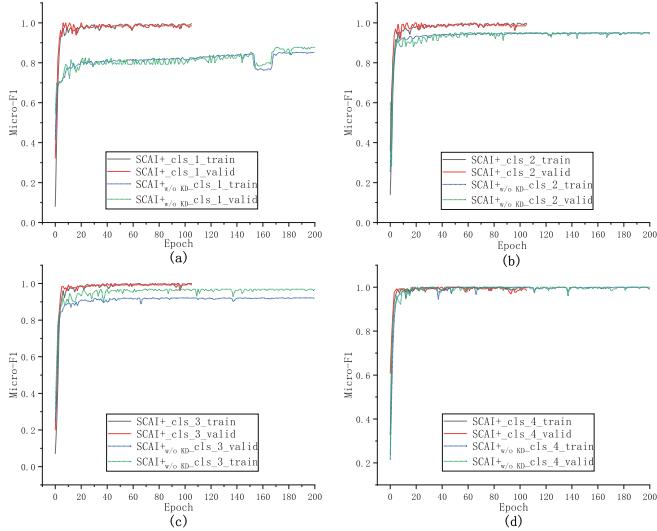


Fig. 7. The train and valid performance curves of different classifiers in SCAI + and SCAI+w/oKD.

The self-distillation training paradigm plays an important role in extending the application capabilities of this model in the portable identification scenario, especially in low-performance smart terminals with high battery life requirements.

4 Conclusion

In this paper, aiming at the computational efficiency problems of deep learning models in the face of portable and rapid liquor identification, we propose an adaptive inference computing architecture to optimize performance and computational efficiency. Specifically, we leverage Early-exit architecture to allocate different computations for different spectral curve samples while better exploiting the computation capability of different portable devices. At the same time, we propose the PA-ResNet and self-distillation learning module greatly improving the efficiency and training speed. The experimental results fully verify the superiority of the method in this paper, especially in the case of a limited computing budget, which is important for extending its applicability to portable devices in identification platform.

Of course, this paper also has some shortcomings, we need to do some further exploration and research. For instance, more architecture design, applicability for more diverse identification tasks, and robustness against adversarial attack.

References

1. Wang, Y., Sun, Y., Fu, Y., Zhu, D., Tian, Z.: Spectrum-BERT: pretraining of deep bidirectional transformers for spectral classification of Chinese liquors. *IEEE Trans. Instrum. Measur.* **73**, 1–13 (2024)

2. Gu, J., et al.: Conformal prediction based on raman spectra for the classification of Chinese liquors. *Appl. Spectrosc. AS.* **73**, 759–766 (2019)
3. Zhao, Z., Liu, Z., Ji, M., Zhao, X., Zhu, Q., Huang, M.: ConInceDeep: a novel deep learning method for component identification of mixture based on Raman spectroscopy. *Chemom. Intell. Lab. Syst.* **234**, 104757 (2023)
4. Ji, H., Pu, D., Yan, W., Zhang, Q., Zuo, M., Zhang, Y.: Recent advances and application of machine learning in food flavor prediction and regulation. *Trends Food Sci. Technol.* **138**, 738–751 (2023)
5. Huang, G., et al.: Glance and focus networks for dynamic visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**, 4605–4621 (2022)
6. Zheng, Z., et al.: Dynamic spatial focus for efficient compressed video action recognition. *IEEE Trans. Circ. Syst. Video Technol.* **1** (2023)
7. Graves, A.: Adaptive Computation Time for Recurrent Neural Networks, <http://arxiv.org/abs/1603.08983> (2017)
8. Sheng, T., Shi, S., Zhu, Y., Chen, D., Liu, S.: Analysis of protein and fat in milk using multiwavelength gradient-boosted regression tree. *IEEE Trans. Instrum. Meas.* **71**, 1–10 (2022)
9. Cozzolino, D.: Advantages, opportunities, and challenges of vibrational spec-troscopy as tool to monitor sustainable food systems. *Food Anal. Methods* **15**, 1390–1396 (2022)
10. He, M., et al.: Identification of liquors from the same brand based on ultraviolet, near-infrared and fluorescence spectroscopy combined with chemometrics. *Food Chem.* **400**, 134064 (2023)
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016)
12. Zhao, M., Zhong, S., Fu, X., Tang, B., Pecht, M.: Deep residual shrinkage networks for fault diagnosis. *IEEE Trans. Industr. Inf.* **16**, 4681–4690 (2019)
13. Huang, G., Chen, D.: Multi-Scale dense networks for resource efficient image classification. In: ICLR 2018 (2018)



EfficientPose: A Lightweight and Efficient Model with Transformer for Human Pose Estimation

Wei Liang¹, Zhang Cheng¹, Junjia Han², and Yanxia Wang^{1(✉)}

¹ College of Computer and Information Science, Chongqing Normal University,
Chongqing 401331, China
wangyanxia@cqnu.edu.cn

² School of Mechanical and Electrical Information Engineering, Shandong University,
Shandong 250100, China

Abstract. The current methods for human pose estimation focus on improving the accuracy of prediction results, but they overlook the significant issues of computational cost and large number of parameters in practical deployment. Although some lightweight pose estimation models have successfully reduced the number of parameters, lightweight models typically employ smaller convolutional kernel to reduce the model size, leading to insufficient capture of contextual information. To address this issue, this paper constructs a lightweight network model EfficientPose. Specifically, to expand the receptive field and acquire richer feature information without increasing computational costs, this paper proposes the Efficient Bottleneck Block (EBB) module. Additionally, to capture global spatial dependencies and enhance the representation capability of low-resolution features, a Transformer encoder is introduced into the model. Meanwhile, to overcome the issue of excessively long training time for lightweight models, a novel iterative training strategy is proposed to fully unleash the potential of EfficientPose. To validate the effectiveness of EfficientPose model, extensive comparative experiments and ablation studies are conducted in this paper. Compared with HRNet-W48, which has the same backbone network, EfficientPose not only reduces the number of parameters by 72% when the input image size is the same but also improves the accuracy by 0.8 and 0.9 percentage points in the validation and test sets of COCO, respectively. Experiments show that the EfficientPose model can maintain high accuracy even with a significant reduction in the number of parameters. This provides the potential for further application in real-world scenarios with limited resources.

Keywords: Lightweight models · Efficient bottleneck block · Transformer encoder · EfficientPose

1 Introduction

Despite significant advances in human pose estimation in recent years, accurate keypoint estimation remains a major challenge in computer vision due to occlusions and complex variations in human pose. Effectively representing keypoints locations during training

and providing strong supervision is a longstanding problem in this field. In human key-point estimation methods, there are two common supervision approaches. One approach is to directly use the keypoint positions as training targets, as seen in methods like DeepPose [23] and IPR [21]. However, this method has limited generalization capabilities as it overlooks the importance of spatial positional information. Another common supervision method involves using heatmaps as training targets, as seen in approaches like Hourglass [17]. While heatmaps retain spatial information during the training phase, they suffer from quantization errors, especially at lower resolutions. To preserve more details and reduce quantization errors, methods like SimpleBase [25], HRNet [20], and HigherHRNet [6] utilize higher-resolution to predict heatmaps. However, the drawbacks of high-resolution prediction are also evident, notably the significantly increased computational costs.

To strike a balance between accuracy and computational cost, we propose a bottleneck module with positional information and multi-scale dilated convolutions. The module combines dilated convolution techniques to enlarge the receptive fields and maintain computational efficiency, so as to better capture contextual information. Additionally, in order to preserve every channel-wise information and reduce computational resource consumption, the bottleneck module is optimized, which helps to better facilitate spatial semantic feature distribution and achieve more accurate key point positioning by regrouping channels and dividing channel dimensions into multiple sub-features. The bottleneck module employs two parallel sub-networks for convolution operations. One sub-network uses a 1×1 convolutional kernel for channel dimension compression, while the other uses a 3×3 convolutional kernel to better capture local features. This combination maintains the model's lightweight nature while achieving precise keypoint localization. Furthermore, to enhance the model's feature representation capability at low resolutions, Transformer [24] is introduced to capture the global context in human pose estimation. Compared to Convolutional Neural Networks (CNNs) with limited receptive fields, Transformers can more effectively capture long-range interactions between any two pixels through self-attention mechanisms. Multiple Transformer encoders are applied to high-level features at low resolutions, making the Transformer sequence shorter in length and lowering computational costs. By integrating Transformer into our proposed network, not only maintain the characteristics of a lightweight model, but also enhance its efficiency. However, due to the relatively weaker generalization ability of lightweight human pose estimation networks, they are more prone to get stuck in local minima. We propose an iterative training strategy that periodically adjusts the learning rate to fully exploit the potential of lightweight networks and achieve significant improvements. In summary, the contributions of this paper are as follows:

- EfficientPose, a lightweight network model for human pose estimation, is constructed in the paper, and a large number of experiments are carried out to verify the effectiveness of the model, and the experimental data show that EfficientPose is able to maintain a high accuracy in the case of a large reduction in the number of parameters.
- A bottleneck module with positional information and multi-scale dilated convolutions is proposed, significantly reducing the computational complexity of the model while achieving more precise keypoint localization.

- The encoder from Transformer is integrated into the EfficientPose network model, enriching its ability to represent global information and capturing global contextual information more effectively. To fully exploit the potential of this network model, an iterative training strategy is proposed to adjust the learning rate periodically.

2 Related Work

2.1 Human Pose Estimation

Deep Convolutional Neural Networks (CNNs) achieve significant achievements in the field of human pose estimation. The popular Convolutional Pose Machines (CPM) employ a method of refining keypoint detection through a series of stages in the network. The stacked Hourglass networks utilize a cascaded implementation of the pose estimation task with an Hourglass structure. Building upon CPM, Yan et al. integrate the concept of Part Affinity Fields (PAF) and propose the OpenPose [2] method. PAF utilizes the detected significant joints to better estimate the less significant joints. This innovation allows for multi-person detection while reducing complexity and computational requirements. The MCAP [7] method relies on Hourglass backbone for pose estimation, which is enhanced by Hourglass Residual Units(HRU) to increase the receptive field, and uses the conditional random field (CRF) for post-processing to assemble the relationship between the detected nodes. However, the CRF increases time complexity. Sun et al. propose HRNet that suggests that human pose estimation should include both high-resolution and low-resolution representations. The method starts from high-resolution and gradually adds low-resolution sub-networks to form multiple stages, performing multi-scale fusion between sub-networks. However, CNNs still struggle to capture the constraint relationships between human keypoints due to their limited receptive field, limiting their ability to understand global spatial relationships. Recent studies that use the Transformer for human pose estimation. TokenPose [13] represents keypoints as token embeddings within the Transformer framework. Transpose [26], on the other hand, combines HRNet with Transformer to utilize context-aware encoders for capturing long-range interaction capabilities.

2.2 Atrous Convolution and ASPP

A key challenge in semantic segmentation and pose estimation methods involving fused convolutional layers is the significant reduction in resolution caused by pooling operations. Fully Convolutional Networks (FCN) [16] employ an upsampling strategy with deconvolution layers to address the issue of reduced resolution. These strategies aim to reverse the convolutional operations, increasing the size of the feature maps to the dimensions of the original image. In the field of semantic segmentation, a commonly used technique is atrous convolution, also known as dilated convolution [4]. The primary goal of dilated convolution is to increase the receptive field size within the network, avoiding downsampling, and constructing a multi-scale processing framework. Taking one-dimensional convolution as an example, the output of the signal can be defined as follows:

$$y[i] = \sum_{l=1}^L x[i + rl] \cdot \omega[l] \quad (1)$$

where r is the dilation rate, $\omega[l]$ is a filter of length L , $x[i]$ is the input, and $y[i]$ is the output. When the dilation rate is 1, the result is a regular convolution operation.

The ASPP (Atrous Spatial Pyramid Pooling) method assembles convolutions with different dilation rates through four parallel branches, followed by an additional eight-fold fast bilinear interpolation to combine them, which restores the feature map to the resolution of the original image. In the context of pose estimation, the ASPP network offers significant advantages in detecting body parts within the context by increasing both resolution and the range of the receptive field.

Table 1. Detailed Network Architectures for EfficientPose-R50 (Based on ResNet-50) and EfficientPose-W48 (Based on HRNet-W48)

Architecture	EfficientPose-R50	EfficientPose-W48
Backbone	Conv-k7-s2-c64,BN,ReLU	Conv-k3-s2-c64,BN,ReLU
	Max Pooling-k3-s2	Conv-k3-s2-c64,BN,ReLU
	3 × Bottleneck-c64	4 × Bottleneck-c64
	Bottleneck-s2-c128	Transition1-Stage2
	3 × Bottleneck-c128	Transition2-Stage3
	Conv-k1-s1-c256	Conv-k1-s1-c92
Encoder	4 × Transformer Encoder d256-h8-h512	6 × Transformer Encoder d96-h1-h192

3 Methods

This section provides a detailed introduction to the proposed bottleneck module with position-aware multi-scale dilated convolutions, the context-aware encoder of Transformer, and a novel iterative training strategy proposed in the paper. Based on these modules, a brand-new lightweight model EfficientPose (Fig. 1) is constructed for human pose estimation.

3.1 EfficientPose Architecture

The EfficientPose model (Fig. 1) mainly consists of three modules. Firstly, the backbone network that is based on adjusted HRNet or ResNet extracts high-level low-resolution feature maps. Secondly, the bottleneck blocks and Encoders. Features extracted through bottleneck blocks are flattened into sequences, and context information is captured using multiple Transformer encoders. Finally, there is the regression head, which primarily utilize 1×1 convolutions to learn Gaussian responses for each position to locate keypoints, and the center point offsets for each keypoint to achieve precise results. The detailed network architectures for the Backbone and Transformer encoders are shown in Table 1.

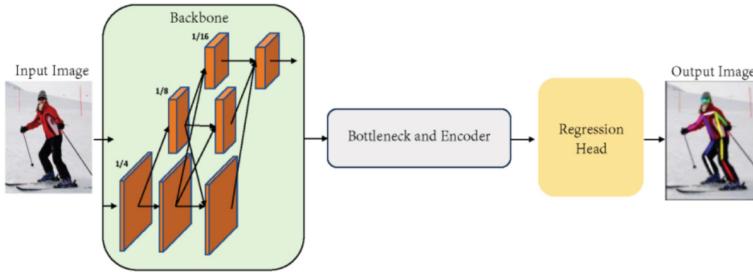


Fig. 1. Overall Architecture of EfficientPose

3.2 Efficient Bottleneck Block (EBB)

The design of EBB (Fig. 2) is inspired by modules such as ASPP, CA [11] and EMA [18]. It maximizes the advantages of the larger receptive field of the ASPP configuration and the size reduction of the cascading approach. Additionally, it incorporates a grouped coordinate attention mechanism, which aims to learn efficient channel representations and generate superior pixel-level attention. Firstly, for a given input feature map $X \in \mathbb{R}^{C \times H \times W}$, EBB integrates multiple-scale features by fusing along four distinct dilated convolution branches and one pooling branch, which enhances the receptive field of the model for a better understanding of spatial structure information. Secondly, to capture inter-channel dependencies and reduce the computational costs, after the concatenation, a grouped coordinate attention mechanism is introduced to divide the concatenated features into G sub-features along the channel dimension to learn different semantic information and extract attention weights for the grouped feature maps. These grouped styles can be represented as $X = [X_0, X_1, \dots, X_{G-1}]$, where $X_i \in \mathbb{R}^{C//G \times H \times W}$. Subsequently, two 1D global average pooling operations are applied separately in the two branches located in the horizontal X and vertical Y directions, encoding channel information along both spatial directions. After concatenating the two encoding features along the image's height dimension, a 1×1 convolutional kernel with shared weight parameters is employed. The output of the 1×1 convolutional kernel is then split into two vectors using two non-linear functions sigmoid to enhance the expressive capability of the neural network. To facilitate interaction features between the X and Y channels, a straightforward multiplication is employed to combine the X and Y channels with the Groups feature map. In addition, in the 3×3 branch, only one 3×3 convolutional kernel is stacked to capture multi-scale feature representations to expand the feature space and acquire localized cross-channel interactions. Through this approach, EBB not only encodes cross-channel information to adjust the importance of different channels, but also preserves precise spatial structural information within the channels.

3.3 Transformer Encoder

Due to the limited receptive field of the CNN, it results in low-resolution features that lack global context information. Therefore, a context-aware encoder is introduced, which mainly consists of multi-head attention, layer normalization, and feed-forward networks.

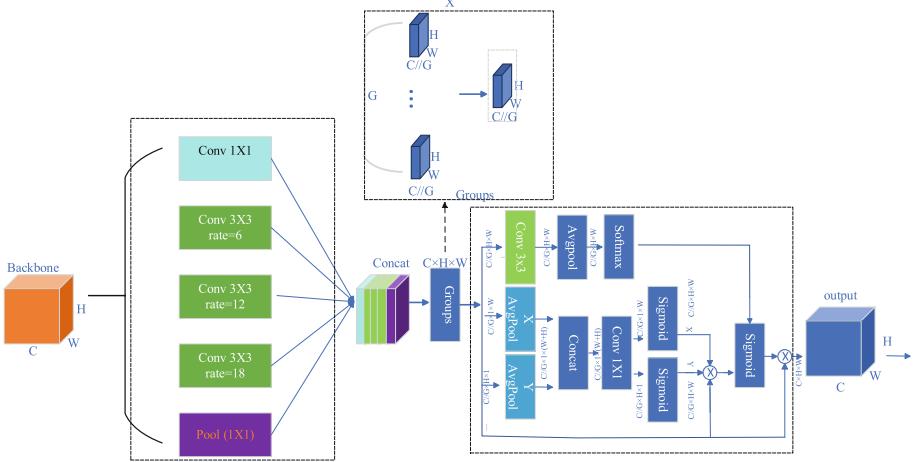


Fig. 2. Schematic diagram of the Efficient Bottleneck Block (EBB). To classify central pixels, multiple parallel filters with different dilation rates are utilized, employing multiscale features. “Groups” in the figure indicates grouping, “X Avg Pool” represents 1D horizontal global pooling, and “Y Avg Pool” represents 1D vertical global pooling.

Layer normalization is applied between the multi-head attention and the feed-forward network. Multi-head attention consists of multiple self-attention layers that encapsulate a variety of complex relationships between different positions in the sequence. The structure of the Transformer encoder is illustrated in Fig. 3. By utilizing multiple Transformer encoders, it not only captures global context information more effectively, enhancing the representation capability of global information, but also reduces computational complexity. The calculation method is as follows:

$$Z = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right) \cdot V \quad (2)$$

In the above equation, Q , K and V denote the representations of the input sequence X by three different transformation matrices. Inside each encoder, the multi-head attention layer is applied first, followed by layer normalization, and then passed through a feed-forward network consisting of two linear layers and a ReLU activation function, represented by the following function:

$$\text{FFN}(X) = W_2 \cdot \text{ReLU}(W_1 X) \quad (3)$$

where W_1 and W_2 are the parameter matrices of the two linear layers. The feed-forward network is used to further process the output from the attention layer, enhancing the representational capacity of the model.

3.4 Iterative Training Strategy

The classic universal approximation theorems [8, 10] demonstrate that as long as a network has a sufficient number of hidden units, it can represent any function without

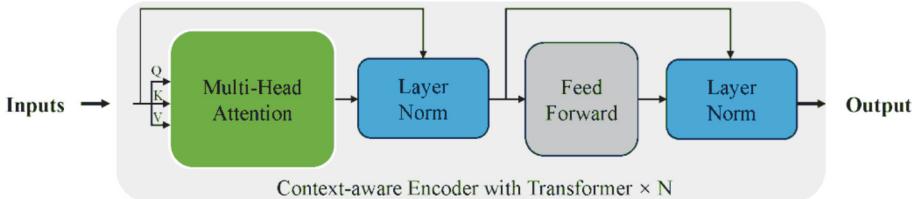


Fig. 3. Transformer Encoder Structure Diagram

being limited by its specific characteristics. In other words, given an adequate number of parameters, the network has the ability to reach a global minimum. However, in the actual optimization process, it often gets trapped in certain local minima. Due to the limited generalization ability of lightweight networks, smaller networks are more prone to getting stuck in poorer local minima. The smaller the network size, the more prominent this situation becomes. Once stuck in a local minimum, it becomes very challenging for optimizers with smaller learning rates to traverse this “ridge”. Therefore, sometimes it is necessary to increase the learning rate again. This paper proposes a new training iteration strategy that periodically adjusts the learning rate. The main idea is to initialize the learning rate with a specific value ($1e-4$) in the first phase and then reduce the learning rate at each milestone. Next, use the best model obtained from the previous phase as a pre-trained model to initialize the parameters, reset the learning rate, and restart the training from a specific epoch, then proceed with the rest of the process.

4 Experiments

To validate the effectiveness of the EfficientPose model, two types of experiments are conducted based on the publicly available pose estimation datasets COCO2017 [15] and MPII [1]. These experiments include comparative experiments with classical methods on the validation and test sets, ablation experiments, and visualization of pose estimation results.

4.1 Implementation Results

This subsection focuses on comparing the experimental results of the classical networks SimpleBase, HRNet and TokenPose with the EfficientPose proposed in the paper on the COCO validation set, they have the same backbone the number of parameters of the models with similar accuracy to the EfficientPose is compared on the COCO test set, while the comparison of the experimental results of models with the same backbone network is carried out on the MPII validation set.

COCO Validation Set. According to the results in Table 2, it can be observed that the model EfficientPose-W48 proposed in this paper, when compared with classical networks using the same backbone network (SimpleBase, HRNet, and TokenPose), outperforms in most evaluation metrics. EfficientPose-W48 is only 0.1 lower in the AP0.5 evaluation metric compared with HRNet-W48, while the other evaluation metrics show improvements. When compared with SimpleBaseline with ResNet50 as the

same backbone network, EfficientPose-R50 achieves a 2.5% increase in AP while reducing parameters by 75.0%. Furthermore, when compared with the latest lightweight networks HRNet-Lite, EfficientPose-W32 reduces parameters by 45%, but the AP metric increases by 0.6%. In comparison with ShiftPose with a similar number of parameters, EfficientPose achieves a 2.4% improvement in AP. This indicates that EfficientPose can still provide high detection accuracy with fewer computational resources.

COCO Test Set. Table 3 demonstrates that EfficientPose has significant advantages over existing state of the art pose estimation methods, both in terms of model parameters and accuracy. For example, compared to the model RMPE, which has the smallest number of parameters, EfficientPose-W32 has a 72% reduction in the amount of parameters and a 1.5% point improvement in the accuracy AP; Compared with the classical network SimpleBase, EfficientPose-W48 achieves an 89% reduction in parameters while slightly improving AP by 0.1% points. Moreover, when compared with HRNet-W48 with the same backbone network and input size, EfficientPose-W48 not only improves AP by 0.9% points but also reduces parameters by 72%. In comparison with the network DARK with better evaluation metrics listed in the table, although the evaluation metrics of EfficientPose are slightly lower, its parameter count is 72% lower than DARK. Additionally, the data in the table indicates that DARK has a higher input image resolution, containing richer features than EfficientPose. Compared to the latest lightweight model HRNet-Lite with the same backbone network, EfficientPose reduces parameters by 45%, and experiences only a minor decrease of 0.3% points in the AP0.5 metric, while outperforming in other evaluation metrics. This highlights that EfficientPose maintains outstanding detection performance even with a smaller parameter count.

MPII Validation Set. Based on the experimental results shown in Table 4, it can be observed that the proposed EfficientPose-W48 network has certain advantages over popular classical networks on the MPII dataset in terms of parameter count and detection accuracy. EfficientPose-W48, with a size similar to the Transpose network, has improved by 0.2 and 0.5% points on the evaluation metrics PCKh@0.5 and Mean@0.1, respectively. Compared with the classic network with the smallest parameter count DLCM, EfficientPose-W48 has improved by 0.7% points on PCKh@0.5 while having a similar parameter count. EfficientPose-R50, compared with LFP, has reduced its parameter count by 56% while maintaining the same level on the PCKh@0.5 evaluation metric. When compared with SimpleBase with the same backbone network, EfficientPose-R50 achieves the same level on PCKh@0.5 with an 88% reduction in parameter count. Moreover, when compared with the best-performing model DARK in the Table 4, EfficientPose-W48 has reduced its model parameters by 62.8% while maintaining an equivalent level of detection accuracy. Compared to ShiftPose, under similar parameter count, AP has increased by 3.2% points. EfficientPose not only has a simple model, but also has better detection performance.

Table 2. The comparison results on the COCO Validation set

Method	Backbone	Input size	#Params	AP	AP 0.5	AP 0.75	AR
CPN [5]	ResNet-50	256×192	27.0M	68.6	—	—	—
SimpleBaseline	ResNet-50	256×192	34.0M	70.4	88.6	78.3	76.3
SimpleBaseline	ResNet-101	256×192	53.0M	71.4	89.3	79.3	77.1
SimpleBaseline	ResNet-152	256×192	68.6M	72.0	89.3	79.8	77.8
HRNet-W32	HRNet-W32	256×192	28.5M	74.4	90.5	81.9	79.8
HRNet-W32	HRNet-W48	256×192	63.6M	75.1	90.6	82.2	80.4
TokenPose	HRNet-W48	256×192	20.8M	75.4	90.0	81.8	80.4
TransPose	HRNet-W48	256×192	17.5M	75.8	90.1	82.1	80.8
HRNet-Lite [14]	HRNet-W32	256×192	14.5M	73.9	89.7	80.9	79.1
ShiftPose [3]	ShiftPose	256×192	10.2M	72.1	91.5	—	—
EfficientPose-R50	ResNet-50	256×192	8.5M	72.9	89.3	80.0	78.4
EfficientPose-W32	HRNet-W32	256×192	8.1M	74.5	89.6	81.2	79.5
EfficientPose-W48	HRNet-W48	256×192	17.9M	75.9	90.5	82.3	81.1

Table 3. The comparison results on the COCO Test set

Method	Backbone	Input size	#Params	AP	AP 0.5	AP 0.75	AR
G-RMI [19]	ResNet-101	353×257	42.6M	64.9	85.5	71.3	69.7
Integral Pose	ResNet-101	256×256	45.0M	67.8	88.2	74.8	—
RMPE [9]	PyraNet	320×256	28.1M	72.3	89.2	79.1	—
CPN	ResNet	384×288	58.8M	72.1	91.4	80.0	78.5
SimpleBaseline	ResNet-152	384×288	68.6M	73.7	91.9	81.1	79.0
HRNet-W32	HRNet-W32	384×288	28.5M	74.9	92.5	82.8	80.1
HRNet-W48	HRNet-W48	256×192	63.6M	74.2	92.4	82.4	—
HRNet-W48	HRNet-W48	384×288	63.6M	75.5	92.5	83.3	80.5
MSPN [12]	ResNet-50×4	384×288	120.3M	76.1	93.4	83.8	81.6
DARK [28]	HRNet-W48	384×288	63.6M	76.2	92.5	83.6	81.1
HRNet-Lite	HRNet-W32	256×192	14.5M	73.3	91.8	81.0	78.5
EfficientPose-W32	HRNet-W32	256×192	8.1M	73.8	91.5	81.3	79.0
EfficientPose-W48	HRNet-W48	256×192	17.9M	75.1	92.1	82.6	80.1

Table 4. The comparison results on the MPII Validation set

Method	#Params	PCKh@0.5	Mean@0.1
Hourglass	25.1M	89.2	—
LFP [27]	28.1M	89.6	—
SimpleBase	68.6M	89.6	—
DLCM [22]	15.5M	89.8	—
Transpose	17.5M	90.3	41.6
HRNet-W32	28.5M	90.3	—
DARK	28.5M	90.6	—
ShiftPose	10.2M	86.4	—
EfficientPose-R50(ours)	8.5M	89.6	39.2
EfficientPose-W48(ours)	17.9M	90.5	42.1

Table 5. Ablation Study of EfficientPose

Model	Backbone	EBB	Encoder	Iterative strategy	AP	#Params	FPS
EfficientPose	✓		✓		72.6	6.0M	114
	✓	✓			72.8	7.8M	116
	✓	✓	✓		72.9	8.5M	105
	✓	✓	✓	✓	72.9	8.5M	126

4.2 Ablations

To further validate the performance of the proposed EfficientPose model, this study conducted ablation experiments on each component using the COCO2017 dataset, and the results are shown in Table 5. Ablation experiments are conducted using EfficientPose-R50 as the subject, with the model consisting of the backbone network and encoder as the baseline, achieving an AP of 72.6 with a parameter count of only 6.0M. This indicates that the encoder can integrate contextual information features. Combining the backbone network with the Efficient bottleneck block increases the AP by 0.2% points compared with the baseline, with a parameter count increase of only 1.8M. This is mainly because the Efficient bottleneck block not only leverages the advantages of multi-scale dilated convolutions but also divides the input feature map into multiple sub-features through feature grouping to learn different semantic information, almost without introducing additional parameters, thereby more accurately estimating keypoint positions. Combining the backbone network, encoder, and bottleneck block increases the AP by 0.3% points compared to the baseline, with a parameter count increase of only 2.5M. Furthermore, it is evident that using the iterative strategy, EfficientPose-R50 with iterative training achieves a 20% speedup compared with EfficientPose-R50 without the iterative strategy. Each component in the EfficientPose model contributes to the improvement in accuracy, mainly due to the effective design of the modules that construct short-range and long-range dependencies, enrich information features, and enable the model to more accurately estimate keypoint positions.

5 Conclusion

In this paper, a lightweight human pose estimation network model EfficientPose is proposed for resource-constrained scenarios. The model consists of three main components, the backbone network, Efficient Bottleneck Block (EBB) modules, and Transformer encoders. To assess the impact of each component on the model, ablation experiments are conducted, demonstrating the significant influence of EBB modules and Transformer encoders on the overall performance. Moreover, extensive comparative experiments are conducted on the COCO and MPII datasets to validate the model's performance. The results show that, compared with popular classical networks and the latest lightweight network models, EfficientPose achieves comparable or higher accuracy with a substantial reduction in parameters. Although EfficientPose demonstrates competitive performance on public datasets, further research is needed to explore its application in real-time detection scenarios.

References

1. Andriluka, M., Pishchulin, L., Gehler, P., Schiele, B.: 2D human pose estimation: new benchmark and state of the art analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3686–3693 (2014)
2. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2D pose estimation using part affinity fields. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7291–7299 (2017)
3. Chen, H., Jiang, X., Dai, Y.: Shift pose: a lightweight transformer-like neural network for human pose estimation. *Sensors* **22**(19), 7264 (2022)
4. 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)
5. Chen, Y., Wang, Z., Peng, Y., Zhang, Z., Yu, G., Sun, J.: Cascaded pyramid network for multi-person pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7103–7112 (2018)
6. Cheng, B., Xiao, B., Wang, J., Shi, H., Huang, T.S., Zhang, L.: Higherhrnet: scale-aware representation learning for bottom-up human pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5386–5395 (2020)
7. Chu, X., Yang, W., Ouyang, W., Ma, C., Yuille, A.L., Wang, X.: Multi-context attention for human pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1831–1840 (2017)
8. Cybenko, G.: Approximation by superpositions of a sigmoidal function. *Math. Control Signals Syst.* **2**(4), 303–314 (1989)
9. Fang, H.S., Xie, S., Tai, Y.W., Lu, C.: RMPE: regional multi-person pose estimation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2334–2343 (2017)
10. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. *Neural Netw.* **2**(5), 359–366 (1989)
11. 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, pp. 13713–13722 (2021)
12. Li, W. et al.: Rethinking on multi-stage networks for human pose estimation. arXiv preprint [arXiv:1901.00148](https://arxiv.org/abs/1901.00148) (2019)
13. Li, Y., et al.: Tokenpose: Learning keypoint tokens for human pose estimation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 11313–11322 (2021)
14. Li, Y., Liu, R., Wang, X., Wang, R.: Human pose estimation based on lightweight basicblock. *Mach. Vis. Appl.* **34**(1), 3 (2023)
15. 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
16. 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)
17. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, the Netherlands, October 11–14, 2016, Proceedings, Part VIII 14, pp. 483–499. Springer (2016)
18. Ouyang, D., et al.: Efficient multi-scale attention module with cross-spatial learning. In: ICASSP 2023–2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1–5. IEEE (2023)

19. Papandreou, G., et al.: Towards accurate multi-person pose estimation in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4903–4911 (2017)
20. Sun, K., Xiao, B., Liu, D., Wang, J.: Deep high-resolution representation learning for human pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5693–5703 (2019)
21. Sun, X., Xiao, B., Wei, F., Liang, S., Wei, Y.: Integral human pose regression. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 529–545 (2018)
22. Tang, W., Yu, P., Wu, Y.: Deeply learned compositional models for human pose estimation. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 190–206 (2018)
23. Toshev, A., Szegedy, C.: Deeppose: human pose estimation via deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1653–1660 (2014)
24. Vaswani, A., et al.: Attention is all you need. *Adv. Neural Inf. Process. Syst.* **30** (2017)
25. Xiao, B., Wu, H., Wei, Y.: Simple baselines for human pose estimation and tracking. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 466–481 (2018)
26. Yang, S., Quan, Z., Nie, M., Yang, W.: Transpose: keypoint localization via transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 11802–11812 (2021)
27. Yang, W., Li, S., Ouyang, W., Li, H., Wang, X.: Learning feature pyramids for human pose estimation. In: proceedings of the IEEE International Conference on Computer Vision, pp. 1281–1290 (2017)
28. Zhang, F., Zhu, X., Dai, H., Ye, M., Zhu, C.: Distribution-aware coordinate representation for human pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7093–7102 (2020)



Enhanced Chinese Named Entity Recognition with Transformer-Based Multi-feature Fusion

Xiaoli Zhang, Quan Zhang^(✉), Kun Liang, and Haoyu Wang

College of Artificial Intelligence, Tianjin University of Science and Technology, Tianjin 300457,
China
zhangquan@mail.tust.edu.cn

Abstract. In the field of Named Entity Recognition (NER), the integration of semantic features is crucial for enhancing model performance. Particularly in Chinese NER tasks, the inclusion of lexical information into character-based models is essential due to the lack of clear word boundaries. Additionally, effectively fusing multiple semantic features is of utmost importance for further performance enhancement. To address these challenges, this paper proposes a novel Chinese Named Entity Recognition model, termed DFL-NER. The model adopts an improved Transformer feature fusion method, beginning with the extraction of word vectors using the pre-trained DeBERTa model. Subsequently, lexical features are integrated into character vectors through an enhanced SoftLexicon algorithm, and ultimately, dynamic word vectors, lexical features, and glyph features are combined through a feature fusion module to enrich semantic representation. Experimental results demonstrate that compared to the baseline model, DFL-NER achieves F1-score improvements of 7.14%, 0.37%, and 1.81% on the Weibo, Resume, and MSRA datasets, respectively, showcasing outstanding performance.

Keywords: Chinese NER · DeBERTa · Lexical Features · Glyph Features ·
Glyph Features

1 Introduction

Named Entity Recognition (NER) is pivotal in natural language processing, commonly framed as a sequence tagging task [1]. Chinese NER confronts challenges due to ambiguous word boundaries, emphasizing the significance of lexical information for character-based models.

Recent studies have shown that lexical enhancement techniques can significantly improve the performance of character-based NER methods. For example, approaches like SoftLexicon [2] and Lattice LSTM [3] have demonstrated the effectiveness of integrating lexical features into NER models. These techniques help to mitigate the issue of missing word boundaries by providing additional context at the character level. In addition to lexical information, the logographic nature of Chinese characters offers another dimension of semantic information. Chinese characters, evolved from pictographs, possess glyph structures that encapsulate rich semantic content. For instance, the use of radicals, which are the building blocks of Chinese characters, can provide valuable clues

about the meaning of a character. Incorporating such glyph-based features into NER models has been explored in recent works [4–9], showing that it can further enhance model performance.

Building on these insights, this paper introduces a novel Chinese NER model, named DFL-NER (DeBERTa Feature Fusion-based Lexical-enhanced NER), which leverages an improved Transformer feature fusion method. The model harnesses the power of the DeBERTa pre-trained model [10] to extract contextualized character embeddings and employs an enhanced SoftLexicon mechanism to incorporate lexical features. Furthermore, it integrates glyph features to further enrich the semantic representation of characters. By combining these multi-level features, DFL-NER aims to provide a more comprehensive and robust solution for Chinese Named Entity Recognition.

2 Related Work

2.1 Chinese NER Based on Lexical Enhancement

In NER tasks, lexical boundaries often play a crucial role in detecting entity boundaries. Therefore, introducing lexical information into character granularity-based NER methods has been a focus of research in recent years. The introduction of lexical information into NER systems is also referred to as lexical enhancement. There are two mainstream approaches to lexical enhancement: the dynamic frame method and the adaptive embedding method. The dynamic frame method requires designing a corresponding dynamic frame to incorporate lexical information. The most representative model of this approach is the Lattice LSTM model proposed by Zhang et al. [3], which was the first model to effectively integrate lexical information into character-based NER. However, Lattice LSTM has limitations, as the model is only adapted to LSTM and lacks portability.

The adaptive embedding method focuses on incorporating lexical information only at the embedding layer and combines it with other general networks for NER tasks. For example, the Soft-lexicon method proposed by Peng et al. [2] introduces simple lexicon utilization at the embedding layer to avoid complex model structures and facilitate migration to other sequence annotation frameworks. This paper has chosen the Soft-lexicon-based lexical enhancement method due to its flexibility and simplicity.

2.2 Chinese NER with Fusion of Glyph Features

Lexical enhancement techniques have improved model performance. Simultaneously, each Chinese character has its structural characteristics. Researchers have explored various methods to incorporate glyph information into NER models to enhance performance. Inspired by Word2Vec, some researchers [11, 12] have used radical embeddings to capture character semantics and improve model performance for Chinese natural language processing tasks.

Another approach to combining glyph information is to process characters in the form of images and utilize convolutional neural networks (CNNs) to extract glyph features from character images. Meng et al. [13] proposed a BERT-based Glyce model that designed a glyph grid to extract features such as strokes and structure of Chinese

characters from images, applying glyph embedding to a wide range of Chinese natural language processing tasks. The proposed ChineseBert model, based on the Glyce model, fused character, glyph, and pinyin (Chinese phonetic alphabets) features into language model pre-training and achieved state-of-the-art performance in many natural language processing tasks [14].

2.3 Feature Fusion in Chinese NER

Feature fusion [15] is a crucial aspect of enhancing NER models by integrating various types of information. In the context of Chinese NER, combining lexical, glyph, and contextual features can provide a more comprehensive representation of the text. Transformer-based models, such as BERT and its variants, have shown great potential in capturing contextual information. However, the direct application of these models may not fully exploit the rich semantic information present in Chinese characters. Therefore, developing improved feature fusion methods that can effectively integrate these diverse features is essential for advancing Chinese NER performance. This paper focuses on an improved Transformer feature fusion method that leverages the strengths of DeBERTa, enhanced lexical information, and glyph features to achieve superior NER performance

3 Method

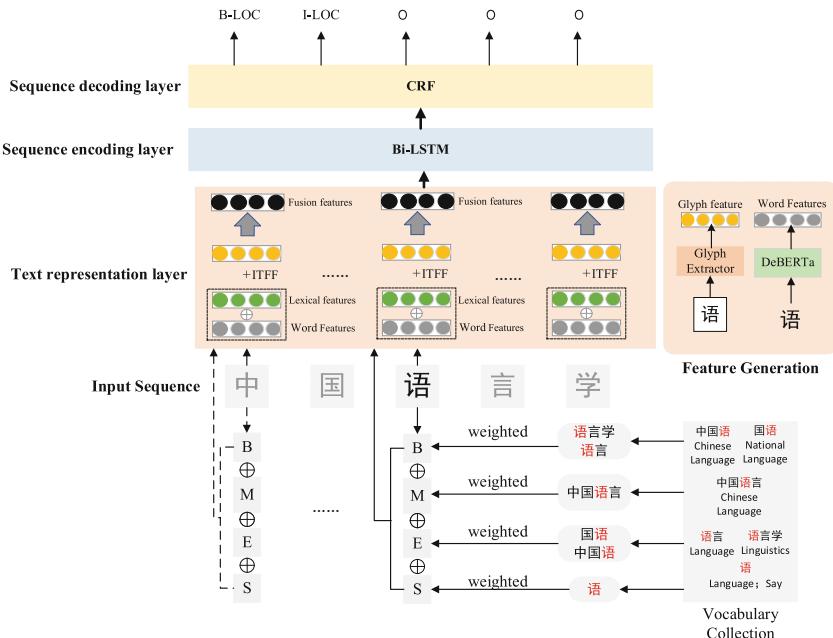


Fig. 1. DFL-NER model structure diagram. The DFL-NER model mainly consists of three layers: the text representation layer, sequence encoding layer, and sequence decoding layer. ITFF stands for Improved Transformer Feature Fusion.

In this paper, we propose a novel Chinese Named Entity Recognition model, named DFL-NER, which incorporates both lexical information and Chinese character glyph features. The architecture of the DFL-NER model is structured into three main layers: the text representation layer, the sequence encoding layer, and the sequence decoding layer (Fig. 1).

3.1 Text Representation Layer

The text representation layer is responsible for converting text into a computer-readable format and includes character-level features and lexical features enhanced by the SoftLexicon method. Specifically, this layer utilizes the DeBERTa pre-trained model to extract contextualized character embeddings. Additionally, glyph features are extracted using a Glyph Extractor to obtain the glyph embeddings of Chinese characters.

- (1) **Word feature acquisition.** The model employs DeBERTa (Decoding-enhanced BERT with Disentangled Attention) [10] to dynamically generate word vectors for each character, which are then used in subsequent feature fusion stages. Initially, the input text data is processed using Relative Position Embeddings to capture the relative positional information between words, which is essential for understanding the text's structural aspects. Type embeddings further introduce a semantic layer for different word types, such as the beginning of sentences ([CLS]) and separators ([SEP]). After these preliminary steps, the data is passed through multiple Transformer layers, which leverage self-attention mechanisms to enhance the model's comprehension of the text. Finally, an enhanced mask decoder refines the model's output, ensuring high accuracy and semantic consistency in the predictions. This comprehensive approach enables DeBERTa to generate highly contextualized and semantically rich word vectors, which are crucial for the subsequent feature fusion stages, as illustrated in Fig. 2.

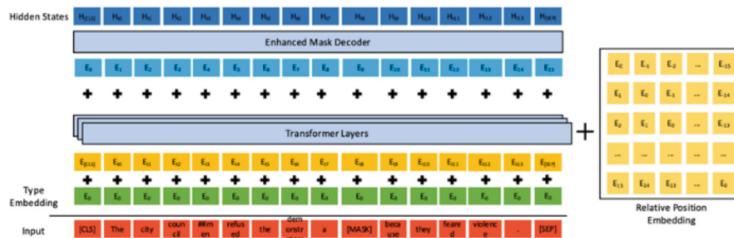


Fig. 2. Input Embedding Structure of DeBERTa.

- (2) **SoftLexicon based lexical features.** SoftLexicon categorizes all potential word segmentation results based on character positions in the vocabulary into four types: beginning, middle, end, and single character, marked respectively as B, M, E, and S. Figure 3 shows the specific SoftLexicon method. The implementation principle is as follows:

1. Vocabulary classification for matching: To retain word segmentation information, characters are matched and labeled using word sets as outlined in Eq. (1)–(4).

$$B(c_i) = \{W_{i,k}, \forall W_{i,k} \in L, i < k \leq n\} \quad (1)$$

$$M(c_i) = \{W_{j,k}, \forall W_{j,k} \in L, 1 \leq j < i < k\} \quad (2)$$

$$E(c_i) = \{W_{j,i}, \forall W_{j,i} \in L, 1 \leq j < i\} \quad (3)$$

$$S(c_i) = \{c_i, \exists c_i \in L\} \quad (4)$$

where L denotes all words in the dictionary, n denotes the length of the sentence, and i, j, k denotes the position of the words in the sentence.

2. word set compression: In this paper, we adopt a word set compression method with improved algorithmic efficiency for word weighting. Moreover, we have enhanced the original algorithm by considering the rich information contained in low-frequency words. Specifically, we use the reciprocal of word frequency as the metric for word weighting to better balance the importance of rare words. The word weighting principle is represented by Eq. (5) and Eq. (6).

$$V^s(S) = \frac{4}{Z} \sum_{w \in S} \frac{e^w(w)}{z(w)} \quad (5)$$

$$Z = \sum_{w \in B \cup M \cup E \cup S} z(w) \quad (6)$$

where $Z(W)$ denotes the word frequency of word w in the data statistics; Z is the sum of the frequency of all words in the word set; $e^w(w)$ is the word vector matrix used for embedding lookup.

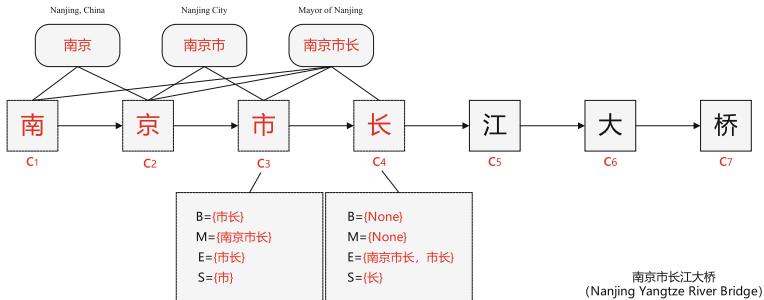


Fig. 3. SoftLexicon method. For the Chinese sequence ‘南京市市长’ (Nanjing Yangtze River Bridge), after word segmentation, each character is obtained. Then, for the current character, all vocabulary sets corresponding to BMES are obtained in turn.

3. Character representation fusion: Combine the representations of the four word sets into a fixed dimensional feature and added to the representation of each character. The addition is done by directly concatenating the four words sets. The final representation of each character is obtained through Eqs. (7) and (8).

$$e^s(B, M, E, S) = [V^s(B); V^s(M); V^s(E); V^s(S)] \quad (7)$$

$$[X^c; e^s(B, M, E, S)] \rightarrow X^c \quad (8)$$

V^s denotes the weighting function in the second step, x^c is the final representation of the character.

- (3) **Glyph feature acquisition** For the selection of character images, in order to enrich the pictorial information of character images, we choose a combination of multiple types of fonts as the image data set, including a total of three different types of Chinese fonts, namely, official script, regular script and Simsun.

In the process of extracting font features, due to the much smaller size of character images compared to those in ImageNet, we built a feature extractor—the Glyph Extractor—to improve computational efficiency. Figure 4 shows the structure of the Glyph Extractor and the process of extracting glyph features.

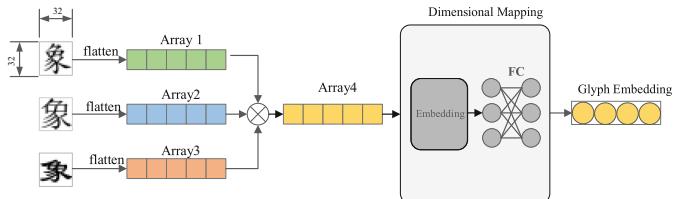


Fig. 4. Glyph Extractor Structure. Glyph features are obtained by sequentially performing image vectorization, vector addition, and dimensional mapping on 32×32 sized glyph images.

- (4) **Feature Fusion Module Based on Improved Transformer** In the improved Transformer model, Eq. (9) realizes the learning of positional encoding, treating it as a learnable embedding matrix. Equation (10) concatenates glyph feature vectors containing lexical features with character vectors, then combines them with learned positional encoding. Then, the multi-head self-attention mechanism is used to capture the interactions between features, and the outputs from different heads are concatenated to form the final fusion vector.

$$\begin{aligned} PE_{(pos)} &= W_{pos}, \\ V_{fusion} &= [V_{glyph}; V_{word}; V_{lex}] + PE_{(pos)}. \end{aligned} \quad (9)$$

Where $W \in \mathbb{R}^{\max_len \times d_{model}}$ is the learnable positional embedding matrix, \max_len is the maximum length of the sequence, d_{model} is the dimension of the embedding

vector, and W_{pos} denotes the pos -th row of the matrix W . V_{glyph} , V_{word} and V_{lex} respectively represent glyph feature vectors, dynamic word vectors, and character vectors with integrated lexical features, and $PE_{(pos)}$ represents the learned positional encoding.

$$\begin{aligned} \text{Attention}_h(Q, K, V) &= \text{softmax}\left(\frac{Q_h K_h^T}{\sqrt{d_k}}\right)V_h, \\ V' &= \text{Concat}(\text{Attention}_1, \dots, \text{Attention}_H)W^O, \\ \text{FFN}(V') &= \text{ReLU}(V'W_1 + b_1)W_2 + b_2. \end{aligned} \quad (10)$$

In the above equation, W_1 , W_2 , b_1 and b_2 are learnable parameters, and V' is the feature fusion vector obtained through the multi-head self-attention mechanism. The output of the feed-forward network can be used as the final feature vector, denoted as $V_F = \text{FFN}(V')$, for subsequent sequence labeling tasks.

3.2 Sequence Encoding Layer

After fusing the multi-feature vectors, the fused vectors are fed to the sequence coding layer, which models the dependencies among the features. In this paper, we employ a time-based bi-directional Long Short-Term Memory Network (Bi-LSTM) as the sequence encoding layer.

3.3 Sequence Decoding Layer

The sequence decoding layer in our model is comprised of a Conditional Random Field (CRF [16]) model, which is a fundamental model extensively used in NLP tasks such as word separation, named entity recognition, and lexical annotation. The CRF model generates optimal tag sequences that conform to grammatical rules by considering the logical relationships between tags.

4 Experiments

In this paper, we evaluate the effectiveness of our model through a series of experiments. We also conduct comparison tests to benchmark the performance of our proposed DFL-NER model against other deep learning-based methods.

4.1 Results and Analysis

The experiment evaluation was conducted on three widely used Chinese datasets: Weibo dataset, Resume dataset, and MSRA dataset. Tables 1, 2, and 3 present the performance results of our method on the Weibo, Resume, and MSRA datasets, respectively, compared to the baseline model.

Weibo: Table 1 presents the F1-score results obtained on the Weibo dataset for named entities (NE), nominal entities (NM), and overall (Overall). Compared to the baseline model on Overall, the F1-score is improved by 7.14%, while the F1-score on NE and NM are improved by 11.4% and 2.96%, respectively.

Table 1. Experimental results of Weibo dataset.

Models	NE	NM	Overall
CAN-NER	55.38	62.98	59.31
LR-CNN	57.14	66.67	59.92
Lattice	53.04	62.25	58.79
Lexicon (LSTM)(Baseline)	59.08	62.22	61.42
BERT + LSTM + CRF	69.65	64.62	67.33
DFL-NER	70.48	65.18	68.56

Resume: From the data in Table 2, it can be observed that on the Resume dataset, the DFL-NER model shows an improvement of 0.37% in F1 score compared to the baseline model (Lexicon (LSTM)), increasing from 95.53% to 95.90%.

Table 2. Experimental results of Resume dataset.

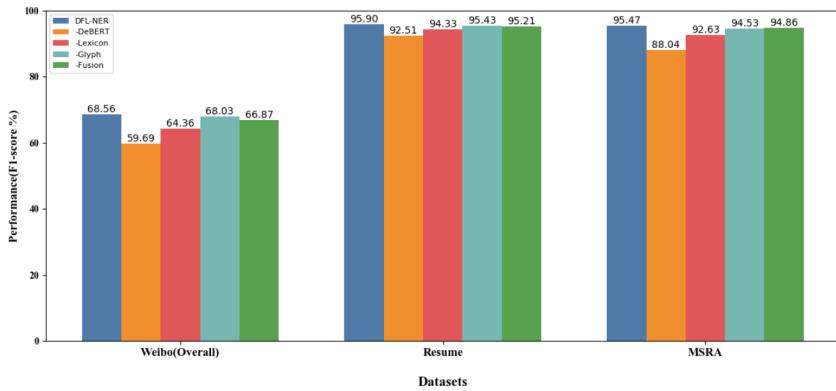
Models	P	R	F1
CAN-NER	95.05	94.82	94.94
LR-CNN	95.37	94.84	95.11
Lattice	94.81	94.11	94.46
Lexicon (LSTM)(Baseline)	95.30	95.77	95.53
BERT + LSTM + CRF	95.75	95.28	95.51
DFL-NER	96.11	95.46	95.90

MSRA: From the data in Table 3, it can be observed that on the MSRA dataset, the DFL-NER model shows an improvement of 1.81% in F1 score compared to the baseline model (Lexicon (LSTM)), increasing from 93.66% to 95.47%.

The ablation experiment results in Fig. 5 show that the model's performance decreases after removing each component. Specifically, excluding the word vectors generated by DeBERTa results in a 3.27% decrease in F1 score; including lexical features increases the F1 score by 1.48%; removing glyph features leads to a 0.35% decrease in F1 score.

Table 3. Experimental results of MSRA dataset.

Models	P	R	F1
CAN-NER	93.53	92.42	92.97
LR-CNN	94.50	92.93	93.71
Lattice	93.57	92.79	93.18
Lexicon (LSTM)(Baseline)	94.63	92.70	93.66
BERT + LSTM + CRF	95.06	94.61	94.83
DFL-NER	95.45	95.48	95.47

**Fig. 5.** Histogram of ablation experiment results

5 Conclusion

This paper introduces DFL-NER, a novel Chinese Named Entity Recognition model, which integrates glyph and lexical information effectively through an improved Transformer feature fusion method, significantly boosting model performance. Experimental results demonstrate F1 score improvements of 7.14%, 0.37%, and 1.81% on the Weibo, Resume, and MSRA datasets, respectively, compared to the traditional SoftLexicon model. This underscores the effectiveness of DFL-NER in Chinese Named Entity Recognition tasks, particularly highlighting the crucial role of its feature fusion method in enhancing model performance.

Future work will continue to explore how to further optimize the feature fusion strategy and how to apply this model to a wider range of natural language processing tasks.

Acknowledgments. This work is supported by the National Natural Science Foundation of China (No. 62377036).

Disclosure of Interests. We declare that we do not have any commercial or associative interest that represents a conflict of interest in connection with the work submitted.

References

1. Liu, P., Guo, Y., Wang, F., et al.: Chinese named entity recognition: the state of the art. *Neurocomputing* **473**, 37–53 (2022)
2. Ma, R., Peng, M., Zhang, Q., et al.: Simplify the usage of Lexicon in Chinese NER. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 5951–5960 (2020)
3. Zhang, Y., Yang, J.: Chinese NER using lattice LSTM. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1554–1564 (2018)
4. Mikolov, T., Sutskever, I., Chen, K., et al.: Distributed representations of words and phrases and their compositionality. *Adv. Neural Inf. Process. Syst.* **26** (2013)
5. Sun, Y., Lin, L., Yang, N., Ji, Z., Wang, X.: Radical-enhanced Chinese character embedding. In: Loo, C.K., Yap, K.S., Wong, K.W., Teoh, A., Huang, K. (eds.) *ICONIP 2014*. LNCS, vol. 8835, pp. 279–286. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-12640-1_34
6. Zhang, J., Yu, X., Wang, Z., et al.: GBWNER: a named entity recognition method based on character glyph and word boundary features for Chinese EHRs. *J. King Saud Univ.-Comput. Inf. Sci.* **35**(8), 101654 (2023)
7. Dong, Y., Bai, J., Wang, L., et al.: Chinese named entity recognition combining prior knowledge and glyph features. *J. Comput. Appl.* **44**(3), 702 (2024)
8. Tan, Q., Li, Y., Xiong, X., et al.: Improving Chinese named entity recognition by glyph and part-of-speech. In: 2023 6th International Conference on Artificial Intelligence and Big Data (ICAIBD), pp. 794–799. IEEE (2023)
9. Song, X., Yu, H., Li, S., et al.: Robust Chinese named entity recognition based on fusion graph embedding. *Electronics* **12**(3), 569 (2023)
10. He, P., Liu, X., Gao, J., et al.: Deberta: Decoding-enhanced bert with disentangled attention. arXiv preprint [arXiv:2006.03654](https://arxiv.org/abs/2006.03654) (2020)
11. Chen, A., Yin, C.: Crw-ner: exploiting multiple embeddings for Chinese named entity recognition. In: 2021 4th International Conference on Artificial Intelligence and Big Data (ICAIBD), pp. 520–524. IEEE (2021)
12. Luo, L., Yang, Z.H., Song, Y.W., et al.: Chinese clinical named entity recognition based on stroke ELMo and multi-task learning. *Chinese J. Comput.* **43**(10), 1943–1957 (2020)
13. Meng, Y., Wu, W., Wang, F., et al.: Glyce: glyph-vectors for Chinese character representations. *Adv. Neural Inf. Process. Syst.* **32** (2019)
14. Sun, Z., Li, X., Sun, X., et al.: ChineseBERT: chinese pretraining enhanced by glyph and pinyin information. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 2065–2075 (2021)
15. Dai, Y., Gieseke, F., Oehmcke, S., et al.: Attentional feature fusion. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 3560–3569 (2021)
16. Lafferty, J., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data (2001)



YOLO-Fire: A Fire Detection Algorithm Based on YOLO

Bo Xu^(✉) Fengyou Hua Qun Yang Tao Wang

Inspur Electronic Information Industry Co., Ltd., Jinan 250101, China

xubo03@ieisystem.com

Abstract. Fire incidents pose a significant threat to public safety and the security of lives and properties. Currently, deep learning methods deployed on edge devices are commonly utilized for fire detection. However, these methods suffer from poor real-time performance, low accuracy, and high false alarm rates. To address these challenges, we propose a novel fire detection algorithm named YOLO-Fire, built upon the foundation of the state-of-the-art YOLOv5s. Firstly, we replace the original YOLOv5's C3 structure with SimpleC3 to reduce the model's parameter count without compromising feature extraction capability. Secondly, we adopt a single-input dynamic upsampler to better preserve crucial features. Lastly, we employ the Focal WIoU-loss function to mitigate penalties for distance and aspect ratio variations, thereby enhancing the detection capability of irregular objects. Experimental results on a fire dataset demonstrate that the YOLO-Fire algorithm achieves a mean Average Precision (mAP) of 58.2% compared to 57.1% of YOLOv5s, with precision (P) increasing from 64% to 65.9%. Furthermore, the YOLO-Fire model's GFLOPs decreases from 15.8GFLOPs of YOLOv5s to 11.2GFLOPs, not only enhancing detection accuracy but also enabling real-time performance on edge devices.

Keywords: YOLO-Fire · fire detection · SimlpeC3 · dynamic upsampler · WIoU

1 Introduction

Fire incidents pose a significant threat to public safety and the security of lives and properties, making timely detection and warning crucial for maintaining public safety and social order. In practical fire detection, external environmental factors such as sunlight, yellow objects, and glass reflections often lead to false alarms. Currently, certain fire detection methods rely on sensor-based approaches [1], which have limited detection range and can only detect fires after they occur, failing to provide early warnings.

Traditional machine learning methods have also found applications in fire detection. Traditional approaches, which focus on image processing-based fire detection methods, primarily rely on recognizing and detecting easily identifiable RGB colors and edge features in fire [2]. Although they partly address interference issues under different lighting conditions, they are greatly influenced by the environment and struggle to adapt to complex scenes, resulting in high false alarm rates.

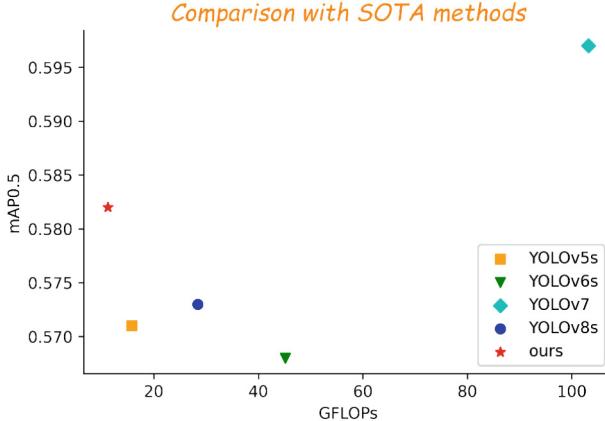


Fig. 1. The figure provides a comparison with current state-of-the-art (SOTA) methods, including YOLOv5s, YOLOv6s, YOLOv7, and YOLOv8s. The comparison is conducted at a consistent input resolution of 640×640 , with a focus on accuracy (mAP0.5). Considering both GFLOPs and mAP0.5, our algorithm demonstrates superior performance.

With the advancement of computer vision technology and improvements in computer hardware, deep learning-based object detection algorithms have been widely applied in fire detection. These algorithms can be categorized into two-stage and one-stage object detection methods based on convolutional neural networks. Classic two-stage object detection algorithms include RCNN [3], Fast-RCNN [4], Faster-RCNN [5], and Mask-RCNN [6]. One-stage object detection algorithms identify objects through regression and include classic algorithms such as YOLOv5 [7], YOLOv6 [8], YOLOv7 [9], the latest YOLOv8 [10], and SSD [11], which integrates Faster-RCNN's object proposal strategy.

While deep learning-based detection algorithms offer higher accuracy and better robustness compared to traditional machine learning methods, they often lack real-time performance, particularly when deployed on edge devices, thereby failing to meet the requirements for real-time fire detection. To address this issue, we propose YOLO-Fire, a lightweight and high-precision deep learning-based fire detection algorithm. YOLO-Fire's detection accuracy and GFLOPs compared to current typical algorithms are shown in Fig. 1.

YOLO-Fire is a novel algorithm based on YOLOv5s, and this paper systematically introduces the rethinking of YOLO-Fire, mainly including the following aspects:

SimpleC3. Adopted to reduce the model parameters without compromising feature extraction capabilities, while the ResConv module is employed to enhance feature extraction capabilities.

Dynamic Upsampler. Employed to better preserve crucial features, reduce artifacts, and improve the model's detection accuracy.

Focal WIoU-loss. Applied to optimize the loss function, reducing penalties for distance and aspect ratio variations in blurry targets, thereby enhancing the detection capability of irregular objects such as fire.

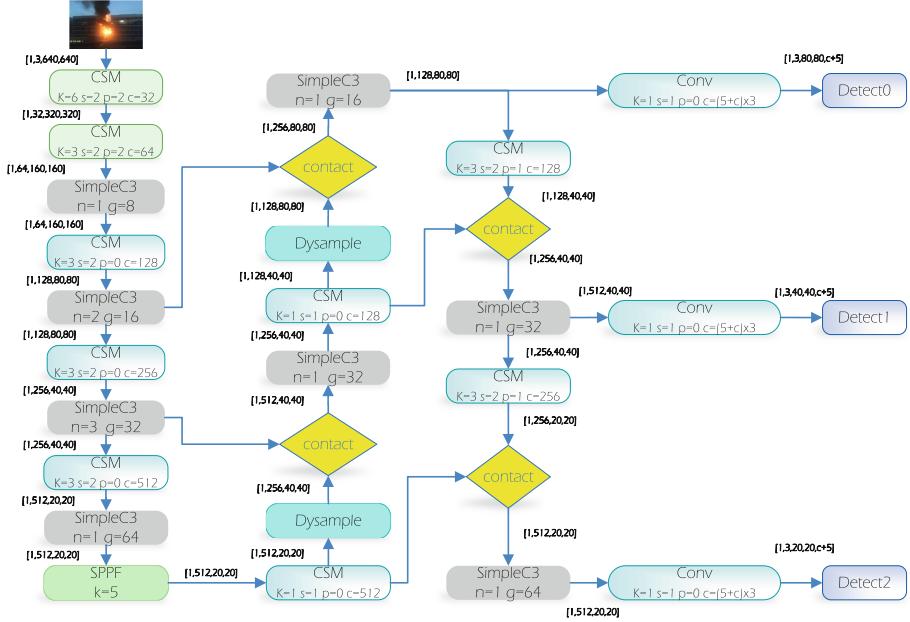


Fig. 2. Network Architecture of YOLO-Fire.

2 YOLO-Fire

The overall framework of YOLO-Fire is based on YOLOv5s, with a series of improvements and adjustments made upon it. The network architecture of YOLO-Fire is illustrated in Fig. 2.

The algorithm employs three different detection layers to detect fire targets, performing sampling at 8x, 16x, and 32x scales respectively. The shallow-level representation feature maps are fused with deep-level high-semantic features using the FPN network [18], which propagates strong semantic features from top to bottom. PANet is utilized to integrate shallow-level representation information, transmitting strong localization features from bottom to top. Ultimately, three detection heads with scale sizes of 20×20 , 40×40 , and 80×80 respectively are obtained for individual target detection.

2.1 SimpleC3

To improve the model's detection efficiency and enable real-time detection, this study introduces a novel network architecture to replace the original feature extraction structure of YOLOv5. The structural diagram of SimpleC3 is depicted in Fig. 3.

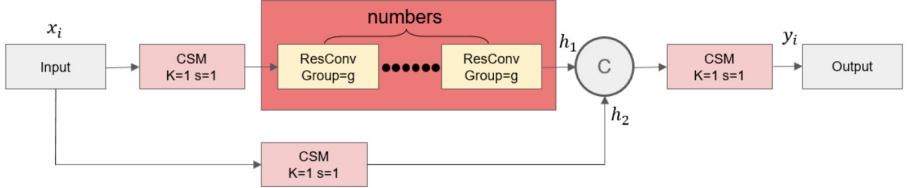


Fig. 3. Network Structure Diagram of SimpleC3, where “numbers” represent the inclusion of a certain number of ResConv modules, and “C” indicates concatenation of two inputs.

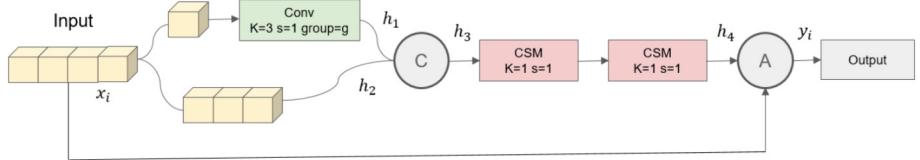


Fig. 4. Network Structure Diagram of the ResConv Module, where “C” represents concatenation of two inputs, and “A” represents element-wise addition of two inputs at corresponding positions.

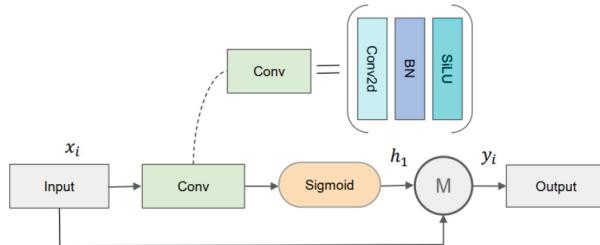


Fig. 5. The network structure diagram of the CSM module, where “M” represents the element-wise multiplication of two inputs at corresponding positions.

The input features are simultaneously extracted and encoded along two different directions, generating two sets of feature maps. One set of feature maps is used to fuse deeper semantic information, while the other set preserves the shallow-level information of the input features. Finally, the two sets of feature maps are aggregated, and the aggregated feature maps are further integrated using the CSM module to obtain the output y_i of the model. To extract deeper semantic information, this paper proposes the ResConv module for feature extraction, which employs a cascaded ResConv module to delve deeper into feature extraction. The final output y_i is obtained as shown in Eq. (1).

$$y_i = CSM([h_1, h_2]) \quad (1)$$

$$h_1 = ResConv(CSM(x_i)) \quad (2)$$

$$h_2 = CSM(x_i) \quad (3)$$

where h_1 represents deeper semantic information and h_2 represents shallow-level information.

To enable cascaded extraction of deeper semantic information while reducing model parameters and computational complexity, this paper designs the structure diagram of the ResConv module for deep feature information extraction, as shown in Fig. 4.

The ResConv module first splits the input features along the channel dimension [12], dividing them into two groups with a ratio of 1:3. The group with the smaller dimension undergoes convolutional operations independently, and the results are concatenated with the group having the larger dimension. The concatenated features then undergo further feature extraction using two sets of 1×1 kernels in the CSM module. Subsequently, the extracted features are added element-wise to the input features, resulting in the output y_i of this module. The output y_i is represented by Eq. (4).

$$y_i = x_i + h_4 \quad (4)$$

$$h_4 = \text{CSM}(\text{CSM}(h_3)) \quad (5)$$

$$h_3 = [h_1, h_2] \quad (6)$$

$$h_2 = x_i[:, \frac{3}{4} * \text{channel}, :, :] \quad (7)$$

$$h_1 = \text{Conv}(x_i[:, \frac{3}{4} * \text{channel} :, :, :]) \quad (8)$$

where “channel” represents the feature dimension of the input features.

To better enable the model to distinguish the boundary information of fire, this paper introduces the CSM (Convolutional-Sigmoid-Multiply) module. The CSM module divides the input features into two directions: one direction undergoes Convolutional (Conv) operations, while the other direction undergoes Sigmoid operations. The outputs of these operations are then multiplied element-wise to obtain the output y_i , as shown in Eq. (9). The structure diagram of the CSM module is depicted in Fig. 5.

$$y_i = x_i \cdot h_1 \quad (9)$$

$$h_1 = \frac{1}{1 + e^{-\text{Conv}(x_i)}} \quad (10)$$

where “Conv” represents a set of processing functions, including 1 group of 2D convolution, Batch Normalization (BN), and SiLU activation function.

2.2 Dynamic Upsampler

Upsampling is an essential component in dense prediction models for gradually restoring feature resolution. The most commonly used upsamplers are nearest neighbor (NN) and bilinear interpolation, which follow fixed rules for interpolating upsampled values. To

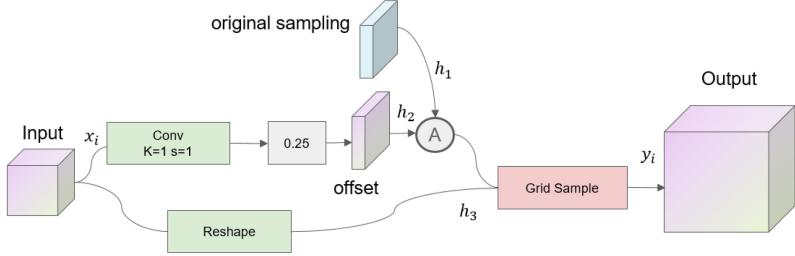


Fig. 6. Network Structure Diagram of the Dynamic Upsampler, Where “offset” denotes the transformation of the input to the size of the output features, and “original sampling” represents generating a tensor of the same size as the output dimensions based on the scaling ratio.

Increase flexibility, reduce artifacts, and better preserve important features, this paper adopts a single-input dynamic upsampler based on the advantages of the single-input dynamic upsampler [13]. The network structure diagram of the dynamic upsampler is shown in Fig. 6.

Algorithm 1 Focal WIoU-loss

Input: detected bounding box: B ; ground truth bounding box: B^{gt} ; width and height of detected bounding box: W_B, H_B ; width and height of ground truth bounding box: $W_{B^{gt}}, H_{B^{gt}}$;

Output: $L_{focalWIoU}$

1: (x^B, y^B) is the central coordinates of box B ; $(x^{B^{gt}}, y^{B^{gt}})$ is the central coordinates of box B^{gt} ;

$$2: S_u = W_B * H_B + W_{B^{gt}} * H_{B^{gt}} - |x^{B^{gt}} - x^B| * |y^{B^{gt}} - y^B|$$

$$3: \text{IoU} = \frac{|x^{B^{gt}} - x^B| * |y^{B^{gt}} - y^B|}{S_u}$$

$$4: L_{\text{IoU}} = 1 - \text{IoU}$$

$$5: R_{WIoU} = \exp\left(\frac{(x^{B^{gt}} - x^B)^2 + (y^{B^{gt}} - y^B)^2}{(W_{B^{gt}}^2 + H_{B^{gt}}^2)}\right)$$

$$6: L_{WIoU} = R_{WIoU} L_{\text{IoU}}$$

$$7: \alpha = e * \beta, 1/e \leq \beta \leq 1$$

$$8: C = (2\alpha \ln \beta + \alpha)/4$$

$$9: L_f = \begin{cases} -\frac{\alpha x^2(2 \ln(\beta x) - 1)}{4}, & 0 < x \leq 1; 1/e \leq \beta \leq 1 \\ -\alpha \ln(\beta)x + C, & 1 < x; 1/e \leq \beta \leq 1 \end{cases}$$

$$10: L_{\text{loca}} = \sum_{i \in \{B_i, B_i^{gt}\}} L_f(|B_i - B_i^{gt}|)$$

$$11: L_{focalWIoU} = (1 - L_{\text{IoU}})^\gamma L_{WIoU}$$

The upsampler first performs feature sampling and deformation operations on the input features separately. Then, it obtains the upsampled features y_i through the Grid

Sample operation, as shown in Eq. (11).

$$y_i = \text{GridSample}(h_1 + h_2, h_3) \quad (11)$$

where h_1 represents the original sampling grid, h_2 represents the result of x_i after convolution, scaling, and offset, and h_3 represents the deformation of x_i for facilitating the Grid Sample operation.

2.3 Focal WIoU-Loss

The loss function for bounding box regression (BBR) is essential to object detection. Its good definition will bring significant performance improvement to the model. Most existing works assume that the examples in the training data are high quality and focus on strengthening the fitting ability of BBR loss. If we blindly strengthen BBR on low-quality examples, it will jeopardize localization performance. Focal-EIoU v1 was proposed to solve this problem, but due to its static focusing mechanism (FM), the potential of non-monotonic FM was not fully exploited. Based on this idea, we propose an IoU-based loss with a dynamic non-monotonic FM named Wise-IoU (WIoU) [14]. The dynamic non-monotonic FM uses the outlier degree instead of IoU to evaluate the quality of anchor boxes and provides a wise gradient gain allocation strategy. The Focal L1 loss function [15] can solve problems such as the number of small samples is small and the pixel quality of small samples is relatively poor compared to large samples.

Taking advantage of the benefits of both the Focal L1 loss function and WIoU, we integrate them to create the focal WIoU loss. The calculation of focal WIoU loss is summarized in Algorithm 1.

Direct adoption of $L_{focalWIoU}$ based on L_{MPDIoU} as mentioned above can lead to a weakening of the weighted effect on the object box. The gradient of $L_{focalL1}$ can be expressed as $\frac{\partial L_{focalL1}(L_{WIoU}(B_i))}{\partial B_i} = \frac{\partial L_{focalL1}(L_{WIoU}(B_i))}{\partial L_{WIoU}(B_i)} \frac{\partial L_{WIoU}(B_i)}{\partial B_i}$, $\frac{\partial L_{WIoU}(B_i)}{\partial B_i}$ are variables, as B_i approaches B_i^{gt} and the difference between them becomes vanishingly small, L_{MPDIoU} approaches 0, the overall gradient will be very small. To address these concerns, we use the weighting factor of L_{WIoU} to effectively reweight the loss. Additionally, a parameter γ is introduced to control the degree of outlier suppression. Equation (12) is used to calculate $L_{focalWIoU}$, which represents the focal WIoU-loss.

$$L_{focalWIoU} = (1 - L_{WIoU})^\gamma L_{WIoU} \quad (12)$$

3 Experiments

3.1 Implementation Details

This experiment utilized the Inspur NE5260M5 edge server equipped with 2 * Intel(R) processors. The software environment included CUDA 11.8, PyTorch 2.0.0, and Python 3.10. YOLOv5s served as the base network for this research, and various feature ablation experiments were conducted on YOLOv5s to compare and analyze the proposed improvements.

The study utilized a proprietary dataset for training the network, conducting ablation experiments, and evaluating the effectiveness of the proposed improvements. Our dataset comprises 23,653 images, with 80% allocated for training, 10% for validation, and 10% for testing. The data was annotated using LabelImg software.

The network training parameters were configured as follows: batch size of 32, maximum iterations of 300, input image size of 640×640 , and a strategy to dynamically adjust the learning rate using cosine annealing with an initial learning rate of 0.01.

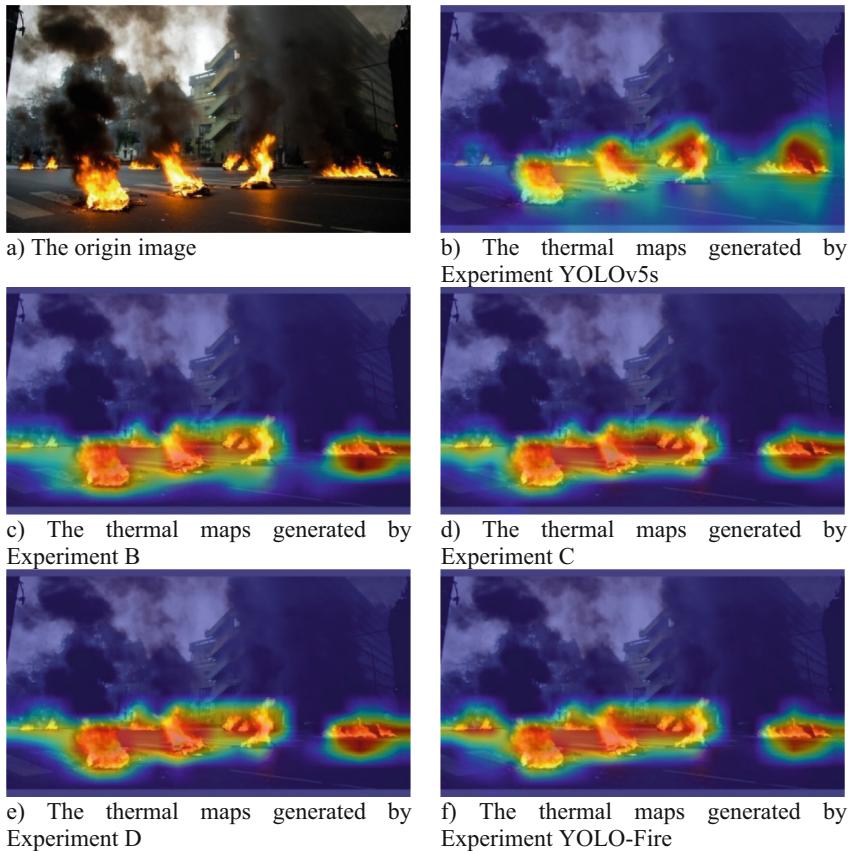


Fig. 7. The thermal maps generated by different experiments.

3.2 Ablation Studies

The experiment was divided into eight groups, with four of them based on YOLOv5s. To illustrate the effectiveness of the improved features, we compared the differences between different experiments using two metrics: mAP and GFLOPs. The improvements and performance comparisons for each experiment are shown in Table 1.

Table 1. Comparative Table of Ablation Experiment Improvements.

Experiments	SimpleC3	Dysample	Focal WIoU-loss	mAP0.5/%	GFLOPs
YOLOv5s	×	×	×	57.1	15.8
YOLOv6s	×	×	×	56.8	45.2
YOLOv7	×	×	×	59.7	103.2
YOLOv8s	×	×	×	57.3	28.4
B	✓	×	×	57.3	11.2
C	✓	✓	×	57.9	11.2
D	✓	×	✓	57.7	11.2
YOLO-Fire	✓	✓	✓	58.2	11.2

Firstly, the feature extraction effectiveness of each experiment was analyzed by visualizing the final feature maps of the models in five experimental groups. The heatmaps of the five algorithms are shown in Fig. 7. From Fig. 7, it can be observed that compared to the original YOLOv5s, YOLO-Fire extracts more target feature points with fewer erroneous target positions, effectively improving the detection accuracy of fire and reducing the model’s false alarm rate (Table 2).

Different methods used the same dataset and parameter settings during the training process. Based on the parameter information during training, we plotted the loss function curves and mAP0.5 curves of the experiments with different algorithms, as shown in Fig. 8. From the graphs, it can be seen that compared to YOLOv5s, YOLO-Fire achieves lower loss function values and converges faster. Its mAP0.5 value has increased by 1.1%.

3.3 Experimental Results Analysis

For each category, the mAP0.5 values on the fire dataset are shown in Table 2. It can be observed from Table 2 that YOLO-Fire achieves significantly better detection values for each category compared to YOLOv5s. Specifically, the accuracy for fire increases from 72.3% to 73.1%, and the accuracy for smoke improves from 41.9% to 43.4%.

The overall performance metrics on the fire dataset are presented in Table 3. It can be seen from Table 3 that YOLO-Fire exhibits improvements in mAP0.5, mAP0.5:0.95, and precision (P). Particularly, there is a 1.1% increase in mAP0.5 and a 3.3% increase in precision (P). Additionally, the model parameters decrease by 1.9 M, and the computational power required by the model decreases by 4.6 GFLOPs.

To better evaluate the generalization ability of YOLO-Fire, two sets of images were randomly extracted from the fire dataset, each representing different scenes, for testing purposes. The detection results of YOLO-Fire and the original YOLOv5s are compared, and the detection effect images are shown in Fig. 9.

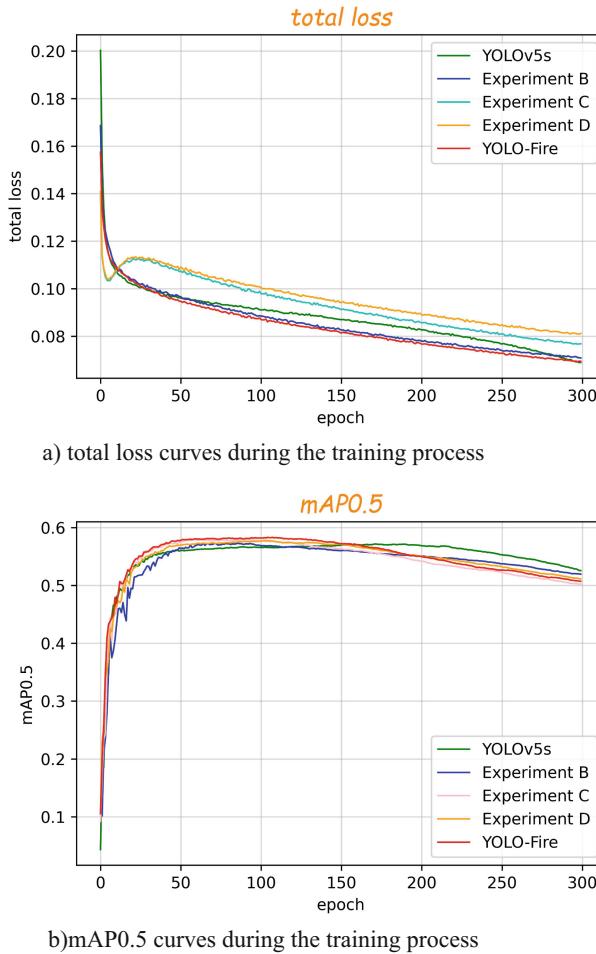


Fig. 8. Main performance curves during the training process.

Table 2. mAP@0.5 values for different categories in each experiment.

Experiments	fire/%	smoke/%
YOLOv5	72.3	41.9
B	72.6	42.0
C	72.6	43.2
D	72.8	42.7
YOLO-Fire	73.1	43.4

Table 3. Overall performance metrics for each experimental method.

Experiments	mAP0.5/%	P/%	R/%	Resolution	Parameters	GFLOPs
YOLOv5	57.1	64.0	55.3	640 × 640	7.1M	15.8
B	57.3	62.1	55.9		5.2M	11.2
C	57.9	67.3	53.1		5.2M	11.2
D	57.7	65.4	53.9		5.2M	11.2
YOLO-Fire	58.2	65.9	54.7		5.2M	11.2



a) The origin image



b) The detection result by YOLOv5s



c) The detection result by YOLO-Fire

**Fig. 9.** Results of different algorithms in different scenarios. The YOLOv5s failed to recognize the fire in the forest and many false detections have occurred in the factory workshop.

4 Conclusion

To address the issues of poor real-time performance, low accuracy, and high false detection rates in existing fire detection algorithms, this paper proposes a new fire detection algorithm tailored for real-time detection on edge devices: YOLO-Fire. Firstly, we replace the original YOLOv5's C3 structure with SimpleC3 to enhance the model's feature extraction capability and reduce its parameter count. Secondly, we employ a dynamic upsample to increase flexibility, reduce artifacts, and better preserve effective features without increasing the model's computational load. Finally, we replace the loss function with Focal WIoU-loss to decrease penalties for distance and aspect ratio variations in blurry targets, thus enhancing the model's generalization ability. Experimental results demonstrate that, on the fire dataset, the YOLO-Fire algorithm not only improves the accuracy of fire detection compared to YOLOv5s but also enhances the real-time performance of the model, making it more suitable for deployment on edge devices for fire detection applications.

References

1. Zhao, K.: Hardware design of smoke detector based on wireless network node. *Control Eng. China* **25**(11), 1998–2002 (2018)
2. Yan, X., Wang, L., Bu, L.: Fuzzy Clustering Segmentation Algorithm of Flame Image Based on Multi-Dimensional Color Vector Space **3**, 368–373 (2012)
3. Girshick, R., et al.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2014)
4. Girshick, R.: “Fast r-cnn”. In: Proceedings of the IEEE International Conference on Computer Vision (2015)
5. Ren, S., et al.: Faster r-cnn: towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.* **28** (2015)
6. He, K., et al.: “Mask r-cnn”. In: Proceedings of the IEEE International Conference on Computer Vision (2017)
7. Ultralytics. YOLOv5 2020. <https://github.com/ultralytics/yolov5>. Accessed 1 Jan 2023
8. Li, C., et al.: YOLOv6: A single-stage object detection framework for industrial applications. arXiv preprint [arXiv:2209.02976](https://arxiv.org/abs/2209.02976) (2022)
9. Wang, C.Y., Bochkovskiy, A., Liao, H.Y.M.: YOLOv7: trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2023)
10. Jocher, G.; Chaurasia, A.; Qiu, J.: YOLO by Ultralytics. GitHub. 1 January 2023. <https://github.com/ultralytics/ultralytics>. Accessed 12 Jan 2023
11. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A.C.: Ssd: Single shot multibox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 21–37. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_2
12. Chen, J., et al.: Run, don't walk: chasing higher FLOPS for faster neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2023)
13. Liu, W., et al.: Learning to upsample by learning to sample. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2023)

14. Tong, Z., et al.: Wise-IoU: bounding box regression loss with dynamic focusing mechanism. arXiv preprint [arXiv:2301.10051](https://arxiv.org/abs/2301.10051) (2023)
15. Zhang, Y.-F., et al.: Focal and efficient IOU loss for accurate bounding box regression. Neurocomputing **506**, 146–157 (2022)



From Vision to Sound: The Application of ViT-LSTM in Music Sequence

Menghao Fang¹, Shuo Zhang², Xia Li², Liangbin Yang¹, and Zixiao Kong¹(✉)

¹ School of Cyber Science and Engineering, University of International Relations,
Beijing 100091, China

ylb@uir.cn, kongzixiao@uir.edu.cn

² Marine Engineering College, Dalian Maritime University, Liaoning 116000, China

Abstract. This study combines the Visual Transformer (ViT) and Long Short-Term Memory (LSTM) networks to propose a deep learning model called ViT-LSTM, aimed at generating music sequences from music data features. We conducted this research using the MAESTRO dataset, which covers various MIDI files primarily focused on piano music. The data preprocessing involved hot encoding techniques and the creation of training and validation datasets using a sliding window strategy. We comprehensively evaluated the performance of this model using two metrics: generation diversity and generation quality. The experimental results demonstrate that ViT-LSTM can generate diverse and high quality music compared to other classical deep learning models, with a higher signal-to-noise ratio in the generated music.

Keywords: Vision Transformer · Long Short-Term Memory · Music Generation · Deep learning · Neural network

1 Introduction

The rapid development of artificial neural networks is gradually blurring the line between art and science. Many research result show how to transform areas previously considered to be entirely human (due to the creative or intuitive nature of tasks) into algorithmic methods. Music is one of these areas.

At first, people explored a variety of artificial intelligence methods for music generation, including rule-based and Markov models. Illiac Suite in 1979 is an example of these early methods, and with the advent of artificial neural networks in the late 1980s, new methods of music generation began to emerge. These early work laid the foundation for the music generation technology based on deep learning that we see today [1, 2].

As a method of music generation, deep learning has gradually become the focus, beyond the traditional prediction and classification tasks. This shift is demonstrated by Google's Magenta Research Group and Spotify's CTRL Research Group, which use deep learning for innovative music applications [3].

M. Fang and S. Zhang—are Equally Contributed

A course on music generation based on deep learning techniques explores the development from artificial neural networks to deep learning, highlighting the progress and transformation of methodology over time [4, 5]. Recurrent neural networks (RNN) [6] and long-term memory (LSTM) [7] networks, especially LSTM networks, have been proved to be effective tools for music generation. In music generation, LSTM network can generate creative new music by learning the patterns and rules of music sequence data. Its long-term dependence and memory ability make it suitable for dealing with complex structures and music development in music, and can produce more coherent and emotional music works. Therefore, LSTM has been widely used in the field of music generation, and has become the core component of many music generation models.

Deep learning techniques are very suitable for dealing with complex music generation tasks [8]. The continuous research and development in this field promotes the possibility of human intelligence-driven music creation. As recently proposed, a LSTM-based PRECON-LSTM model is used to generate music and use ABC [9] notation to represent music. The purpose of this model is to learn the patterns of notes and use feature level modeling to predict notes. According to the subjective evaluation of peers, the model can produce pleasant and high-quality music. And you can find some selected music tracks generated by PRECON-LSTM on the music sharing community (AI music generation website) [10].

However, the music works generated by neural networks usually rely on training data, which may lack real innovation or individuality, and their performance greatly depends on the quality and diversity of training data. If the dataset is biased or limited, the generated music may reflect these limitations, resulting in a single style or deviation from the expected type of music. Therefore, we use the ViT-LSTM model, and we compare ViT-LSTM with LSTM, VAE [11], Transformer [12] and other models to show the advantages of this technology.

2 Related Work

In the field of music and language generation, although artificial neural networks show many advantages, there is still the problem of generating statistically similar but actually wrong or boring sequences. As early as 1971, Xenakis used Markov chains to artificially create music [13], they were used to predict output based on previous inputs in the Markov chain. But the Markov model cannot produce novel music. Sinith *et.al* introduces a method to identify wavelets and filter banks in musical instruments such as violins and flutes, and uses these wavelets to automatically synthesize music using hidden Markov models [14].

Dieleman et al. try to synthesize the original audio signal using *WaveNet* architecture [15], but the original audio signal contains so much information that the current neural network can not fully understand it. Later, *Kingma* and *Welling* introduced variable automatic encoders to improve efficiency. *Karpathy* [16] developed a LSTM model called *CharRNN*, which takes a large text corpus as input for training and predicts the output character by character.

Johnston et al. [17] and *Huang et al.* [18] use the model introduced by *Karpathy* to generate music. In these works, music is represented by ABC notation, which is the textual representation of music. These works predict the following notes word by word.

Sturm et al. [19] use a similar model, but do very basic preprocessing of the input ABC dataset. The problem with the above works is that they are prone to ABC notation with grammatical errors, which can lead to unpleasant music synthesis or even no music synthesis.

Therefore, this paper studies the combination of LSTM and visual attention converter (ViT). LSTM networks are good at dealing with long-term time-dependent serial data [20], which is very important for music generation. Music usually contains complex time structures and repetitive patterns, and LSTM can effectively capture these long-term dependencies to produce coherent and structured music clips. The ViT [21] uses attention mechanisms to identify and process important features [7–21].

3 Methods

This paper constructs an innovative deep learning model, which combines LSTM with ViT [20] and works with MAESTRO datasets to generate rhythmic music. By combining these two strong neural network architectures, the model can simultaneously capture the time series characteristics of music data and the relationship between data blocks, thus achieving greater expressiveness and creativity in music generation. The overall frame diagram is shown in Fig. 1.

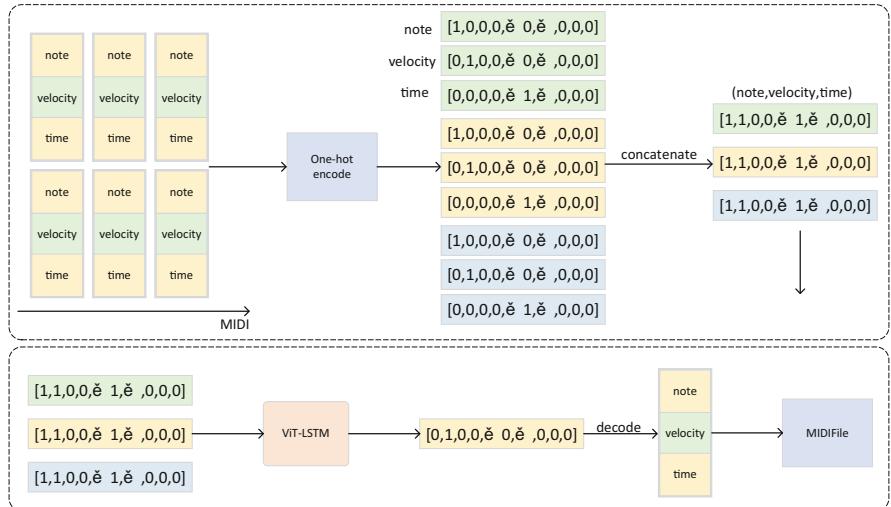


Fig. 1. Overall Frame Diagram

As shown in the Fig. 1, the music data in MIDI format, including note, strength and time attributes, are converted into one-hot coding to form an independent attribute sequence. These sequences are then spliced into a larger sequence and then input into the ViT-LSTM pattern. This model combines the global feature understanding of ViT and the long-term dependency capture ability of LSTM, and is used to generate music.

The output after ViT-LSTM processing is a decoding sequence, which still retains the one-hot encoding form. Eventually, the decoding sequence is converted back to note, strength, and time information, and a new MIDI file is generated.

3.1 Dataset Definition

The key features of MIDI [21] datasets mainly include notes, strength and time. The combination of these features constitute the basic elements of music data. In order to train the model, it needs to learn the internal relationship between these notes, strength and time, so that it can predict the following notes, strength and time series. In the process of model training, the input data x is composed of the sequence from time step i to time step $i + n_{p\text{rev}}$, and the corresponding label y is the sequence after time step $i + n_{p\text{rev}}$. In this way, the model can try to predict future sequences of musical features by looking at previous sequences of notes, dynamics, and timing, so as to generate music or perform other music-related tasks.

3.2 Model Construction

Structure of ViT-LSTM

1. ViT is a switch model based on the attention mechanism, which was originally used to deal with visual data. By applying it to music generation, the model can extract and pay attention to important notes, rhythms and melodies from music clips, and realize the global perception of music structure 7.
2. By combining LSTM and ViT, this paper proposes a novel way to explore the method of music generation, which opens up a new possibility for the application of deep learning in the field of creativity.
3. First of all, we have 30 sets of music data features, each containing notes, note intensity (velocity), and time information. These features are arranged in a data sequence, forming a total of 30 such sequences. Subsequently, these music data feature sequences are sent into the LSTM structure. As a kind of recurrent neural network, LSTM shows a good effect in modeling sequence data. The LSTM network processes the input data and produces the corresponding output results by learning the patterns and dependencies in the music sequence. After processing of the data sequence of the LSTM structure, the output is reshaped by a full connection layer to form a tensor in the shape of (128, 128, 3).

Algorithm 1: Data Training of Algorithm: ViT-LSTM

Require: a generative model

1: Input: (note, velocity, time) sequence of MIDI file

2: Output: ViT-LSTM Model

3: One-hot: sample=One-hot(note, velocity, time) Divide time values into discrete intervals.

4:X: $\{x_i\}_{i=1}^N$, Y: $\{y_j\}_{j=1}^N$ where $x_i = [m_i, m_{i+1}, \dots, m_{i+n_{prev}}]$, $y_i = m_{i+n_{prev}}$

5: Training: network model ViT-LSTM

6: for $i <$ Number of iteration do $\hat{y} = \text{ViT} - \text{LSTM}(x)$

$$L = L_n(\hat{y}_1, y_1) + L_n(\hat{y}_2, y_2) + L_n(\hat{y}_3, y_3)$$

7:end

Next, the tensor of the shape (128, 128, 3) is passed as input to a structure called ViT. ViT, that is, visual-semantic converter, is essentially a converter architecture based on self-attention mechanism. In this process, we set the parameters of ViT: the image size is 128×128 , the size of the image is divided into small pieces 4×4 , the output layer size num-classes is 128, the dim is 32, the depth is 6, the number of heads is 16, the mlp-dim is 64, the dropout is 0.1, and the emb-dropout is 0.1. The ViT structure uses the self-attention mechanism to process the input tensor to capture the global and local information. Finally, the ViT structure produces a processed output that may be applied to music generation or other related tasks. Algorithm 1 demonstrates the training process of ViT-LSTM.

Loss Function Analysis

The data in this paper is processed by one-hot coding before use. This coding method c this characteristic of the data, this article chooses the NLLLoss [22] as the loss function converts the category label into vector form and is used to provide training to the model. In view function of model training. NLLLoss is suitable for multi-classification problems, especially when the output is to predict the probability of each category.

The input feature of the model consists of three parts, which are (x_1, x_2, x_3) , corresponding to $(note, velocity, time)$. On the other hand, the label (y_1, y_2, y_3) represents the $(note, velocity, time)$ sequence data of the $n + i$ time.

$$L_n(\log(\text{softmax}(x)), y) = - \sum_{i=1}^n \text{OneHot}(y)_i \times \log(\text{softmax}(x)_i) \quad (1)$$

$$L = L_n(x_1, y_1) + L_n(x_2, y_2) + L_n(x_3, y_3) \quad (2)$$

where L_n is the NLLLoss loss function, and One-Hot is the encoding method of the data in this paper.

4 Experiment

In this paper, the dataset is encoded first. Excellent coding helps to eliminate the size relationship between (note, velocity, time) features, and can increase dimensional features to better represent different categories. This makes it easier for models to learn patterns

and differences between data. Then, we use the ViT-LSTM model to capture the intrinsic features of MIDI data in order to generate higher quality music works. Finally, we evaluate the performance of the model by verifying the accuracy, and compare it with LSTM, VAE, Transformer and other.

4.1 Data Processing

MAESTRO dataset: The study uses Google's MAESTRO dataset [23], which focusses on piano music and is a collection containing a large number of MIDI files. It aims to support research and application in the fields of music generation, music understanding and machine learning. The core content covers the key elements of piano music, such as notes, pitch, volume and rhythm, etc. These MIDI files come from music works that are professionally trained and performed, from classical concerts and professional piano performances, and cover music works of different composers, times and styles.

One-hot: One-hot coding is a commonly used data coding technique, which is used to convert classified variables into vector representations for machine learning algorithms to process. It maps each different category to a binary vector, where only one element is 1 and the other elements are 0. This coding method makes the classification variables easier to process and express in the machine learning model [24].

$$X_{encoded}, Y_{encoded} = \text{One-hot_encode}(note, velocity, time) \quad (3)$$

First of all, this paper discretizes the time information in the music data set, divides the time value into multiple intervals, and converts the time into discrete type variables according to the interval. Secondly, according to the pitch, volume and discretized time value of each note, one-hot coding is used to convert them into three one-hot vectors respectively. These three vectors are then merged into one vector. The number of categories is determined according to note, velocity, time range, and the scope of different MIDI files is different. For example, suppose that each note, velocity, time has only five categories, as in the Table 1.

Table 1. Feature Coding Diagram.

note	velocity	time	(note, velocity, time)
10000	10000	10000	100001000010000
01000	01000	01000	010000100001000
00100	00100	00100	001000010000100
00010	00010	00010	000100001000010
00001	00001	00001	000010000100001

Data Partition: The purpose of this article is to construct the data set needed for training and verification of the preprocessed music data. Specifically, the sliding-window strategy

is adopted to segment the encoded note data according to a specific length to form the corresponding input and output data pairs.

In this method, firstly, the one-hot coding technology is used to process the original music data, so that the music data can be converted into a more suitable one-hot coding format. Then, according to the set sliding window length (n_{prev}), the processed data set is iteratively processed. In each iteration, 30 consecutive sets of notes are selected as input attributes (X), but the next note of the note is taken as the corresponding target attribute (Y). At the same time, the time information in the target attribute is processed and converted into discrete variables.

Suppose that the note sequence of the original music data set is m_1, m_2, \dots, m_n , where m_i represents the i note and n represents the length of the note sequence. Using the one-hot coding technique to process the music data set, we get $m_i = [0, 0, 0, \dots, 1, \dots, 0, 0, 0]$, (a total of k zeros or ones), where k represents the total number of notes. In each iteration, a continuous n_{prev} note is selected as the input attribute X , and the next note of the note is taken as the corresponding target attribute Y . Therefore, in the iterative process, it can be expressed as follows, and finally get the data set $X : \{x_i\}_{i=1}^N, Y : \{y_j\}_{j=1}^N$ [25].

4.2 Performance Evaluation

In this paper, Generation Diversity (GD) and Generation Quality (GQ) are used as the evaluation indicators of this study.

Generate diversity: GD refers to whether the model can produce a variety of different but still valid outputs.

In this paper, dynamic time warping (DTW) is used to compare the similarity between generated music. DTW can effectively deal with the problem of sequence matching at different speeds, allowing sequences to bend and stretch on the timeline for better alignment and comparison.

$$DTW(i, j) = |x[i] - y[j]| + \min(DTW(i - 1, j), DTW(i, j - 1), DTW(i - 1, j - 1)) \quad (4)$$

$$Novelty = 1 - DTW(i, j) \quad (5)$$

where the $DTW(i, j)$ in the formula represents the minimum distance when matching the first i element in the sequence x with the first j element in the sequence y . In this recursive definition, $x[i]$ and $y[j]$ represent the elements of sequences x and y at indexes i and j , respectively. $|x[i] - y[j]|$ indicates the distance or difference between the i element in the sequence x and the j element in the sequence y . $\min(DTW(i - 1, j), DTW(i, j - 1), DTW(i - 1, j - 1))$ means to calculate the minimum distance by recursively selecting elements in different positions in the sequence x and y to match. At each step, select the minimum distance (that is, the minimum of the three cases), and add the distance of the current position to the cumulative distance of the previous position to get the minimum matching distance of the current position.

Generation Quality: The generated GQ refers to the sound quality of the generated music and whether there is noise in it. Although some models can generate music with

high diversity, that is, high GD values, there may be a large number of noise components. Therefore, it is necessary to evaluate the quality of the generated music.

In this paper, signal-to-noise ratio (SNR) is used to evaluate the sound quality of music. The signal energy can be calculated by the sum of the squares of the amplitude of the signal.

$$E_s = \sum_{i=1}^N x^2(i) \quad (6)$$

where E_s is the energy of the signal, $x(i)$ is the first signal of the sequence, and N is the number of samples of the signal. The noise energy can also be calculated by the sum of squares of the amplitude of the noise signal.

The noise energy can also be calculated by the sum of squares of the amplitude of the noise signal.

$$E_n = \sum_{i=1}^N n^2(i) \quad (7)$$

where E_n is the energy of noise, $n(i)$ is the first signal of the sequence, and N is the number of samples of the signal.

The signal-to-noise ratio is the ratio of signal energy to noise energy, which can be calculated using the following formula.

$$SNR = 10 \log_{10} \left(\frac{E_s}{E_n} \right) \quad (8)$$

$$GQ = \frac{SNR_{mean}}{SNR_{max}} \quad (9)$$

Among them, SNR is the signal-to-noise ratio, E_s is the energy of the signal, E_n is the energy of noise, the mean signal-to-noise ratio of the SNR_{mean} model, and the maximum value of all the contrast model SNR_{max} of the SNR_{max} . In this formula, GQ is the generated quality of music.

MIDI Music File Generation

After data preprocessing, this study will train and generate music data based on the combination of Transformer and LSTM model in deep learning, namely ViT-LSTM model, so that it can effectively learn and generate the sequence of note, volume and time information [12].

In the process of training, the ViT-LSTM model generates sequences with certain sound and music characteristics by learning music patterns and structures in the data set. The generated sequence will be converted into the standard MIDI audio file format by Midi-File function and Midi-Track function, which can be widely used in music creation and appreciation. The generated music sequence has specific music features trained based on the model, as shown in Fig. 2.

According to image analysis, this picture shows a data flow chart that describes the conversion process from the data sequence to the MIDI file. The process begins with a “Data Sequence” (data sequence), which is the input to the process. This data sequence

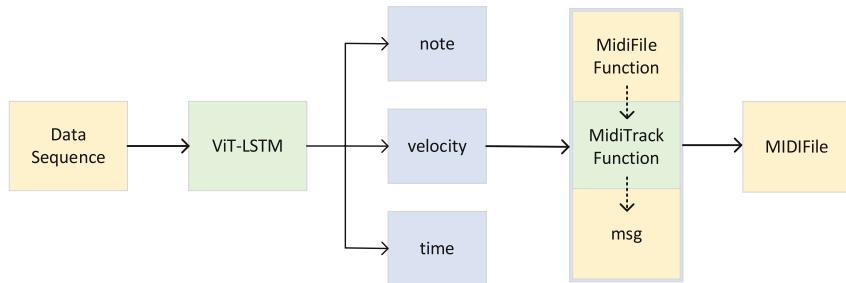


Fig. 2. Conversion of Data Format To MIDI Process Diagram

enters a processing module called “ViT-LSTM”, which may refer to a hybrid neural network structure that combines ViT and LSTM models. From the ViT-LSTM module, the data is divided into three different outputs: note, velocity (speed of sound or intensity) and time. These three outputs are then passed to two different functional modules, one is the “Midi-File Function” and the other is “Midi-Track Function” (MIDI track function), which process the input and eventually generate “msg” (message), indicating further data conversion. Eventually, these processed messages are integrated into the “MIDI-File”, the MIDI file. This means that the output of the entire process is to create a music MIDI file, which can be used to play back music or further processed by music editing software.

Result Analysis

In terms of evaluation indicators, Generation Diversity (GD) is used to evaluate the types of generated music, and Generation Quality (GQ) is used to evaluate the sound quality of generated music, such as Table 2. In the field of music generation, evaluation metrics are extremely limited. However, this limitation does not imply a reduction in the complexity of experiments but may lead to relatively fewer data presentations in research papers.

Table 2. Model Performance Evaluation Form

Algorithm	GD	GQ
Vit-LSTM	0.8234	0.7231
LSTM	0.6746	0.5546
VAE	0.7983	0.6215
Transformer	0.8046	0.7199
Vision Transformer	0.8012	0.7033

The ViT-LSTM model performs best in Generation Diversity, as shown in Fig. 2. Its high score indicates that the generated music sequences have low similarity, which shows the color table of the model in generating music diversity. In addition, the ViT-LSTM model also has a higher score on the Generation Quality index, as shown in Fig. 3, which

indicates that the music sequence generated by the model has higher sound quality and less noise.

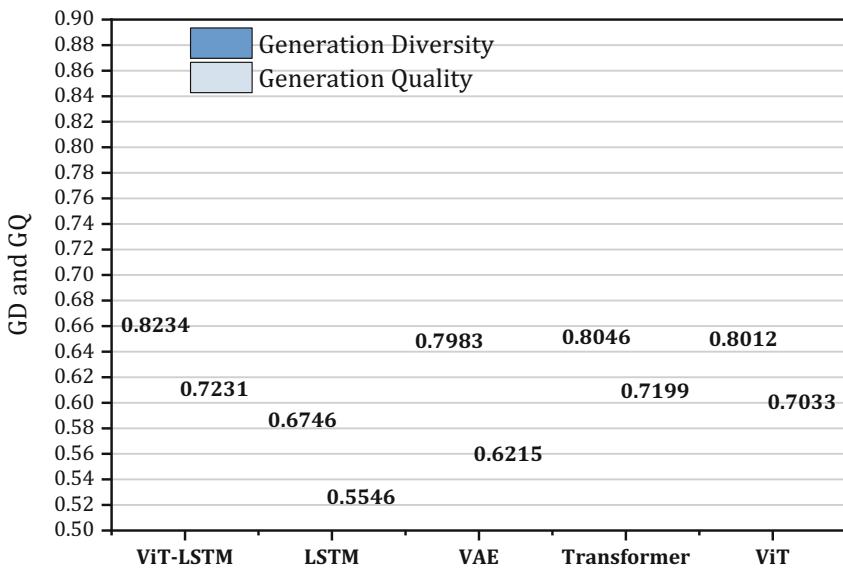


Fig. 3. Comparison Of Model Generation Quality And Music Diversity

5 Conclusion

In this study, we validated the immense potential of deep learning models in the field of music generation, particularly focusing on the outstanding performance of the ViT-LSTM model in capturing musical sequence information. By utilizing the MAESTRO dataset for music generation, we obtained music compositions with rich rhythms and comprehensively evaluated the music generation capability of the ViT-LSTM model from various perspectives such as Generation Diversity and Generation Quality. Compared to common VAE, LSTM, and Transformer music generation models, ViT-LSTM exhibited unique advantages in overall performance. Additionally, a comparison of ViT-LSTM's composition components indicated the effectiveness of our approach. While this study made significant progress, different types of MIDI data may require diverse encoding methods. Future work will concentrate on designing a ViT-LSTM model suitable for various MIDI music data to enhance its applicability and performance.

Acknowledgments. This work was supported by Fundamental Research Funds for the Central Universities, University of International Relations (3262024T01, 3262024T25), and the Teaching Reform and Innovation Project, University of International Relations (2023030, 2023029).

References

1. Alles, H.: Music synthesis using real time digital techniques. Proc. IEEE **68**(4), 436–449 (1980)
2. Schwanauer, S., Levitt, D.: A method for composing simple traditional music by computer. In: Machine Models of Music, pp. 243–262. MIT Press (1993)
3. Miller, A.I.: 17 Project magenta: AI creates its own music. In: The Artist in the Machine: The World of AI-Powered Creativity, pp.137–144. MIT Press (2019)
4. Meng, C., Chenglong, W., Qingtian, Z., Haizhong, Z.: Research and application of music digitalization based on matlab. In: International Conference on Advanced Design and Manufacturing Engineering (2013)
5. He, H., Yuan, M., Liu, X.: Research on surface defect detection method of metal workpiece based on machine learning. IEEE (2021)
6. Sajad, S., Dharshika, S., Meleet, M.: Music generation for novices using recurrent neural network (rnn). In: 2021 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES), pp. 1–6 (2021)
7. Remesh, A., Sinith, M.S.: Symbolic domain music generation system based on LSTM architecture. In: 2022 Second International Conference on Next Generation Intelligent Systems (ICNGIS), pp. 1–4. IEEE (2022)
8. Shi, L., Li, C., Tian, L.: Music genre classification based on chroma features and deep learning. In: 2019 Tenth International Conference on Intelligent Control and Information Processing (ICICIP), pp. 81–86 (2019)
9. <http://abcnotation.com/blog/2010/01/31/how-to-understand-abc-the-basics/>
10. <https://soundcloud.com/sarthak-agarwal/sets/ai-music>
11. Jiang, J., Xia, G.G., Carlton, D.B., Anderson, C.N., Miyakawa, R.H.: Transformer vae: A hierarchical model for structure-aware and interpretable music representation learning. In: ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 516–520 (2020)
12. Mou, L., et al.: Memomusic 3.0: Considering context at music recommendation and combining music theory at music generation. In: 2023 IEEE International Conference on Multimedia and Expo Workshops (ICMEW), pp. 296–301 (2023)
13. Cuenat, S., Couturier, R.: Convolutional neural network (cnn) vs vision transformer (vit) for digital holography. In: 2022 2nd International Conference on Computer, Control and Robotics (ICCCR), pp. 235–240 (2022)
14. Xenakis, I., Kanach, S.: Formalized music: Thought and mathematics in composition (1971)
15. Sinith, M., Murthy, K.: Automatic music composition based on hmm and identified wavelets in musical instruments. In: 2011 International Conference on Signal Processing, Communication, Computing and Networking Technologies, pp. 163–166 (2011)
16. van den Oord, A., et al.: Wavenet: A generative model for raw audio, ArXiv, vol. abs/1609.03499 (2016)
17. Mohanty, R., Dubey, P.P., Sandhan, T.: Temporally conditioning of generative adversarial networks with lstm for music generation. In: 2023 10th International Conference on Signal Processing and Integrated Networks (SPIN), pp. 526–530 (2023)
18. Agarwala, N., Inoue, Y., Sly, A.: Music composition using recurrent neural networks (2017)
19. Sturm, B.L., Santos, J.F., Korshunova, I.: Folk music style modelling by recurrent neural networks with long short term memory units (2015)
20. Zeng, W., Han, D., Cui, M., Wu, Z., Han, B., Zhou, H.: Iflv: Wireless network intrusion detection model integrating fcn, lstm, and vit. In: 2023 IEEE 10th International Conference on Cyber Security and Cloud Computing (CSCloud)/2023 IEEE 9th International Conference on Edge Computing and Scalable Cloud (EdgeCom), pp. 470–475 (2023)

21. Ginanjar, R., Iskandar, I.: Midi conversion to musical notation. In: 2011 First International Conference on Informatics and Computational Intelligence, pp. 97–99 (2011)
22. Liu, M., et al.: Hybrid spatiotemporal graph convolutional network for detecting landscape pattern evolution from long-term remote sensing images. *IEEE Trans. Geosci. Remote Sens.* **60**, 1–16 (2022)
23. Lu, I., Ying, B., Che, X., Jin, Z., Wang, M., Wu, S.: Ddos attack detection method based on one-hot coding and improved resnet18. In: 2023 4th International Seminar on Artificial Intelligence, Networking and Information Technology (AINIT), pp. 196–199 (2023)
24. He, C., Huang, Y., Wang, C., Wang, N.: Dynamic data partitioning strategy based on heterogeneous flink cluster. In: 2022 5th International Conference on Artificial Intelligence and Big Data (ICAIBD), pp. 355–360 (2022)
25. Wang, M., Xiao, Y., Zheng, W., Jiao, X.: Rndm: a random walk method for music recommendation by considering novelty, diversity, and mainstream. In: 2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI), pp. 177–183 (2018)



Graph Convolution Recommendation Algorithm Integrating Multi-relationship Preferences

Jing Su, Tianfeng Zhao, Jianghong Wu^(✉), and Peixuan Shi

School of Artificial Intelligence,
Tianjin University of Science and Technology, Tianjin 300457, China
wujh@tust.edu.cn

Abstract. With the continuous development of Internet technology, existing data forms are becoming more and more complex. However, traditional recommendation algorithms have limitations when processing complex structured data. This paper improves the message propagation layer by considering user social relationships and item similarity relationships. Firstly, An item similarity calculation method that combines user activity is proposed, which combines social relationships and improved item similarity information with interaction information to represent the embedding of users and items. Secondly, the attention mechanism is integrated in the process of aggregating information to make the model more interpretable. Finally, high-order transfer of information is achieved by stacking graph convolutional layers. And perform inner product operation on the user and item representations obtained in the propagation layer to achieve prediction. Experimental verification was conducted on the Lastfm data set. The experimental results show that the performance of this method is the best, proving the effectiveness of the method.

Keywords: Recommendation algorithm · Graph convolutional neural network · Social relationship · Item similarity · Attention mechanism

1 Introduction

Recommendation algorithm can help users filter out redundant information from numerous options and achieve personalized recommendations. From the earliest applied collaborative filtering algorithm [1] to the recommendation framework based on deep learning, technical research is constantly developing. Ahmadian et al. [1] used deep neural networks to model trust relationships and label information between users. Among them, the algorithm evolved from Graph Neural Network (GNN) uses graph embedding technology [3, 4], which has certain advantages in processing graph-structured data and can better represent features. Kipf et al. [5] proposed applying convolutional neural networks to graph data in 2016, and proposed graph convolutional neural networks (GCN). It can model the relationship between users and items, obtain vector representation of high-order information through convolution iteration, and achieve good results. However, as the number of graph convolution layers increases, over-smoothing problems will occur. Chen et al. [6] proposed a graph convolution model of residual network for recommendation to alleviate this problem.

Auxiliary information such as social relationships between users and item attribute information are also helpful in building preference features. Researchers combine GNN with social networks. In traditional social recommendations, matrix decomposition [7], collaborative filtering and other methods are used to mine user features from interactive information and social networks to implement recommendations. Wu et al. [8] proposed the DiffNet algorithm, which uses graph neural networks to aggregate high-order neighbor information of target nodes in the social graph. The graph attention network uses the attention mechanism to distinguish the preferences of target nodes and neighbor nodes for recommendation, reducing the impact of popularity bias. Mu et al. [9] used graph attention network to obtain feature vectors, and then obtained prediction scores through neural collaborative filtering. Alleviating data sparseness through contrastive learning is also a commonly used training method in machine learning. Inspired by contrastive learning, Zou et al. [10] proposed a multi-layer cross-view contrastive learning mechanism.

Based on the research on the above recommendation algorithm, the main contents of this article are as follows: Propose an item similarity construction method that integrates user activity levels. When embedding propagation, user social information and improved item similarity information are integrated to represent features. The attention mechanism is then combined to make the model more interpretable. Finally, high-order information in the graph structure is obtained by stacking graph convolution layers, and prediction scores are obtained to implement recommendations.

2 Graph Convolution Recommendation Algorithm Integrating Multi-relationship Preferences

This paper proposes a recommendation algorithm that integrates multi-relationship preferences based on graph convolutional neural network. In the graph convolution layer, user social information and similarity multi-relationship information between items are integrated by considering. Refining the vector embedding to more accurately represent user and item features.

Graph embedding technology provides powerful tools that can map complex graph structure data to low-dimensional space to facilitate various graph analysis tasks. After data conversion, it can usually be used as input to a machine learning model and then applied to recommendation tasks. It can be divided into multiple categories and is widely used in highly scalable recommendation fields. Through graph embedding technology, the structure and relationships between graph nodes can be captured to better perform graph representation learning. This paper uses the neural network graph embedding method to embed graph data into vector space, and obtain initial embeddings e_u^* and e_i^* of user u and item i respectively.

2.1 User Feature Embedding Propagation

The user features propagation graph is shown in the Fig. 1. In user-item interaction networks, embedding propagation refines users' embeddings by aggregating the embeddings of interactive items.

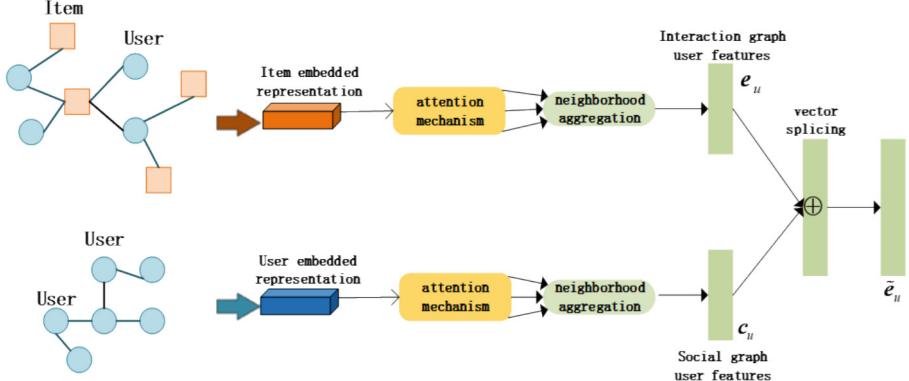


Fig. 1. User features propagation graph

Based on the principle of graph attention network, the attention scores of user u and neighbor i in the interaction graph are calculated as follows:

$$p_{ui} = \text{LeakyRelu}(e_u \parallel e_i) \quad (1)$$

where \parallel represents the splicing of vectors. The obtained attention scores are weighted and normalized to obtain the weight of each neighbor node. Calculated as follows:

$$\alpha_{ui} = \frac{\exp(p_{ui})}{\sum_{w \in R(u)} \exp(p_{uw})} \quad (2)$$

where $R(u)$ is the neighbor set of node u . After that, combined with the weight of neighbor nodes, The first-order aggregation of users is obtained as:

$$e_u = \sum_{i \in H_u} \alpha_{ui} e_i^* \quad (3)$$

where H_u represents the user's first-order neighbor set. In the user's social network, the user's embedding is refined by aggregating friends. The same principle as Formula 2 is used to calculate the weight α_{uv} of neighborhood friend node v to target node u , and after aggregation, the first-order representation of the user in the social graph is obtained as:

$$c_u = \sum_{v \in H_u} \alpha_{uv} e_v^* \quad (4)$$

For any user u , the embedding representation of k th layer is used for aggregation to obtain the embedding representation of $k + 1$ th layer. The $k + 1$ th layer embedding representation of users and items is obtained as:

$$e_u^{k+1} = \sum_{i \in H_u} \frac{1}{\sqrt{|H_u|} \sqrt{|H_i|}} \alpha_{ui} e_i^k \quad (5)$$

$$c_u^{k+1} = \sum_{v \in H_u} \frac{1}{\sqrt{|H_u|} \sqrt{|H_v|}} \alpha_{uv} c_v^k \quad (6)$$

where H_u represents the first-order neighbor set of user u ; H_v represents the first-order neighbor set of user v . The embeddings obtained from each propagation layer are weighted and fused to obtain the final user embedding. The fusion process is:

$$\tilde{e}_u^k = \sigma(\Gamma(e_u^k, c_u^k)) \quad (7)$$

$$\tilde{e}_u = \sum_{k=0}^K \alpha_k \tilde{e}_u^k \quad (8)$$

where \tilde{e}_u represents the final user embedding representation; Γ represents vector splicing operation; $\sigma()$ represents the row regularization operation; K represents the number of layers; α_k is the weight of k th layer when aggregating user embeddings.

2.2 Item Feature Embedding Propagation

The item features propagation graph is shown in the Fig. 2:

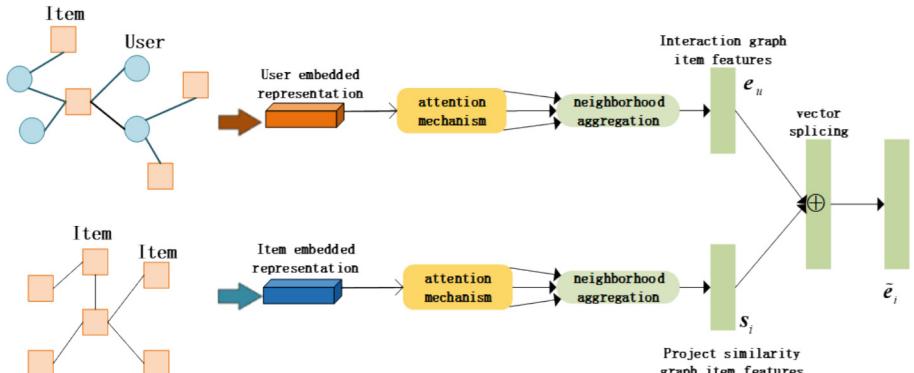


Fig. 2. Item features propagation graph

This paper proposes an item similarity construction method that integrates user activity. Calculate item similarity by combining user activity score and Jaccard coefficient. The user activity score is calculated as follows:

$$f(u) = \frac{1}{\sqrt{1 + |T(u)|}} \quad (9)$$

where $|T(u)|$ is the total number of items that user u has interacted. Combined with the Jaccard coefficient, the item similarity is calculated and the item similarity network is constructed accordingly.

$$J'_{i,j} = \frac{\sum_{p \in R(i) \cap R(j)} f(p)}{\sum_{q \in R(i) \cup R(j)} f(q)} \quad (10)$$

where $J_{i,j}$ represents the Jaccard coefficient between items; $R(i)$ and $R(j)$ represent the set of all users who have interacted with items i and j respectively. $f(p)$ and $f(q)$ are the activity weights of user p and user q respectively. When the coefficient is $J_{i,j} > 0.5$, it is considered that there is a similar relationship. Similar to user feature propagation, the weights α_{iu} and α_{ij} of the domain nodes of item node i in the interaction graph and item similarity graph are calculated respectively. Then combined with the attention weight, the first-order embedding representations e_i and s_i of the item in the interaction graph and item similarity graph are obtained.

$$e_i = \sum_{u \in H_i} \alpha_{iu} e_u^* \quad (11)$$

$$s_i = \sum_{j \in H_i} \alpha_{ij} s_j^* \quad (12)$$

For any item i , use the embedding representation of the k th layer to aggregate to obtain the embedding representation of the $k+1$ th layer. The $k+1$ th layer embedding representation of item i in the interaction graph and item similarity graph is obtained as:

$$e_i^{k+1} = \sum_{u \in H_i} \frac{1}{\sqrt{|H_i|} \sqrt{|H_u|}} \alpha_{iu} e_u^k \quad (13)$$

$$s_i^{k+1} = \sum_{j \in H_i} \frac{1}{\sqrt{|H_i|} \sqrt{|H_j|}} \alpha_{ij} s_j^k \quad (14)$$

where H_i represents the first-order neighbor set of item i ; H_j represents the first-order neighbor set of item j . The final item embedding is obtained by weighted fusion of the embeddings obtained from each propagation layer. The fusion process is:

$$\tilde{e}_i^k = \sigma(\Gamma(e_i^k, s_i^k)) \quad (15)$$

$$\tilde{e}_i = \sum_{k=0}^K \alpha_k \tilde{e}_i^k \quad (16)$$

where \tilde{e}_i represents the final item embedding representation; Γ represents vector splicing operation; $\sigma()$ represents the row regularization operation; K represents the number of layers; α_k is the weight of k th layer when aggregating item embeddings.

2.3 Predict

The predicted score y_{ui} is obtained in the form of inner product, and the calculation method is as follows:

$$y_{ui} = \tilde{e}_u^T \cdot \tilde{e}_i \quad (17)$$

where \hat{y}_{ui} represents the score of the predicted score, and \mathbf{T} represents the vector transpose.

2.4 Model Optimization

The model mainly uses Bayesian loss function and contrastive learning loss function for adjustment. The contrastive learning task compares the embedding representation of the intermediate layer obtained through different layers of propagation with the final embedding representation. The specific calculation formula is as follows:

$$L = L_{BPR} + \eta L_{cl} + \lambda ||\Theta||^2 \quad (18)$$

where L_{BPR} is the Bayesian loss function; L_{cl} represents the total contrast loss; η represents the weight of the comparative learning loss function; Θ represents the training data; $\lambda ||\Theta||^2$ represents the regularization term, λ is the regularization coefficient.

3 Experimental Design and Result Analysis

3.1 Experimental Data Set

The experiment was completed based on the open source data set in the recommended scenario, and the public data set Lastfm was selected for the experiment. The data set is collected from user activities of the online music service platform LastFM. It mainly contains data such as interactive relationships between 1892 users and 17632 projects, as well as social relationships between users. Select 80% of them as the training set and the remaining 20% as the test set.

3.2 Evaluation Indicators

The core of the recommendation algorithm is to predict the possible interactions of the user. The accuracy of the algorithm usually refers to the ability to predict user interests or preferences in the recommendation system. The evaluation of model performance uses common indicators Normalized Discounted Cumulative Gain (NDCG), Recall, and Precision to evaluate the recommendation performance of the model. Relevant introduction is as follows:

NDCG is an indicator used to evaluate the ranking quality of recommendation systems. It takes into account the actual relevance and ranking order of each item. The value of NDCG ranges from 0 to 1. The closer the value is to 1, the better the ranking quality of the recommendation list. i indicates the position number in the recommendation list; $r(i) = 1$ indicates that the recommended item has an interactive relationship with the target user; $|REL|$ indicates the number of the top N recommended items sorted by $r(i)$.

$$NDCG = \frac{\sum_{i=1}^N \frac{r(i)}{\log_2(i+1)}}{\sum_{i=2}^{|REL|} \frac{r(i)}{\log_2 i}} \quad (19)$$

Recall is the proportion of correctly predicted samples among all real positive samples. TP represents the predicted correct positive Number of samples; FN represents the number of incorrectly predicted negative samples.

$$\text{Recall} = \frac{TP}{TP + FN} \quad (20)$$

Precision is the proportion of samples that are correctly predicted to be positive to the total number of samples that are predicted to be positive. FP represents the number of incorrectly predicted negative samples.

$$\text{Precision} = \frac{TP}{TP + FP} \quad (21)$$

3.3 Experimental Settings

This experiment was completed under the Windows operating system, the language used was Python, and the framework used was PyTorch1.7.0. Set the embedding vector dimension of the model to 64, use the Adam method for optimization, set the default learning rate to 0.001, and the amount of data processed each time is 1024. The importance of each layer is set to the same, the number of layers K is set to 3, η set to 0.15, and the regularization coefficient is set to $1e^{-4}$. The comparison models selected for the experiment are as follows:

SBPR [11]: Social recommendation method based on matrix factorization.

NGCF [12]: Based on the idea of graph convolution network, the user-item interaction information is expanded into vector embedding.

LightGCN [13]: By reducing the amount of parameters and computational complexity while maintaining high recommendation performance.

MHCN [14]: Social recommendation method based on hypergraph convolutional network, which models the correlation between users with hyperedges.

SocialLGN [15]: Propagate social graph information and interaction graph information in the LightGCN convolutional layer at the same time.

3.4 Experimental Results

Result Analysis

This paper conducts a large number of experiments on the data set and compares it with other baseline methods, showing the values of three evaluation indicators in Top-10 and Top-20 recommendations. The experimental results are shown in Table 1. The SBPR model based on traditional matrix decomposition has the worst recommendation effect. The main reason is that the model in this paper uses graph embedding technology to map user, project and other information into vector space in the embedding layer, which effectively alleviates the negative impact of data sparsity on the model. The NGCF model and LightGCN model based on graph neural networks have also improved compared to SBPR. The algorithm processes of the two models are similar, but the simplified LightGCN model has better performance. Both the SocialLGN model and

the MHCN model introduce auxiliary information of social relationships. Comparative experimental results show that it is helpful to improve recommendation accuracy. The model in this paper uses social relationships and the improved Jaccard coefficient to map the item similarity relationship into the embedding propagation, enriching the training data. In addition, the reference of the attention network enables the aggregation of information to rely on the feature expression between nodes and is independent of the graph structure, capturing more representative neighbor influences. In summary, the model in this paper effectively improves the recommendation effect of the algorithm and has better performance than other models.

Table 1. Performance comparison with other models

Model Name	NDCG (TOP-10)	Recall (TOP-10)	Precision (TOP-10)	NDCG (TOP-20)	Recall (TOP-20)	Precision (TOP-20)
SBPR	0.1992	0.1592	0.1556	0.2034	0.2083	0.1056
NGCF	0.2297	0.1786	0.1758	0.2557	0.2572	0.1265
LightGCN	0.2523	0.1953	0.1945	0.2774	0.2759	0.1355
MHCN	0.2545	0.1998	0.1958	0.2798	0.2786	0.1367
SocialLGN	0.2562	0.2019	0.1980	0.2830	0.2802	0.1381
This paper	0.2607	0.2085	0.2021	0.2895	0.2860	0.1407

3.5 Ablation Analysis

Study the impact of key modules on recommendation performance and the effectiveness of each module. Several variants are obtained by deleting some modules: a. Remove the attention mechanism during aggregation; b. Remove the similarity relationship between items; c. Remove the layer comparison learning loss part. Using Top-10 recommendations, the results in the data set are shown in Fig. 3. It can be observed that the recommendation performance after removing several modules has decreased, which verifies that these parts can play a positive role in improving recommendation performance.

Influence of Regularization Coefficient

The regularization coefficient is used to balance the relationship between the regularization term and the loss function. Top-10 recommendations are used, and the comparison results are shown in Fig. 4. Performance is best when the value is $1e^{-4}$. But over-regularization can have the opposite effect.

The Influence of the Number of Graph Convolution Layers

Determine the best settings by changing the number of convolutional layers. Using Top-10 recommendations, the experimental results are shown in Fig. 5. If there are too many layers, over-fitting will occur, the expressions between nodes will be over-smoothed, and there will be a lack of differentiation between nodes. The final number of layers is set to 3 layers.

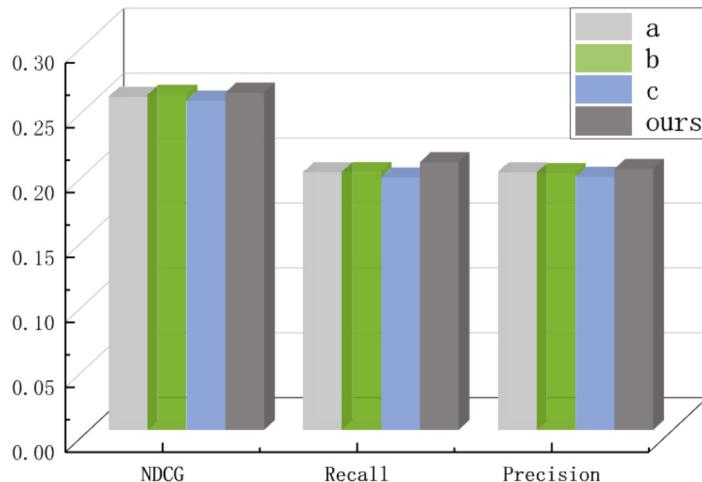


Fig. 3. Experimental comparison of model settings without some modules

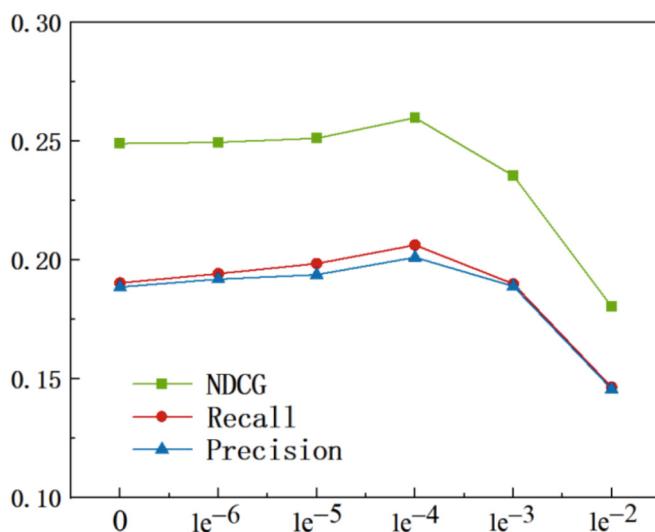


Fig. 4. Comparison of regularization coefficient settings

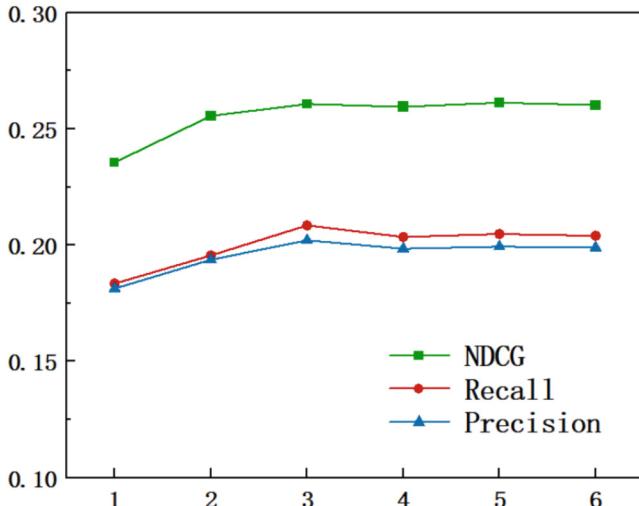


Fig. 5. Comparison of different convolution layer settings

4 Conclusion

This paper proposes a recommendation algorithm based on graph convolutional neural network. In order to better extract information features, an improved Jaccard coefficient method is proposed to construct the item similarity network. By integrating user social information and item similarity information to learn the embedding representation of users and items to alleviate the problem of data sparseness. The attention mechanism is introduced in the aggregation process to improve feature expression capabilities. Finally, the model is jointly trained using contrastive learning. The results show that compared with existing methods, our method achieves better performance in the data set.

There are still some shortcomings in the research of this article. When processing graph structure information, the interactive relationship between users and projects is processed indiscriminately. In real life, operations on music software, including playback, collection, adding to playlists, etc., all have different degrees of reliability. In future work, considering these different relationships can lead to more reasonable feature extraction, which can further improve the accuracy of recommendations.

Acknowledgments. The authors are sincerely grateful to the anonymous reviewers for their insightful comments and suggestions, which greatly improve this paper. This work was financially supported by the National Natural Science Fund(Grant No. 62377036) and the Tianjin Science and Technology Plan Project(Grant No. 22KPXMRC00210).

References

1. Ekstrand, M.D.: Collaborative filtering recommender systems. Found. Trends® Hum.-Comput. Interact. **4**(2), 175–243 (2011)
2. Ahmadian, S., Ahmadian, M., Jalili, M.: A deep learning based trust-and tag-aware recommender system. Neurocomputing **488**, 557–571 (2021)
3. Mikolov, T., Chen, K., Corrado, G., et al.: Efficient estimation of word representations in vector space. arXiv preprint arXiv, 1301.3781 (2013). <https://doi.org/10.48550/arXiv.1301.3781>
4. Grover, A., Leskovec, J.: Node2vec: scalable feature learning for networks. In: 22nd SIGKDD 2016, pp. 855–864. ACM, San Francisco (2016)
5. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv, 1609.02907 (2016). <https://doi.org/10.48550/arXiv.1609.02907>
6. Velickovic, P., Cucurull, G., Casanova, A., et al.: Graph attention networks. In: 6th ICLR 2018, pp. 1–12. Vancouver, Open Review (2018)
7. Cheng, H.T., Koc, L., Harmsen, J., et al.: Wide & deep learning for recommender systems. In: 1st Workshop on Deep Learning for Recommender Systems 2016, pp. 7–10. ACM, Boston (2016)
8. Wu, L., Sun, P., Fu, Y., et al.: A neural influence diffusion model for social recommendation. In: 42nd SIGIR 2019, pp. 235–244. ACM, Paris (2019)
9. Mu, N., Zha, D., He, Y., et al.: Graph attention networks for neural social recommendation. In: 31st IEEE 2019, pp. 1320–1327. IEEE, Portland (2019)
10. Zou, D., Wei, W., Mao, X.L., et al.: Multi-level cross-view contrastive learning for knowledge-aware recommender system. In: 45th SIGIR 2022, pp. 1358–1368. ACM, Madrid (2022)
11. Zhao, T., Mcauley, J.L., King, I.: Leveraging social connections to improve personalized ranking for collaborative filtering. In: 23rd CIKM 2014, pp. 261–270. ACM, Shanghai (2014)
12. Wang, X., He, X.N., Wang, M., et al.: Neural graph collaborative filtering. In: 42nd SIGIR P2019, pp. 156–174. ACM, Paris (2019)
13. He, X.N., Deng, K., Wang, X., et al.: LightGCN: simplifying and powering graph convolution network for recommendation. In: 43rd SIGIR 2020, pp. 639–648. ACM, New York (2020)
14. Yu, J.L., Yin, H.Z., Li, J.D., et al.: Self-supervised multi-channel hypergraph convolutional network for social recommendation. In: Web Conference (WWW) 2021, pp. 413–424. ACM, Ljubljana (2021)
15. Liao, J., Zhou, W., Luo, F.J., et al.: SocialLGN: light graph convolution network for social recommendation. Inf. Sci. **589**, 595–607 (2022)



Temporal Sequential Wave Neural Network for Solving the Optimal Cognitive Subgraph Query Problem

Jiaqian Bi^{1(✉)}, Zhilei Xu², and Qing Yu¹

¹ School of Computer Science and Engineer, Tianjin University of Technology, Tianjin 300382, China

jiaqianbi@163.com

² School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China

Abstract. The Optimal Cognitive Subgraph query problem (MCSQR) aims to identify the subgraph structure that achieves the highest global recognition in a static social network. In this paper, we propose the Temporal Wave Neural Network (TSWNN) framework, which, differs from traditional neural networks, does not require any training. All neurons in TSWNN are computed in parallel and consist of input, wave receiver, neuron state memory, wave generator, wave transmitter, and output. The fundamental idea behind TSWNN is based on parallel wave transmission where each neuron can receive waves sent by all precursor neurons before activation occurs. Each neuron transmits its path and knowledge to its successor through waves while objective neurons calculate final recognition based on received waves and output optimal solutions. Evaluation using four public datasets shows that TSWNN outperforms A*, Dijkstra, Label, and TDNN.

Keywords: Temporal Sequential Wave Neural Network · Optimal Cognitive Subgraph · Subgraph Query

1 Introduction

Subgraph querying is crucial in graph data processing, involving the search for isomorphic subgraphs within a larger data graph [1]. It is widely used in areas such as social networks [2], knowledge graphs [3] and protein networks [4]. Traditional approaches often overlook dynamic information propagation and node perception, focusing solely on topology matching. To address this, we propose the concept of optimal cognitive subgraph, which considers both graph topology and dynamic information propagation, aiming to maximize overall cognition within a specific period.

Optimal cognitive subgraph research, rooted in traditional graph topology, introduces the concept of node cognitive information degrees, and explores information propagation path diversity. However, in the current subgraph query field, researchers have studied the problem of subgraph isomorphism [5], and the early subgraph query research mainly focuses on the theoretical basis of graph theory and focuses on the subgraph isomorphism problem in graph structure. The Ullmann algorithm [6], VF2 [7], VF2++ [8], VF3,

VF3-Light [9], and various join optimization techniques have been proposed, such as k-GPM, D-Join [10], CFL [11]. In addition, various approximation algorithms have been proposed to solve the subgraph matching problem, including G-Finder [12], Saga [13], and PG-N [14]. However, these algorithms mainly focus on the matching of graph structure, ignoring the information interaction and cognitive differences between nodes. Huang et al.'s [15] first used neural networks to solve the time-varying shortest path problem. The optimal cognitive subgraph query problem and the time-varying shortest path problem are like some extent, both involving the problem of searching for paths or subgraphs that satisfy specific conditions in a network. Hence, neural networks are introduced into optimal cognitive subgraph queries.

The rest of the paper is organized as follows. Section 2 presents the definition of the problem in question, Sect. 3 proposes the temporal sequential wave neural network framework and related algorithms, Sect. 4 gives numerical examples, Sect. 5 conducts comparative experiments, and Sect. 6 gives concluding remarks.

2 Definition of the Problem

In this section, we will give definitions of the relevant maximal cognitive subgraph models. Some of the notations used in this study are summarized in Table 1.

Table 1. Explanation of symbols

Symbols	Explanation
G	<i>inputGraph</i>
V	<i>setofnodes</i>
E	<i>setofedges</i>
T	<i>giventime</i>
t	<i>currenttime</i>
v_s	<i>startnode</i>
v_d	<i>targetnode</i>
V_i^S	set of predecessor nodes of node i
V_i^P	set of successor nodes of node i
Wi	<i>setofwaverceivedbynodi</i>
w_{ij}	<i>wavefromnodeitonodej</i>
y_i	<i>whethernodeiisactivatedornot</i>
c_i	<i>cognitivedegreeofnodei</i>
c_{ij}	<i>awarenessofwavefromnodeitonodej</i>
l_{ij}	<i>transmissiontimeofwavefromnodeitonodej</i>
P_{ij}	<i>pathfromnodeitonodej</i>
y_{ij}	<i>awarenessofnodeitonodejwave</i>
Y_i	<i>activationtimeofnodei</i>

Definition 1 (Node Optimal Cognition). Let G be a static network, v_i be a node of G , w_{ij} be the wave from v_i to v_j , c_{ij} be the weight of the wave w_{ij} , $C_j(i)$ be the cognitive degree that v_j receives from v_i , and $C_j(t)$ be the cognitive degree of v_j at the moment t . When a node receives a wave from a source, it updates its cognitive level for that source. If the wave has been received from this source before, the cognitive degree of the newly received wave and the previously stored cognitive degree are compared, and the larger value is taken. The cognition of v_j is given by:

$$C_j(i) = \max(C_j(i), c_{ij} * C_i(t - l_{ij})) \quad (1)$$

$$C_j(t) = \sum_{i \in N} C_j(i) \quad (2)$$

Definition 2 (Optimal Cognitive Subgraph). In a static network G , where v_s is the starting node and v_d is the destination node, and with a finite time T , $P_{sd} = < v_s, \dots, v_d >$ represents the paths from v_s to v_d within T . The final cognition of the destination node is the sum of the cognitions along all the paths leading to the destination node. The optimal cognitive subgraph is defined by the paths $P_{sd} = < v_s, \dots, v_d >$ and the network formed by the final cognition of each node.

3 Temporal Sequential Wave Neural Network

In this section, we outline the TSWNN structure design, algorithm steps, and provides time complexity.

3.1 Design of Temporal Sequential Wave Neural Network

TSWNN is a new type of neural network composed of neurons that does not require any training. For simplicity, a social network can be viewed as a temporal wave neural network, where each user corresponds to a neuron and each arc corresponds to a connection between two users. Neuron in a neural network communicate with each other through waves. Each wave contains three parts of information. Once the wave is accepted by the neuron, the life cycle of the wave is over.

Definition 3 (Temporal Wave). Let G be a random network, v_s, v_i, v_j are the nodes of G , (v_i, v_j) is the starting node for an arc from v_i to v_j . A wave from v_i to v_j is defined as w_{ij} , a temporal wave w_{ij} that is generated when v_i is activated and sent to the succeeding node v_j . The timing wave w_{ij} can be written as $\{c_{ij}, l_{ij}, (v_i, v_j)\}$, c_{ij} representing the cognitive degree of the wave, l_{ij} representing the transmission time of the wave, (v_i, v_j) representing the wave being sent from v_i to v_j .

When and only when a neuron receives a wave for the first time, that neuron is activated, and the neuron can receive temporal waves from different precursor neurons multiple times. Where the start neuron is activated and if the destination neuron is activated, a maximal cognitive subgraph from the start to the end point is obtained. Figure 1 depicts the general structure of a temporal wave neuron, comprising six parts: input, wave receiver, neuron state memory, wave generator, wave transmitter and output.

- 1) Input: Receive waves from precursor neurons.
- 2) Wave receiver: The wave receiver can get the cognitive degree c_i of the precursor neuron from the input, the transmission time w_{ij} of the wave, the cognitive degree y_{ij} of the wave, and the arrival time t_j of the wave, which is expressed as follows:

$$w_{ij} = \begin{cases} l_{ij}, & \text{if the output receives a wave} \\ 0, & \text{if the output not receives a wave} \end{cases} \quad (3)$$

$$y_{ij} = \begin{cases} c_{ij}, & \text{if } w_{ij} \neq 0 \\ 0, & \text{if } w_{ij} = 0 \end{cases} \quad (4)$$

- 3) Neuronal state memory: Storage neuron state and cognition.

$$y_j = \begin{cases} 0, & \text{if } y_j = 0 \text{ and node } v_j \text{ does not receive the wave} \\ 1, & \text{if } v_j \text{ node } v_j \text{ receives the wave} \end{cases} \quad (5)$$

$$C_j(i) = \max(C_j(i), y_{ij} * C_i(t - l_{ij})) \quad (6)$$

$$C_j(t) = \sum_{i \in N} C_j(i) \quad (7)$$

- 4) Wave generator: When a neuron is activated, that neuron generates a temporal wave. The generated wave is transmitted to the inactivated succeeding neuron.

$$w_{jk} = l_{jk} \quad (8)$$

$$y_{jk} = \begin{cases} c_{jk}, & \text{if } w_{jk} \neq 0 \\ 0, & \text{if } w_{jk} = 0 \end{cases} \quad (9)$$

- 5) Wave transmitter: The wave transmitter sends waves to inactivate successor neuron. If the successor neuron is inactive, the wave is sent; if it's active, the wave isn't sent. It can be seen as the inverse process of the wave transmitter.

$$w_{jk} = \begin{cases} 0, & \text{if the output does not send a wave} \\ l_{jk}, & \text{if the output sends a wave} \end{cases} \quad (10)$$

- 6) Output: The output sends waves to succeeding neurons.

3.2 TSWNN Algorithm

The basic idea of temporal sequential wave neural networks is: The waves are transmitted in parallel; each neuron can receive waves sent by all precursor neurons before it is activated; each neuron computes the cognitive degree and transmits the path as well as the cognitive degree to its successor neuron via waves; the destination neuron calculates the final cognition of all received waves and outputs the optimal solution.

TSWNN abstracts the social network as a temporal wave neural network, and first performs the initialization of the network model, including parameter initialization and

neuron initialization. After determining the start and end nodes, the system randomly generates the inter-node transmission time as well as the cognitive level of the nodes, and the start node is set to an activated state and transmits the wave to its inactivated successor node. This process is carried out according to the principle of shortest transmission time and simultaneously passes the wave to other succeeding nodes. When the succeeding nodes receive the wave, they are activated and update their cognitive degrees based on the cognitive degrees of the predecessor nodes and the cognitive degrees of the edges. This process continues until all waves are received at the end point. The awareness of each node will be accumulated based on the information it receives from the wave and reflect the final awareness of the endpoint. Finally, all start-to-finish paths will be merged into an overall network. Algorithms 1–4 describe the TSWNN process.

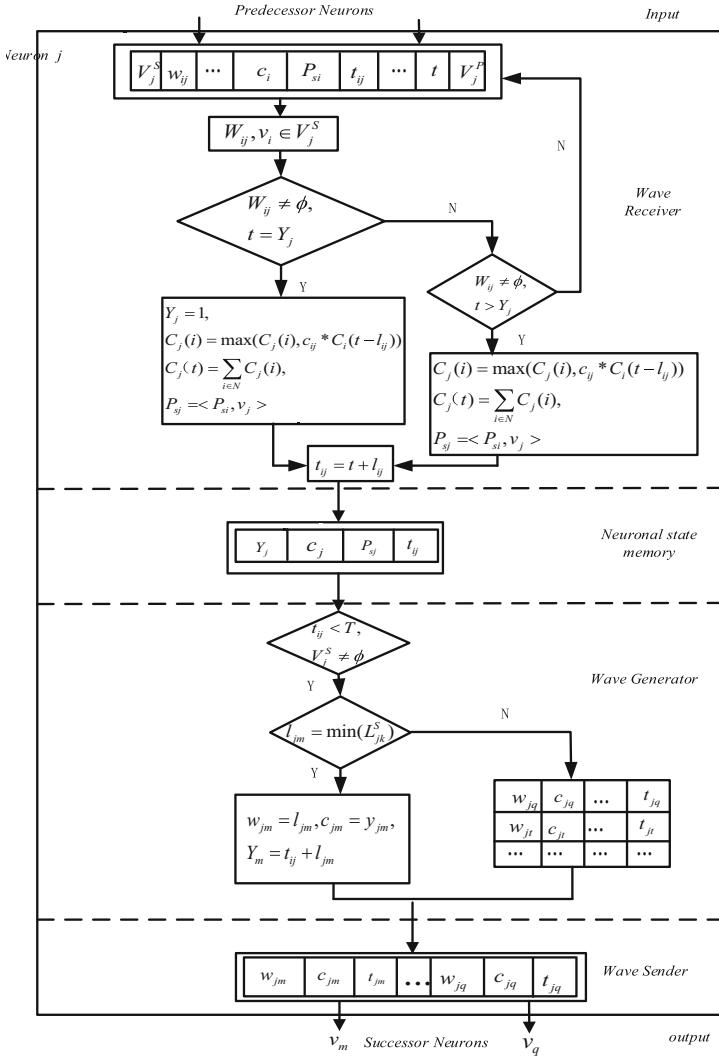


Fig. 1. Structure of temporal sequential wave neuron

Algorithm 1: Temporal Sequential Wave Neural Network

Input: v_s, v_d, t, G , where $G = (V, E, L, C)$.

Output: P_d, M_d .

1. Set $t = 0, P_{sd} = \phi, T = M$.

2. Initialize all neurons with INA algorithm.

3. When ($t < T$)

Update neurons and paths using UNA.

4. Output P_d based on OMC.

Algorithm 2: Initialization Algorithm (NIA)

Input: v_s, v_d, G .

Output: $c_i, V_i^p, V_i^S, y_i, l_{ij}, P_{ij}$

For any $v_i \in V, v_j \in V$

Set $c_s = 1, c_i = 0, t = 0, y_s = 1, y_i = 0$.

Set $V_i^p = \varphi, V_i^S = \varphi, P_{ij} = \varphi$.

For any $v_i \in V, e_{ij} \in E$

Set $\{v_j\} \cup V_i^S$

Algorithm 3: Update Neuron Algorithm (UNA)

Input: $T, G, t, v_s, v_d, y_i, c_i$.

Output: $c_i, V_i^p, V_i^S, y_i, P_{ij}, Y_i, W_i, W'_i, v_i \in V, v_j \in V$,

1. If $v_i = v_s, t < T, y_j = 0, v_j \in V_s^S, j = 1, 2, \dots, x$

Set $w_{sj} = l_{sj}, y_{sj} = c_{sj}, w_{sj} \cup W_j, Y_j = l_{sj}$

2. If $t < T, Y_j = t, v_j \in V, w_{ij} \in W_j$

Set $y_i = 1, c_j = c_i * w_{ij} + c_j$.

Set $w_{jm} = l_{jm}, y_{jm} = c_{jm}, w_{jm} \cup W_m, Y_m = l_{jm}$.

Set $P_{ij} = P_i \cup \{v_i, v_j\}$

3. If $t < T, Y_j < t, w_{ij} \in W_j, w_{ij} \notin W'_j$ (When j is activated)

Set $c_j = c_i * w_{ij} + c_j$.

Set $w_{jm} = l_{jm}, y_{jm} = c_{jm}, w_{jm} \cup W_m, Y_m = l_{jm}, W_i = W_i - \{w_{ij}\}$

Set $P_{ij} = P_i \cup \{v_i, v_j\}$

4. If $t < T, Y_j < t, w_{ij} \in W_j, w_{ij} \in W'_j$ (When j is activated)

Set $C_j(i) = \max(C_j(i), c_{ij} * C_i(t - l_{ij}))$

$C_j(t) = \sum_{i \in N} C_j(i)$

Set $w_{jm} = l_{jm}, y_{jm} = c_{jm}, w_{jm} \cup W_m, Y_m = l_{jm}, W_i = W_i - \{w_{ij}\}$

Set $P_{ij} = P_i \cup \{v_i, v_j\}$

If $t > T$

$t++$

Algorithm 4: Output Maximum Cognitive Algorithm (OMC)

```

Input:  $v_s, v_d, G$ 
Output:  $R_d, M_d$ 
1. Set  $R_d = \phi, M_d = 0$ 
2. While  $t=T$ 
    If  $R_d \neq \phi$ 
        Set  $R_d = R_d \cup_{t=0}^T P_d(t) \quad M_d = C_d(t)$ 
        If  $R_d \neq \phi, t --$ 
        Until  $R_d \neq \phi$ 
        Set  $R_d = R_d \cup_{t=0}^T P_d(t) \quad M_d = C_d(t)$ 
Output  $R_d, M_d$ 

```

3.3 Time Complexity Analysis of TSWNN Algorithm

Theorem 1 (Time complexity of TSWNN). Given a network, let n be the number of nodes (neurons), m be the number of arcs (edges), T be the bounded time, and Δt be the update step of the neural network. Then the time complexity of TSWNN is $O(n^2)$.

Proof: The algorithm TSWNN consists of INA, UNA, and OMC.

For the INA algorithm, each neuron necessitates 5 assignment statements per neuron, resulting in a time complexity of $O(5 * n) = O(5n)$. Traversing all edges and adding neighboring nodes sums up to $O(n + m)$.

For the UNA, the time complexity for the starting node, occurring once, is $O(n)$, and for non-initial nodes, it remains $O(n)$ for altering activation state, cognition, and transmission time. In the worst-case scenario, with $n-1$ operations performed to add waves to the wave collection and set activation time, the total time complexity becomes $O(n^2)$. If a node receives the same wave, it needs to compare cognitive degrees to select the optimal wave, with a time complexity of $O(m)$ where m is the number of same waves received. After processing the wave, the node continues to propagate it to subsequent nodes, requiring $O(n^2)$ operations in the worst-case scenario where each node connects to all others. Considering the process needs to be completed in T time, with each stage executed at most once in each time step, and $t++$ required after each execution, the time complexity of one execution is denoted as $O(n + m) + O(n^2) + O(n^2) + O(n^2) + O(m) + O(n^2)$. Thus, the time complexity of UNA is $O(T * (n + n^2 + n^2 + m + n^2))$.

For the OMC algorithm, the time complexity of the algorithm is $O(T)$ when traversing the path information of the end point from the moment T .

Therefore, the overall time complexity of the algorithm is $O(n + m) + O(T * (n + n^2 + n^2 + m + n^2)) + O(T) \leq O(n^2)$. Theorem 1 is proved.

4 Example of TSWNN Algorithm

To validate our TSWNN algorithm, we use an example with a network structure shown in Fig. 2 and Table 2. It has 5 vertices and 7 arcs. Assuming node A is the start and node D is the end, arc lengths and their cognitive degrees are randomly generated. Table 1

outlines all steps for running TSWNN with a time limit of 10. The maximum cognitive subnetwork is $\{<A, B>, <A, D>, <B, D>, <B, E>, <D, E>\}$ with a cognitive degree of 0.548.

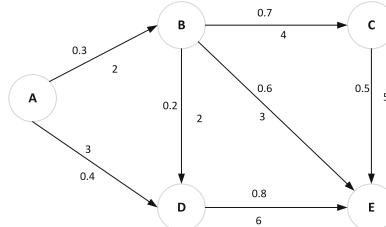


Fig. 2. Example diagram network structure

5 Experimentation

In this section, we assess TSWNN against established algorithms like Dijkstra, A*, Label, and TDNN across four datasets: Facebook, wiki-Vote, hamsterster, and dolphins (Table 3). To demonstrate its effectiveness, we randomly generate root and destination nodes, as well as awareness and transmission times between nodes. We evaluate performance based on both runtime and cognitive metrics.

5.1 Experimental Results with Different Number of Nodes

As shown in Fig. 3, the TSWNN algorithm outputs better solutions than the other four algorithms in terms of running time, which is the shortest in the same dataset and increases for each algorithm as the number of nodes increases. TSWNN's relatively minimal running time increases with more nodes due to its parallel computing strategy, which reduces the impact of network complexity on solution speed.

As shown in Table 4, TSWNN outperformed the other four algorithms in terms of cognitive performance, achieving optimal results compared to the others. As the number of nodes increased, the optimal cognitive subgraph cognition also increased for each algorithm. However, when comparing the four algorithms, TSWNN consistently achieved optimal cognition, while the other three algorithms showed similar results.

5.2 Experimental Results with Different Number of Sides

Figure 4 demonstrate that our proposed TSWNN consistently exhibits the shortest running time on the same dataset, highlighting its efficiency on the problem of querying in large-scale graphs. In contrast, Dijkstra's time complexity is typically $O((V + E) * \log(V))$, making it relatively slow for large-scale graphs. Algorithm A*, which combines heuristic search and Dijkstra, has a similar time complexity. The Label algorithm and TDNN algorithm also have similar time complexities, around $O((V + E) * \log(V))$ and

Table 2. TSWNN process for solving OMC

Neur on	Neuron State	Neuron Cognition	Sequence Wave	Sequence Wave Cognition	Neuronal Pathway
t=0					
A	$y_A = 1$	$c_A = 1$	$w_{AB} = 2$, $w_{AD} = 3$	$c_{AB} = 0.3$, $c_{AD} = 0.4$	A
B	$y_B = 0$	$c_B = 0$	\emptyset	\emptyset	\emptyset
C	$y_C = 0$	$c_C = 0$	\emptyset	\emptyset	\emptyset
D	$y_D = 0$	$c_D = 0$	\emptyset	\emptyset	\emptyset
E	$y_E = 0$	$c_E = 0$	\emptyset	\emptyset	\emptyset
t=2					
A	$y_A = 1$	$c_A = 1$	\emptyset	\emptyset	A
B	$y_B = 1$	$c_B = 0.3$	$w_{BD} = 2$, $w_{BE} = 3$, $w_{BC} = 4$	$c_{BD}=0.2$, $c_{BE} = 0.6$, $c_{BC} = 0.7$	$\langle A,B \rangle$
C	$y_C = 0$	$c_C = 0$	\emptyset	\emptyset	\emptyset
D	$y_D = 0$	$c_D = 0$	\emptyset	\emptyset	\emptyset
E	$y_E = 0$	$c_E = 0$	\emptyset	\emptyset	\emptyset
t=3					
A	$y_A = 1$	$c_A = 1$	\emptyset	\emptyset	A
B	$y_B = 1$	$c_B = 0.3$	\emptyset	\emptyset	$\langle A,B \rangle$
C	$y_C = 0$	$c_C = 0$	\emptyset	\emptyset	\emptyset
D	$y_D = 1$	$c_D = 0.4$	$w_{DE} = 6$	$c_{DE} = 0.8$	$\langle A,D \rangle$
E	$y_E = 0$	$c_E = 0$	\emptyset	\emptyset	\emptyset
t=4					
A	$y_A = 1$	$c_A = 1$	\emptyset	\emptyset	A
B	$y_B = 1$	$c_B = 0.3$	\emptyset	\emptyset	$\langle A,B \rangle$
C	$y_C = 0$	$c_C = 0$	\emptyset	\emptyset	\emptyset
D	$y_D = 1$	$c_D = 0.46$	$w_{DE} = 6$	$c_{DE} = 0.8$	$\langle A,D \rangle \langle A,B \rangle \langle B,D \rangle$
E	$y_E = 0$	$c_E = 0$	\emptyset	\emptyset	\emptyset
t=5					
A	$y_A = 1$	$c_A = 1$	\emptyset	\emptyset	A
B	$y_B = 1$	$c_B = 0.3$	\emptyset	\emptyset	$\langle A,B \rangle$
C	$y_C = 0$	$c_C = 0$	\emptyset	\emptyset	\emptyset
D	$y_D = 1$	$c_D = 0.46$	\emptyset	\emptyset	$\langle A,D \rangle \langle A,B \rangle \langle B,D \rangle$
E	$y_E = 1$	$c_E = 0.18$	\emptyset	\emptyset	$\langle A,B \rangle \langle B,E \rangle$
t=6					
A	$y_A = 1$	$c_A = 1$	\emptyset	\emptyset	A
B	$y_B = 1$	$c_B = 0.3$	\emptyset	\emptyset	$\langle A,B \rangle$
C	$y_C = 1$	$c_C = 0.21$	$w_{CE} = 5$	$c_{CE} = 0.5$	$\langle A,B \rangle \langle B,C \rangle$
D	$y_D = 1$	$c_D = 0.46$	\emptyset	\emptyset	$\langle A,D \rangle \langle A,B \rangle \langle B,D \rangle$
E	$y_E = 1$	$c_E = 0.18$	\emptyset	\emptyset	$\langle A,B \rangle \langle B,E \rangle$
t=9					
A	$y_A = 1$	$c_A = 1$	\emptyset	\emptyset	A
B	$y_B = 1$	$c_B = 0.3$	\emptyset	\emptyset	$\langle A,B \rangle$
C	$y_C = 1$	$c_C = 0.21$	\emptyset	\emptyset	$\langle A,B \rangle \langle B,C \rangle$
D	$y_D = 1$	$c_D = 0.46$	\emptyset	\emptyset	$\langle A,D \rangle \langle A,B \rangle \langle B,D \rangle$
E	$y_E = 1$	$c_E = 0.5$	\emptyset	\emptyset	$\langle A,B \rangle \langle A,D \rangle \langle B,E \rangle \langle D,E \rangle$
t=10					
A	$y_A = 1$	$c_A = 1$	\emptyset	\emptyset	A
B	$y_B = 1$	$c_B = 0.3$	\emptyset	\emptyset	$\langle A,B \rangle$
C	$y_C = 1$	$c_C = 0.21$	\emptyset	\emptyset	\emptyset
D	$y_D = 1$	$c_D = 0.46$	\emptyset	\emptyset	$\langle A,D \rangle \langle A,B \rangle \langle B,D \rangle$
E	$y_E = 1$	$c_E = 0.548$	\emptyset	\emptyset	$\langle A,B \rangle \langle A,D \rangle \langle B,E \rangle \langle D,E \rangle$

Table 3. Data Sets

Data Set	Node	Edge
dolphins	62	159
Facebook	333	5038
wiki-Vote	889	2915
hamsterster	2426	16630

Table 4. Comparison of Perceptions

Algorithm	Data Set				
		dol	fac	wiki	ham
TSWNN	0.0375	0.0808	0.5068	0.7609	
other	0.0191	0.0227	0.1433	0.4374	

$O(c * n)$ respectively. However, TSWNN achieves relatively low time complexity by using a neural network model to process path selection and timing information.

Table 5 indicates that in networks with varying numbers of arcs, TSWNN consistently outperforms the other four algorithms in terms of cognitive performance, while the remaining algorithms show similar performance. As the number of arcs increases, the optimal cognitive subgraph cognition of the network generally increases, suggesting that the cognition of each algorithm is affected by the size of the network.

Table 5. Comparison of Perceptions

Algorithm	Data Set				
		dol	wiki	fac	ham
TSWNN	0.0408	0.5068	0.5392	0.7609	
other	0.0166	0.1433	0.4217	0.4374	

5.3 Experimental Results with Different Limiting Times

Figure 5 illustrates that under the same constraints, TSWNN performs the fastest on the wiki-Vote dataset, while the other algorithms are relatively slower. As time limits increase, solution times for all algorithms rise, but TSWNN maintains its speed advantage. Dijkstra algorithm's time complexity is typically $O((V + E) * \log(V))$, resulting in long runtimes on large graphs. Algorithm A* and Label algorithm share a similar time complexity. TDNN's involvement of neural networks leads to a higher time complexity

of $O(c * n)$. In contrast, TSWNN leverages neural networks for efficient path and timing processing, ensuring optimal solutions across different time constraints.

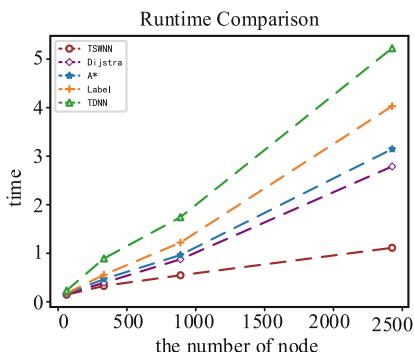


Fig. 3. Runtime Comparison

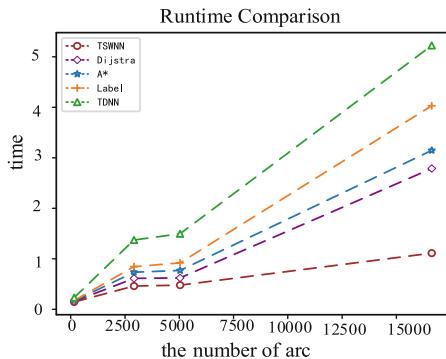


Fig. 4. Runtime Comparison

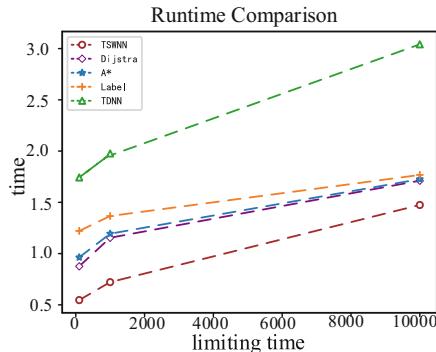


Fig. 5. Runtime Comparison

Under the same time constraints, TSWNN achieved maximum awareness compared to the other algorithms, which had similar levels of awareness. As time constraints increased, all algorithms exhibited a decrease in required cognition, yet TSWNN maintained maximum cognition. Despite decreasing information timeliness, TSWNN consistently found globally optimal solutions. Detailed results are presented in Table 6, highlighting TSWNN's superior cognitive performance across various time constraints.

Table 6. Comparison of Perceptions

Algorithm	T		
	100	1000	10000
TSWNN	0.6757	0.5068	0.2241
other	0.2031	0.1433	0.0369

The identical highest cognitive awareness in the comparative experiments is due to the utilization of shortest path-based methods. The higher the cognitive awareness, the better, as it facilitates broader dissemination of information.

6 Conclusion

In this paper, the proposed TSWNN is essentially a training-free temporal sequential wave neural network framework, which is utilized to solve the optimal cognitive subgraph query problem using the proposed TSWNN. The TSWNN algorithm has several advantages over other methods. It consistently achieves optimal solutions. TSWNN also boasts faster computational speed due to its parallel computing nature, making it suitable for large-scale networks. Additionally, its performance superiority is validated through time complexity analyses and comparative experiments. Given that real-life networks are not static, future research aims to incorporate the time factor to enhance the structure and framework of TSWNN for addressing optimal cognitive subgraph queries in time-varying environments.

References

1. Xiaohuan, S., Chunjie, J., Linlin, D., Xingyan, D., Baoyan, S.: Dynamic Top-K interesting subgraph query on large-scale labeled graphs. *Information* **10**(61) (2019)
2. Guanfeng, L., et al.: Multi-constrained graph pattern matching in large-scale contextual social graphs. In: International Conference on Data Engineering, pp. 351–362. IEEE (2015)
3. Jing, Z., Xiaokang, Z., Jifan, Yu, Jian, T., Jie, T., Hong, C.: Subgraph retrieval enhanced model for multi-hop knowledge base question answering. *ACL*, 5773–5784 (2022)
4. Yuxuan, W., Ying, X., Junchi, Y., et al.: ZeroBind: a protein-specific zero-shot predictor with subgraph matching for drug-target interactions. *Nat. Commun.* **14**, 7861 (2023)
5. Linhao, C., Jia, Y., Xinrui, G.: Privacy preserving subgraph isomorphism query for dynamic graph database. *Journal* **211**, 103562 (2023)
6. Ullmann, J.R.: An Algorithm for Subgraph Isomorphism. *ACM* **23**(1), 31–42 (1976)
7. Cordella, L.P., Foggia, P., Sansone, C.: A (sub) graph isomorphism algorithm for matching large graphs. *IEEE Trans. Pattern Anal. Mach. Intell.* **26**(10), 1367–1372 (2004)
8. Jüttner, A., Madarasi, P.: VF2++—an improved subgraph isomorphism algorithm, discrete applied mathematics. *Discret. Appl. Math.* **242**, 69–81 (2018)
9. Vincenzo, C., Pasquale, F., Antonio, G.: VF3-Light: a lightweight subgraph isomorphism algorithm and its experimental evaluation. *Pattern Recogn. Lett.* **125**, 591–596 (2019)
10. Lei, Z., Lei, C., Ozs, M.T., Dongyan, Z.: Answering pattern match queries in large graph databases via graph embedding Proc. VLDB Endow **21**(1), 97–120 (2012). <https://doi.org/10.1007/s00778-011-0238-6>
11. Fei, B., Lijun, C., Xuemin, L., et al.: Efficient subgraph matching by postponing cartesian products. In: Proceedings of ACM SIGMOD 2016, pp. 1199–1214 (2016)
12. Lihui, L., Boxin, D., et al.: G-Finder: approximate attributed subgraph matching. In: 2019 IEEE International Conference on Big Data (Big Data), pp. 513–522 (2019)
13. Yuanyuan, T., Richard, C.M., Carlos, S., et al.: SAGA: a subgraph matching tool for biological graphs. *Bioinformatics* **23**(2), 232–239 (2007)
14. Anjan, D., Josep, L., Horst, B., Umapada, P.: Product graph-based higher order contextual similarities for inexact subgraph matching. *Pattern Recognit.* **76**, 596–611 (2018)
15. Wei, W., Chunwang, Y., Jinsong, W., et al.: A time-delay neural network for solving time-dependent shortest path problem. *Neural Netw.* **90**, 90:21–28 (2017)



Joint Prior Relation Enhancement and Non-autoregressive Decoding for Document-Level Event Extraction

Xue Kang^{1,2,3}, Yan-Ni Han^{1,2,3}(✉), Wen Zhang^{1,2,3}, Han Jiang^{1,2,3}, Yi Xiang⁴,
and Shu-Guang Liu⁵

¹ Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
hanyanni@iie.ac.cn

² School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

³ Key Laboratory of Cyberspace Security Defense, Beijing, China

⁴ Chongqing University of Science and Technology, Chongqing, China

⁵ Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences,
Chongqing, China

Abstract. Document-level event extraction (DEE) aims to extract all structured event information from a given document. This research currently faces three main challenges: (1) Event arguments often scatter across multiple sentences; (2) A document typically comprises multiple interrelated events; (3) Traditional autoregressive decoding extraction methods result in exceedingly slow extraction speeds. To address these challenges, this paper proposes a Document-Level Event Extraction method by Joint Prior Relation Enhancement and Non-Autoregressive Decoding (PRENA). Specifically, we propose a Prior Relationship Enhancement Network (PREN) to explicitly model multiple prior dependencies between entity mentions and sentences, effectively capturing global interactions between cross-sentence arguments and sentences, and modeling dependencies between multiple correlated events. In addition, we propose a non-autoregressive decoding algorithm based on an optimized pseudo triggers selection strategy, which efficiently extracts multiple event argument combinations in a joint parallel manner, significantly improving the efficiency of model extraction. Comparative experiments validate the outstanding performance of the proposed PRENA in enhancing both the accuracy and efficiency of event extraction.

Keywords: Event extraction · Prior relation · Non-autoregressive · pseudo triggers

1 Introduction

Event extraction is a core task and difficult problem in information extraction, aiming to extract all structured event information from unstructured text, greatly promoting the development of academic fields such as information retrieval [1, 2], public opinion monitoring [3, 4], and knowledge graph [5, 6].

Based on the different granularity of extracted text, event extraction can be divided into two categories: sentence-level event extraction (SEE) [7–11] and document level event extraction (DEE) [12–14]. In comparison to SEE, DEE is typically associated with challenges of “across-sentence” and “multievent” issues: (1) In DEE, event arguments usually scatter across various sentences. As illustrated in Fig. 1b, for the “EU” event type, the event arguments “4000000” and “Jan. 7, 2009” are in Sentence①, “6740000” is in Sentence④, while “Hu Meizhen” is in both Sentence① and Sentence④. This necessitates the capability to capture global contextual information interactions of arguments that scatter across multiple sentences. (2) Typically, a document describes multiple related events. Analyzing sentences ① to ④ in Fig. 1b, two event types are identified: “EU” and “EO”, with the “EO” type corresponding to two event records. In addition, these different event records are interrelated and share common arguments. This presents a requirement for us to recognize the global relevance and interdependency among multiple related events.

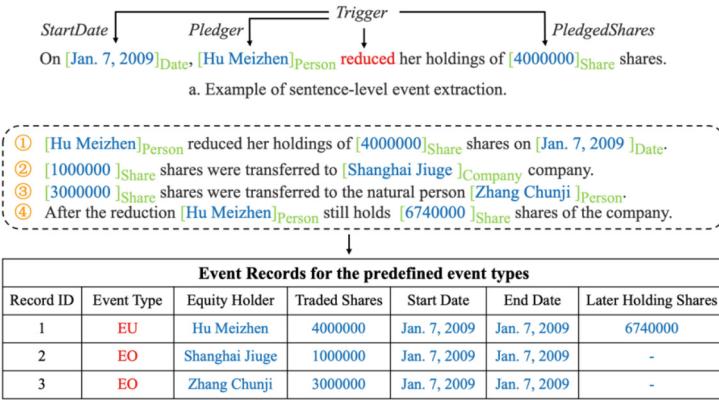


Fig. 1. Example of event extraction.

To address these challenges, previous DEE methods were mainly based on Directed Acyclic Graphs (DAG). However, such methods necessitate significant computing resources and memory space due to their reliance on autoregressive decoding algorithm, resulting in extremely low training and inference efficiency.

Considering the above issues, this paper proposes a Document-Level Event Extraction method by Joint Prior Relation Enhancement and Non-Autoregressive Decoding (PRENA). The main contributions of this paper are summarized as follows:

- 1) We introduce a prior dependency mechanism, and propose a prior relation enhancement network to model multiple prior dependencies of entity mentions and sentences, in order to capture global information interactions among cross-sentence arguments, and model dependencies between multiple related events.
- 2) We propose a non-autoregressive decoding algorithm based on an optimized pseudo triggers selection strategy to jointly extract events, which promotes bidirectional

information interaction between multiple related events while explicitly improving extraction efficiency.

- 3) We validate the effectiveness of the PRENA model on the widely used datasets, which achieves highly competitive extraction performance while greatly improving the training and inference speed of the model.

2 Related Work

Currently, many studies focus on document-level event extraction. Yang et al. [12] propose the DCFEE model, which initially identifies the core arguments of the main sentence, then automatically extracts and fills in missing arguments from the surrounding sentences. Zheng et al. [13] propose the Doc2EDAG model, which makes significant improvements by constructing an argument directed acyclic graph (DAG). A series of DAG-based models have been derived from this: Xu et al. [14] put forward GIT model that further utilizes a heterogeneous graph convolutional network to capture the global context of entities and sentences. The ReDEE model [16] introduces argument relation information into the event extraction domain for the first time. The PTPCG model [15] innovatively proposes an event extraction framework based on pseudo-trigger-aware pruned complete graph. Overall, although there has been some progress in document-level event extraction research, most models more or less overlook the importance of global interaction and relation features between cross-sentence arguments and sentences, and methods based on DAG decoding significantly reduce the extraction speed of the models. Our research aims to optimize the accuracy of model extraction and the efficiency of training and inference.

3 Methodology

In this section, we provide a comprehensive overview of PRENA. We first present the overall structure of our model, then we discuss in detail the specific implementation approaches for each component of the model.

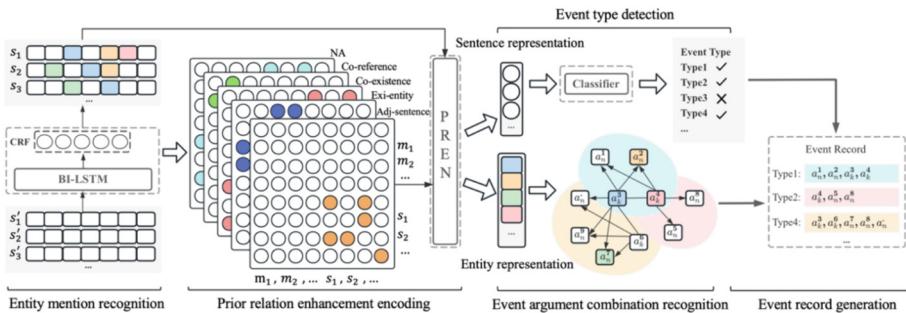


Fig. 2. Overview of our PRENA.

As shown in Fig. 2, our model can be divided into five key components: (1) Entity mention recognition: Identify all entity mentions in the document as candidate event arguments. (2) Prior relation enhancement encoding: Construct a prior relation enhancement network to model the interactions between entity mentions and sentences. (3) Event argument combination recognition: Optimize the pseudo triggers selection strategy and adopt a non-autoregressive algorithm to decode all event argument combinations. (4) Event type detection: Detect all possible event types in the document. (5) Event record generation: Identify the roles played by each argument in the event argument combinations corresponding to specific event types.

3.1 Entity Mention Recognition

We first employ a BI-LSTM architecture to encode document D at the sentence level, obtaining the token-level sequence representations of sentence s_i as $\{w_i\}_{i=1}^{|s_i|}$. We then connect a Conditional Random Field (CRF) [18] to the output layer of the BI-LSTM, converting the entity recognition task into a sequence labeling task with BIO (Begin, Inside, Other) scheme [13] to obtain the entity labeling sequence with the maximum probability. The training function for this module is:

$$L_{ner} = - \sum_{i \in |D|} \log P(y_{s_i} | s_i) \quad (1)$$

where y_{s_i} is the ground truth label sequence for s_i . Since an entity mention may span multiple tokens, we perform a max-pooling operation on token-level representations and aggregate it with the type embedding t_i , to obtain the representation of the entity mention m_i . This is formally denoted as $m_i = \text{MaxPooling}\left(\{w_i\}_{i=1}^{|m_i|}\right) \| t_i \in \mathbb{R}^{d_m}$, where $|m_i|$ denotes the total number of tokens in entity mention m_i , and d_m represents the embedding dimension of the entity mention. In addition, we concatenate the final hidden states from each direction of the BI-LSTM to obtain the sentence representation $s_i = \overrightarrow{w}_{|s_i|} \| \overleftarrow{w}_0 \in \mathbb{R}^{d_m}$.

3.2 Prior Relation Enhancement Encoding

Due to the fact that DEE tasks typically involve multiple interrelated events, and the event arguments of each event often scatter across multiple sentences. In addition, an event argument, representing an entity object, may correspond to multiple entity mentions across sentences. So, capturing the global interaction between across-sentence entity mentions and sentences is crucial for document level event extraction tasks. Therefore, we introduce a prior dependency mechanism between entity mentions and sentences, and construct a Prior Relation Enhancement Network (PREN) to explicitly model the multiple prior relations between entity mentions and sentences.

Prior Dependency Mechanism. Through entity mention recognition, we can obtain the initial entity mention representation matrix for document D as $M = \{m_i\}_{i=1}^{|d_p|} \in \mathbb{R}^{d_p \times d_m}$ and the initial sentence representation matrix as $S = \{s_i\}_{i=1}^{|d_q|} \in \mathbb{R}^{d_q \times d_m}$, where d_q denotes the number of entity mentions, d_p denotes the number of sentences. To capture global interactions between entity mentions and sentences, we introduce a prior dependency

mechanism between entity mention nodes and sentence nodes, encompassing five types of prior relations:

Co-reference: If an entity object is referenced multiple times, there exists a co-reference dependency relation between its entity mentions, which is effective in facilitating co-reference resolution. For instance, “Co-reference” relation exists between “Hu Meizhen” entity mention nodes in sentences 1 and 4 in Fig. 3.

Co-existence: There is a co-existence relation between a sentence and all entity mentions within that sentence, which allows for modeling contextual information of entity mentions within the sentence, thereby capturing sentence-level semantic information. For instance, “Co-existence” relation exists between sentence node “2” and “1000000” or “Shanghai Jiuge” entity mention nodes within that sentence in Fig. 3.

Adj-Sentence: Refer to the relation between any two adjacent sentences in a document. By utilizing Adj-sentence, contextual information of sentences can be modeled, thereby capturing document-level semantic information. For instance, “Adj-sentence” relation exists between sentence node “3” and sentence node “4” in Fig. 3.

Exi-Entity: Model the dependency among all entity mentions within the same sentence, thereby capturing entity level semantic information. For instance, “Exi-entity” relation exists between “3000000” and “Zhang Chunji” entity mention nodes in sentence 3 in Fig. 3.

NA: Model other entity mention entity mention, entity mention-sentence, and sentence-sentence node pairs that do not belong to any of the four types of dependency relations mentioned above.

On this basis, we construct a prior relation matrix $D^{pr} \in \mathbb{R}^{n \times (d_p+d_q) \times (d_p+d_q)}$ to map the prior dependencies between entity mentions and sentences in the document, where n represents the number of prior dependency relations. For each element $d_{n,i,j}^{pr}$ in D^{pr} , we use a binary value to denote whether there is a n^{th} prior dependency relation between $node_i$ and $node_j$, formalized as:

$$d_{n,i,j}^{pr} = \begin{cases} 1, & node_i - node_j \text{ have the } n^{th} \text{ prior dependency} \\ 0, & node_i - node_j \text{ do not have the } n^{th} \text{ prior dependency} \end{cases} \quad (2)$$

where $node_i$ refers to either an entity mention node or a sentence node.

Prior Relation Enhancement Network. To effectively encode the prior dependency relations between entity mentions and sentences, we design the **Prior Relation Enhancement Network** (PREN). As shown in Fig. 3, PREN innovatively integrates the prior relation matrix D^{pr} into the computation process of the attention mechanism, building upon the foundation of the native Transformer architecture [19].

Specifically, we first aggregate entity mention and sentence embedding representations to obtain the initial input embeddings for PREN as $X = M \| S \in \mathbb{R}^{(d_p+d_q) \times d_m}$.

We then apply a linear transformation to acquire the prior relation enhancement query matrix $Q_{pr} = XW_q^{pr}$ and key matrix $K_{pr} = XW_k^{pr}$. Here, $W_q^{pr}, W_k^{pr} \in \mathbb{R}^{d_m \times d_m}$ are the prior relation enhancement weight matrices. Next, we calculate the prior relation enhancement attention scores A_{pr} as follows:

$$A_{pr} = \text{softmax} \left(\frac{\sum_{i=1}^n Q_{pr} W[i,:,:] K_{pr}^T \odot D^{pr}[i,:,:]}{\sqrt{d_m}} + B \right) \quad (3)$$

where $W \in \mathbb{R}^{n \times d_m \times d_m}$ represents the trainable weight matrices, B is a bias vector, and \odot denotes element-wise multiplication. Next, we define the self-attention weight matrices $W_q^{sf}, W_k^{sf} \in \mathbb{R}^{d_m \times d_m}$ used to generate the self-attention query matrix $Q_{sf} = XW_q^{sf}$ and key matrix $K_{sf} = XW_k^{sf}$. We then calculate the self-attention scores A_{sf} :

$$A_{sf} = \text{softmax}\left(\frac{Q_{sf}K_{sf}^T}{\sqrt{d_m}}\right) \quad (4)$$

Finally, we introduce the weight matrix $W_v^{res} \in \mathbb{R}^{d_m \times d_m}$ to obtain the value matrix $V_{res} = XW_v^{res}$. And we combine the prior relation enhancement attention scores A_{pr} with the self-attention scores A_{sf} , to compute the final output A_{res} of the PREN model attention calculation module.

$$A_{res} = (A_{pr} + A_{sf})V_{res} \quad (5)$$

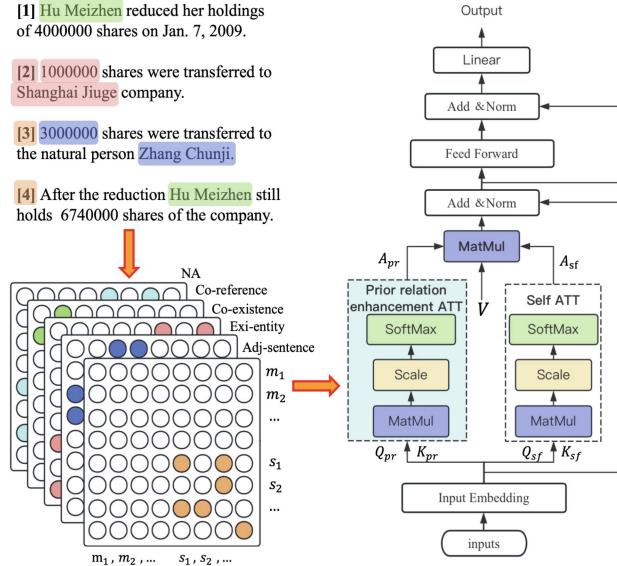


Fig. 3. The PREN architecture.

3.3 Event Argument Combination Recognition

Argument Semantic Graph. Due to the semantic space distances between various event arguments within the same event are closer, we add an entity semantic distance measurement layer after the prior relation enhancement representation layer, to obtain the semantic space distance between argument entities.

Given the entity matrix representation enhanced by prior relations $E = \{\varepsilon_i\}_{i=1}^{|\varepsilon|} \in \mathbb{R}^{|\varepsilon| \times d_m}$, we use the dot product similarity function to calculate the semantic distance

score matrix $\tilde{N}_{i,j}$, and set a threshold δ to transform the score matrix $\tilde{N}_{i,j}$ into a binary entity adjacency matrix $\tilde{N}_{i,j}$ formalized as:

$$\tilde{\varepsilon}_i = W_p^T \varepsilon_i + b_p, \tilde{\varepsilon}_j = W_q^T \varepsilon_j + b_q \quad (6)$$

$$\tilde{N}_{i,j} = \text{sigmoid}\left(\frac{\tilde{\varepsilon}_i^T \tilde{\varepsilon}_j}{\sqrt{d_m}}\right), N_{i,j} = \begin{cases} 1, & \text{if } \tilde{N}_{i,j} \geq \delta \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

where $W_p, W_q \in \mathbb{R}^{d_m \times d_m}$ and $b_p, b_q \in \mathbb{R}^{d_m}$ are trainable parameter weight matrices. Clearly, by directing $entity_i$ towards $entity_i$ corresponding to $N_{i,j} = 1$ we can further construct the complete event argument semantic graph G of the document based on the obtained entity adjacency matrix $N_{i,j}$. The loss function is defined as:

$$L_{\text{sem}} = \frac{\sum_j \sum_i [y_{i,j}^N \log N_{i,j} + (1 - y_{i,j}^N) \log (1 - N_{i,j})]}{-|N|} \quad (8)$$

Non-autoregressive Algorithm. To achieve efficient event argument combination extraction, we refer to the Pseudo-Trigger-aware Pruned Complete Graph (PTPCG) method [15] and employ a non-autoregressive combination decoding algorithm. PTPCG selects a set of pseudo-trigger (core argument combinations) for each event type in the document, assuming all pseudo-trigger in the same event are bidirectionally connected to each other, while there is only a unidirectional connection from pseudo-trigger to any other ordinary arguments. This allows for the joint identification of all event argument combinations from the graph G . However, unlike PTPCG, we propose an optimized pseudo-trigger selection strategy for more accurate identification of pseudo-trigger, thereby efficiently decoding more complete event argument combinations.

Specifically, we obtain the importance score for pseudo-trigger by fusing Information Gain $G(R, T)$, Existence $Exi^{(i)}(R)$, and Distinguishability $Dis^{(i)}(R)$. The existence and distinguishability inherited from PTPCG are used to measure whether R can recognize and distinguish combinations, respectively. Formally, given event type $t_i, R = \{r_j\}_{j=1}^{|R|}$ represents a subset of predefined argument roles for t_i . Information Gain $IG(R, T)$ is used to measure the amount of information provided by R in determining different event type combinations in a document:

$$H(T) = -\sum_{t_i} p(t_i) \log p(t_i), H(T|R) = -\sum_{r_j \in R} p(r_j) \sum_{t_i} p(t_i|r_j) \log p(t_i|r_j) \quad (9)$$

$$IG(R, T) = H(T) - H(T|R) \quad (10)$$

where $p(t_i)$ is the probability of occurrence of event type t_i , $p(r_j)$ is the probability that the argument corresponding to role r_j is not null, and $p(t_i|r_j)$ is the probability that the event type is t_i under the aforementioned condition. The complete importance score as shown in (11):

$$Impo = \alpha IG(R, T) + \beta Exi^{(i)}(R) + \gamma Dis^{(i)}(R) \quad (11)$$

where α, β , and γ are hyper-parameters, ultimately selects the candidate with the highest $Impo$ as the pseudo-trigger. This non-autoregressive decoding algorithm that does not rely on DAG can greatly improve model training and inference speed.

3.4 Event Type Detection

Through prior relation enhancement modeling, we obtain a sentence embedding representation matrix S . For each predefined event type t_i , we use a binary classifier to predict whether the corresponding event is identified. The training loss L_{type} is:

$$L_{type} = - \sum_{t_i} \log P(y_i | S) \quad (12)$$

where y_i represents the label for the i -th event type.

3.5 Event Record Extraction

After identifying candidate event types $T_c = \{t_i\}_{i=1}^{|T_c|}$ and argument combinations $A = \{a_j\}_{j=1}^{|A|}$ through event type detection and event argument combination recognition, we can obtain all type-combination pairs $\{(t_i, a_j) | t_i \in T_c, a_j \in A\}$. Next, we use a fully connected feedforward neural network as a classifier. Based on the predefined argument roles for event type t_i , we classify the roles of all arguments ε_j in the argument combination a_j , in order to identify the event argument combination and roles for each specific event type in the document. The training function is:

$$L_{rec} = - \sum_j \sum_i [y^{(i,j)} \log p^{(i)}(t_i | a_j) + (1 - y^{(i,j)}) \log (1 - p^{(i)}(t_i | a_j))] \quad (13)$$

where $y^{(i,j)}$ is the golden answers that most match a_j . The overall loss function of the model is designed as the sum of the losses of four subtasks with different weights:

$$L_{total} = \alpha_1 L_{ner} + \alpha_2 L_{sem} + \alpha_3 L_{type} + \alpha_4 L_{rec} \quad (14)$$

where $\alpha_1, \alpha_2, \alpha_3$ and α_4 are hyper-parameters.

4 Experiments

4.1 Datasets

We conduct evaluations on the widely-used document-level event extraction datasets ChiFinAnn [13] in the financial domain. ChiFinAnn is currently the largest trigger-free document-level event extraction dataset, consisting of 32040 financial announcements, of which approximately 29% of documents contain multiple event records, and 98% of event arguments are scattered in different sentences.

4.2 Baselines and Metric

Baselines: The DCFEE model proposed by Yang et al. [12] is a representative model oriented towards main sentence filling, which has two variants: DCFEE-O and DCFEE-M. The Doc2EDAG model proposed by Zheng et al. [13] first propose DEE method by constructing argument directed acyclic graphs, and derived a series of DAG-based

models: GreedyDec greedily extracts individual event record from document. GIT [14] uses heterogeneous graph convolutional network to capture the global context of entities and sentences. PTPCG [15] proposes a lightweight event extraction framework based on pseudo-trigger pruning complete graph.

Metrics: We follow the evaluation metric setup by Zheng et al. in Doc2EDAG. For each event record, we calculate the macro and micro precision, recall and F1 score at the argument role level.

4.3 Main Results

Overall Performance. The experimental results in Table 1 indicate that our PRENA has the best Macro F1 and Micro F1 scores compared to previous models, thanks to the prior relation enhancement and an optimized non-autoregressive algorithm. Compared with the previous advanced model GIT, PRENA increased by 1.7% and 0.4% in Macro-F1 and Micro-F1, respectively, demonstrating highly competitive extraction capabilities.

From a model scale perspective, PRENA is remarkably lightweight, with its parameter magnitude constituting only 29.6% of GIT, yet achieving performance higher than GIT. Additionally, experimental results reveal that the highest Macro-F1 and Micro-F1 scores for PRENA are attained when the semantic graph is pruned to $|R| = 1$. Therefore, unless explicitly stated otherwise, subsequent experiments are based on the scenario where $|R| = 1$.

Table 1. Comparison between PRENA and baselines on ChiFinAnn. w/oE is the number of parameters without vocabulary embeddings.

Model	#Params(w/oE)	Macro			Micro		
		P	R	F1	P	R	F1
DCFEE-O	32M(16M)	59.5	49.0	53.7	67.7	54.4	60.3
DCFEE-M	32M(16M)	57.4	46.7	51.5	58.1	55.2	56.6
GreedyDec	64M(48M)	73.2	39.1	51.0	80.4	49.1	61.0
Doc2EDAG	64M(48M)	78.5	68.2	73.0	80.3	75.0	77.5
PTPCG	32M(16M)	83.7	65.4	73.4	84.2	68.1	79.4
GIT	97M(81M)	79.1	70.1	74.3	82.3	78.4	80.3
PRENA $ R =All$	40M(24M)	79.7	63.2	70.5	80.1	65.8	72.2
PRENA $ R =2$		82.1	65.8	73.1	82.3	72.8	77.3
PRENA $ R =1$		84.8	68.9	76.0	85.7	76.2	80.7

Argument Scattering and Multi Events. The cross-sentence issue is a significant challenge in document-level event extraction tasks. To effectively evaluate the capability of our PRENA in addressing this issue, we divide the dataset into four groups based on the degree of argument scatter across sentences: I, II, III, and IV, with Group IV representing the document sets involving the highest number of sentences per event record. According

Table 2. Comparison between PRENA and baselines on four sets with an increasing number of sentences involved in event records.

Model	I	II	III	IV
DCFEE-O	64.6	70.0	57.7	52.3
DCFEE-M	54.8	54.1	51.5	47.1
GreedyDec	67.4	68.0	60.8	50.2
Doc2EDAG	79.6	82.4	78.4	72.0
PTPCG	80.6	83.7	79.2	74.5
GIT	81.9	85.7	80.0	75.7
PRENA	82.7	86.5	82.9	79.5

to Table 2, PRENA outperforms the baseline models in Groups I, II, III, and IV, with F1 score improvements of 0.8%, 0.8%, 2.9%, and 3.8% respectively. This indicates that PRENA can effectively capture global interactions between cross-sentence arguments and sentences, thereby mitigating the cross-sentence issue.

The experimental results in Table 3 indicate that PRENA achieve the best Macro-F1 and Micro-F1 scores on the document set with only a single event record (S), registering an increase of 2.7% and 2.0% in F1 scores, respectively. On the document set with multiple event records (M), PRENA’s Micro-F1 score is the best, while Macro-F1 score decreased by 0.3% compared to GIT. This indicates that PRENA can effectively mitigate multi-event issues to a certain extent, but there is still room for improvement.

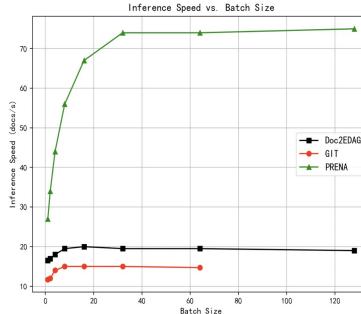
Table 3. Comparison between PRENA and baselines on documents with only a single event record (S) and multiple event records (M) on ChiFinAnn.

Model	Macro			Micro		
	S	M	S&M	S	M	S&M
DCFEE-O	62.5	47.0	53.7	69.0	50.3	60.3
DCFEE-M	57.9	47.7	51.5	63.2	49.4	56.6
GreedyDec	75.1	38.1	51.0	77.8	37.0	61.0
Doc2EDAG	81.1	67.1	73.0	81.0	67.4	77.5
PTPCG	84.1	66.1	73.4	88.2	69.1	79.4
GIT	83.1	70.1	74.3	87.6	72.3	80.3
PRENA	85.8	69.8	76.0	89.6	73.4	81.0

Comparison on Speed. The results in Table 4 indicate that PRENA, with its nonautoregressive advantage, significantly surpasses the others in training speed. Doc2EDAG and GIT require 6.3 days and 6.6 days, respectively, to train 100 epochs using four GPUs,

Table 4. Comparison of training time between PRENA and baselines.

Model	pfs-days	Best Epoch	One Epoch (minutes)	Best (hours)	Total (days)
Doc2EDAG	0.131	82	90.6	123.8	6.3
GIT	0.137	96	95.1	152.2	6.6
PRENA	0.009	63	18.2	19.1	1.5

**Fig. 4.** Comparison of inference speed between PRENA and baselines.

whereas PRENA completes training in just 1.5 days using a single GPU. Furthermore, in terms of pfs-days comparison, PRENA is respectively 14.6 and 15.2 times faster than Doc2EDAG and GIT.

Figure 4 illustrates the performance comparison between PRENA and the baselines in terms of inference speed. As the batch size increases, the inference speed of the PRENA model gradually accelerates, ultimately stabilizing at a peak of 75 docs/s. In contrast, Doc2EDAG and GIT do not show significant speed-up in inference rate with increasing batch sizes, peaking at 19 docs/s and 15 docs/s respectively, which are 3.9 and 5.0 times slower than PRENA.

4.4 Ablation Study

We design a series of model ablation experiments on the ChiFinAnn dataset: 1) -PREN: we remove the prior dependency construction and replace the prior relation enhancement network with a BI-LSTM; 2) -NAO: we remove the non-autoregressive algorithm based on optimized pseudo-trigger selection strategy; 3) -PREN-NAO: we simultaneously remove the above two modules in the original model.

As shown in Table 5, the experimental results demonstrate that both the PREN and NAO modules significantly and positively impact the overall performance of the model, which perfectly validates the effectiveness of each sub-module proposed in our research.

Table 5. Ablation study of PRENA variants.

Model	Macro			Micro		
	P	R	F1	P	R	F1
PRENA	85.7	76.2	80.7	71.6	56.5	63.2
-PREN	-1.9	-2.5	-2.3	-2.6	-2.4	-2.6
-NAO	+ 0.7	-2.4	-1.1	+ 0.4	-0.8	-0.4
-PREN-NAO	-1.5	-3.5	-2.7	-4.1	-1.9	-2.9

5 Conclusions

This study proposes a novel method for document-level event extraction: PRENA. PRENA innovatively proposes a Prior Relation Enhancement Network and a non-autoregressive decoding framework based on optimized pseudo-trigger selection strategy, which is highly competitive in extraction performance while also exhibiting significant advantages in training and inference speed. Future research can focus on further enhancing the model’s capability in handling complex multi-event scenarios.

Acknowledgments. This work is supported by the Cooperation project between Chongqing Municipal undergraduate universities and institutes affiliated to CAS (HZ2021015).

References

1. Fan, Y., et al.: Pre-training methods in information retrieval. *Foundations and Trends® in Information Retrieval*, vol. 16, no. 3, pp. 178–317 (2022)
2. Lu, Y., et al.: Unified structure generation for universal information extraction,” arXiv preprint [arXiv:2203.12277](https://arxiv.org/abs/2203.12277) (2022)
3. Meng, F., Xiao, X., Wang, J.: Rating the crisis of online public opinion using a multi-level index system. arXiv preprint [arXiv:2207.14740](https://arxiv.org/abs/2207.14740) (2022)
4. Wang, Z., Zhang, S., Zhao, Y., Chen, C., Dong, X.: Risk prediction and credibility detection of network public opinion using blockchain technology. *Technol. Forecast. Soc. Chang.* **187**, 122177 (2023)
5. Wu, X., Wu, J., Fu, X., Li, J., Zhou, P., Jiang, X.: Automatic knowledge graph construction: a report on the 2019 icdm/icbk contest. In: 2019 IEEE International Conference on Data Mining (ICDM). IEEE, pp. 1540–1545 (2019)
6. Bosselut, A., Le Bras, R., Choi, Y.: Dynamic neuro-symbolic knowledge graph construction for zero-shot commonsense question answering. *Proceedings of the AAAI Conference on Artificial Intelligence* **35**(6), 4923–4931 (2021)
7. Li, F., et al.: Event extraction as multi-turn question answering. In: Findings of the Association for Computational Linguistics: EMNLP 2020, pp. 829–838 (2020)
8. Paolini, G., et al.: Structured prediction as translation between augmented natural languages, arXiv preprint [arXiv:2101.05779](https://arxiv.org/abs/2101.05779) (2021)
9. Lu, Y., et al.: Text2event: Controllable sequence-to-structure generation for end-to-end event extraction, arXiv preprint [arXiv:2106.09232](https://arxiv.org/abs/2106.09232) (2021)

10. Lv, J., et al.: Trigger is non-central: Jointly event extraction via label-aware representations with multi-task learning. *Knowl.-Based Syst.* **252**, 109480 (2022)
11. Li, H., Gao, T., Wang, J., Li, W.: Joint event extraction via structural semantic matching. arXiv preprint [arXiv:2306.03469](https://arxiv.org/abs/2306.03469) (2023)
12. Yang, H., Chen, Y., Liu, K., Xiao, Y., Zhao, J.: Dcfee: a document-level Chinese financial event extraction system based on automatically labeled training data. In: Proceedings of ACL 2018, System Demonstrations, pp. 50–55 (2018)
13. Zheng, S., Cao, W., Xu, W., Bian, J.: Doc2edag: an end-to-end document-level framework for chinese financial event extraction. arXiv preprint [arXiv:1904.07535](https://arxiv.org/abs/1904.07535) (2019)
14. Xu, R., Liu, T., Li, L., Chang, B.: Document-level event extraction via heterogeneous graph-based interaction model with a tracker. arXiv preprint [arXiv:2105.14924](https://arxiv.org/abs/2105.14924) (2021)
15. Zhu, T., et al.: Efficient document-level event extraction via pseudo-trigger-aware pruned complete graph, arXiv preprint [arXiv:2112.06013](https://arxiv.org/abs/2112.06013) (2021)
16. Liang, Y., Jiang, Z., Yin, D., Ren, B.: Raat: relation-augmented attention transformer for relation modeling in document-level event extraction, arXiv preprint arXiv: 2206.03377 (2022)
17. Tong, M., et al.: Docee: A large-scale and fine-grained benchmark for document-level event extraction. Association for Computational Linguistics (2022)
18. Graves, A., Graves, A.: Long short-term memory, Supervised sequence labelling with recurrent neural networks, pp. 37–45 (2012)
19. Vaswani, A., et al.: Attention is all you need. Advances in neural information processing systems, vol. 30 (2017)



Intrusion Detection System Based on ViTCycleGAN and Rules

Menghao Fang¹, Xia Li², Yuanyuan Wang³, Qiuxuan Wang⁴, Xinlei Sun⁵,
and Shuo Zhang²(✉)

¹ School of Cyber Science and Engineering, University of International Relations,
Beijing 100091, China

² Marine Engineering College, Dalian Maritime University, Liaoning 116026, China
2220210726.zs@dltmu.edu.cn

³ School of Mathematics and Statistics, Anyang Normal University, Anyang 455099, China

⁴ Navigation College, Dalian Maritime University, Liaoning 116026, China

⁵ Information Science and Technology College, Dalian Maritime University,
Liaoning 116026, China

Abstract. This paper explores the application of deep learning techniques in the field of intrusion detection and its potential problems. Possible challenges of deep learning in intrusion detection include handling the imbalance between positive and negative samples, which leads to unstable model performance in distinguishing normal and abnormal traffic. To address this issue, the paper proposes combining deep learning-based intrusion detection techniques with a rule-based approach to enhance the system's adaptability and intelligence. The specific scheme includes four Vision Transformer models, two generators, and two discriminators. The discriminators are used to differentiate normal traffic and detect abnormal behaviors, following strict detection rules to reach a final determination. Through validation on the NSL-KDD dataset and CIC-DDOS2019 dataset, the proposed scheme achieves accuracies of 98.32% and 99.23%, respectively, providing new research insights and solutions in the field of intrusion detection.

Keywords: Deep Learning · Intrusion Detection · Neural Network · Information Security

1 Introduction

With the popularity of the Internet, networks are growing rapidly, connecting people around the globe, but also affecting the personal and property security of users. An attack on the network can lead to irreparable damage to personal property [1]. In May 2021, Colonial Pipeline suffered a major ransomware attack. Using ransomware called Dark-Side, hackers successfully breached Colonial Pipeline's computer network, encrypted the company's data, and demanded a ransom to decrypt it. This incident forced Colonial Pipeline to shut down its pipeline system, halting the delivery of petroleum products and causing a fuel crisis in the eastern United States. Internet security protection is urgent.

M. Fang, X. Li—These authors contributed equally to this work.

Common traffic attacks include Distributed Denial of Service Attacks (DDoS Attacks), Distributed Reflection Amplification Attacks (DRDoS Attacks), SYN Flooding Attacks, UDP Flooding Attacks, and ICMP Flooding Attacks, etc. [2]. These traffic attacks can cause the target network or server to become overloaded, preventing normal users from accessing the service normally, resulting in service unavailability. In this case, users may not be able to complete online transactions, access website content, or use network services, affecting user experience and normal business operations. It brings great losses to users and enterprises.

Deploy an effective firewall and intrusion detection and prevention system (IDS/IPS). Firewalls can effectively prevent attack traffic from entering the network system by filtering network traffic and identifying and blocking malicious traffic [3]. In addition, intrusion detection systems (e.g., neural network-based IDSs) that incorporate deep learning techniques can monitor network traffic in real time and use deep learning algorithms to deeply analyze and identify traffic data to discover potential abnormal traffic and attacks [4]. Deep learning models can learn the complex patterns and characteristics of network traffic to improve detection of unknown attacks while reducing false positives and effectively reducing the reliance on human intervention.

In this paper, we propose an innovative Vision Transformer [5] based Cycle-Consistent Generative Adversarial Network [6] model called ViTCycleGAN (Vision Transformer Cycle-Consistent Generative Adversarial Network). The model incorporates deep learning and pattern recognition approaches, aiming to identify and prevent malicious behaviors, thus securing the network. Specifically, in the ViTCycleGAN model, this paper designs four Vision Transformer network models, including two generators and two discriminators. The role of the generators is to optimize the representation of data features, which enables the discriminators to capture data features more comprehensively. The discriminators, on the other hand, have different functions, one for recognizing positive samples and the other for recognizing negative samples, with the final decision made by a set of rules.

Our study shows through sufficient experimental validation that the proposed intrusion detection framework provides unique advantages in malicious traffic detection and network traffic monitoring. Validation against the CIC-DDOS2019 dataset [7] and the NSL-KDD dataset [8] shows that our model achieves 99.23% and 98.32% accuracy, respectively. These experimental results fully demonstrate that our model performs excellently in the intrusion detection domain.

2 Related Work

An intrusion detection system is designed to protect computer systems and network resources from malicious attacks, unauthorized access, or other security threats. It can help organizations detect and respond to potential vulnerabilities, attacks, or abnormal activity in a timely manner, thereby ensuring the security and reliability of information systems.

With the rapid development of artificial intelligence and deep learning technologies, deep learning techniques are beginning to be introduced into the field of intrusion detection, such as the use of models such as convolutional neural network (CNN) [9] and

recurrent neural network (RNN) [10] for network traffic analysis and anomaly detection. With the improvement of computational power and hardware equipment, improved Generative Adversarial Networks [11] and Attention Mechanism [12] have been gradually applied to the field of intrusion detection.

Rule-based intrusion detection techniques use a predefined set of rules to identify anomalous behaviours or attacks. For example, rules are set to detect abnormal login behaviours, illegal access attempts, etc. *Nitin Naik et al.* [13] introduced the Dynamic Fuzzy Rule Interpolation (D-FRI) method into the intrusion detection system, which reduces the rate of false positives and misses. *Giuseppina Andresini et al.* [14] proposed a new intrusion detection method, which analyses the network flow data based on stream features and learns an intrusion detection model using the original deep metric learning method combining Auto Encoders and Triplet networks.

Currently, more and more scholars are combining rule-based intrusion detection techniques with deep learning-based intrusion detection techniques, for example, *Subham Kumar Gupta et al.* [15] proposed a hybrid optimized and deep learning centred IDS to achieve higher accuracy. *P Rajesh Kanna et al.* [16] proposed a new approach to intrusion detection by using optimized CNN (OCNN) and a unified model of Hierarchical Multiscale LSTM (HMLSTM) to propose a highly accurate IDS model to extract and learn spatio-temporal features efficiently.

There are two common problems with GAN-based intrusion detection systems, the first is the pattern collapse problem of the generator, which can lead to inadequate capture of data features by the discriminator. The second problem is the imbalance between the discriminator's detection of positive and negative samples, e.g., the model may be able to effectively discriminate between normal traffic but may not be able to discriminate between anomalous traffic, or the model may detect anomalous traffic but may also misclassify some of the normal traffic as anomalous.

To address this problem, this paper trains two discriminators, one for distinguishing normal traffic and the other for detecting abnormal behavior. These discriminators follow a strict set of detection rules, including multi-dimensional analysis based on traffic characteristics, behavioral patterns, and abnormal flags. Ultimately, the system is able to accurately determine the presence of malicious behavior.

3 Methodologies

In this paper, we propose an innovative cycle-consistent generative adversarial network model known as ViTCycleWGan (Vision Transformer Cycle-Consistent Generative Adversarial Network). The model's two generators and two discriminators are composed of Vision Transformer models, which are used to train two discriminators to detect anomalous behavior in traffic. In this paper, two discriminators are trained where one discriminator is used to judge normal traffic and the other discriminator is used to judge abnormal traffic, and the two discriminators make a judgment together to improve the accuracy of the intrusion detection system. Finally, this paper verifies the robustness and effectiveness of the model with the CiC-DDOS2019 dataset and the NSL-KDD dataset. The general framework of the method is shown in Fig. 1.

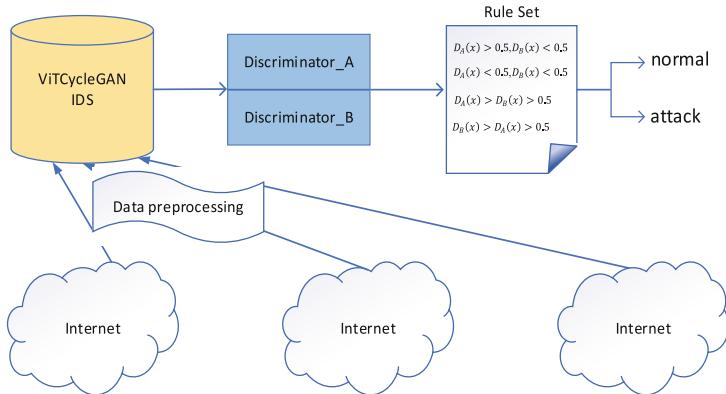


Fig. 1. Overarching Framework

3.1 Cycle-Consistent Generative Adversarial Network (CycleGAN)

CycleGAN is a deep learning model for unsupervised image transformation. It can transform one type of image into another without the need for training data pairs, which makes it useful in many practical applications. And in our field of intrusion detection, it is often used to generate normal or anomalous samples for data enhancement.

3.2 Vision Transformer

Vision Transformer (ViT) is a deep learning model based on an attention mechanism originally developed for image classification tasks. However, it can also be applied to intrusion detection, especially in processing image data and achieving efficient feature extraction. With ViT, we can achieve efficient feature extraction and representation learning on image data, thus improving the recognition ability of intrusion detection systems for both normal and abnormal images. The multi-scale processing capability of ViT allows the system to comprehensively analyze the image data and achieve more accurate target detection and anomaly detection. By using pre-trained ViT models and migration learning methods, the deployment and training process of the intrusion detection system can be accelerated to improve detection performance and effectively respond to network security threats.

3.3 Intrusion Detection System: ViTCycleGAN

Four network models, including two generators and two discriminators, were designed in this study. These generator and discriminator networks all use the Vision Transformer network structure. In the field of intrusion detection, traditional neural network models often suffer from vision blindness, resulting in inadequate detection of sample features or inadequate detection of either positive or negative samples. For example, the model may be able to effectively distinguish normal traffic but not abnormal traffic; or the model may detect abnormal traffic but also misclassify some of the normal traffic as abnormal. The solution proposed in this paper effectively addresses this problem by training two

discriminator models, one discriminator model focused on judging positive samples and the other discriminator model focused on judging negative samples. Moreover, these discriminators are constructed using the Vision Transformer model, which has excellent feature extraction capability. Finally, the final result is determined by the set of intrusion detection rules based on the values of the two discriminators.

3.3.1 The Discriminators and Generators Composed of ViT

In the intrusion detection model based on ViTCycleGAN, the network structure of the two generators and the two discriminators adopts the Vision Transformer structure. For example, after the traffic extracted from the NSL-KDD dataset is preprocessed, it first passes through a Linear layer, which is reshaped into the form of (128,128,3), and then passes through a Patch layer, where the Patch consists of a Conv2d and a Flatten layer, and then proceeds to the Transformer Encoder.

3.3.2 Design of Optimization Objectives for Flows

Since both ViT and CycleGAN are network models in the field of graph generation, they need to be conditioned and transformed into intrusion detection models. In this paper, four ViT models are designed as generators G_A , G_B and discriminator D_A , D_B , then the training set traffic is divided into abnormal traffic $\{x_{\text{attack}}\}_1^N$ and normal traffic $\{x_{\text{normal}}\}_1^N$. The purpose of the generator is to convert the normal traffic x_{normal} into fake abnormal traffic $x_{\text{fake-attack}}$ and the purpose of the generator G_B is to convert the abnormal traffic x_{attack} into fake normal traffic $x_{\text{fake-normal}}$.

$$x_{\text{fake-attack}} = G_A(x_{\text{normal}}), x_{\text{fake-normal}} = G_B(x_{\text{attack}}) \quad (1)$$

The purpose of the discriminator D_A is whether the sample is normal traffic or not, and the purpose of the discriminator D_B is to determine whether the sample is abnormal traffic or not. Then the generator G_A converts the false normal traffic to cyclic abnormal traffic, and the generator G_B converts the false abnormal traffic to cyclic normal traffic.

$$x_{\text{cycle-attack}} = G_A(x_{\text{fake-normal}}) \quad (2)$$

$$x_{\text{cycle-normal}} = G_B(x_{\text{fake-attack}}) \quad (3)$$

The training process diagram is shown in Fig. 2, which contains the process of passing data between the two generators and the two discriminators.

Finally cyclic consistency loss function L_{cyc} to calculate the similarity of cyclic anomaly vectors $x_{\text{cycle-attack}}$ to the anomaly vectors x_{attack} and the similarity of cyclic normal vectors $x_{\text{cycle-normal}}$ to the normal vectors x_{attack} . Where $E_{x_{\text{attack}} \sim P_{\text{data}}}$ is the data distribution of the abnormal flow and $E_{x_{\text{normal}} \sim P_{\text{data}}}$ is the data distribution of the normal flow.

$$\begin{aligned} L_{\text{cyc}} = & E_{x_{\text{normal}} \sim P_{\text{data}}(x_{\text{normal}})} [\|G_B(G_A(x_{\text{normal}})) - x_{\text{normal}}\|_1] \\ & + E_{x_{\text{attack}} \sim P_{\text{data}}(x_{\text{attack}})} [\|G_A(G_B(x_{\text{attack}})) - x_{\text{attack}}\|_1] \end{aligned} \quad (4)$$

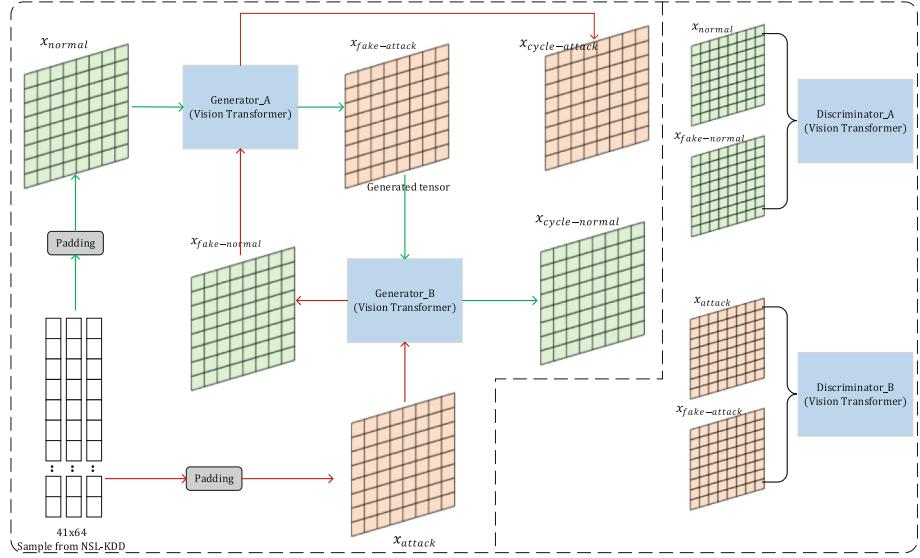


Fig. 2. ViTCycleGAN Training Graph

and yet

$$L_{cyc} = \|x_{cycle-normal} - x_{normal}\|_1 + \|x_{cycle-attack} - x_{attack}\|_1 \quad (5)$$

where the cyclic consistency loss function uses the L1 loss function for the samples and predicted values, x and y is the sample and predicted value, the L1 loss function is as follows:

$$loss(x, y) = \frac{1}{n} \sum_{i=1}^n |y_i - f(x_i)| \quad (6)$$

In order to achieve the optimal results, both " $\|x_{cycle-normal} - x_{normal}\|_1$ " and " $\|x_{cycle-attack} - x_{attack}\|_1$ " must be optimized using the L1 loss function.

There are two types of adversarial loss functions, which are $L_{normal \rightarrow attack}$ and $L_{attack \rightarrow normal}$. The first adversarial loss function is the discriminator D_B to distinguish whether the traffic is anomalous or not and the formula is shown below

$$\begin{aligned} L_{normal \rightarrow attack}(G_A, G_B, x_{normal}, x_{attack}) &= E_{x_{attack} \sim P_{data}(x_{attack})} [\log D_B(x_{attack})] \\ &\quad + E_{x_{normal} \sim P_{data}(x_{normal})} [\log(1 - D_B(G_A(x_{normal})))] \\ &\quad + E_{x_{normal} \sim P_{data}(x_{normal})} [\log(1 - D_B)] \end{aligned} \quad (7)$$

where $E_{x_{attack} \sim P_{data}}$ is the data distribution for abnormal traffic and $E_{x_{normal} \sim P_{data}}$ is the data distribution for normal traffic. The second adversarial loss function, which is the discriminator D_A to distinguish whether a flow is normal or not, is publicised as follows.

$$\begin{aligned} L_{normal \rightarrow attack}(G_A, G_B, x_{normal}, x_{attack}) &= E_{x_{normal} \sim P_{data}(x_{normal})} [\log D_A(x_{normal})] \\ &\quad + E_{x_{attack} \sim P_{data}(x_{attack})} [\log(1 - D_A(G_B(x_{attack})))] \end{aligned}$$

$$+ E_{x_{attack} \sim P_{data}(x_{attack})} [\log(1 - D_A(x_{attack}))] \quad (8)$$

In this case, the adversarial loss function often uses the MSE loss function, publicized as follows.

$$MSE = \frac{1}{n} \sum (x_i - y_i)^2 \quad (9)$$

where x and y are the sample and predicted values, respectively.

Therefore, this paper derives the loss function formula for training the generator as follows.

$$\begin{aligned} L_{Generator} = & E_{x_{normal} \sim P_{data}(x_{normal})} [\log(1 - D_B(G_A(x_{normal})))] \\ & + E_{x_{attack} \sim P_{data}(x_{attack})} [\log(1 - D_A(G_B(x_{attack})))] + L_{cyc} \end{aligned} \quad (10)$$

The loss function formula for the discriminator is as follows. The loss function for distinguishing normal traffic.

$$\begin{aligned} L_{D_A} = & E_{x_{normal} \sim P_{data}(x_{normal})} [\log D_A(x_{normal})] \\ & + E_{x_{attack} \sim P_{data}(x_{attack})} [\log(1 - D_A(G_B(x_{attack})))] \\ & + E_{x_{attack} \sim P_{data}(x_{attack})} [\log(1 - D_A(x_{attack}))] \end{aligned} \quad (11)$$

The loss function for distinguishing abnormal traffic.

$$\begin{aligned} L_{D_B} = & E_{x_{attack} \sim P_{data}(x_{attack})} [\log D_B(x_{attack})] \\ & + E_{x_{normal} \sim P_{data}(x_{normal})} [\log(1 - D_B(G_A(x_{normal})))] \\ & + E_{x_{normal} \sim P_{data}(x_{normal})} [\log(1 - D_B(x_{normal}))] \end{aligned} \quad (12)$$

3.3.3 Intrusion Detection Rules Collection

In this paper, we use trained discriminator and discriminator for malicious behavior identification. The role of discriminator is to determine whether the unknown traffic is normal traffic or not, and the role of discriminator is to determine whether the unknown traffic is malicious traffic or not. And build the following ruleset:

- 1) If $D_A(x) > 0.5$ $D_B(x) < 0.5$, then it indicates that the traffic is normal traffic.
- 2) If $D_A(x) < 0.5$, $D_B(x) > 0.5$, then the flow is an abnormal flow.
- 3) If $D_A(x) > D_B(x) > 0.5$, then it indicates that the traffic is normal traffic.
- 4) If $D_B(x) > D_A(x) > 0.5$, then the flow is an abnormal flow.

4 Experiments

In this paper, the CIC-DDOS2019 dataset and the NSL-KDD dataset are selected as the training and testing sets of the model for experiments. Then, the data are preprocessed by completing the missing values, removing outliers, normalizing, and converting the data

that cannot be processed by computer into data that can be processed, e.g., anonymizing IP addresses and normalizing port numbers. This article conducted experiments comparing ViTCycleGAN with visual converters and MLP as the internal structures of CycleGAN and GAN. The results showed that ViTCycleGAN outperformed any single structure in classification tasks.

4.1 Dataset

CIC-DDOS2019 is an open-access cybersecurity dataset released in 2019 by a team of researchers at Concordia University. The NSL-KDD dataset is a classic network intrusion detection dataset designed to provide more realistic and diverse network traffic data to evaluate and improve the performance of intrusion detection systems. As shown in Table 1 and Fig. 3.

Table 1. CIC-DDOS 2019 Common Types of Attacks

CICDDOS-2019	SYN Flood	Slowloris	Ping of Death
	UDP Flood	RUDY	TCP SYNACK Flood
	ICMP Flood	NTP Amplification	UDP Fragment Flood
	HTTP Flood	SSL Renegotiation	HTTP Slow Post
	DNS Amplification	LAND	UDP Fragmentation Flood

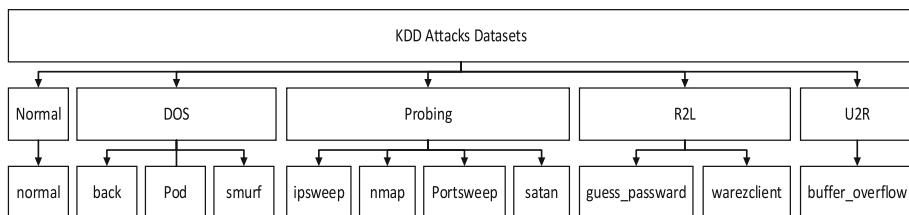


Fig. 3. KDD Dataset By Attack Type

4.2 Data Preprocessing

Before the data can be prepared for use, it must first be pre-processed. The preprocessing steps include removing outliers, normalizing the data, converting text data to numeric form, and converting IP addresses to numbers. In this study, a set of rules was used to remove outliers and the Z-Score method was used to normalize the data.

Conversions are performed for certain features, such as converting certain character-based data to floating point, and converting IP addresses to numbers. The conversion formula is shown below:

$$\text{Integer Value} = a \times 256^3 + b \times 256^2 + c \times 256 + d \quad (13)$$

where a , b , c , and d make up the IP address, representing each of the four parts of the IP address, the first part of the IP address is multiplied by 256^3 , i.e. $a \times 256^3$ to indicate that the value of that part occupies the highest place in the integer. The second part of the IP address is multiplied by 256^2 , i.e. $b \times 256^2$, to indicate that the value of that part occupies the second-highest place in the integer. The third part of the IP address is multiplied by 256, i.e. $\times 256$, indicates that the value of the part occupies the third place in the integer. The value of the fourth part of the IP address is added directly to the integer, i.e. d . By weighting each part by multiplying it by the appropriate base number and summing the results, an integer value is obtained that uniquely represents the original IP address.

Z-Score normalization (also known as standardization or Z-Score normalization) is a common method of data standardization that converts data to a standard normal distribution with a mean of 0 and a standard deviation of 1.

$$z = \frac{x - \mu}{\sigma} \quad (14)$$

where x is the eigenvalue of the original data, μ is the mean of the original data feature, and σ is the standard deviation of the original data feature.

4.3 Performance Evaluation of ViTCycleGAN Intrusion Detection System

In order to objectively evaluate the effectiveness of our proposed method, we conducted comparative experiments and compared our model with state-of-the-art intrusion detection systems. The experimental results indicate that our method has unique advantages. The experimental results are shown in Fig. 4.

Table 2 and Fig. 3 shows the binary classification results of ablation trials performed on the NSL-KDD dataset using different algorithms. We used several major algorithms for comparison, including Vision Transformer, MLP-CycleGAN, GAN, and our proposed algorithm. In terms of accuracy, our algorithm performs well, reaching 0.9832, which is higher than the other algorithms. Meanwhile, our algorithm also achieves a significant advantage in false negative rate (FNR) and false positive rate (FPR), which are 0.0124 and 0.01131, respectively, indicating that our algorithm has a better control on the false positive rate. The comprehensive evaluation index, F1-score, also shows that our algorithm has a unique advantage in comprehensive performance, reaching 0.9723, which further proves the effectiveness and superiority of our proposed method on the NSL-KDD dataset.

Table 3 and Fig. 4 summarizes the binary classification results of ablation experiments performed on the CIC-DDOS dataset. We compare several important algorithms, including Vision Transformer, MLP-CycleGAN, GAN, and our proposed algorithm. The results show that our algorithm achieves 0.9923 accuracy, which is much higher than other algorithms. In addition, our algorithm also performs well in terms of false negative rate (FNR) and false positive rate (FPR), which are 0.0116 and 0.0067, respectively, indicating that our algorithm has a low risk of misclassification. The comprehensive evaluation metric, F1-score, further confirms the superiority of our algorithm at 0.9911, showing the excellent performance and effectiveness of our algorithm on the CIC-DDOS dataset.

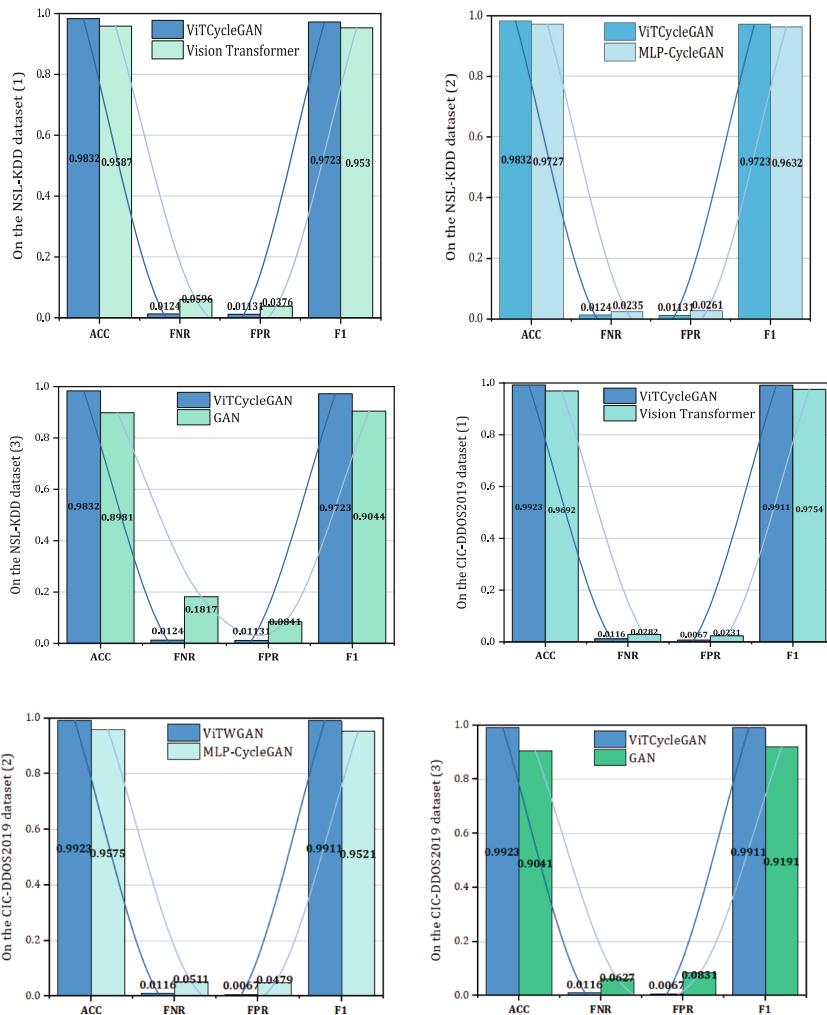


Fig. 4. Ablation experiments on the NSL-KDD & CIC-DDOS2019 datasets

Table 2. Binary classification results for the NSL-KDD dataset of ablation tests

Arithmetic	Accuracy	FNR	FPR	F1-score
Vision Transformer	0.9587	0.0596	0.0376	0.9530
MLP-CycleGAN	0.9727	0.0235	0.0261	0.9632
GAN	0.8981	0.1817	0.0841	0.9044
Our	0.9832	0.0124	0.0113	0.9723

Table 3. CIC-DDOS dataset binary classification results for ablation trials

Arithmetic	Accuracy	FNR	FPR	F1-score
Vision Transformer	0.9692	0.0282	0.0231	0.9754
MLP-CycleGAN	0.9575	0.0511	0.0479	0.9521
GAN	0.9041	0.0627	0.0831	0.9191
Our	0.9923	0.0116	0.0067	0.9911

5 Conclusion

In this study, we propose an Intrusion Detection System based on ViTCycleGAN and rules, which distinguishes between normal and abnormal traffic using two discriminators. One discriminator focuses on detecting normal behavior, while the other focuses on detecting abnormal behavior. By combining these two discriminators to detect anomalous behavior, we achieve effective monitoring and protection of network security.

Our research has achieved some insights, discriminator dual validation, by introducing two discriminators, we achieve dual validation of network traffic, which improves the accuracy and reliability of the intrusion detection system.

References

1. Jin, K., Ye, D.: Optimal innovation-based stealthy attacks in networked LQG systems with attack cost. *IEEE Trans. Cybern.* **54**(2), 787–796 (2024)
2. Hwang, R.-H., Lee, C.-L., Lin, Y.-D., Po-Chin Lin, Hsiao-Kuang Wu, Yuan-Cheng Lai, C.K. Chen,
3. Quincozes, S.E., Raniery, C., Nunes, R.C., Albuquerque, C., Passos, D., Mossé, D.: Counselors network for intrusion detection. *Int. J. Netw. Manag.* **31**(3), May/June 2021
4. Host-based intrusion detection with multi-datasource and deep learning. *J. Inf. Secur. Appl.* **78** (2023), 103625, ISSN 2214-2126
5. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Houlsby, N.: An image is worth 16x16 words: transformers for image recognition at scale (2020)
6. Zhu, J.-Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, pp. 2242–2251 (2017)
7. <https://www.unb.ca/cic/datasets/ddos-2019.html>
8. <https://www.unb.ca/cic/datasets/nsl.html>
9. S.V. Pingale, Sanjay R. Sutar, Remora whale optimization-based hybrid deep learning for network intrusion detection using CNN features, *Expert Systems with Applications*, Volume 210, 2022, 118476, ISSN 0957-4174,
10. Sheikhan, M., Jadidi, Z., Farrokhi, A.: Intrusion detection using reduced-size RNN based on feature grouping. *Neural Comput. & Applic.* **21**, 1185–1190 (2012)
11. Giuseppina Andresini, Annalisa Appice, Luca De Rose, Donato Malerba, GAN augmentation to deal with imbalance in imaging-based intrusion detection, *Future Generation Computer Systems*, Volume 123, 2021, Pages 108–127, ISSN 0167-739X
12. Laghrissi, F., Douzi, S., Douzi, K., et al.: IDS-attention: an efficient algorithm for intrusion detection systems using attention mechanism. *J Big Data* **8**, 149 (2021)

13. Naik, N., Diao, R., Shen, Q.: Dynamic Fuzzy Rule Interpolation and Its Application to Intrusion Detection. *IEEE Trans. Fuzzy Syst.* **26**(4), 1878–1892 (2018)
14. Andresini, G., Appice, A., Malerba, D.: Autoencoder-based deep metric learning for network intrusion detection. *Inf. Sci.* **569**, 706–727 (2021). ISSN 0020–0255
15. Subham Kumar Gupta, Meenakshi Tripathi, Jyoti Grover, Hybrid optimization and deep learning based intrusion detection system, *Computers and Electrical Engineering*, Volume 100, 2022, 107876, ISSN 00457906
16. P Rajesh Kanna, P Santhi, Unified Deep Learning approach for Efficient Intrusion Detection System using Integrated Spatial–Temporal Features, *Knowledge-Based Systems*, Volume 226, 2021, 107132, ISSN 09507051



DFT-3DLaneNet: Dual-Frequency Domain Enhanced Transformer for 3D Lane Detection

Kaijiang Li, Yuling Liu, Peisen Wang^(✉), XiangQian Liu, Xichen Liu, ChunYi Guo,
and Bing Zhou

Zhengzhou University, Zhengzhou, China
wps28501@gs.zzu.edu.cn

Abstract. In autonomous driving, 3D lane detection using a monocular camera presents significant challenges. In complex driving environments, spatial-domain image features may have certain limitations. To address these challenges, we propose a DFT-3DLaneNet model that uses a dual-frequency domain enhancement 3D unified deformable attention for monocular 3D lane detection. The model utilizes a learnable filter radius to extract dynamic high-frequency and low-frequency features, enhancing both the lanes and 3D space. On the one hand, we propose a dual-channel high-frequency feature enhancement module (DHF) to enhance lane features. On the other hand, we propose a cross-channel low-frequency attention module (CLA) to enhance 3D spatial perception. Low-frequency features not only alleviate the problem of imbalanced lane type distribution in the dataset but also improve the capture of 3D lane features and make the model more robust. Experimental results show that our method outperforms existing state-of-the-art approaches in terms of F-score.

Keywords: Autonomous driving · Monocular 3D lane detection · Low-pass filtering · High-pass filtering · Deformable attention

1 Introduction

With advancements in computing power, artificial intelligence, and sensor technology, autonomous driving reaches a breakthrough stage. One critical component is lane detection, which plays a crucial role in ensuring the safety, and accuracy of autonomous driving. Lane detection provides vehicles with rich road information, enabling them to perceive their surroundings and follow the predetermined driving trajectory.

In flat road scenes, lane detection can be approximated as a 2D task. However, real traffic scenes are complex, including mountain roads with uphill and downhill slopes, road surface depressions, and gravel roads. In complex scenarios, 2D lane detection can only capture planar features without depth information. To address the lack of depth information, some methods employ monocular images and utilize geometric features such as curve fitting and anchor points to achieve 3D lane detection [1, 9]. PersFormer [2] employs a novel feature transformation method that utilizes camera parameters as a reference and BEV features generated from the front view for 3D lane detection. However, projecting camera parameters onto BEV may cause 3D space displacement due to

uneven road surfaces. Anchor3Dlane [9] introduces a BEV-independent 3D lane detection approach by predefining lane anchors in 3D space. LATR [11] performs 3D lane detection directly from front-viewed images by querying lane perception and embedding 3D ground spatial positions. Despite significant progress in 3D lane detection independent of BEV, inferring the underlying 3D structure from a single monocular image remains a challenging task.

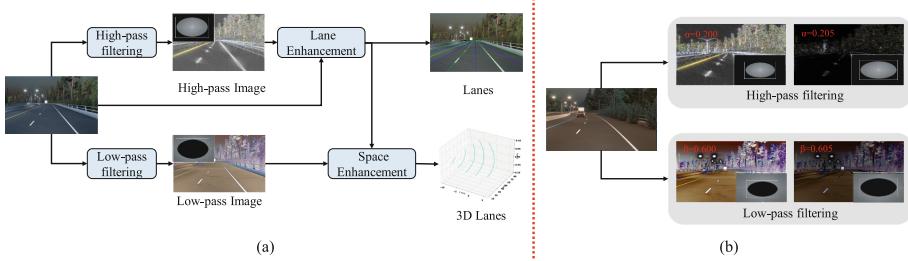


Fig. 1. (a) Frequency-domain filtering is applied to the 3D lane perception data processing pipeline. After high-pass filtering in the frequency domain, the edge information of lane texture is more abundant, providing a greater advantage than using the original image for lane perception. For images that have undergone low-pass filtering, spatial information is refined, which is beneficial for 3D spatial perception after removing strong contrast edge information. Combining these two types of filtered images is more advantageous for 3D lane perception [5]. (b) The perception of lane lines varies with different radii of high-pass filtered images. Similarly, the perception of 3D spatial features varies with different low-pass filtered images. Here, α and β represent the ratio of filter radius for high-pass and low-pass filters, respectively. Best viewed on screen.

To address these challenges, we separate image features based on the characteristics of high-pass and low-pass filtering in the frequency domain. As shown in Fig. 1(a), the high-frequency component of the image after high-pass filtering contains lane textures and edge details, which have significant advantages for lane perception. Additionally, after low-pass filtering, the image edges tend to be smoothed, which is more conducive to the perception of 3D spatial structures [5]. As shown in Fig. 1(b), due to the dynamic distribution of lane features at different frequencies in different images caused by various factors such as scenes and weather conditions, different filter radii are required to extract ideal high-frequency and low-frequency features. Therefore, we use a neural network to dynamically predict the filter mask radius range to enhance the robustness of the entire network model in noisy environments. Specifically, we first predict the high-pass and low-pass filtered images in the image that are beneficial for 3D lane perception. Next, we propose a dual-channel high-frequency feature enhancement module (DHF) to enhance lane line features. We also design a cross-channel low-frequency attention module (CLA) to enhance 3D spatial perception. Finally, we employ a transformer decoder composed of DHF and CLA with dual-frequency domain deformable attention for predicting 3D lane lines.

Our contributions are as follows:

- We propose DFT-3DLaneNet, an end-to-end transformer 3D lane detection framework that enhances the dual-frequency domain. By combining dynamic frequency-domain features with the task characteristics of 3D lane lines and avoiding the embedding of any geometric features, we prevent the problem of missing 3D lane features caused by the diversity of lane lines and scenes.
- We propose a method that utilizes dynamically learnable high-frequency and low-frequency feature extraction. Moreover, based on the characteristics of the two frequency-domain features, we propose a dual-channel high-frequency feature enhancement module (DHF) to enhance lane line features and a cross-channel low-frequency attention module (CLA) to enhance 3D spatial perception.
- We validate our framework on benchmark datasets Apollo Synthetic and OpenLane, and our proposed DFT-3DLaneNet achieves state-of-the-art performance.

2 Related Work

2.1 2D Lane Detection

2D lane detection involves detecting and tracking the position, direction, and shape of lanes to assist vehicles in staying within the correct lane. With the development of deep learning, using convolutional neural networks (CNN) for lane detection has become the mainstream method [14]. Lane2Seq [15] proposes a novel reinforcement learning-based multi-format model adaptation method that integrates task-specific knowledge into the detection network, achieving sequence-based lane detection. ClrNet [8] improves lane detection confidence by combining a new lane detection metric, LaneIoU, with local lane angles. While 2D lane detection made progress, it also faces many challenges such as lane occlusion, slopes and high-speed curves, uneven road surfaces, and curved road lines, which can lead to failures in lane detection for autonomous driving. To address these challenges, researchers proposed 3D lane detection.

2.2 3D Lane Detection

The latest 3D lane detection methods are mainly based on deep learning and sensor fusion. These methods combine sensor data from cameras, lidars, and other sources, and use deep learning algorithms to detect 3D lanes. However, the high cost and sparsity of lidar data limit its practical application. LATR [11] utilizes monocular images to construct cross-attention for 3D lane detection using a lane-aware query generator and dynamic 3D ground positional embedding.

However, constructing a 3D lane directly from 2D images is a highly challenging task. Complex scenarios such as curved entrances, intersections, and forks exceed the scope of simplified geometric priors, making it difficult to accurately identify lane lines. Utilizing a high-pass filter can strengthen lane textures, while a low-pass filter can aid in perceiving the 3D space by smoothing the image. Therefore, we propose an end-to-end 3D lane detection model, DFT-3DLaneNet, that combines both high-pass and low-pass filters to overcome these challenges.

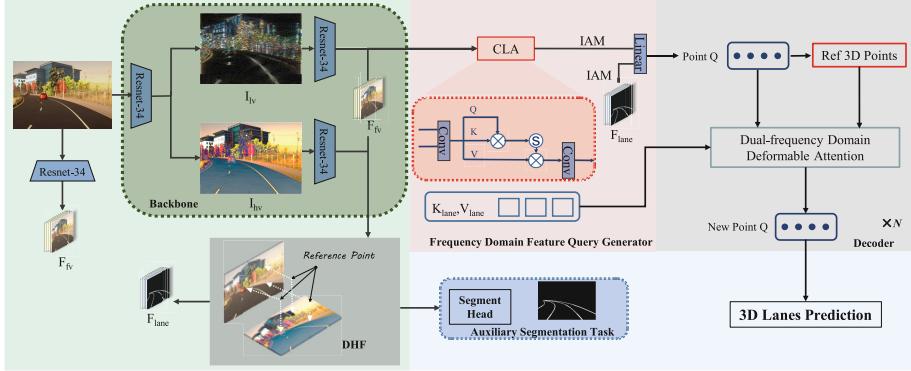


Fig. 2. The overall architecture. DFT-3DLaneNet is a novel 3D lane detection framework that uses the front-viewed image as input and separates its high-frequency and low-frequency features through learnable high-pass/low-pass processing. The high-frequency and front-viewed features are fused using DHF fusion and utilized for sublane segmentation to perceive the lane. The low-frequency and front-viewed features are processed using CLA to enhance the perception of 3D features. Finally, the two fused features are combined using IAM for query generation fusion. The fused features are used as Point Q for dual-frequency domain deformable attention lane query, enabling 3D lane detection.

2.3 Frequency Domain Image Processing

The utilization of various filtering methods in digital image processing can enhance image quality. Low-pass filters can smooth images and reduce noise, while high-pass filters can provide information related to details such as edges and textures. ClarifyNet [13] achieves single-image dehazing by fusing and processing ground truth haze-free images, low-pass filtered images, and high-pass filtered images through multiple stages of filtering and attention. Guo et al. [5] propose a new 3D low-pass filter based on non-uniform spectrum synthesis, obtaining highly smooth 3D medical models with geometric accuracy and volume preservation. Based on this, we improve the model's perception of lane lines and 3D space by using a combination of dynamic high-frequency and low-frequency features.

3 Method

The overall framework of DFT-3DLaneNet is illustrated in Fig. 2 Sect. 3.1 describes the use of learnable filter radii to extract high-frequency and low-frequency features. In Sect. 3.2, we introduce the dual-channel high-frequency feature enhancement module (DHF), which is designed to build the lane segmentation subtask and aid comprehensive perception of lane information. Section 3.3 presents cross-channel low-frequency attention module (CLA) for enhancing 3D spatial perception. In Sect. 3.4, we introduce a dual-frequency domain deformable attention that effectively aggregates high-frequency lane perception and low-frequency 3D spatial features into queries, enabling 3D lane prediction.

3.1 Frequency Domain Feature Extraction

We enhance the features of lane lines and 3D space through high-frequency and low-frequency characteristics. We employ learnable filter radii to filter the frequency-domain features of the front-viewed image $I_{fv} \in R^{3 \times H \times W}$. Specifically, we use Resnet-18 [7] as the backbone network to predict the filter radius of the front-viewed image. The specific description is as follows:

$$\alpha, \beta = MLP_{reg}(Resnet_{18}(I_{fv})), \quad (1)$$

where α and β represent the filter radius ratios for high-pass and low-pass filtering, respectively. Next, we extract high-frequency and low-frequency images using the predicted radius ratios α and β . The filtered high-frequency and low-frequency images are denoted as I_{hv} and I_{lv} , respectively. The details are described below:

$$\begin{cases} I_{hv} = IFFT(FFT(I_{fv}), Mask(H, W, \alpha)), \\ I_{lv} = IFFT(FFT(I_{fv}), \overline{Mask}(H, W, \beta)), \end{cases} \quad (2)$$

where H and W represent the size of I_{fv} . $Mask$ denotes the filtering mask, \overline{Mask} denotes the complement of the filtering mask. FFT represents the fast Fourier transform, which converts spatial-domain signals into frequency-domain signals. $IFFT$ represents the inverse Fourier transform, which converts frequency-domain signals into spatial-domain signals. Finally, we employ the CNN backbone network, specifically ResNet-34 [7], to extract features from I_{fv} , I_{hv} , and I_{lv} , resulting in the corresponding front-viewed features F_{fv} , high-frequency features F_{hv} , and low-frequency features F_{lv} .

3.2 Dual-Channel High-Frequency Feature Enhancement

We utilize the high-frequency feature F_{hv} to enhance lane perception. To achieve this, we devise a dual-channel high-frequency feature enhancement module (DHF). DHF operates by utilizing the edge features of the high-frequency feature F_{hv} to augment the perception of lane lines in the front-viewed feature F_{fv} on a pixel-to-pixel basis. The formula for computing the enhanced feature F_{lane} is as follows:

$$F_{lane} = \mathcal{G}(F_{fv}) \odot \sigma(\mathcal{G}(F_{hv})), \quad (3)$$

where σ represents the sigmoid function, \mathcal{G} represents several convolutional layers, and \odot denotes element-wise multiplication.

3.3 Cross-channel Low-Frequency Attention

In order to enhance the spatial perception ability in 3D space, we employ a cross-channel low-frequency attention module (CLA) to capture the spatial relationship between front-viewed feature F_{pv} and low-frequency feature F_{lv} . Firstly, we expand the fused feature along the channel direction 3 times by using multilayer convolutions and use it as the Q, K, V for multi-head attention. The specific description is as follows:

$$\{Q, K, V\} = \mathcal{G}([F_{pv}, F_{lv}]), \quad (4)$$

where $[,]$ represents concatenation along the channel dimension. Next, we utilize a multi-head channel attention mechanism to extract 3D features F_{space} from F_{pv} and F_{lv} . We slice them along the channel dimension to divide them into different attention heads. For each attention head, we use the following formula to calculate the attention:

$$F_{chan} = \text{Softmax}\left(QK^T/\gamma\right)V, \quad (5)$$

where γ is a learnable parameter. Finally, we concatenate the results F_{chan} from multiple heads to obtain the attention feature F_{space} .

3.4 Dual-Frequency Domain Deformable Attention

Frequency-domain Feature Query Generator. We utilize the instance activation map (IAM) [3] to generate a Point Query Q_{point} that is both lane-aware and 3D spatial-aware. Specifically, we generate the corresponding Query for the lane-aware feature F_{lane} and spatial-aware feature F_{space} separately by applying IAM. The IAM approach highlights the critical regions of the feature map by assigning higher weights to relevant activations and suppressing irrelevant ones. The description of IAM is presented below:

$$\begin{cases} Q_{lane} = \sigma(\mathcal{G}([F_{lane}, P_{2d}])) \otimes F_{lane}^T, \\ Q_{space} = \sigma(\mathcal{G}([F_{space}, P_{2d}])) \otimes F_{space}^T, \end{cases} \quad (6)$$

where \otimes denotes matrix product operation, and P_{2d} represents the embedding of spatial coordinates [10]. The spatial feature Q_{point} for lane perception is obtained by fusing Q_{lane} and Q_{space} through a learnable linear layer. The description is provided below:

$$Q_{point} = \text{Linear}([Q_{lane}, Q_{space}]). \quad (7)$$

3D Unified Deformable Attention. We design a Transformer-based architecture with N layers for our decoder. Specifically, in each layer, we utilize multiple fully connected layers based on the Q_{point} to generate our 3D Reference Points P_{3d} . The calculation for P_{3d} is described as follows:

$$P_{3d} = \text{MLP}(Q_{point}). \quad (8)$$

Next, we employ deformable attention [16] to generate the fused 3D perception feature Q . In the first layer of our decoder, we use Q_{point} as the Q for the deformable attention fusion. The iterative process for the Q feature is described as follows:

$$Q_i = \begin{cases} \text{DeformAttn}(Q_{point}, P_{3d}, F_{lane} + E_m), & i = 1, \\ \text{DeformAttn}(Q_{i-1}, P_{3d}, F_{lane} + E_m), & i > 1, \end{cases} \quad (9)$$

where i represents the $i - th$ layer of the decoder, and E_m represents Sine Positional Encoding.

3.5 Prediction and Loss

Prediction Head. During training, DFT-3DLaneNet includes a segmentation auxiliary task for lane perception. We use multiple layers of CNN layers as the segmentation head to predict the lane lines S_{pred} . To ensure model efficiency, we downsample the ground truth lane segmentation map to the same resolution as F_{lane} for prediction. The computation for predicting the segmentation feature map S_{pred} is as follows:

$$S_{pred} = \mathcal{G}(F_{lane}). \quad (10)$$

For 3D lane prediction, we utilize the features Q generated by the decoder as input and predict the x, z , lane point visibility, and lane category using MLP . The process is described in detail below:

$$\begin{cases} [\hat{x}, \hat{z}, \hat{v}] = MLP_{reg}(Q), \\ \hat{c} = MLP_{cls}(Pool_{max}(Q)). \end{cases} \quad (11)$$

Loss Function. In general, the Loss is composed of two components: the 2D lane segmentation loss and the 3D lane loss. We have:

$$\begin{aligned} \mathcal{L}_{lane} &= w_{seg}\mathcal{L}_{seg} + w_{iou}\mathcal{L}_{iou}, \\ \mathcal{L} &= w_x\mathcal{L}_x + w_z\mathcal{L}_z + w_v\mathcal{L}_v + w_c\mathcal{L}_c + w_{lane}\mathcal{L}_{lane}, \end{aligned} \quad (12)$$

where different loss weights, denoted as w_* , are defined as follows: $w_{seg} = 5$, $w_{iou} = 2$, $w_x = 2$, $w_z = 10$, $w_v = 1$, $w_c = 10$, and $w_{lane} = 2$. For learning losses in the x and z directions, we use L1 loss, denoted as \mathcal{L}_x and \mathcal{L}_z , respectively. We use focal loss, denoted as \mathcal{L}_c , to learn the lane category. For both lane point visibility and lane segmentation, we use BCE loss as the loss function, denoted as \mathcal{L}_v and \mathcal{L}_{seg} , respectively. Additionally, we optimize the segmentation performance using IOU-based loss, denoted as \mathcal{L}_{iou} .

4 Experiment

In this section, we present our experimental settings, including the dataset, evaluation metrics, and algorithm implementation details. Next, we compare DFT-3DLaneNet with state-of-the-art methods to demonstrate its superiority. Finally, we conduct ablation experiments to validate the contributions of each module.

4.1 Datasets and Metrics

Apollo Synthetic. [6] comprises a collection of over 10.5K high-resolution images synthesized using the Unity 3D engine. Images depict diverse 3D environmental scenes in Silicon Valley under various weather conditions, including highways, urban areas, residential zones, and city centers. Moreover, the Apollo Synthetic dataset comprehensively covers various scenarios in lane detection, following three primary scene classifications: balanced, rarely observed, and visual variation scenes.

Table 1. Comparison with state-of-the-art methods on Apollo Synthetic dataset with three different scenes. The symbols “N” and “F” respectively denote the short for “near” and “far”. The symbol “–” represents missing results from the original paper.

Scene	Methods	F1	AP	x err/N(m)	x err/F(m)	z err/N(m)	z err/F(m)
Balanced Scene	3D-LaneNet [4]	86.4	89.3	0.068	0.477	0.015	0.202
	Gen-LaneNet [6]	88.1	90.1	0.061	0.496	0.012	0.214
	PersFormer [2]	92.9	-	0.054	0.356	0.010	0.234
	CurveFormer [1]	95.8	97.3	0.078	0.326	0.018	0.219
	Anchor3Dlane [9]	95.6	97.2	0.052	0.306	0.015	0.223
	LATR [11]	96.8	97.9	0.022	0.253	0.007	0.202
	DFT-3DLaneNet	96.9	98.1	0.021	0.251	0.007	0.201
Rare Subset	3D-LaneNet [4]	74.6	72.0	0.166	0.855	0.039	0.521
	Gen-LaneNet [6]	78.0	79.0	0.139	0.903	0.030	0.539
	PersFormer [2]	87.5	-	0.107	0.782	0.024	0.602
	CurveFormer [1]	95.6	97.1	0.182	0.737	0.039	0.561
	Anchor3Dlane [9]	94.4	96.9	0.094	0.693	0.027	0.579
	LATR [11]	96.1	97.3	0.050	0.600	0.015	0.532
	DFT-3DLaneNet	96.3	97.4	0.048	0.598	0.013	0.529
Visual Variations	3D-LaneNet [4]	74.9	72.5	0.115	0.601	0.032	0.230
	Gen-LaneNet [6]	85.3	87.2	0.074	0.538	0.015	0.232
	PersFormer [2]	89.6	-	0.074	0.430	0.015	0.266
	CurveFormer [1]	90.8	93.0	0.125	0.410	0.028	0.254
	Anchor3Dlane [9]	91.8	92.5	0.047	0.327	0.019	0.219
	LATR [11]	95.1	96.6	0.045	0.315	0.016	0.228
	DFT-3DLaneNet	95.2	96.7	0.043	0.314	0.014	0.216

OpenLane [2] represents the largest real-world 3D lane dataset to date. It is derived from the Waymo Open dataset [12] and comprises 150K training frames and 40K testing frames. This dataset encompasses over 880K lane annotations across 200K frames, spanning 14 distinct lane categories that include various types encountered in different scenarios.

Table 2. Comparison with state-of-the-art methods on the OpenLane validation dataset. Acc. Denotes category accuracy. The symbols “N” and “F” respectively denote the short for “near” and “far”. The symbol “—” represents missing results from the original paper

Methods	F1	Acc	x err/N(m)	x err/F(m)	z err/N(m)	z err/F(m)
3D-LaneNet [4]	44.1	-	0.479	0.572	0.367	0.443
Gen-LaneNet [1]	50.5	-	0.340	0.772	0.207	0.651
PersFormer [2]	50.5	92.3	0.485	0.553	0.364	0.431
Anchor3Dlane [9]	54.3	90.7	0.275	0.310	0.105	0.135
LATR [11]	61.9	92.0	0.219	0.259	0.075	0.104
DFT-3DLaneNet	62.4	92.3	0.211	0.255	0.073	0.099

Evaluation Metrics. We employ the evaluation metrics proposed in Gen-LaneNet [6], including F-score, Average Precision (AP), as well as x/z errors at near range (0–40 m) and far range (40–100 m), to assess our Model on the aforementioned dataset.

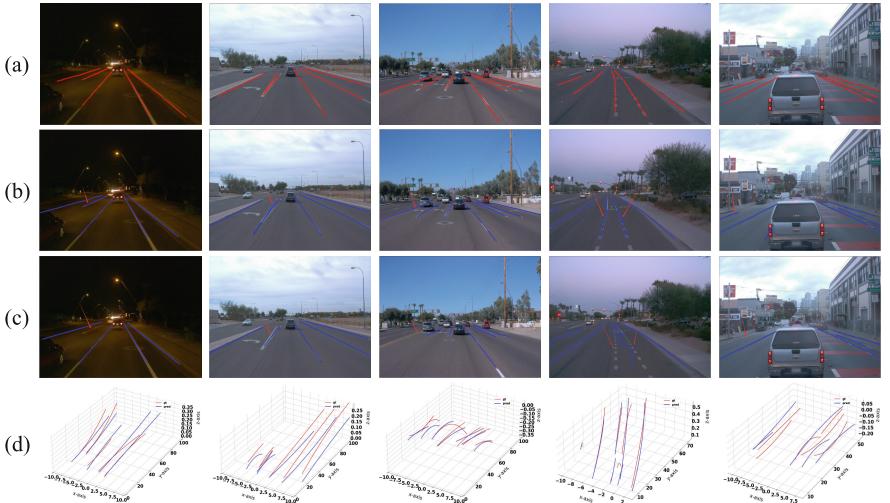


Fig. 3. Qualitative evaluation on the OpenLane validation set is presented in rows (a), (b), and (c), respectively showing the ground truth 2D lane and the 2D projection predictions of DFT and LATR [12] for the lane lines. Row (d) demonstrates the ground truth (red) and prediction of DFT (blue) in 3D space. Best viewed on screen.

4.2 Implementation Details

In DFT-3DLaneNet, we resize the input image to 720×960 and employ ResNet-18 as the backbone network for predicting the frequency-domain filtering radius. Subsequently,

high-frequency and low-frequency images are obtained through frequency-domain transformation. We utilize ResNet-34 as the backbone network for extracting high-frequency, low-frequency, and front-viewed features. The number of lane queries is set to 40 using IAM transformation. In the dual-frequency domain deformable attention, 4 attention heads, 8 sampling points, and 512-dimensional embedding deformable attention are adopted [16].

During the training, we employ a cosine annealing scheduler. The learning rate is initialized to $1e-4$, and we utilize the Adam optimizer with a weight decay of 0.01. The model is trained on 4 V100 GPUs with a batch size of 12. The epochs are set to 48 and 180 in OpenLane and Apollo Synthetic, respectively.

4.3 Comparisons with State-of-the-Arts

Quantitative Comparison. We compare several state-of-the-art methods using F-score, AP [6]/category accuracy [2], x errors, and z errors as evaluation metrics. Table 1 reports the quantitative results on the Apollo Synthetic dataset, which verify that our method outperforms the state-of-the-art methods. DFT-3DLaneNet achieves the best performance in each scene, with the F-score metric improving by 0.1%, 0.2%, and 0.1% in the three scenes, respectively. The results on the rare subset and visual variations datasets also demonstrate that the use of high-frequency and low-frequency features improves the robustness of the model. As shown in Table 2, similar conclusions can be drawn on the OpenLane dataset.

Qualitative Comparison. We conduct a qualitative analysis and compare our approach with LATR [12] on the OpenLane dataset. This dataset contains real-world autonomous driving scenarios with significant noise and complex environments. To enhance the model’s robustness, we first apply a high-frequency filter for lane line enhancement and use a low-frequency filter to smooth image noise. As shown in Fig. 3, our model outperforms the baseline in both offset position and prediction accuracy.

Table 3. The ablation of high-frequency and low-frequency features on the rare subset of Apollo Synthetic dataset. “H-freq” denotes high-frequency features, and “L-freq” denotes low-frequency features.

H-freq	L-freq	F1	AP	x err/N(m)	x err/F(m)	z err/N(m)	z err/F(m)
		95.0	96.5	0.054	0.605	0.019	0.537
		95.9	97.0	0.052	0.601	0.015	0.532
		95.7	96.8	0.050	0.600	0.014	0.530
		96.3	97.4	0.048	0.598	0.013	0.529

4.4 Ablation Studies

In this section, we analyze the impact of the proposed components on the model through several ablation studies on the Apollo Synthetic dataset [6].

Table 4. Experimental results of using different high-frequency fusion strategies on the balanced scene of the Apollo Synthetic dataset. "H-Fusion" refers to the high-frequency feature fusion strategy.

H-Fusion	F1	AP	x err/N(m)	x err/F(m)	z err/N(m)	z err/F(m)
CNN	95.9	97.2	0.076	0.326	0.010	0.232
SpaceAtt	96.5	97.9	0.024	0.254	0.008	0.213
DHF	96.9	98.1	0.021	0.251	0.007	0.201

Frequency Domain. We compare the effects of high-frequency and low-frequency features on 3D lane detection. Considering the robustness of frequency-domain processing, we evaluate the model's performance on the rare subset of the Apollo Synthetic dataset. As shown in Table 3. The results indicate a significant improvement in F-score and AP metrics after incorporating high-frequency features, with increases of 0.9% and 0.5%, respectively. The addition of low-frequency features lead to a more noticeable improvement in x and z errors, reducing errors by 0.004–0.005 m and 0.005–0.007 m, respectively. Optimal performance in 3D lane detection is achieved when both high-frequency and low-frequency features are simultaneously enhanced.

High-Frequency Fusion Strategy. To analyze the impact of different high-frequency feature fusion strategies on the model, we conduct a comparative study of CNN, spatial attention, and our proposed DHF fusion strategies on the balanced scene of the Apollo Synthetic dataset. Since lane lines tend to be pixel-to-pixel mappings in feature representation, spatial attention has an advantage over multi-layer CNN fusion in lane detection. Therefore, our DHF fundamentally constructs a pixel mapping between high-frequency and front-viewed features. As shown in Table 4, DHF consistently exhibits significant advantages in various evaluation metrics.

Table 5. Experimental results of using different low-frequency fusion strategies on the visual variations of Apollo Synthetic dataset." L-Fusion" refers to the low-frequency feature fusion strategy.

L-Fusion	F1	AP	x err/N(m)	x err/F(m)	z err/N(m)	z err/F(m)
CNN	91.6	92.3	0.049	0.331	0.021	0.229
SpaceAtt	94.6	96.1	0.045	0.319	0.017	0.225
CLA	95.2	96.7	0.043	0.314	0.014	0.216

Low-Frequency Fusion Strategy. To analyze the impact of different low-frequency feature fusion strategies on the model, we conduct comparative analyses of CNN, spatial attention, and the proposed CLA on the visual variations of the Apollo Synthetic dataset. The 3D space itself is sensitive to spatial attention, but in IAM processing, positional spatial information is already embedded. Therefore, Query-based variable attention is

more acute towards channel features. As shown in Table 5, our proposed CLA achieves optimal results across all evaluation metrics. This also demonstrates that CLA is more adept at extracting 3D spatial features from low-frequency fusion features using channel-aware attention.

5 Conclusions

In this paper, we propose DFT-3DLaneNet, a simple yet efficient end-to-end 3D lane detection framework. We enhance lane features and 3D spatial features by utilizing high-frequency and low-frequency features separately. On the one hand, we enhance lane features through the designed dual-channel high-frequency feature enhancement module (DHF). On the other hand, we introduce a cross-channel low-frequency attention module (CLA) to boost 3D spatial perception. Lastly, we introduce a dual-frequency domain deformable attention mechanism, effectively aggregating high-frequency lane perception and low-frequency 3D spatial features into queries to achieve 3D lane prediction. Extensive experiments demonstrate the significant progress achieved by DFT-3DLaneNet, establishing the state-of-the-art on two public datasets.

References

1. Bai, Y., Chen, Z., Fu, Z., Peng, L., Liang, P., Cheng, E.: Curveformer: 3d lane detection by curve propagation with curve queries and attention. In: 2023 IEEE International Conference on Robotics and Automation (ICRA), pp. 7062–7068. IEEE (2023)
2. Chen, L., et al.: Persformer: 3d lane detection via perspective transformer and the openlane benchmark. In: European Conference on Computer Vision, pp. 550–567. Springer (2022)
3. Cheng, T., et al.: Sparse instance activation for real-time instance segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4433–4442 (2022)
4. Garnett, N., Cohen, R., Pe’er, T., Lahav, R., Levi, D.: 3d-lanenet: end-to-end 3d multiple lane detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 2921–2930 (2019)
5. Guo, Y., Su, Z., Lin, S., Lu, J., Zhong, X., Luo, X.: 3d medical model low-pass filtering based on non-uniform spectral synthesis. Comput. Aided Des. **104**, 27–35 (2018)
6. Guo, Y., et al.: Gen-lanenet: a generalized and scalable approach for 3d lane detection. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXI 16, pp. 666–681. Springer (2020)
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
8. Honda, H., Uchida, Y.: Clrernet: improving confidence of lane detection with laneiou. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 1176–1185 (2024)
9. Huang, S., et al.: Anchor3dlane: learning to regress 3d anchors for monocular 3d lane detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 17451–17460 (2023)
10. Liu, R., et al.: An intriguing failing of convolutional neural networks and the coordconv solution. Advances in neural information processing systems 31 (2018)

11. Luo, Y., et al.: Latr: 3d lane detection from monocular images with transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 7941–7952 (2023)
12. Sun, P., et al.: Scalability in perception for autonomous driving: Waymo open dataset. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2446–2454 (2020)
13. Susladkar, O., et al.: Clarifynet: A high-pass and low-pass filtering based cnn for single image dehazing. *J. Syst. Architect.* **132**, 102736 (2022)
14. Tian, Y., et al.: Lane marking detection via deep convolutional neural network. *Neurocomputing* **280**, 46–55 (2018)
15. Zhou, K.: Lane2seq: towards unified lane detection via sequence generation. arXiv preprint [arXiv:2402.17172](https://arxiv.org/abs/2402.17172) (2024)
16. Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint [arXiv:2010.04159](https://arxiv.org/abs/2010.04159) (2020)



Correlation Matters: A Stock Price Predication Model Based on the Graph Convolutional Network

Chengkun Xin¹ , Qian Han³, and Gang Pan^{2(✉)}

¹ College of Intelligence and Computing, Tianjin University, Tianjin, China

² School of Future Technology, Tianjin University, Tianjin, China

pangang@tju.edu.cn

³ Lingnan College, Sun Yat-Sen University, Guangzhou, China

Abstract. In the financial markets, accurate prediction of stocks is crucial for formulating investment strategies. Previous research predominantly relied on a stock’s historical information for prediction, but overlooked the cross-effects between stocks. However, stocks are closely connected rather than independent of each other. This work introduces a deep learning framework named Stock-GCN for stock prediction, which can be easily extended by integrating other modules. By constructing a stock graph structure, the model transforms the prediction of individual stocks into the prediction of the entire graph. Experiments show that StockGCN effectively captures comprehensive spatio-temporal correlations through modeling multi-scale stock networks and consistently outperforms state-of-the-art baselines on real-world stock datasets.

Keywords: Stock Prediction · Graph Convolutional Network · Interrelationship · Industry Graph

1 Introduction

In the challenging financial markets, accurate stock prediction is the foundation for effective investment strategies. Even slight improvements in predictive accuracy have the potential to optimize investors’ strategies and achieve significant profits [9]. Recently, with the advancements of deep learning, more effective prediction models have emerged [2, 6, 7, 15], indicating that deep learning models can effectively simulate the complex behaviors of the market. However, general methods operate under the assumption that stocks are independent, thereby neglecting the interrelationships among them. This means that the prediction results for a specific stock rely solely on its own historical data. It is clear that these studies have overlooked the cross-effects among stocks, which dynamically impact stock prices [14].

The interrelationships among stocks arise from the complex relationships between companies in reality, such as supply, competition, and information sharing, among others. These tangible relationships suggest that the future price of one stock is invariably connected to other stocks. A plethora of stock market research has been conducted

to demonstrate the existence of lead-lag effects [14] as well as sector rotation effects. Consequently, stocks are closely interconnected, and the entire stock market can be viewed as a dynamic graph.

Following this idea, due to the characteristics of the stock market, graph analysis can be applied to stock prediction tasks. A common approach in these studies is to construct a graph, either manually or automatically. In such a graph, the nodes represent stocks and the stock information serves as node features. The weights of the edges represent the interrelationships between stocks. Through edges, the known labels of nodes are disseminated to neighboring nodes. These graph-based methods represent an improvement compared to independent prediction models relying solely on historical data. However, while these methods are typically neighbor-based, they do not train and optimize based on current states of nodes and neighbors. This makes it difficult to uncover deeper relationships.

Fortunately, the emergence of Graph Neural Network (GNN) has significantly advanced the field of graph analysis, enabling the integration of GNN into the field of stock prediction. GNN encodes nodes in a graph as vectors or features, enabling the capture of node attribute information. Through message propagation and feature aggregation, GNN can learn complex relations between nodes. Therefore, the prediction of stock prices using GNN is both feasible and essential.

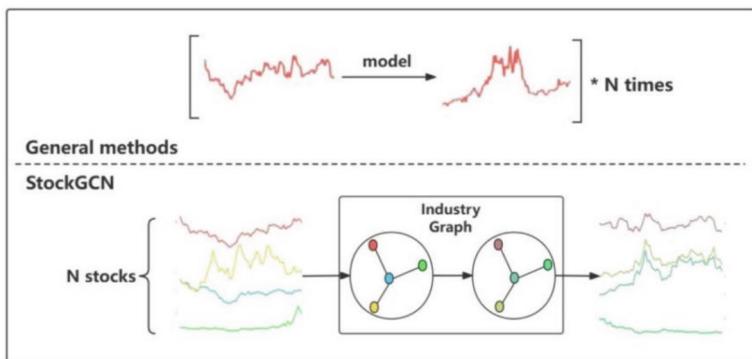


Fig. 1. Differences between StockGCN and General Stock Prediction Methods. General methods predict individual stocks separately without involving other stocks. StockGCN combines stock data into a graph. In fact, time series prediction is transformed into a graph prediction problem.

This work introduces a stock prediction model called StockGCN, which is a general framework for stock price prediction and can be easily optimized and extended. StockGCN employs deep semi-supervised learning algorithms to generate a model for predicting stock prices in the next time step. This work also proposes an effective graph construction method based on sector rotation effect and lead lag effect, which can be effectively constructed with small dataset size. Specifically, this work combines prior financial knowledge to create a predefined adjacency matrix. This graph, based on the industry's upstream and downstream relationships, forms a chain-like network by

connecting companies in different sectors through supply relationships. By constructing the Industry graph, StockGCN transforms the stock price prediction into a graph-based time series prediction task. This allows StockGCN to treat multiple stocks as a whole for prediction rather than predicting single stocks in isolation. Figure 1 demonstrates these distinctions. Moreover, the model incorporates real-world industry supply chain relationships and data correlations in order to more accurately depict real-world scenarios.

2 Related Work

2.1 Stock Price Prediction

Stock price prediction is an application field of time series forecasting. In the stock market, stock prices can be viewed as time series data that varies over time.

In recent years, numerous deep learning approaches have been advanced to extract valuable features from diverse information sources, aiming to improve prediction accuracy. For example, methods that combine text information, Causal Weight Adjustment (CWA) and attention mechanisms [13]. Combining Deep Neural Networks (DNN) and Adaptive Regression Splines (MARS) to predict closing prices has yielded promising results [1]. The prediction model based on multi-graph convolutional neural networks has also shown remarkable effectiveness in stock prediction tasks, which defines shareholding graph, industry graph and topicality graph. The model adjusts on the basis of the basic propagation layer, combining predefined types of graphs and incorporating them into the model for multi-graph convolution. This convolutional approach endows the model with enhanced performance and scalability.

However, most of the studies [2, 5, 7, 12] focus on integrating historical stock prices with other information while ignoring the interactions between stocks. This work proposes a stock prediction model based on Graph Convolutional Network (GCN) and introduces a method of constructing a graph based on prior knowledge. By performing message-passing and feature aggregation on the graph, the model can learn the cross-effects between stocks, thereby improving the accuracy of predictions.

2.2 Graph Convolutional Network

Graphs can represent the relationships between stocks. GCN has gained increasing popularity due to its outstanding performance in node classification tasks [11].

GCN has achieved great success in addressing spatial dependencies between entities in networks. It operates under the assumption that the state of a node is influenced by the state of its neighboring nodes.

Researchers have recently applied Graph Convolutional Networks (GCN) to stock prediction, addressing the spatial dependencies between stocks. Traditional models based on company shareholding graphs fall short due to sparse cross-shareholding relationships. This study uses an industry graph to better reflect the underlying dynamics of price fluctuations, thereby enhancing model performance.

3 Problem Formulation

Let $Z_t \in R_N$ denote the value of a multivariate variable of dimension N at time step t , where $Z_t[i] \in R_N$ denote the value of the i^{th} variable at time step t . Given a sequence of historical p time steps of a variable, $X = \{Z_{t_1}, Z_{t_2}, \dots, Z_{t_p}\}$, the goal is to predict the value of next time stamp $Y = \{Z_{t_{p+1}}\}$. A mapping $f(\cdot)$ from X to Y can be established by minimizing the loss.

Here are the formal definitions of some concepts related to graphs:

Definition(3.1)(Graph). Graph is denoted as $G = (V, E)$, where E is the set of edges, and $V = v_1, v_2, \dots, v_p$ is the set of nodes. N represents the number of nodes in a graph.

Definition(3.2)(Adjacency Matrix). Adjacency matrix is a matrix that represents the adjacency relationship between vertices, denoted as $A \in R_{N \times N}$ with $A_{i,j} > 0$ if $(V_i, V_k) \in E$ and $A_{i,j} = 0$ if $(V_i, V_k) \notin E$.

4 StockGCN

The overall process of StockGCN is to convolve the preprocessed graphs and data for prediction (Fig. 2). The model framework mainly includes a predefined adjacency matrix, graph convolution layers, and residual connections. To effectively discover hidden correlations between nodes, it is necessary to construct a special adjacency matrix. This work utilizes the Spearman's coefficient to construct a multi-sector industry graph. The resulting adjacency matrix is then utilized as input for all graph convolution layers. Stacked graph convolution layers are able to expand the information fusion range to uncover deeper dependencies. To avoid gradient vanishing, the model introduces residual connections and aligns dimensions through fully connected layers. More detailed introductions of the model framework are shown in Sect. 4.2.

4.1 Graph Construction

The interrelationship of stock returns over time has long been confirmed in the stock market [14]. To capture the lead-lag effect of stocks and the sector rotation effect generated by stocks from different sectors, the industry graph $G_I = (V, E_I)$ can represent the relationships between stocks. Here, $|V| = N$ refers to the number of public companies(N), and $A = (a_{i,j})_{N \times N}$ is the adjacency matrix that represents connections in the stock network. The element $a_{i,j}$ indicates whether stocks are connected and quantifies the strength of these connections.

When a sector is relatively independent in the stock market, graph construction involves selecting all stocks within that sector. Stocks with similar historical price trends, indicating shared user demographics and policy sensitivities, are expected to maintain this similarity. This relationship can be quantified using Spearman's coefficient:

$$\rho = \frac{\sum_i^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_i^n \sqrt{(x_i - \bar{x})^2} \sum_i^n \sqrt{(y_i - \bar{y})^2}} \quad (1)$$

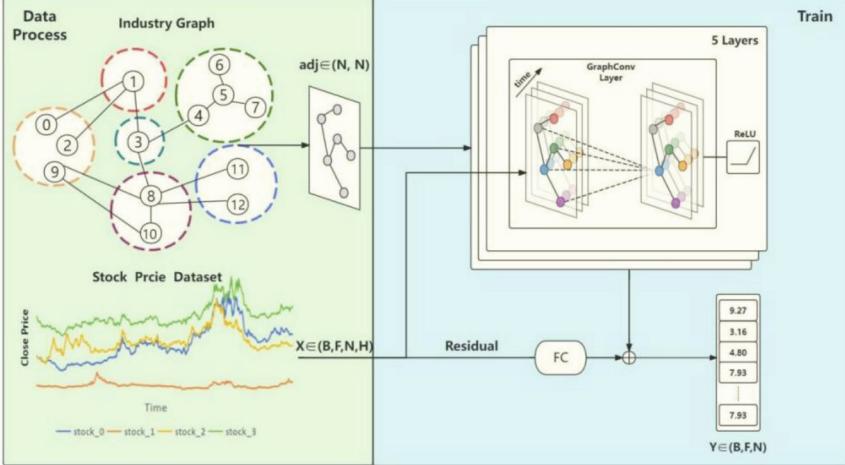


Fig. 2. Overall process of StockGCN. Industry Graph: $G_I = (V, E_I)$, using adjacency matrix $adj \in R_{N \times N}$ to store the relationships between companies. Each node with a number represents stock i , $i \in (0, \text{numberofnodes})$. The circles represent sectors. The connecting lines between the nodes represent the cooperative relationships between them. FC: fully connected layer.

We then performing the top-k operation on the correlation coefficient ρ to sparsize the adjacency matrix and highlight strongly correlated relationships. Additionally, the Spearman's coefficients should be normalized to the range $[0, 1]$.

In more common scenarios, however, stocks often span across multiple sectors, thereby weakening the impact of the lead-lag effect [8]. The main factor lies in sector rotation and the structure of the industrial chain. Fluctuations in the operational conditions of upstream enterprises will impact downstream enterprises. To address this general scenario, this work constructs an industrial graph $G_I = (V, E_I)$ to capture the effects of sector rotation.

$a_{i,j}$ can be caculated by the following formula:

$$a_{i,j} = \begin{cases} \rho & \text{if } n_i \leftrightarrow n_j, \\ 0 & \text{else} \end{cases} \quad (2)$$

where $a_{i,j}$ refers to the elements of adjacency matrix A , ρ refers to the Spearman's coefficient, and operator \leftrightarrow denotes the supply-demand relationship between two companies in the industrial chain.

4.2 Framework of StockGCN

Figure 3 showcases the framework of StockGCN. It consists of residual connections, stacked graph convolutional layers, and fully connected layers. By stacking graph convolutional layers, individual nodes can integrate information from more distant nodes, thus capturing deep spatial dependencies.

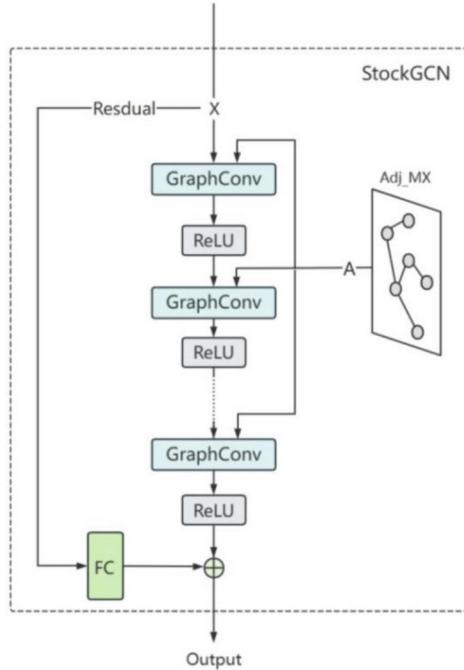


Fig. 3. The framework of StockGCN. FC: fully connected layer. Adj_MX: adjacency matrix of Industry graph. GraphConv: Graph convolutional layer. The stock historical data and adjacency matrix are input into the graph convolutional layer. Residual connections are introduced to prevent over-smoothing.

The input of StockGCN is the historical stock prices of all stocks. $Input = x \in [B, F, N, H] = [\text{batchsize}, \text{features}, \text{num_nodes}, \text{seq_length}]$. GraphConv layer utilizes convolution with a symmetrically normalized Laplacian matrix. $adj_mx \in [N, N]$ is the preprocessed adjacency matrix of the industry graph. The graph convolutional layer takes historical stock prices and the adjacency matrix of the industry graph as input for convolution. With each convolutional layer, the model can capture deeper spatial dependencies. Residual connections can prevent the vanishing gradient problem and improve training efficiency. Lastly, the model outputs the predicted stock price $y \in [B, F, N]$ for the next time step.

In general, the model can be summarized by the following formula:

$$X_t^{GCN} = \sigma(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} X_t W) \quad (3)$$

where $X_t^{GCN} \in R^{N \times F}$ refers to the price predicted by GCN at time t , $\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$ is the symmetric normalized Laplacian matrix, $X_t = [X_{t-1}, X_{t-2}, \dots, X_{t-his}] \in R^{his \times N \times F}$ refers to the set of features of all nodes at the previous history length moments. W represents all trainable parameters.

5 Experiments

5.1 Datasets

The model is validated on two stock datasets from CSMAR¹ and IFIND² respectively for the Chinese stockmarket, which are derived from the Shanghai Stock Exchange and Shenzhen Stock Exchange. Stock information is summarized at regular intervals.

CSMAR. The data consists of 1584 records for a total of 11 shanghai-listed stocks, which are classified into five different sectors: oil, investment and financing, new materials, household appliances and high-technology. The dataset covers a time span from 3/14/2022 to 4/29/2022.

IFIND. The data set comprises 19 shenzhen-listed stocks from the coal industry, encompassing coal mining, processing, and downstream power and heating sectors. Each stock includes 243 records spanning from 1/4/2021 to 12/31/2021.

5.2 Preprocessing

In order to verify the impact of industrial chain on stock volatility, we removed delisted stocks. CSMAR contains 11 stocks, while IFIND contains 19 stocks.

5.3 Baselines

The model is compared with the following baselines:

GRU [4]. Gated Recurrent Unit network, a special RNN model.

CNN [16]. In addition to processing images, CNN can also be used to predict time series data through the perceptual ability of convolutional kernels.

LSTM [17]. Long Short Term Memory network, a special RNN model.

RGSL [18]. A regularized graph structure learning model with semantic knowledge for multivariate time series forecasting.

5.4 Result Analysis

To validate the model's superiority, the following questions need to be addressed:

1. Has incorporating the graph structure into stock prediction led to improved performance in contrast to conventional models that consider stocks as independent entities?
2. Does StockGCN outperform the baseline models?
3. Does the model validate the interconnections among stocks?
4. How does the length of historical information affect the accuracy?

¹ <https://data.csmar.com/>

² <http://ft.10jqka.com.cn/>

The model carried out a significant number of experiments and obtained results from two actual datasets. Figure 4 and Table 2 answer questions 1 and 2. Figure 5 addresses question 3. Table 3 addresses question 4.

Benefits of Using Graphs. This study uses spatial topologies to model stock interdependencies and employs graph neural networks to enhance stock predictions by updating node representations through graph convolutions. As the network deepens, its perceptual scope increases, improving predictive performance over non-graph models. As shown in Table 2 and Fig. 4, StockGCN outperforms conventional models like CNN and LSTM, and also exceeds the RGSL model (Table 1).

Table 1. Specific information about the two stock datasets.

Dataset	CSMAR	IFIND
Type	Time series data	Time series data
Number of Stocks	11	19
Length	1584	243
Total Records	17424	4617
Time Interval	15 min	24 h
Time Span	14/3/2022 - 29/4/2022	4/1/2021 - 31/12/2021

Table 2. Performance comparison of different models on the datasets.

Model	CSMAR		IFIND	
	MAE	RMSE	MAE	RMSE
GRU [4]	0.83	1.32	1.23	3.19
CNN [16]	0.66	1.38	1.00	2.67
LSTM [17]	0.73	1.65	1.64	3.84
RGSL [18]	0.89	1.35	2.04	6.69
StockGCN	0.63	1.15	0.94	2.25

The experimental results mentioned above clearly demonstrate that incorporating the spatial topological structure can significantly improve the performance.

Interconnections Among Stocks. In the IFIND dataset, stock 0, upstream in the supply chain and historically 92% similar to stock 3, reacts preemptively to factors like policies and epidemics, influencing stock 3's prices. StockGCN experiments confirm this, showing in Fig. 5 that fluctuations in stock 0 between days 7 and 25 are reflected in stock 3's prices. StockGCN accurately predicts these trends, especially on days 9 to 13 and 18 to 22, demonstrating significant interactions between the stocks that impact their prices.

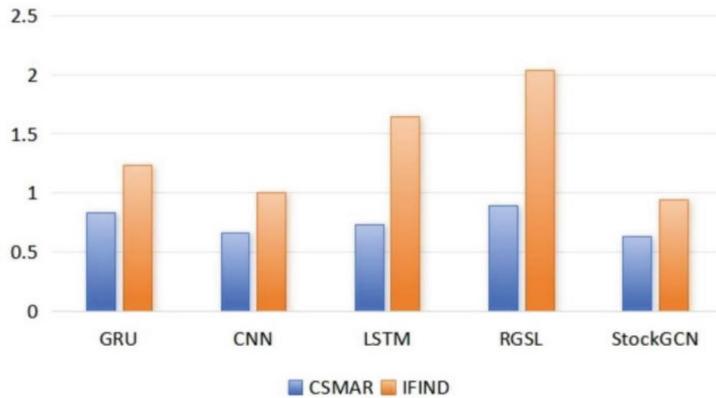


Fig. 4. MAE on datasets. On both datasets, StockGCN achieved the best predictive performance.

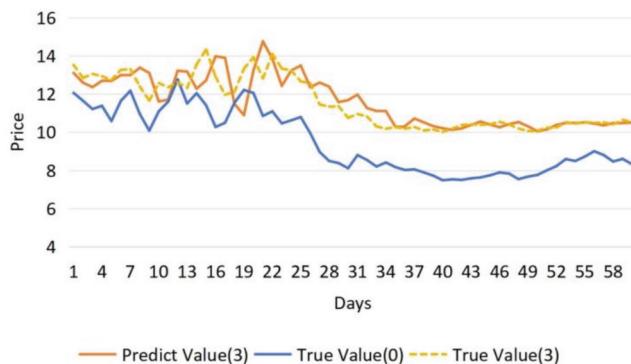


Fig. 5. Prices of stock 0 and stock 3. In the first 60 days, the price changes of the two stocks reflect the cross-effects between them.

Table 3. Performance of StockGCN with different history length.

Length	CSMAR			IFIND		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE
5	0.54	0.0170%	1.03	1.67	0.090%	4.43
6	0.44	0.0138%	0.84	0.61	0.040%	1.75
7	0.42	0.0132%	0.79	0.57	0.038%	1.59
8	0.36	0.0113%	0.69	0.49	0.037%	1.35
9	0.42	0.0132%	0.77	0.65	0.040%	1.87
10	0.66	0.0208%	1.20	0.83	0.057%	2.01

The Impact of History Length. The model was experimented with different history lengths, specifically [5–10] previous days. As shown in Table 3, the model achieved optimal performance at days = 8. When adopting such a history length, StockGCN obtained MAE = 0.36, RMSE = 0.69, MAPE = 0.0113% on the CSMAR dataset and MAE = 0.49, RMSE = 1.35, MAPE = 0.037% on the IFIND dataset. In addition, the predictive performance gradually declined as the history length decreased or increased. Hence, the length of the historical information significantly affects the predictive performance.

5.5 Ablation Study

Based on the different components, StockGCN is divided into the following categories:

- w/o Residual: StockGCN without residual connection.
- fewer-layers: StockGCN with 2,3 and 4 graph convolution layers.
- more-layers: StockGCN with 6 and 7 graph convolution layers.

Numerous studies have shown that Graph Convolutional Networks (GCN) can perform well with just a few layers, but generally, increasing the number of layers within a reasonable range can improve performance. Additionally, residual connections help maintain effective training. In this study, ten experiments with 1000 epochs each were conducted to determine the optimal layer number, confirming the benefits of residual connections. The ablation study (Table 4) indicates that a 5-layer network is effective, and omitting residual connections reduces performance.

Table 4. The performance of various ablation methods on different datasets.

Layers	CSMAR			IFIND		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE
w/o Res	3.30	0.11%	5.28	1.84	0.10%	4.81
2-layers	0.59	0.02%	1.12	0.95	0.06%	2.62
3-layers	0.49	0.12%	0.91	5.48	0.48%	11.99
4-layers	0.39	0.01%	0.73	0.98	0.05%	2.55
6-layers	1.87	0.06%	2.98	6.16	0.45%	11.58
7-layers	0.63	0.02%	1.11	3.90	0.21%	9.91
StockGCN	0.44	0.01%	0.84	0.61	0.04%	1.75

6 Conclusion

This work demonstrates that taking into consideration the cross-effects between stocks can effectively enhance prediction accuracy because the price movement of an individual stock is inevitably influenced by other related stocks. The contribution of the model lies

in proposing a graph construction method based on cross-effects between stocks. It uses prior knowledge to select nodes and quantifies the interrelationship between stocks with the Spearman's coefficients. In the case of a small-scale dataset, this method enables more accurate predictions by using fewer nodes to simulate market behavior. The model can be further optimized by acquiring more diverse node features and integrating additional external factors.

Acknowledgements. This work was supported by the Natural Science Foundation of Tianjin (No. 21JCYBJC00640) and by the 2023 CCF-Baidu Songguo Foundation (CCF-BAIDU 202311).

References

1. Bose, A., Hsu, C.H., Roy, S.S., Lee, K.C., Mohammadi-Ivatloo, B., Abimannan, S.: Forecasting stock price by hybrid model of cascading multivariate adaptive regression splines and deep neural network. *Comput. Electr. Eng.* **95**, 107405 (2021)
2. Chen, C., Zhao, L., Bian, J., Xing, C., Liu, T.Y.: Investment behaviors can tell what inside: Exploring stock intrinsic properties for stock trend prediction. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2376–2384 (2019)
3. Chen, Y., Wei, Z., Huang, X.: Incorporating corporation relationship via graph convolutional neural networks for stock price prediction. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 1655–1658 (2018)
4. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint [arXiv:1406.1078](https://arxiv.org/abs/1406.1078) (2014)
5. Ding, X., Zhang, Y., Liu, T., Duan, J.: Deep learning for event-driven stock prediction. In: Twenty-Fourth International Joint Conference on Artificial Intelligence (2015)
6. Feng, F., Chen, H., He, X., Ding, J., Sun, M., Chua, T.S.: Enhancing stock movement prediction with adversarial training. arXiv preprint [arXiv:1810.09936](https://arxiv.org/abs/1810.09936) (2018)
7. Hoseinzade, E., Haratizadeh, S.: Cnnpred: Cnn-based stock market prediction using a diverse set of variables. *Expert Syst. Appl.* **129**, 273–285 (2019)
8. Hou, K.: Industry information diffusion and the lead-lag effect in stock returns. *Rev. Financ. Stud.* **20**(4), 1113–1138 (2007)
9. Jiang, F., Liu, H., Yu, J., Zhang, H.: International stock return predictability: the role of us uncertainty spillover. *Pac. Basin Financ. J.* **82**, 102161 (2023)
10. Kim, R., So, C.H., Jeong, M., Lee, S., Kim, J., Kang, J.: Hats: A hierarchical graph attention network for stock movement prediction. arXiv preprint [arXiv:1908.07999](https://arxiv.org/abs/1908.07999) (2019)
11. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint [arXiv:1609.02907](https://arxiv.org/abs/1609.02907) (2016)
12. Liu, H., Song, B.: Stock trends forecasting by multi-layer stochastic ann bagging. In: 2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI), pp. 322–329. IEEE (2017)
13. Liu, Z., Zhang, Q., Huang, D., Wu, D.: Stprformer: a stock price prediction model based on convolutional attention mechanism. In: International Conference on Intelligent Computing, pp. 433–444. Springer (2023)
14. Lo, A.W., MacKinlay, A.C.: When are contrarian profits due to stock market overreaction? *Rev. Financ. Stud.* **3**(2), 175–205 (1990)
15. Ma, Y., Mao, R., Lin, Q., Wu, P., Cambria, E.: Multi-source aggregated classification for stock price movement prediction. *Inf. Fusion* **91**, 515–528 (2023)

16. Mehtab, S., Sen, J.: Stock price prediction using convolutional neural networks on a multivariate timeseries. arXiv preprint [arXiv:2001.09769](https://arxiv.org/abs/2001.09769) (2020)
17. Selvin, S., Vinayakumar, R., Gopalakrishnan, E., Menon, V.K., Soman, K.: Stock price prediction using LSTM, RNN and CNN-sliding window model. In: 2017 International Conference on Advances in Computing, Communications and Informatics (icacci), pp. 1643–1647. IEEE (2017)
18. Yu, H., et al.: Regularized graph structure learning with semantic knowledge for multi-variate time-series forecasting. arXiv preprint [arXiv:2210.06126](https://arxiv.org/abs/2210.06126) (2022)



Coformer: Collaborative Transformer for Medical Image Segmentation

Yafei Gao^{1,2}, Shichao Zhang^{1,2}, Dandan Zhang³, Yucheng Shi³, Guohua Zhao⁴,
and Lei Shi^{1,2(✉)}

¹ School of Cyber Science and Engineering, Zhengzhou University, Zhengzhou 460002, China
{yfgao, shilei}@zzu.edu.cn

² SongShan Laboratory, Zhengzhou 450001, China

³ School of Computer and Artificial Intelligence,
Zhengzhou University, Zhengzhou 450001, China
zhang520422@gs.zzu.edu.cn

⁴ Department of Magnetic Resonance Imaging, The First Affiliated Hospital of Zhengzhou
University, Zhengzhou 450004, China

Abstract. Transformer has shown significant power for medical image analysis. However, the inherent design of the Transformer limits the ability to extract local features, thereby potentially affecting the performance. To address the above limitations, a Collaborative Transformer (Coformer) is proposed for medical image segmentation. In detail, the Multiscale Representation Fusion (MRF) module is designed to extract the semantic information of the fused features. During the encoding phase, local and global multi-scale feature representations are extracted by incorporating with Swin Transformer. Then, the semantic features are deeply extracted by the MRF module based on the cross-attention mechanism in the decoding phase. Comparative experiments on the well-known public Synapse Multi-Organ Segmentation dataset have demonstrated that Coformer achieves 82.39% dice score with 1.7%-12.62% improvements over the state-of-the-art methods.

Keywords: Transformer · Medical image segmentation · Cross attention

1 Introduction

Vision Transformer (ViT) [1] is shown to be competitive in processing images. It has enabled Transformers to exhibit impressive performance in computer vision [2–5]. Consequently, using Transformers to tackle the issue of long-dependency feature extraction in medical image segmentation tasks has become mainstream. For instance, TransUNet [5] employs a method that initially uses the convolutional neural network (CNN) during the encoding phase to extract local features. It then utilizes the Transformer to extract long sequence features from the output. During the decoding phase, it adopts a decoding structure similar to U-Net, leveraging upsampling for multi-scale feature fusion. The approach has produced significant results on abdominal multi-organ segmentation

datasets and cardiac automatic diagnosis challenge datasets. Similarly, Fat-Net [6] introduces a dual-branch encoding structure that harnesses Transformer and CNN for feature extraction. It also designs an efficient Feature Adaptive Module (FAM) for the decoding phase, facilitating more efficient multiscale feature fusion. The method has proven highly successful on ISIC 2017 [7] and ISIC 2018 [8]. HiFormer [9], during the encoding phase, inputs the feature output of each layer into the corresponding Swin Transformer [10] for more efficient feature extraction. It also designs a Double-Level Fusion module (DLF) that uses the Transformer for long-range dependencies extraction after the mixed module outputs the features. The module employs cross-attention [11] for extraction, enhancing the efficiency of multi-scale feature extraction. While the above methods mitigate the issue of a single mechanism’s inability to simultaneously extract local and global features, they still face limitations: Multi-scale feature correlation is not used for semantic information extraction.

In this paper, we introduce a novel network structure, Coformer, which is based on the encoder-decoder structure. The network utilizes the correlation of semantic information between multi-scale features to extract and fuse image features efficiently, enhancing the precision of medical image segmentation tasks. The encoding phase of our model comprises three modules: a CNN module, a Swin Transformer module, and an MRF module. The MRF module receives multi-scale features from the outputs of the CNN and Swin Transformer modules. MRF is designed to extract and fuse the relevant semantic information between multi-scale features through the cross-attention mechanism. Finally, it outputs three different scale-calibrated feature maps to the decoder to produce the final segmentation map. It outperforms most mainstream models across various indicators.

The main contributions are as follows:

- Coformer, as a novel segmentation model, has addressed the limitation of traditional single-mechanism methods that are only effective for single-level features.
- MRF is proposed as a Multi-scale Representation Fusion module based on the cross-attention mechanism. It uses the correlation between scale features for semantic information extraction to obtain better segmentation results.
- Experimental results on public medical image datasets demonstrate the effectiveness and superiority of Coformer compared to other mainstream methods.

The rest of the paper is organized as follows. Section 2 describes the related work of medical segmentation. In Sect. 3, detailed information about Coformer and its core modules is provided. Section 4 presents the experimental details and analysis of the results. Section 5 concludes the paper and proposes prospects.

2 Related Work

While CNN has achieved certain results in medical image segmentation tasks, the limitations lie in its inability to extract contextual information. On the other hand, models based on the attention mechanism of the Transformer have found success in the field of NLP due to their ability to handle long sequence dependencies. ViT has demonstrated the feasibility of using Transformers in image processing, leading to their widespread use in the field of medical image segmentation. For instance, Swin-Unet [12], inspired

by the symmetrical structure of U-Net, proposes a pure Transformer segmentation structure based on Swin Transformer. It fuses multi-scale features from the encoding phase during decoding through skip connections. TransUNet [5] first uses CNN to extract local features from images, then extracts contextual information from the output feature maps, and finally fuses multi-scale features from CNN during the decoding phase through skip connections. UNETR [13] uses Transformer to extract contextual information from 3D images during the encoding phase, and after convolution processing during the skip connection phase, it fuses and concatenates multi-scale features for decoding.

Despite the proposal of many hybrid methods of CNN and Transformer to address the issue of single-type feature extraction, these methods and models have not effectively fused the extracted features. To address the above issues, we propose a novel model based on cross-attention. The model can effectively fuse features extracted from different scales by CNN and Transformer and uses a novel fusion mechanism to further extract effective features, enhancing the performance of 2D medical image segmentation.

3 Method

In this section, we present an overview of Coformer. As depicted in Fig. 1, it employs an end-to-end strategy. It begins by extracting features from the input image using two modules: the Swin Transformer module for contextual information extraction and the CNN module for local feature extraction. The outputs from these modules are then fed into a three-layer fusion module (MRF) to acquire a more effective feature representation. Finally, the representation is decoded to produce the final result.

3.1 Encoder

As depicted in Fig. 1, our proposed encoder is composed of three parts: the Swin Transformer, CNN, and a designed MRF. Given that CNN and Transformer specialize in extracting local and global features, we employ a hybrid approach that extracts both types of features from the image simultaneously. Both the CNN and Transformer modules are structured into three layers. The features of the CNN layers are fused with the corresponding features of the Swin Transformer layers using skip connections to produce the output for this layer. These outputs, classified as P^s , P^m , and P^l based on the scale of the feature representation, serve as inputs for the MRF module. To minimize the loss of semantic information, we process the image using a single convolution in the first module, the output of which is then fed into the Swin Transformer. The output from the first Swin Transformer layer is used as the input for the second layer, enabling the Swin Transformer to extract fine-grained features from the original image. The outputs from the second CNN layer and the second Swin Transformer layer are fused and then fed into the third Swin Transformer layer to extract image features.

In the first layer of the CNN module, a 1×1 convolution is applied to the output, generating a feature map with dimensions $h/4$, $w/4$, and a channel number of D' . It is fused with the output from the first layer of the Swin Transformer, resulting in the first layer output, P^l . The patch-merge operation is then employed, reducing the resolution of the Swin Transformer's output and simultaneously doubling the channel number to $2D'$.

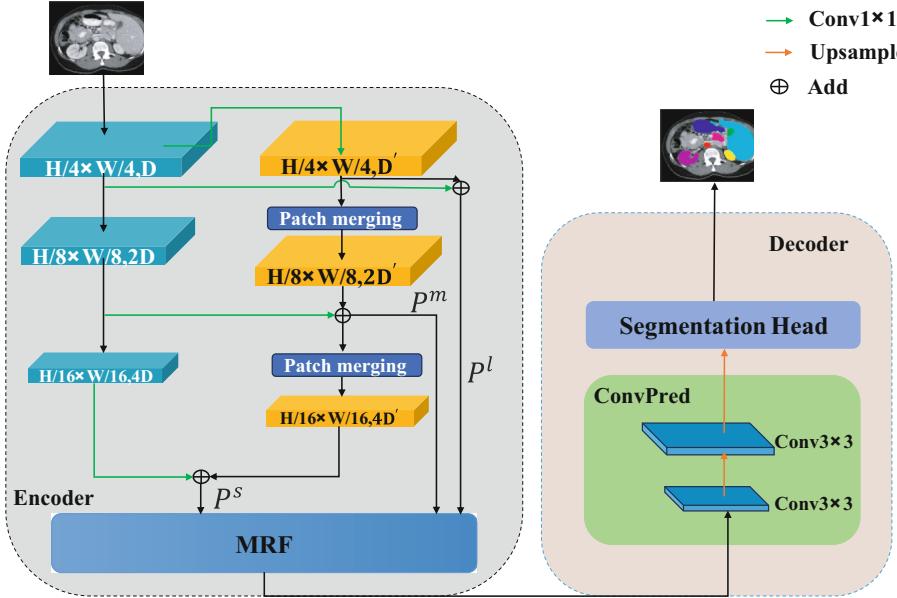


Fig. 1. The framework of Coformer. It consists of 2 modules: 1) The encoder module in the gray area, where cyan is the CNN module, yellow is the Swin Transformer module, and blue is the MRF module; 2) The decoder module in the salmon pink box, where the green module is composed of 2 CNN layers, and the blue is the segmentation head block. (Color figure online)

Then, the feature map from the CNN is fused with the Swin Transformer's feature map. The fused features are fed into the patch-merge operation, serving as the input for the third layer. Using similar fusion processes, the features from the CNN and Swin Transformer are combined to yield P^m and P^s , representing the intermediate and low-level features, respectively.

3.2 Multi-scale Representation Fusion Module (MRF)

Efficiently fusing multi-scale features while extracting semantic information through the correlation between scale features poses an inevitable challenge. The conventional method directly upsamples and fuses features extracted at the same scale. However, due to ignoring the correlation of semantic information between scale features, the inability to effectively combine multi-scale features becomes the bottleneck of the model. To address the above, we propose a novel MRF module. As shown in Fig. 2(b). The module takes as inputs the features output at three levels smallest (P^s), medium (P^m), and largest (P^l), and employs a cross-attention mechanism for hierarchical feature fusion.

Feature maps at different scales possess varying characteristics. Typically, large-scale feature maps contain general features and limited semantic information. In contrast, small-scale feature maps are rich in advanced features and semantic information. Direct interactions between these different feature maps may have little effect because of their significant level differences. It is evident in the proposed DLF in HiFormer [9], where

only the largest and smallest scale feature maps are used for interaction. To address the above, We propose a method for hierarchical fusion using correlations at different scales. In the MRF module, global average pooling is applied to feature maps at each level. It produces a cls token for each level, recorded as cls^s , cls^m and cls^l .

$$cls = \frac{1}{n} \sum_{i=1}^n \frac{x_i - \frac{\sum_{j=1}^n x_j}{n}}{\sqrt{\frac{\sum_{k=1}^n (x_k - \frac{\sum_{l=1}^n x_l}{n})^2}{n-1}} + \epsilon} \quad (1)$$

where x_i , x_j , x_k and x_l respectively represent the i-th, j-th, k-th and l-th features of the feature sequence, n represents the total number of features in the sequence, ϵ ensures numerical stability.

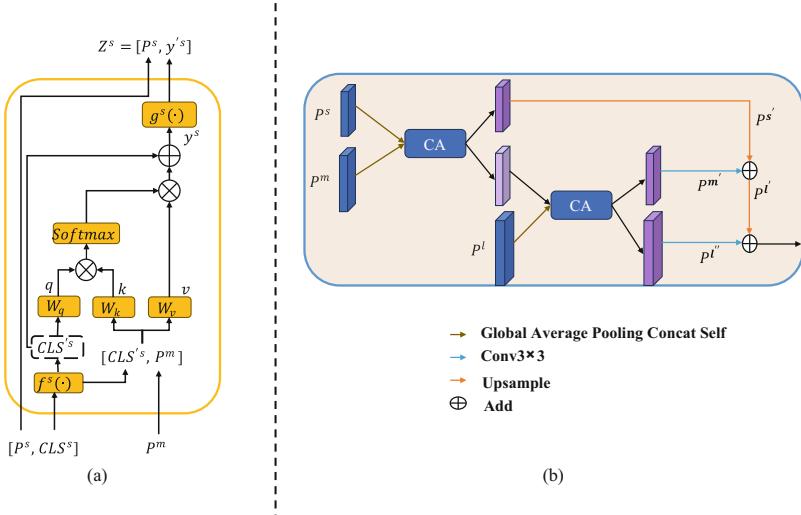


Fig. 2. (a) **The overview of the cross-attention mechanism.** The steps: 1) the low-level class token is projected to ensure dimension consistency before being concatenated to P^m ; 2) The resulting embedding acts as both a key and a value, while the projected CLS'^S functions as a query; 3) Z^S is obtained through inverse projection and concatenation with P^S . (b) **The overview of MRF module where.** CA is composed of 2 cross attention blocks.

We employ a cross-attention module to fuse the features of two adjacent levels. Specifically, we initiate the fusion process by swapping the cls tokens of two levels. It involves exchanging the position of a cls token from one level with the cls token from another level. We map and concatenate with the token of the other level and input it into the cross-attention module for further feature fusion extraction. Swapping and concatenating cls tokens across adjacent levels proves to be an efficient strategy for fusing feature information.

The cross-attention operates as depicted in Fig. 2. Initially, the cls token from a lower level is projected to a higher dimension, resulting in CLS'^S . The CLS'^S is then

concatenated with the embedding from a higher level to form a new token. The new token acts as the key and value inputs for the module, while CLS^S serves as the query. The final output can be expressed as follows:

$$Q = f^s(\text{cls}^s)W_q, K = (f^s(\text{cls}^s) \odot P^m)W_k, V = (f^s(\text{cls}^s) \odot P^m)W_v \quad (2)$$

$$y^s = \text{softmax}\left(\frac{QK^T}{\sqrt{\frac{C}{h}}}\right)V \oplus f^s(\text{cls}^s) \quad (3)$$

$$Z^s = (P^s \odot g^s(y^s)) \quad (4)$$

where $f^s(\cdot)$ and $g^s(\cdot)$ are the projection and back projection functions respectively to align the dimensions, W_q, W_k and $W_v \in R^{c \times (c/h)}$ are learnable weights, C and h are the embedding dimension and number of heads, \odot is the concatenation operations, \oplus is the residual connection operation.

The cross-attention module (CA) consists of two cross-attention blocks. We designed a method that fuses the output features of three levels P^s , P^m , and P^l into a feature. The process begins with the receipt of three scales of feature maps (P^s , P^m , and P^l) from the CA module. The P^s ($h/16, w/16$) is fed into a ConvUp module, which applies convolutions with kernel sizes of 3x3, bilinear upsampling, group normal [14], and relu to achieve a resolution of ($h/8, w/8$), denoted as $P^{s'}$. The P^m is processed through a Conv module that uses convolutions with kernel sizes of 3x3, group normal, and relu to maintain the resolution at ($h/8, w/8$), denoted as $P^{m'}$. After the fusion of $P^{s'}$ and $P^{m'}$, they are input into a ConvUp module to attain a resolution of ($h/4, w/4$), denoted as $P^{l'}$. The P^l is then input into a convolution with kernel sizes of 3x3 module to maintain the resolution at ($h/4, w/4$), denoted as $P^{l''}$. $P^{l'}$ and $P^{l''}$ are fused as the final output of MRF.

3.3 Decoder

Following the fusion of $P^{l'}$ and $P^{l''}$, they are passed through a ConvPred module, which employs a convolution with kernel sizes of 3×3 , bilinear upsampling, and relu to obtain the final feature map with a resolution of (h, w). The final feature map is then input into the segmentation head, which employs a convolution with kernel sizes of 3×3 to produce the final segmentation result map.

3.4 Loss Function

Because Multi-Organ Segmentation is a multi-classification problem, we choose the cross-entropy loss function as a component in the loss function; the L_{ce} is defined as:

$$L_{ce} = - \sum_{i=1}^C y_i \ln \frac{e^{z_i}}{\sum_{k=1}^C e^{z_k}} \quad (5)$$

where y denotes the true label value, C denotes the total number of classes and z is the prediction score.

To make our model converge faster and less prone to overfitting, we choose the Dice loss function as another component in our loss function, the L_{dice} is defined as:

$$L_{dice} = \frac{1}{C} \sum_{k=1}^C \left(1 - \frac{2 \sum_{i=1}^N y_i^k p_i^k + \delta}{\sum_{i=1}^N y_i^k + \sum_{i=1}^N p_i^k + \delta} \right) \quad (6)$$

where C denotes the number of classes, N denotes the number of samples, and y_i^k and p_i^k denote the label and predicted value of the i -th sample for the k -th class, respectively. The δ is introduced to prevent the denominator from becoming zero.

The final loss function is composed of cross-entropy loss and dice loss function, the L is defined as:

$$L = \lambda_1 L_{ce} + \lambda_2 L_{dice} \quad (7)$$

where λ_1 and λ_2 are set to 0.4 and 0.6 respectively.

4 Experiments

4.1 Dataset

The performance of the method was evaluated using the benchmark Synapse Multi-Organ Segmentation dataset [15]. The dataset comprises 30 cases, amounting to a total of 3779 axial clinical CT images of the abdomen. These cases were divided into two sets: 18 cases for training and 12 cases for testing. Each CT image was segmented into 85 ~ 198 slices of 512×512 pixels. The spatial resolution ranged from 0.54×0.54 to 0.98×0.98 , with a slice thickness between 2.5 mm and 5 mm.

4.2 Implementation Details

The method was implemented using PyTorch and trained on a single Nvidia RTX 3090 GPU with 24 GB of memory. We set the input image size to 224×224 and employed data augmentation techniques, such as random flipping and rotation, to enrich the training data. The model's optimizer was the SGD, with a momentum of 0.9 and weight decay of $1e-4$. We set the initial learning rate to 0.01 and the batch size to 10, and trained the model for a total of 400 epochs. Importantly, the CNN and Transformer modules of our model utilized weights that were pre-trained on ImageNet as their initial weights.

4.3 Evaluation Results

The performance of the model is evaluated using different evaluation metrics on the dataset. For the Synapse Multi-Organ Segmentation dataset, we employ the Dice Similarity Coefficient (DSC) and 95% Hausdorff Distance (HD) as our metrics. Furthermore, we benchmark the Coformer model against traditional deep learning methods and hybrid CNN-Transformer methods for a comprehensive comparison. The performance of Coformer is compared with other advanced methods in Table 1. The comparison is based on two key metrics, the average DSC and the average HD, across eight organs

Table 1. The results of different methods on the Synapse dataset. Here, DSC refers to the Dice Similarity Coefficient, and HD denotes the Hausdorff Distance. The abbreviations Gall, L-Kid, R-Kid, and Panc are used to represent the gallbladder, left kidney, right kidney, and pancreas, respectively. In this comparison, the best results are indicated in bold.

Methods	DSC↑	HD↓	Aorta	Gall	L-Kid	R-Kid	Liver	Panc	Spleen	Stomach
DARR [16]	69.77	-	74.74	53.77	72.31	73.24	94.08	54.18	89.90	45.96
U-Net [17]	76.85	39.70	89.07	69.72	77.77	68.60	93.43	53.98	86.67	75.58
Att-UNet [18]	77.77	36.02	89.55	68.88	77.98	71.11	93.57	58.04	87.30	75.75
R50 ViT [5]	71.29	32.87	73.73	55.13	75.80	72.20	91.51	45.99	81.99	73.95
TransUNet [5]	77.48	31.69	87.23	63.13	81.87	77.02	94.08	55.86	85.08	75.62
Swin-Unet [1]	79.13	21.55	85.47	66.53	83.28	79.61	94.29	56.58	90.66	76.60
LeViT-UNet [19]	78.53	16.84	78.53	62.23	84.61	80.25	93.11	59.07	88.86	72.76
DeepLabv3+ [20]	77.63	39.95	88.04	66.51	82.76	74.21	91.23	58.32	87.43	73.53
HiFormer-B [9]	80.39	14.70	86.21	65.69	85.23	79.77	94.61	59.52	90.99	81.08
Ours	82.39	17.51	87.13	71.34	86.40	81.12	94.88	66.27	90.37	81.63

Table 2. The impact of the MRF module on the Synapse dataset and the best results are highlighted.

Methods	MRF	DSC↑	HD↓	Aorta	Gall	L-Kid	R-Kid	Liver	Panc	Spleen	Stomach
Coformer		80.77	19.76	87.53	70.98	83.64	79.00	94.18	61.72	90.58	78.56
Coformer		82.39	17.51	87.13	71.34	86.40	81.12	94.88	66.27	90.37	81.63

in the Synapse Multi-Organ Segmentation datasets. Coformer outperforms traditional deep learning methods. When compared with U-Net, it shows a 5.54% increase in DSC and a 22.19% decrease in average HD. Compared to SwinUnet, it exhibits a 3.26% improvement in DSC and a 4.04% reduction in average HD. Even when compared with hybrid CNN-Transformer models, Coformer still leads. However, when compared with HiFormer, while the DSC increased by 2%, the HD increased by 2.81%. The increase in HD may be attributed to a lack of boundary extraction capability. In terms of organ segmentation, Coformer surpasses in DSC, particularly with organs such as the kidneys, liver, and pancreas. Figure 3 provides a visual comparison of the segmentation results from different methods for a given input image. Our method delivers significant segmentation results, even in complex scenarios. As shown in Table 1, Coformer leads other methods in terms of DSC. The results of ablation experiments for.

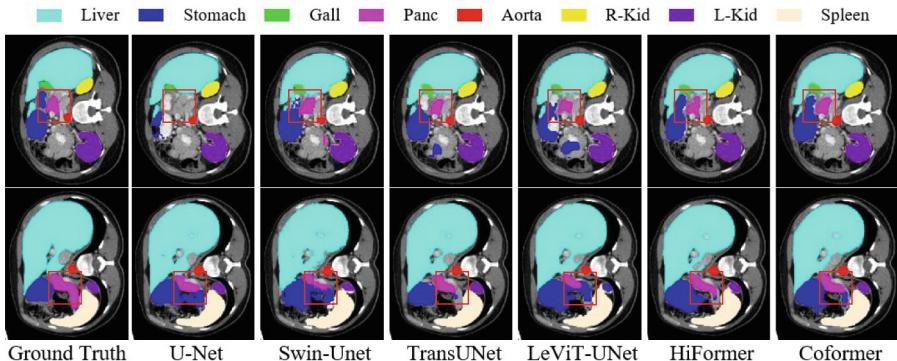


Fig. 3. Visualization results of different methods on Synapse Multi-Organ Segmentation dataset. The abbreviations Gall, L-Kid, R-Kid, and Panc are used to represent the gallbladder, left kidney, right kidney, and pancreas, respectively.

Table 3. On the Synapse dataset, compare the parameters and results of other mainstream methods; the best results are highlighted.

Methods	# Params (M)	DSC ↑	HD ↓
TransUNet [5]	105.28	77.48	31.69
Swin-Unet [1]	27.17	79.13	21.55
LeViT-UNet [19]	52.17	78.53	16.84
DeepLabv3+ [20]	59.50	77.63	39.95
HiFormer-S [9]	23.25	80.29	18.85
HiFormer-B [9]	25.51	80.39	14.70
HiFormer-L [9]	29.52	80.69	19.14
Coformer	25.53	82.39	17.51

MRF are in Table 2. The tabulated results confirm the superiority of the MRF module in fusing and extracting multi-scale features for medical image segmentation. As shown in Table 3, we compare the parameters of the current mainstream methods and coformer, as well as the performance results on the Synapse dataset.

5 Conclusion

In this research, we propose Coformer, a novel model based on the cross-attention mechanism for medical image segmentation. Differing from traditional methods, it adopts different strategies for different scale features when upsampling. We believe that learning with different strategies in upsampling surpasses a single unified strategy. The MRF module interacts with adjacent-level feature maps during upsampling to acquire more semantic information. Experimental results on the Synapse Multi-Organ Segmentation

dataset demonstrate the superior performance of Coformer compared to state-of-the-art methods. Although the method has impressive capabilities in medical image segmentation, it still has a large number of parameters. In future work, we aim to refine the model to enhance its computational efficiency by reducing its complexity.

Acknowledgements. This work was supported by the Nature Science Foundation of China (62006210), the Project of Joint Graduate-student Education Base in Henan Province (YJS2023JD04), the Key Project of Collaborative Innovation in Nanyang (22XTCX12001), the Research Foundation for Advanced Talents of Zhengzhou University (32340306), Pre-research Project of Songshan Laboratory (YYJC022022001), and Supported Project by Songshan Laboratory (232102210154).

References

1. Dosovitskiy, A., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint [arXiv:2010.11929](https://arxiv.org/abs/2010.11929) (2020)
2. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European conference on computer vision, pp. 213–229. Springer (2020)
3. Zheng, S., et al.: Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6881–6890 (2021)
4. 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)
5. Chen, J., et al.: Transunet: transformers make strong encoders for medical image segmentation. arXiv preprint [arXiv:2102.04306](https://arxiv.org/abs/2102.04306) (2021)
6. Wu, H., Chen, S., Chen, G., Wang, W., Lei, B., Wen, Z.: Fat-net: feature adaptive transformers for automated skin lesion segmentation. Med. Image Anal. **76**, 102327 (2022)
7. Codella, N.C., et al.: Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (isbi), hosted by the international skin imaging collaboration (isic). In: 2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018), pp. 168–172. IEEE (2018)
8. Codella, N., et al.: Skin lesion analysis toward melanoma detection 2018: a challenge hosted by the international skin imaging collaboration (isic). arXiv preprint [arXiv:1902.03368](https://arxiv.org/abs/1902.03368) (2019)
9. Heidari, M., et al.: Hiformer: hierarchical multi-scale representations using transformers for medical image segmentation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 6202–6212 (2023)
10. 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)
11. 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, pp. 357–366 (2021)
12. 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)
13. Hatamizadeh, A., et al.: Unetr: transformers for 3d medical image segmentation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 574–584 (2022)

14. Wu, Y., He, K.: Group normalization. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 3–19 (2018)
15. Zhao, Y., Li, J., Hua, Z.: Mpsht: multiple progressive sampling hybrid model multi-organ segmentation. *IEEE J. Transl. Eng. Health Med.* **10**, 1–9 (2022)
16. Fu, S., et al.: Domain adaptive relational reasoning for 3d multi-organ segmentation. In: Medical Image Computing and Computer Assisted Intervention–MICCAI 2020: 23rd International Conference, Lima, Peru, October 4–8, 2020, Proceedings, Part I 23, pp. 656–666. Springer (2020)
17. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, Oc tober 5–9, 2015, Proceedings, Part III 18, pp. 234–241. Springer (2015)
18. Schlemper, J., et al.: Attention gated networks: Learning to leverage salient regions in medical images. *Med. Image Anal.* **53**, 197–207 (2019)
19. Xu, G., Zhang, X., He, X., Wu, X.: Levit-unet: make faster encoders with transformer for medical image segmentation. In: Chinese Conference on Pattern Recognition and Computer Vision (PRCV), pp. 42–53. Springer (2023)
20. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 801–818 (2018)



FRMM: Feature Reprojection for Exemplar-Free Class-Incremental Learning

Hao Wang¹ and Jing Chen^{1,2(✉)}

¹ School of Artificial Intelligence and Computer Science, Jiangnan University, Wuxi 214122, Jiangsu, China
chenjing@jiangnan.edu.cn

² Jiangsu Provincial Engineering Laboratory of Pattern Recognition and Computing
Intelligence, Jiangnan University, Wuxi Jiangsu 214122, China

Abstract. Class-incremental learning involves handling a sequence of classification tasks arriving chronologically, with each task containing different classes. The model must gradually expand its knowledge by learning these tasks successively to acquire the ability to classify all encountered classes. Unlike exemplar-based class-incremental learning (EBCIL) which allows storing some old samples, exemplar-free class-incremental learning (EFCIL) faces a more severe forgetting problem due to the complete prohibition on accessing old data. Some previous methods freeze the feature extractor after the initial stage to enhance the model's stability on the base classes, but this also leads to a larger distribution discrepancy between the feature vectors of base and incremental classes. We propose to insert a reprojection layer between the feature extractor and the classifier to project feature vectors onto a unified distribution, facilitating subsequent classification. Feature reprojection improves the performance of both linear and nearest class mean (NCM) classifier, but their outputs exhibit different biases. We treat them as different experts by ensembling their predictions to combine their strengths. Additionally, we introduce a momentum update to alleviate the linear classifier's inherent forgetting. Across three public datasets of varying scales, our method, named FRMM, outperforms others in most cases while requiring only 1/256 of the storage by the state-of-the-art method that saves a covariance matrix for each class. Code is available at <https://github.com/BlacknoSheep/CIL>.

Keywords: incremental learning · continual learning · catastrophic forgetting · ensemble learning

1 Introduction

Traditional classification models are trained on static datasets, limiting their abilities to the classes present during training [1]. However, new data is continuously generated in the open world, and when novel classes emerge, retraining the model on all available data is typically required to extend its classification capabilities, which is very costly. A solution is to finetune the model only on new data, known as class-incremental learning (CIL). The primary challenge in CIL is catastrophic forgetting [2], where the model

forgets previous knowledge during training due to a lack of supervision from old classes, degrading its performance on those classes. The simplest way to mitigate forgetting is to store exemplars for each old class [3, 4], but storage constraints and privacy concerns often preclude saving past data. Consequently, current research focuses on exemplar-free class-incremental learning (EFCIL) without storing any old samples [5].

Early incremental learning methods penalized changes to important weights via regularization or used knowledge distillation to reduce activation shifts [6, 7], but they are generally underperformed for class-incremental tasks. Some methods employed generative models like GAN [8] to synthesize old class samples [9, 10], avoiding direct exemplar storage. However, such models are difficult to train, inefficient, and face forgetting issues when learning to generate new classes [11]. Inspired by transfer learning, recent EFCIL methods freeze the feature extractor after the initial stage to preserve stability on base classes, then train a linear classifier with prototype augmentation or use the nearest class mean (NCM) classifier which requires no training [12–14]. With a frozen feature extractor, some methods use classifiers based on analytic learning [15], which can hold absolute memorization of previous knowledge [16–18]. These methods can maintain stable and high accuracy even in scenarios with a very large number of incremental learning tasks.

This work focuses on EFCIL with a frozen feature extractor, involving an initial stage with typically more classes (base classes) to provide a good start, followed by multiple incremental stages with few classes each (incremental classes) to assess continual learning capability [14]. Figure 1 shows the feature vector distributions extracted by the frozen feature extractor when trained on the first 50 classes or all 100 classes of CIFAR-100 [23]. Figure 1(a) shows training on classes 0–49, while Fig. 1(b) depicts the upper bound after training on all classes 0–99, the ultimate CIL goal. In Fig. 1(a), the base class feature vectors maintain a good distribution since the feature extractor is trained on them. The incremental classes also exhibit separability, explaining why previous frozen feature extractor methods are effective. However, compared to the upper bound in Fig. 1(b), the incremental class distributions are more scattered with increased class overlap, straining the classifier. In this scenario, the Euclidean distance-based NCM classifier handles base classes well but struggles with incremental classes. The linear classifier has more trainable parameters providing higher plasticity than NCM, but the large distribution discrepancy between base and incremental classes makes it difficult to accommodate both simultaneously. Based on this analysis, we propose the following improvements:

- (1) Inserting a reprojection layer after the feature extractor to optimize the incremental class distributions without disrupting the base class distributions, enhancing the separability of feature vectors and reducing classifier burden. Experiments show substantial performance gains for NCM and linear classifiers with feature reprojection.
- (2) NCM and linear classifier predictions often exhibit different biases, with NCM significantly more accurate on base classes indicating high stability, while linear classifiers have higher plasticity. Inspired by ensemble learning, we integrate NCM and linear classifiers as different experts by combining their outputs to leverage their strengths.

- (3) Although prototype augmentation reduces forgetting for the linear classifier, its accuracy on base classes still drops significantly when learning incremental classes. We propose a momentum update scheme for parameters related to old classes in the linear classifier to mitigate forgetting.
- (4) The proposed method FRMM achieves the highest or second-highest accuracy across multiple datasets while only requiring storage of prototypes and standard deviations, 1/256 of the state-of-the-art method which saves covariance matrices.

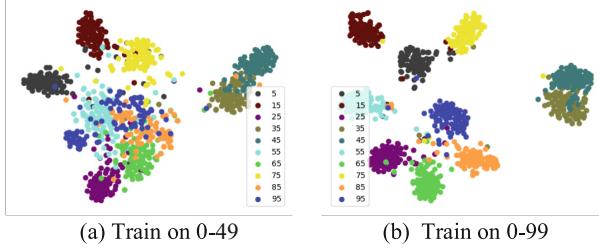


Fig. 1. Feature vector distributions extracted by the feature extractor after training on half or all CIFAR-100 classes, visualized by t-SNE [19].

2 Methods

2.1 Problem Statement and Framework

The CIL problem consists of a series of classification tasks $\{1, 2, \dots, T\}$. The model needs to learn these tasks sequentially. For task t , the dataset is denoted as D_t , and C_t is the classes it contains. In typical CIL settings, there is no overlap between different tasks, which means $C_i \cap C_j = \emptyset, i \neq j$. During the training of task t , data from previous $t - 1$ tasks $\bigcup_{i=1}^{t-1} D_i$ is not accessible. But during testing, the model needs to classify images from all the classes it has learned. The basic CIL model contains a feature extractor and a classifier. Specifically, the feature extractor F_θ , which is parameterized by θ , maps the input image $x \in \mathbb{R}^{H*W*3}$ to the feature vector $v \in \mathbb{R}^d$, where d is the output dimension of F_θ . The classifier H_φ , which is parameterized by φ , classifies v to the label $y_{pred} \in \bigcup_{i=1}^t C_i$. The aim of CIL is to minimize expected risk on all seen classes:

$$\underset{\theta, \varphi}{\operatorname{argmin}} \mathbb{E}_{(x, y) \sim D_1 \cup D_2 \cup \dots \cup D_t} I(y \neq F_\theta(H_\varphi(x))) \quad (1)$$

where \mathbb{E} denotes the expected risk, I is the indicator function which outputs 1 if the argument in it is true, otherwise it outputs 0.

The framework of our method FRMM is shown in Fig. 2. A projection layer is inserted between the feature extractor and the classifier. After passing through the reprojector, feature vectors become more separable for classification. There is an initial stage before incremental stages, which contains more classes. In the initial stage, the model is trained

on the base classes with a linear classifier, like general image classification models. After training, to reduce forgetting of old classes during incremental stages, we memory some information about the old classes. As an EFCIL method, FRMM does not store any samples. Instead, we memorize a prototype [12] and a standard deviation for each class, which can be computed by the following formula:

$$\boldsymbol{\mu}_y = \frac{1}{N_y} \sum_{i=1}^{N_y} F_\theta(\mathbf{x}_{y,i}) \quad (2)$$

$$\sigma_y = \sqrt{\frac{1}{N_y} \sum_{i=1}^{N_y} (F_\theta(\mathbf{x}_{y,i}) - \boldsymbol{\mu}_y) \circ (F_\theta(\mathbf{x}_{y,i}) - \boldsymbol{\mu}_y)} \quad (3)$$

where $\boldsymbol{\mu}_y$ and σ_y denotes the prototype and standard deviation of class $y \in C_t$, and $\mathbf{x}_{y,i}$ is a sample of class y . The operator \circ represents the Hadamard product.

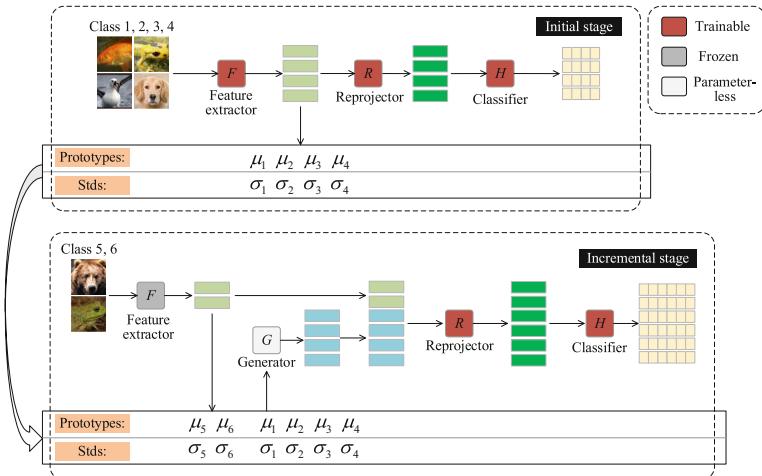


Fig. 2. The framework FRMM. The feature extractor will be frozen after the initial stage to ensure stability. The reprojector and the classifier are always trainable to ensure plasticity. During incremental stages, a parameterless generator is used to generate pseudo feature vectors of old classes learned in previous tasks.

$\boldsymbol{\mu}_y$ and σ_y have the same dimension as the feature vectors, so the space requirement is $O(d)$. Some methods store the covariance matrix instead of the standard deviation to capture a more precise distribution [14, 20, 21], but it requires $O(d^2)$ space, which is costly, even more than some EBCIL methods. In our case, storing standard deviation is sufficient. In incremental stages, the feature extractor is frozen, while the reprojector and the classifier remain trainable. The prototypes and standard deviations of feature vectors from new classes are calculated and stored before training. To mitigate forgetting, we use a generator to generate pseudo feature vectors for each old class. Like PASS [12], we generate by prototype augmentation:

$$\hat{\mathbf{v}}_{old} = \boldsymbol{\mu}_{old} + \sigma_{old} \circ \varepsilon \quad (4)$$

where $\boldsymbol{\epsilon} \sim N(0, \mathbf{I})$ is sampled from the standard Gaussian distribution.

2.2 Feature Reprojection

The goal of feature reprojection is to make the distribution of incremental classes more concentrated while not significantly altering the distribution of base classes. During inference, it is unknown which class a feature vector belongs to. Therefore, we propose a reprojection method that only requires the information from the feature vector itself:

$$\mathbf{v}_{RP} = \frac{\mathbf{v} - E[\mathbf{v}]}{\sqrt{Var[\mathbf{v}]} + \delta} \quad (5)$$

where $\mathbf{v} = \{v_1, v_2, \dots, v_d\}$ denotes the feature vector before the reprojection layer. $E[\mathbf{v}] = \frac{v_1 + v_2 + \dots + v_d}{d}$ is the mean of \mathbf{v} , and $Var(\mathbf{v}) = \frac{\sum_{i=1}^d (v_i - E[\mathbf{v}])^2}{d}$ is the standard deviation of \mathbf{v} . δ is a tiny value used to avoid the denominator being zero. To improve the plasticity, we use a set of learnable parameters to adjust the reprojected feature vectors:

$$\mathbf{v}_{RP} = \alpha \circ \mathbf{v}_{RP} + \beta \quad (6)$$

where α and β are learnable while training and they both have the same dimension as the feature vectors. The operator \circ represents the Hadamard product. The combination of Eq. 4 and Eq. 5 is formally equivalent to LayerNorm [22].

2.3 Ensemble of Two Experts

The NCM classifier performs classification based on Euclidean distance. We take the negative of the distance between the feature vector of a sample and the prototype as the logits for the NCM classifier.:

$$l_{x,y}^{ncm} = -(\mathbf{F}_\theta(\mathbf{x}) - \boldsymbol{\mu}_y)^T (\mathbf{F}_\theta(\mathbf{x}) - \boldsymbol{\mu}_y) \quad (7)$$

$$\mathbf{l}_x^{ncm} = \left\{ l_{x,1}^{ncm}, l_{x,2}^{ncm}, \dots \right\} \quad (8)$$

where $l_{x,y}^{ncm}$ denotes the NCM logit of sample \mathbf{x} and class y . For the linear classifier, its logits are computed by the following formula:

$$l_{x,y}^{linear} = \mathbf{W}^T \mathbf{x} + b \quad (9)$$

$$\mathbf{l}_x^{linear} = \left\{ l_{x,1}^{linear}, l_{x,2}^{linear}, \dots \right\} \quad (10)$$

where $\mathbf{W} \in \mathbb{R}^{d*c}$ and $\mathbf{b} \in \mathbb{R}^c$ are the weights and biases of the fully connected layer, c is the output dimension, i.e., the number of classes that need to be classified.

We treat the NCM classifier and linear classifier as two experts, and the ensemble of their logits can be computed as:

$$l_x^{esm} = \frac{Softmax(l_x^{ncm}) + Softmax(l_x^{linear})}{2} \quad (11)$$

where the Softmax is to convert these two logits to the same scale. Unlike other ensemble learning methods, these two experts share the same feature extractor, and the NCM classifier is parameterless, so the proposed ensemble scheme is zero-cost.

2.4 Momentum Update

The supervision for the old classes comes from the pseudo feature vectors generated by prototype augmentation, while the supervision for the new classes comes from real images, which is imbalanced. Therefore, we propose to update the parameters related to the old classes in the linear classifier with momentum:

$$\mathbf{w}_y = \begin{cases} m \cdot \mathbf{w}_y^{old} + (1 - m) \cdot \mathbf{w}_y^{new}, & y \in C_{0 \sim t-1} \\ \mathbf{w}_i^{new}, & y \in C_t \end{cases} \quad (12)$$

$$b_y = \begin{cases} m \cdot b_y^{old} + (1 - m) \cdot b_y^{new}, & y \in C_{0 \sim t-1} \\ b_y^{new}, & y \in C_t \end{cases} \quad (13)$$

where $\mathbf{w}_y \in \mathbf{W}$ and $b_y \in \mathbf{b}$ are the parameters related to class y , $C_{0 \sim t-1}$ is old classes from task $\{1, 2, \dots, t-1\}$ and C_t is new classes from the task t . The parameters related to new classes update without momentum, because they need more plasticity.

3 Experiments

3.1 Experimental Setup

Datasets. We conduct experiments on three public datasets: (1) CIFAR-100 [23], consisting of 100 classes. Each class has 500 training samples and 100 test samples. The image size is 32×32 pixels. (2) TinyImageNet [24], consisting of 200 classes. Each class has 500 training samples and 50 test samples. The image size is 64×64 pixels. (3) ImageNet-Subset [25], consisting of 100 classes. Each class has 1300 training samples and 50 test samples, with various sizes. We follow the classical EFCIL protocols to divide the dataset [14]. For CIFAR-100 and ImageNet-Subset, the protocols are: (1) $T = 5$, , 50 base classes and 10 incremental classes each task, (2) $T = 10$, , 50 base classes and 5 incremental classes each task, (3) $T = 20$, , 40 base classes and 3 incremental classes each task. For TinyImageNet, the protocols are: (1) $T = 5$, , 100 base classes and 20 incremental classes each task, (2) $T = 10$, , 100 base classes and 10 incremental classes each task, (3) $T = 20$, 100 base classes and 5 incremental classes each task.

Evaluation Metrics. The average accuracy is the widest used metric in CIL, it is computed as:

$$\bar{A} = \frac{1}{T} \sum_{t=0}^T A_t \quad (14)$$

where A_t is the accuracy after learning task t , including the initial stage $t = 0$. Another metric is the accuracy curves over all tasks, which show the accuracy evolution during the whole CIL process. Due to the large number of classes contained in the base class, the overall accuracy of the model is greatly influenced by the base class. Therefore, we also compared the last accuracy of the model on base classes and on incremental classes after completing the entire incremental learning process. Due to the significantly higher storage space required for the covariance-based method compared to other methods, for fair comparison, we also evaluated the storage requirement.

Implementation Details. All experiments are run on an RTX 4060Ti 16G, implemented with PyCIL [26] framework. Same as the methods we compare, ResNet18 [27] is used as the feature extractor. We use SGD optimization for training. For the initial stage, the learning rate is 0.1 and the weight decay of 0.0001, with 250 epochs. For incremental stages, the learning rate is 0.05 and the weight decay is 0.0001, with 50 epochs. The batch size is set to 128 for CIFAR-100 and TinyImageNet, and 384 for ImageNet-Subset. The momentum for updating linear classifier is set to 0.99.

3.2 Results

We compare our method FRMM with other EFCIL methods. Note that FeCAM [14] has two settings: one stores a single covariance matrix sharing by all classes, we mark it as FeCAM-s, and another stores a separate covariance matrix for each class, we mark it as FeCAM-e. It should be pointed out that saving covariance matrices requires a large amount of storage space. For FeCAM-e, the storage requirement is hundreds of times larger than methods that only save prototypes or standard deviation, like PASS [12], SSRE [28], FeTrIL [13] and FRMM. Even for FeCAM-s, its storage requirement is still several times that of other methods. Besides the compared methods, a joint training on all data is presented as a reference, which is the upper bound of CIL.

Table 1. Average top-1 accuracy (%) of EFCIL methods with different numbers of incremental tasks in CIFAR-100 and TinyImageNet. Best results - in **bold**, second best - underlined.

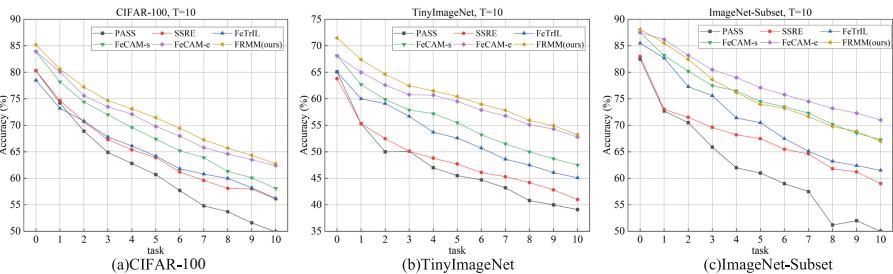
Methods	CIFAR-100			TinyImageNet		
	T = 5	T = 10	T = 20	T = 5	T = 10	T = 20
EWC [6]	24.5	21.2	15.9	18.8	15.8	12.4
LwF-MC [3]	45.9	27.4	20.1	29.1	23.1	17.4
DeeSIL [29]	60.0	50.6	38.1	49.8	43.9	34.1
LUCIR [30]	51.2	41.1	25.2	41.7	28.1	18.9
MUC [31]	49.4	30.2	21.3	32.6	26.6	21.9
PASS [12]	63.5	61.8	58.1	49.6	47.3	42.1
IL2A [20]	66.0	60.3	57.9	47.3	44.7	40.0
SSRE [28]	65.9	65.0	61.7	50.4	48.9	48.2
PRAKA[32]	70.0	68.9	65.9	53.3	52.6	49.8
FeTrIL [13]	66.3	65.2	61.5	54.8	53.1	52.2
FeCAM-s [14]	68.8	68.6	<u>67.4</u>	56	55.7	55.5
FeCAM-e [14]	<u>70.9</u>	<u>70.8</u>	69.4	<u>59.6</u>	<u>59.4</u>	<u>59.3</u>
FRMM (ours)	72.5	72.0	69.4	61.3	60.8	60.3
Upper bound	81.4	81.4	82.3	69.0	69.4	69.4

Table 2. Average top-1 accuracy (%) of EFCIL methods with different numbers of incremental tasks in ImageNet-Subset. Best results - in **bold**, second best - underlined.

Methods	ImageNet-Subset		
	T = 5	T = 10	T = 20
DeeSIL [29]	67.9	60.1	50.5
LUCIR [30]	56.8	41.4	28.5
PASS [12]	64.4	61.8	51.3
SSRE [28]	—	67.7	—
PRAKA[32]	—	69.0	—
FeTrIL [13]	72.2	71.2	67.1
FeCAM-s [14]	75.8	75.6	<u>73.5</u>
FeCAM-e [14]	78.3	78.2	75.1
FRMM (ours)	<u>76.5</u>	<u>76.0</u>	71.9
Upperbound	85.8	85.7	86.3

Accuracy. The comparison of average accuracy are shown in Table 1 and Table 2. The proposed method FRMM outperforms all other methods on both CIFAR-100 and TinyImageNet. On the ImageNet-Subset dataset, the average accuracy of FRMM is suboptimal, being lower than that of FeCAM-e, and comparing to FeCAM-s, FRMM has better performance for $T = 5$ and $T = 10$, while lower for $T = 20$. However, the storage requirement of FRMM is significantly less.

The comparison the accuracy of each incremental task is shown in Fig. 3. Similarly to the result of average accuracy, FRMM achieves higher accuracy on every incremental task compared to other EFCIL methods on CIFAR-100 and TinyImageNet. And on ImageNet-Subset, the performance of FRMM is lower than that of FeCAM-e and is similar to that of FeCAM-s.

**Fig. 3.** Accuracy of each incremental task.

In summary, compared to methods that have similar storage space requirement to FRMM, like FeTrIL, our approach significantly outperforms them in terms of accuracy. Compared to methods with storage space requirements far exceeding FRMM, our approach exhibits slightly lower performance. A detailed comparison regarding storage space requirement will be provided in the next section.

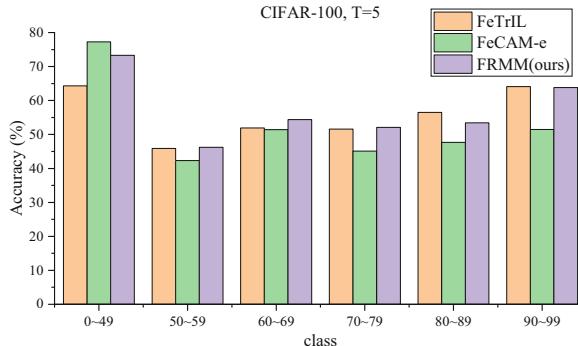


Fig. 4. The comparison of the last accuracy between base classes (0–49) and incremental classes (50–99).

Since the feature extractor is trained on base classes, the model tends to perform better on base classes compared to incremental classes. Therefore, we also compared the accuracy on base classes and incremental classes after completing all incremental tasks. The result is shown in Fig. 4. FeCAM [14] exhibits high accuracy on base classes but much lower accuracy on incremental classes, which means that it lacks plasticity. FRMM exhibits the same accuracy as FeTrIL [13] on incremental classes but achieves higher accuracy on base classes, which implies that FeTrIL has lower stability. In conclusion, FRMM achieves a better balance between stability and plasticity.

Storage Requirement. Since the same feature extractor Resnet18 is used, the space occupied by models of different methods is essentially the same. Therefore, we compare the storage requirement excluding the model itself. The result is shown in Table 3. FeTrIL [13] only stores a 512-dimensional prototype for each class, requiring very small storage. For FRMM, a 512-dimensional standard deviation is also stored, so the storage is double, but still small. In contrast, FeCAM-e [14] not only requires storing a prototype but also saving a 512×512 dimensional covariance matrix for each class, resulting in storage requirement hundreds of times larger than FeTrIL and FRMM. Even for FeCAM-s [14], which only stores one covariance matrix, the required storage is still large, and its precision significantly decreases compared to FeCAM-e.

3.3 Ablation Experiments

The ablation study about different components of FRMM is shown in Table 4. The contribution of feature reprojection is the most significant, indicating that the optimization of the distribution of feature vectors by the reprojector is notable, as shown in Fig. 5. Due

Table 3. Analysis of storage requirement (MB) across tasks for CIFAR-100, with $T = 5$.

Methods	Task 0	Task 1	Task 2	Task 3	Task 4	Task 5
FeTrIL [13]	0.10	0.12	0.14	0.16	0.18	0.20
FeCAM-s [14]	1.10	1.12	1.14	1.16	1.18	1.20
FeCAM-e [14]	50.10	60.12	70.14	80.16	90.18	100.20
FRMM (ours)	0.20	0.23	0.27	0.31	0.35	0.39

Table 4. Ablation of the performance indicating the contribution from the different components of our proposed method FRMM. (%)

FR	Momentum	Ensemble	CIFAR-100, T = 5	
			Average Acc	Last Acc
×	×	×	64.5	54.6
✓	×	×	70.2	60.6
×	✓	×	63.8	53.1
×	×	✓	65.2	55.0
✓	✓	×	70.9	62.1
✓	×	✓	72.0	62.6
×	✓	✓	64.7	54.0
✓	✓	✓	72.5	63.6

to the reprojection layer, the difficulty of subsequent classification is reduced, so we set the momentum for updating the classifier to a high value (0.99). Noting that this setting isn't applicable when the reprojection layer isn't used. The ensemble strategy improves the model's accuracy across various settings because the NCM classifier and the linear classifier have different biases. There-fore, merging the outputs of both classifiers is meaningful.

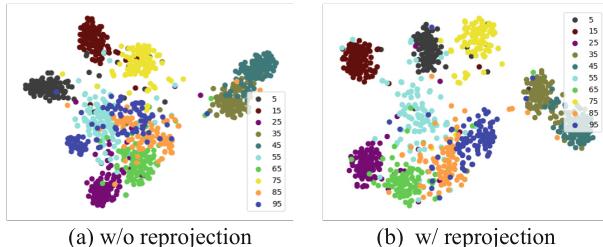
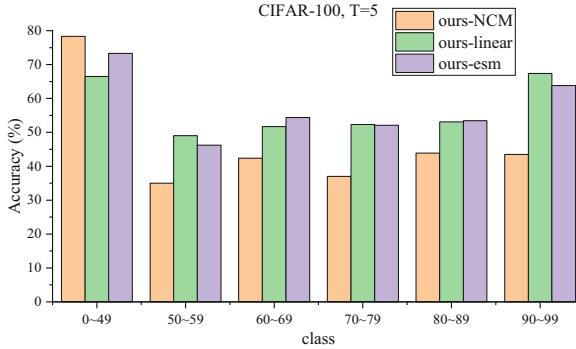
**Fig. 5.** Distribution of feature vectors from base classes (0–49) and incremental classes (50–99) with and without reprojection, visualized by t-SNE [19].

Table 5. Average top-1 accuracy (%) of our method before and after ensemble.

Methods	CIFAR-100			TinyImageNet		
	T = 5	T = 10	T = 20	T = 5	T = 10	T = 20
Ours-NCM	70.1	69.9	67.0	58.2	58.1	57.9
Ours-linear	70.2	69.5	67.3	60.3	59.3	58.6
Ours-esm	72.5	72.0	69.4	61.3	60.8	60.3

The results in Table 5 indicate that our method achieves high accuracy even without ensemble, when directly using the NCM classifier or linear classifier. This also indicates the effectiveness of the feature reprojection. By employing ensemble, the accuracy can be further improved.

The results in Fig. 6 display the last accuracy when using the NCM classifier or the linear classifier separately, as well as the last accuracy after ensemble. In Table 5, the NCM classifier and the linear classifier show close average accuracies. But in Fig. 6, the NCM classifier exhibits significantly higher accuracy on base classes (0–49) compared to the linear classifier, indicating higher stability. However, on incremental classes (50–99), the linear classifier shows significantly higher accuracy than the NCM classifier, suggesting that the linear classifier possesses greater plasticity. By ensemble of both, the model achieves a better balance between stability and plasticity.

**Fig. 6.** The comparison of the last accuracy before and after ensemble.

4 Conclusion and Future Work

In this paper, we analyzed the distribution of feature vectors for base and incremental classes when using a frozen feature extractor. We propose to use a reprojection layer to adjust the feature vectors, thereby alleviating the burden on the classifier. Additionally, we propose to leverage the strengths of the NCM classifier and the linear classifier as two experts with different capabilities by combining them using the principles of ensemble

learning. To address the forgetting issue of the linear classifier, we further propose to perform momentum update on the weights related to old classes in the fully connected layer. The proposed method FRMM achieves optimal or suboptimal results on several datasets, and compared to the current state-of-the-art methods, it requires much less storage space.

Limitations. The proposed method is based on a frozen feature extractor, which imposes a limit on the capacity of the model. In future research, it is advisable to provide more learnable parameters for incremental stages to enhance the plasticity of the model.

References

1. De Lange, M., et al.: A continual learning survey: defying forgetting in classification tasks. *IEEE Trans. Pattern Anal. Mach. Intell.* **44**(7), 3366–3385 (2021)
2. Kim, G., Xiao, C., Konishi, T., Ke, Z., Liu, B.: A theoretical study on solving continual learning. *Adv. Neural. Inf. Process. Syst.* **35**, 5065–5079 (2022)
3. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: Icarl: Incremental classifier and representation learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2001–2010 (2017)
4. Zhou, D.W., Wang, Q.W., Ye, H.J., Zhan, D.C.: A model or 603 exemplars: Towards memory-efficient class-incremental learning. In: The Eleventh International Conference on Learning Representations (2022)
5. Wang, L., Zhang, X., Su, H., Zhu, J.: A comprehensive survey of continual learning: theory, method and application. *IEEE Trans. Pattern Anal. Mach. Intell.* **46**, 5362–5383 (2024)
6. Kirkpatrick, J., et al.: Overcoming catastrophic forgetting in neural networks. *Proc. Natl. Acad. Sci.* **114**(13), 3521–3526 (2017)
7. Li, Z., Hoiem, D.: Learning without forgetting. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) *ECCV 2016. LNCS*, vol. 9908, pp. 614–629. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_37
8. Goodfellow, I., et al.: Generative adversarial networks. *Commun. ACM* **63**(11), 139–144 (2020)
9. Shin, H., Lee, J.K., Kim, J., Kim, J.: Continual learning with deep generative replay. *Adv. Neural Inf. Process. Syst.* **30** (2017)
10. Ostapenko, O., Lesort, T., Rodríguez, P., Arefin, M.R., Douillard, A., Rish, I., Charlin, L.: Continual learning with foundation models: an empirical study of latent replay. In: Conference on Lifelong Learning Agents, pp. 60–91. PMLR (2022)
11. Verma, V.K., Liang, K.J., Mehta, N., Rai, P., Carin, L.: Efficient feature transformations for discriminative and generative continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13865–13875 (2021)
12. Zhu, F., Zhang, X.Y., Wang, C., Yin, F., Liu, C.L.: Prototype augmentation and self-supervision for incremental learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5871–5880 (2021)
13. Petit, G., Popescu, A., Schindler, H., Picard, D., Delezoide, B.: Fetril: feature translation for exemplar-free class-incremental learning. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 3911–3920 (2023)
14. Goswami, D., Liu, Y., Twardowski, B., van de Weijer, J.: Fecam: exploiting the heterogeneity of class distributions in exemplar-free continual learning. In: Oh, A., Neumann, T., Globerson, A., Saenko, K., Hardt, M., Levine, S. (eds.) *Advances in Neural Information Processing Systems*. vol. 36, pp. 6582–6595. Curran Associates, Inc. (2023)

15. Zhuang, H., Lin, Z., Toh, K.A.: Correlation projection for analytic learning of a classification network. *Neural. Process. Lett.* **53**, 3893–3914 (2021)
16. Zhuang, H., Weng, Z., Wei, H., XIE, R., Toh, K.A., Lin, Z.: ACIL: analytic class-incremental learning with absolute memorization and privacy protection. In: Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A. (eds.) *Advances in Neural Information Processing Systems*, vol. 35, pp. 11602–11614. Curran Associates, Inc. (2022)
17. Zhuang, H., Weng, Z., He, R., Lin, Z., Zeng, Z.: GKEAL: Gaussian kernel embedded analytic learning for few-shot class incremental task. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7746–7755 (2023)
18. Zhuang, H., He, R., Tong, K., Zeng, Z., Chen, C., Lin, Z.: DS-AL: a dual-stream analytic learning for exemplar-free class-incremental learning. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 15, 17237–17244 (2024)
19. Van der Maaten, L., Hinton, G.: Visualizing high-dimensional data using t-sne. *J. Mach. Learn. Res.* **9**(11), 2579–2605 (2008)
20. Zhu, F., Cheng, Z., Zhang, X.Y., Liu, C.I.: Class-incremental learning via dual augmentation. *Adv. Neural. Inf. Process. Syst.* **34**, 14306–14318 (2021)
21. Rypeść, G., Cygert, S., Khan, V., Trzcinski, T., Zieliński, B.M., Twardowski, B.: Divide and not forget: ensemble of selectively trained experts in continual learning. In: *The Twelfth International Conference on Learning Representations* (2023)
22. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. *Stat* **1050**, 21 (2016)
23. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images. Master's thesis, University of Tront (2009)
24. Le, Y., Yang, X.: Tiny imagenet visual recognition challenge. *CS 231N* **7**(7), 3 (2015)
25. Russakovsky, O., et al.: Imagenet large scale visual recognition challenge. *Int. J. Comput. Vision* **115**(3), 211–252 (2015). <https://doi.org/10.1007/s11263-015-0816-y>
26. Zhou, D.W., Wang, F.Y., Ye, H.J., Zhan, D.C.: Pycil: a python toolbox for class-incremental learning. *Sci. China Inf. Sci.* **66**(9), 197101–197102 (2023). <https://doi.org/10.1007/s11432-022-3600-y>
27. 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)
28. Zhu, K., Zhai, W., Cao, Y., Luo, J., Zha, Z.J.: Self-sustaining representation expansion for non-exemplar class-incremental learning. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9296–9305 (2022)
29. Belouadah, E., Popescu, A.: Deesil: deep-shallow incremental learning. In: Leal-Taixé, L., Roth, S. (eds.) *ECCV 2018. LNCS*, vol. 11130, pp. 151–157. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-11012-3_11
30. Hou, S., Pan, X., Loy, C.C., Wang, Z., Lin, D.: Learning a unified classifier incrementally via rebalancing. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 831–839 (2019)
31. Liu, Y., Parisot, S., Slabaugh, G., Jia, X., Leonardis, A., Tuytelaars, T.: More classifiers, less forgetting: a generic multi-classifier paradigm for incremental learning. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) *ECCV 2020. LNCS*, vol. 12371, pp. 699–716. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58574-7_42
32. Shi, W., Ye, M.: Prototype reminiscence and augmented asymmetric knowledge aggregation for non-exemplar class-incremental learning. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1772–1781 (2023)



PCQ: Emotion Recognition in Speech via Progressive Channel Querying

Xincheng Wang, Liejun Wang^(✉), Yinfeng Yu, and Xinxin Jiao

School of Computer Science and Technology, Xinjiang University, Urumqi, China
{wljxju,yuyinfeng}@xju.edu.cn

Abstract. In human-computer interaction (HCI), Speech Emotion Recognition (SER) is a key technology for understanding human intentions and emotions. Traditional SER methods struggle to effectively capture the long-term temporal correlations and dynamic variations in complex emotional expressions. To overcome these limitations, we introduce the PCQ method, a pioneering approach for SER via **Progressive Channel Querying**. This method can drill down layer by layer in the channel dimension through the channel query technique to achieve dynamic modeling of long-term contextual information of emotions. This multi-level analysis gives the PCQ method an edge in capturing the nuances of human emotions. Experimental results show that our model improves the weighted average (WA) accuracy by 3.98% and 3.45% and the unweighted average (UA) accuracy by 5.67% and 5.83% on the IEMOCAP and EMODB emotion recognition datasets, respectively, significantly exceeding the baseline levels.

Keywords: Speech emotion recognition · Channel querying · Progressive

1 Introduction

Emotion recognition technology identifies and understands an individual's emotional state by analyzing human behavioural manifestations, including speech, facial expressions, body movements, and language. This technology is essential in some fields, such as HCI [1] and health monitoring [2]. Especially in specific scenarios, like call centres [3] and telemedicine, speech becomes the preferred or the only feasible modality for emotion recognition due to other modalities' unavailability or practical limitations, such as text and images. This modal dependence highlights speech emotion recognition's unique value and challenges in specific applications. Given this, this paper will investigate unimodal speech emotion recognition.

Much attention has been paid to the temporal nature of speech signals [4], a crucial property for understanding and processing speech data. Our research focuses on discrete speech emotion recognition (DSER) [5], i.e., in this framework, we assume that each sentence expresses a single emotion. However, the expression of human emotions is a dynamic process: emotions are not fully revealed instantly but gradually revealed over time. This dynamism means that although our task is to identify the dominant emotion in each sentence, we still need to pay attention to temporal variations in the speech signal to

capture subtle clues about how the emotion develops over time. The temporal character of emotional expression exacerbates the difficulty of accurately dynamic modelling emotional information in speech. To address this challenge, current research approaches face several problems. Firstly, although the Transformer [6] based approach enables global modelling, the approach may not be applicable when dealing with relatively small unimodal speech emotion recognition data. Therefore, current research in SER focuses mainly on capturing contextual emotional information using self-attention or cross-attention mechanisms. Hu et al. in [7] applied different cross-attention modules to a joint speech emotion recognition network and achieved good results in a speaker-independent setting. In [8], Naderi and Nasersharif proposed an approach that combines the features extracted by Wav2Vec 2.0 with rhyming features and introduces a novel attention mechanism that effectively integrates these features to improve the accuracy of cross-corpus speech emotion recognition. In addition, Xu et al. proposed a head fusion strategy in [9], which solves the limitation that the multi-head attention mechanism can only focus on a single information point and makes the model able to concentrate on multiple important information points at the same time, which in turn optimizes the model performance. While these attention methods can achieve global attention to some extent, they inevitably require more parameters.

Meanwhile, Convolutional Neural Networks (CNNs) are also prevalent in the research of speech emotion recognition. For example, Zhao et al. in [10] combined an LSTM network and a full convolutional network (FCN), the latter extracting features by concatenating multiple 2D convolutional layers. Aftab et al. in [11] designed a lightweight FCN to extract features in-depth by increasing the number of convolutional layers, and Mekruksavanich et al. in [12] explored the possibility of applying a 1D convolutional method applied to Thai sentiment recognition. Although these multilayer convolutional network strategies are effective in feature extraction, they usually focus only on the output of the last layer of the CNN, which may ignore the fine-grained information in the shallow layers, which limits the model's ability to capture long-term contextual information.

To address the shortcomings in the existing methods mentioned above, our study proposes a novel approach to speech emotion recognition based on the progressive channel querying technique. By combining the dual perspectives of speech and spectrogram and introducing the channel query module, this method can model speech emotion signals dynamically in the channel dimension in a progressive manner. Our main contributions are as follows:

- We designed a **Multilayer Lightweight CNN (MLCNN)** branch to extract feature outputs from different layers. Further, we introduce the overall framework of the PCQ network. This framework merges the MLCNN branch, using spectrogram as input, with the WavLM pre-trained model branch, using original speech as input.
- We designed a **Channel Semantic Query (CSQ)** module for querying and integrating semantically similar sentiment features from neighbouring layers of an MLCNN in the channel dimension. This module exploits the temporal properties of speech signals to dynamically model speech emotion, thereby capturing the temporal variation

of emotional information in the channel dimension. We achieve step-by-step acquisition of emotional information by integrating multiple CSQ modules in the PCQ framework.

- Our method achieves good results on the IEMOCAP dataset and the EMODB dataset.

2 Proposed Method

This section provides a comprehensive overview of our proposed approach, as shown in Fig. 1. Section 2.1 describes the structure of the sub-network MLCNN. Section 2.2 describes the WavLM pre-trained model and the processing of its output features. Section 2.3 introduces the CSQ module. Section 2.4 outlines the overall network structure of PCQ.

2.1 MLCNN Branch

As shown in Fig. 1(B), the MLCNN we developed consists of four layers with {16, 32, 48, 64} channels per layer. For the layer setting of the MLCNN, we have explained in detail and ablation experiments in Sect. 4. As the network depth increases, the MLCNN will output deeper semantic features layer by layer. Specifically, the network contains four layers whose outputs are x_1, x_2, x_3, x_4 and the feature of the mth pair of adjacent layers is denoted as f_m . Thus, we define: $f_1 = [x_1, x_2], f_2 = [x_2, x_3], f_3 = [x_3, x_4]$, where the $[\cdot, \cdot]$ symbols are used to denote the selected two neighbouring layer features. In addition, each layer contains a Parameter-efficient Depth Convolution (PDC) block, which consists of two point convolutions, a depth convolution, and a channel attention block, as shown in Fig. 1(C). The input feature is $\chi \in \mathbb{R}^{C \times H \times W}$, where C denotes the number of channels, H is the height, W is the width, and the c th channel of the input feature map is denoted as χ_c . For each feature map, the global average pooled feature $Gap(\chi)$ is obtained by computing the average of all its elements. $Gap(\chi)$ is then fed into a fully connected layer Υ and processed by a sigmoid function δ to learn the importance of the sentiment information in each channel and generate the channel weights ω accordingly. Then, by multiplying the weights ω with the input feature χ , we obtain the output feature y . The above process is shown in the following equation. In Eq. (1), $(\chi_c)_{i,j}$ is the element with position (i, j) on the c th channel.

$$Gap(\chi) = \frac{1}{W \times H} \sum_{i=1}^W \sum_{j=1}^H (\chi_c)_{i,j}, \quad (1)$$

$$\omega = \delta(\Upsilon(Gap(\chi))), \quad (2)$$

$$y = \omega \chi. \quad (3)$$

Therefore, the attentional mechanism enables the model to understand and emphasize key emotional information in each channel more accurately. The PDC module adopts an architecture that sequentially connects dot convolution, deep convolution, channel attention mechanism, and dot convolution. This design not only significantly improves

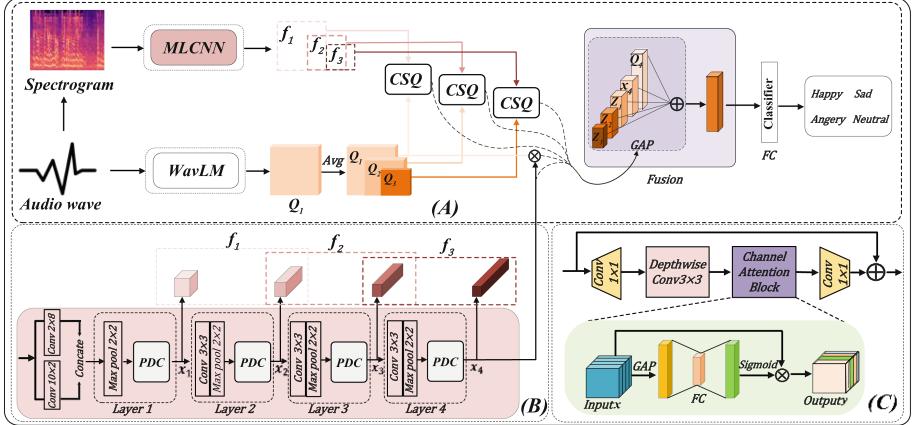


Fig. 1. (A) The overall PCQ framework. (B) The MLCNN network. (C) The PDC module.

the performance of the model in capturing emotionally relevant information, but also effectively reduces noise interference.

The PDC module maintains an equal number of input and output channels, denoted as C . Under equal conditions, traditional 3×3 convolutions require $9C^2$ parameters, while the parameter count of the PDC module is $(16/3)C^2 + 18C$. From the analysis, it is found that the number of parameters of the PDC module will be lower than the traditional 3×3 convolution when $C > 4.90$. The relevant parameter count comparisons are detailed in Table 3 in Sect. 4. In addition, in Table 4 in Sect. 4, we also show that the PDC module improves in terms of accuracy compared to the traditional 3×3 convolution.

2.2 WavLM Branch

As shown in (A) of Fig. 1, the WavLM pre-trained model, serving as an encoder within a branching network that takes speech signals as inputs, effectively models long-term contextual sequences. Second, in the work [13], Zhao et al. revealed the layer-to-layer variability in the pre-trained model: the part closer to the output layer tends to contain rich task-specific information, whereas the underlying layers closer to the input capture more generic features. These bottom layers capture a wide range of features, while the top layer focuses on high-level abstract features that are closely related to a specific task. In this study, we utilized the WavLM pre-trained model for the speech emotion recognition task. Compared to the bottom layer of the model, the top layer of WavLM is more focused on modeling speech emotion information. Therefore, we selected features from the final output layer of the WavLM model and enhanced the network's overall recognition of emotions through multiple average pooling operations.

2.3 CSQ Module

In Fig. 2, we design the Channel Semantic Query (CSQ) Module. CSQ module takes three inputs, with the first two being shallow-level features $X_l \in \mathbb{R}^{C_l \times H_l \times W_l}$ and deep-level features $X_h \in \mathbb{R}^{C_h \times H_h \times W_h}$. To aggregate the semantically similar information

on these two different feature scales, we first employ convolution and bilinear interpolation to resize the deep-level speech features to match the size of the shallow-level speech features, $C_h = C_l$, $H_h = H_l$, $W_h = W_l$. Next, we evenly divide the channels of the two features into four groups (*Group* = 4) for positional encoding. We introduced a channel query token Q , which queries and aggregates channel features with identical positional encodings, creating $\varpi^i \in \mathbb{R}^{(\text{Group}/2+1) \times H_l \times W_l}$, $i = [1, 2, 3, 4]$. As shown in Eq. (4), In each ϖ^i , Q efficiently synthesizes similar emotional information of the same positional encoding, utilizing its pre-trained knowledge. In high-dimensional channels, deeper semantic information exists. To further extract the deeper semantic feature $\eta^i \in \mathbb{R}^{(\text{Group}/2+1) \times H_l \times W_l}$, for four blocks from different channel dimensions, we use 3×3 dilation convolution with dilation rates of $d_i = [7, 5, 2, 1]$, as shown in Eq. (5).

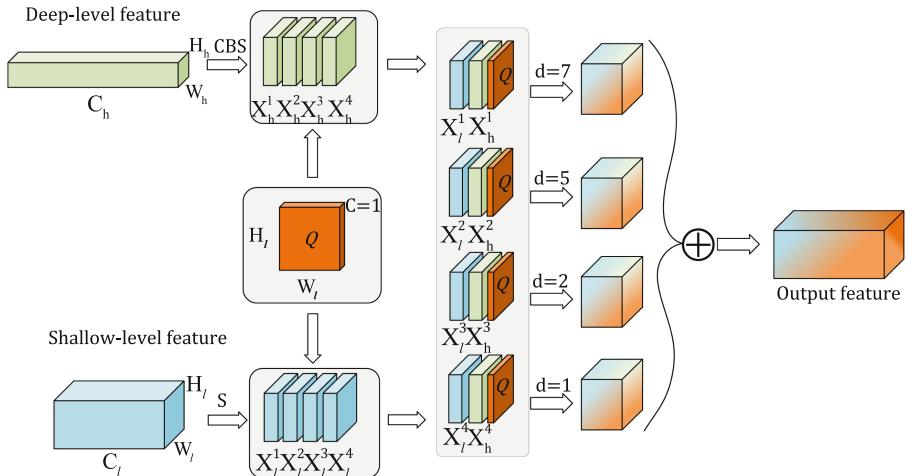


Fig. 2. CSQ module. The letters on the arrows in the diagram represent the following: **C** denotes that the convolution operation is first performed to match the number of channels C_h of the high-level features with the number of channels C_l of the low-level features; **B** denotes that the height H and width W of the feature map are adjusted to keep the high-level features (H_h, W_h) and the low-level features (H_l, W_l) the same using bilinear interpolation; and **S** denotes that the channel dimensions are segmented; **d** is the dilation rate of the convolution.

Based on the correlation between emotion and time, we concatenate η^i to obtain $\hat{\eta} \in \mathbb{R}^{C_l \times H_l \times W_l}$. Next, a convolution operation highlights the emotional information in $\hat{\eta}$.

$$\varpi^i = \text{Concat}\left(X_l^i, X_h^i, Q\right). \quad (4)$$

$$\eta^i = \text{Conv}_{3 \times 3}^{d_i}(\varpi^i). \quad (5)$$

2.4 The Proposed Overall Framework (PCQ)

In Fig. 1(A), PCQ is composed of three main components: the MLCNN sub-branch, the WavLM pre-training branch, and the CSQ module. Firstly, spectrogram features are input into the MLCNN, as illustrated in Fig. 1(B). The MLCNN produces three sets of outputs, namely, f_1, f_2 , and f_3 . These outputs are then sequentially used as inputs for the three independent CSQ modules. After the speech signal passes through the WavLM encoder, it first obtains the channel query token Q_1 with channel number 1. Then, Q_1 undergoes two adaptive pooling operations to obtain the other two channel query tokens, Q_2 and Q_3 . These tokens are pre-trained features with global sentiment information. Next, we need to use this pre-training information to assist the PCQ network in achieving global sentiment perception. Therefore, Q_1, Q_2 , and Q_3 are input as query tokens to the three CSQ modules sequentially, as shown in Eq. (6). Next, the CSQ modules will sequentially generate three progressive features z_1, z_2 and z_3 , which are different degrees of perception of global emotion information. Q_1 obtains its self-attention features through an attention mechanism weighted by x_4 , as shown in Eq. (7).

$$z_j = CSQ(f_j, Q_j), j = 1, 2, 3. \quad (6)$$

$$Q_4 = x_4 \times Q_1. \quad (7)$$

To enhance feature fusion, the *Gap* (Global Average Pooling) operation is applied to Q_1, z_1, z_2, z_3 and x_4 , as shown in Eq. (8). These fused features are fed into a classifier comprising multiple linear layers for emotion prediction, as shown in Eq. (9).

$$y_{fusion} = \text{Concat}(\text{Gap}(z_1), \text{Gap}(z_2), \text{Gap}(z_3), \text{Gap}(x_4), \text{Gap}(Q_1)). \quad (8)$$

$$\hat{y} = \text{Classifier}(y_{fusion}). \quad (9)$$

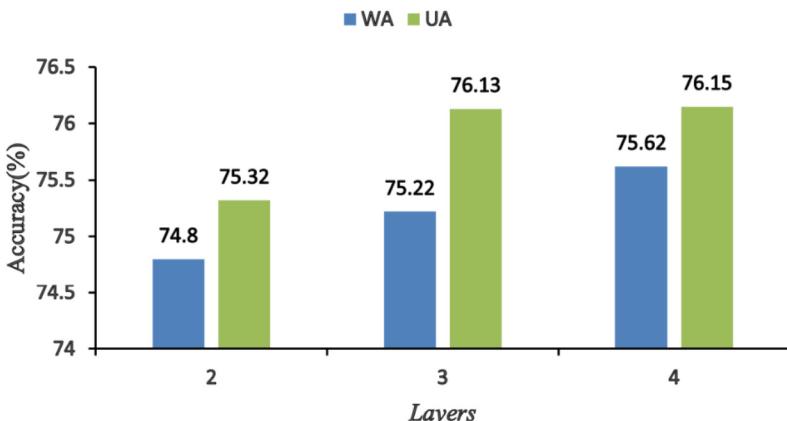


Fig. 3. Visualisation of MLCNN results for different layers on the IEMOCAP dataset: blue for weighted accuracy (WA), green for unweighted accuracy (UA). (Color figure online)

Table 1. Comparison of baseline and state-of-the-art methods on the IEMOCAP dataset. All (\uparrow) means in the experimental table indicate that higher is better, and (\downarrow) means indicate that lower is better. (\dagger) indicates data derived directly from the paper.

Models	WA (\uparrow)	UA (\uparrow)
HNSD [14] †	70.50	72.50
GLAM [15] †	73.70	73.90
SMW_CAT [16] †	73.80	74.25
GLNN [17] †	71.83	65.39
AMSNet [18] †	69.22	70.51
Cross-fusion [19] †	72.04	73.26
MFCC + Spectrogram + W2E(base) [20] †	71.64	72.70
PCQ(ours)	75.62	76.15

3 Experiment

3.1 Datasets

IEMOCAP: This dataset is an English corpus. In total, it contains about 12 h of audio-visual data, of which audio data has been widely used in automatic emotion recognition research. We identify four main emotions: anger, sadness, happiness and neutrality. Considering the unbalanced distribution of emotion categories in the dataset, we merge “happy” and “excited” into “happy”. **EMODB:** This dataset is a German language speech library recorded by 10 participants (5 males and 5 females) covering seven different emotional expressions: anger, disgust, fear, happiness, sadness, surprise, and neutrality. The entire database contains about 535 audio clips, each ranging from 1 to 10 s long, providing a rich sound sample for emotion recognition studies.

3.2 Experimental Setup

In this study, we sampled the raw audio signal at 16 kHz and segmented it into 3-s segments, with underfilled segments using zero padding. The final prediction is based on the judgement of all the segments. Through a series of 40 ms Hamming windows, we generate spectrogram features. Each window was discrete Fourier transformed (DFT) as a frame to obtain an 800-point frequency domain signal, and the first 200 were taken as input features. In this way, we obtained a spectrogram of 300×200 size corresponding to each audio clip. To evaluate the model performance, we use both Weighted Accuracy (WA) and Unweighted Accuracy (UA) metrics and use 10-fold cross-validation to ensure that the results are reliable. Our emotion classification task uses a cross-entropy loss function. Our system is implemented in PyTorch. The batch sizes are 16 and 32 for the IEMOCAP dataset and EMODB dataset, respectively. The early stop is 20 epochs. We use the AdamW optimiser, and the learning rate is 1e-5. All experiments were conducted on an NVIDIA 4090 24G GPU.

Table 2. Comparison of baseline and state-of-the-art methods on the EMO-DB dataset. (*) indicates that all results are reproduced using publicly available source code and the original hyperparameters. (†) indicates data derived directly from the paper.

Models	WA (\uparrow)	UA (\uparrow)
TSP + INCA [21] [†]	90.09	89.47
GM-TCN [22] [†]	91.39	90.48
LightSER [11] [†]	94.21	94.15
TIM-Net [23] [†]	95.70	95.17
MFCC + Spectrogram + W2E(base) [20] [*]	90.17	89.65
PCQ(ours)	95.84	95.48

Table 3. Comparison of our proposed method with the baseline network in terms of parameter size on the IEMOCAP dataset.

Models	Parameters(\downarrow)
MFCC + Spectrogram + W2E(base) [20]	174.62M
PCQ(ours)	97.99M
AlexNet(base) [20]	2.47M
MLCNN(ours)	0.092M

4 Experimental Results and Analysis

Figure 3 shows the performance of MLCNN with the number of layers 2, 3 and 4 on the IEMOCAP dataset. The results show that both performance metrics improve significantly with the increasing number of layers, especially when the model reaches 4 layers, both WA and UA metrics reach the highest, 75.62% and 76.15%, respectively. This trend demonstrates that increasing network depth effectively enhances model accuracy. However, to ensure comparability with the baseline AlexNet network, we decided against further increasing the number of layers, opting instead for the same network depth to facilitate effective model comparison under similar conditions. Therefore, a 4-layer MLCNN network was used for all subsequent experiments.

4.1 Results and Comparisons

Table 1 shows the performance of our proposed PCQ network compared to the three-branch network used in the baseline study described in [20] on the IEMOCAP dataset. Our method achieves significant improvements of 3.98% and 3.45% in terms of WA and UA metrics, respectively. In addition, Table 1 also shows some current state-of-the-art speech emotion recognition models. Compared to these models, our PCQ network exhibits a superior performance. On the EMODB dataset, as shown in Table 2, the

PCQ network achieved high accuracy. Compared with the baseline network [20], PCQ improved 5.67% and 5.83% on WA and UA, respectively, further demonstrating the effectiveness and applicability of the methodology. In the upper part of Table 3, we compare in detail the total number of parameters of the benchmark network used on the IEMOCAP dataset with our newly designed PCQ network. From the data, the number of parameters of the PCQ network is reduced by 43.98% compared to the benchmark network. In the lower part of Table 3, we compare the branching networks designed based on spectrograms. Compared to the baseline AlexNet, our designed MLCNN substantially reduces the number of parameters, which fully proves the lightweight feature of our proposed PCQ network and MLCNN network.

Table 4. Results of the ablation study of the components of the proposed frame on the IEMOCAP dataset. “w/o” denotes without, and “w/” denotes with.

Models	WA (\uparrow)	UA (\uparrow)
PCQ	75.62	76.15
w/o PDC(w/conv3 \times 3)	75.20	75.81
w/o CSQ	74.65	74.73
w/o WavLM	57.66	57.68
w/o WavLM(w/ W2E)	70.19	71.77
Spectrogram + W2E(base) [20]	70.05	71.30

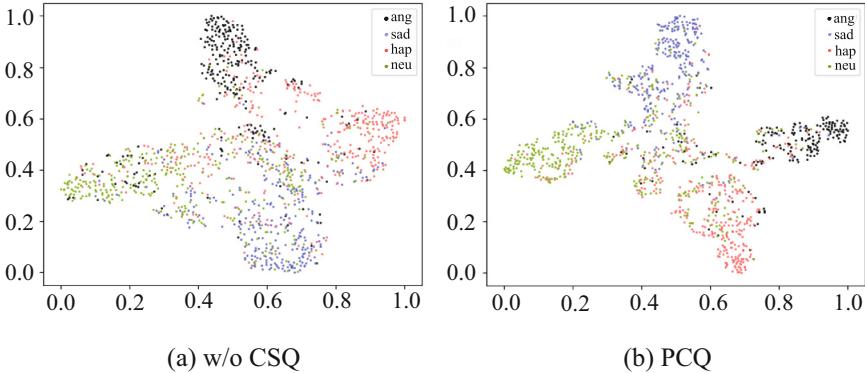


Fig. 4. The t-SNE visualization of feature distribution on the IEMOCAP dataset.

Table 4 details the results of the ablation experiments for each component of the PCQ network on the IEMOCAP dataset. Replacing the PDC module with the 3×3 Conv2d resulted in a reduction of WA and UA by 0.42% and 0.34%, respectively, despite an increase in the number of network parameters. This clearly demonstrates the advantages of the PDC module in terms of lightweight and efficiency. Without the CSQ module, the WA and UA of the network decreased by 0.97% and 1.42%, respectively. As shown

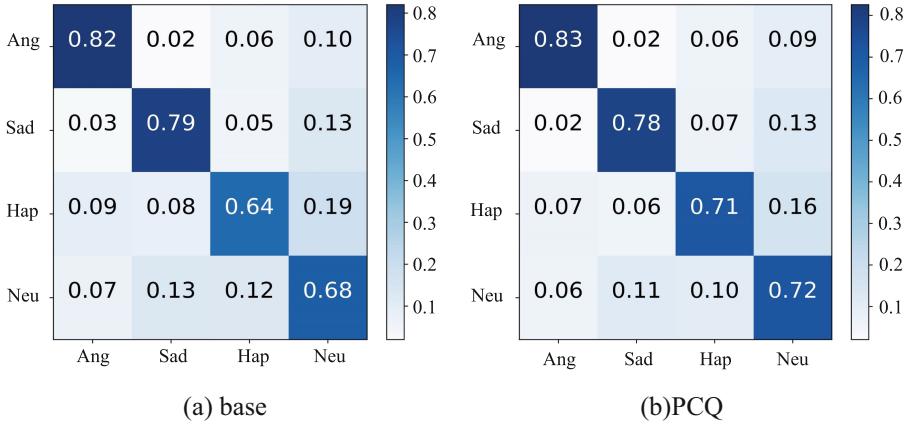


Fig. 5. Comparison of normalized confusion matrices on the IEMOCAP dataset.

in the third-to-last row of Table 4, without the CSQ module and the WavLM branch, the training accuracy of the network is significantly reduced. While the WA and UA when using Wav2Vec 2.0 as the pre-training encoder increase by 0.14% and 0.47% respectively compared to the baseline two-branch network, they are still 5.43% and 4.38% lower compared to our PCQ network. This further demonstrates that both our CSQ model and the WavLM pre-training model are indispensable key components in the PCQ backbone network. As shown in Fig. 4, the t-SNE visualisation results on the IEMOCAP dataset show that the PCQ method with the integrated CSQ module has clearer classification boundaries than the network without the CSQ module. Furthermore, in Fig. 5, the normalised confusion matrix shows that the PCQ method significantly improves the recognition of “happy” and “neutral” emotions on the IEMOCAP dataset.

5 Conclusion

In this study, we propose a new framework for speech emotion recognition named Progressive Channel Querying (PCQ). The method mainly queries and integrates similar sentiment features in the channel dimension through the CSQ (Channel Semantic Query) module. Applying the CSQ module at different layers in the PCQ framework enables a gradual enhancement of the understanding of the sentiment information, thus allowing the model to acquire sentiment features in a progressive manner. Experimental results on the IEMOCAP and the EMODB datasets show that our method achieves significant improvements on the SER task compared to existing techniques. SER tasks are used in a particular scenario. For future research, we will work on multi-scene SER research, i.e., multimodal emotion recognition, where the input is speech, image, text, and other modalities.

Acknowledgments. The following projects jointly supported this work: the Tianshan Excellence Program Project of Xinjiang Uygur Autonomous Region, China (2022TSYCLJ0036), the Central Government Guides Local Science and Technology Development Fund Projects

(ZYYD2022C19), and the National Natural Science Foundation of China (62303259), the Graduate Student Research and Innovation Program in the Xinjiang Uygur Autonomous Region (XJ2024G089).

References

1. Sasikumar, M., Khanna, P.: Emotion recognition in human computer interaction. In: *Research Trends in Information Technology* (2007)
2. Zheng, W., Yan, L., Wang, F.Y.: Two birds with one stone: knowledge-embedded temporal convolutional transformer for depression detection and emotion recognition. *IEEE Trans. Affect. Comput.* **14**(4), 2595–2613 (2023)
3. Morrison, D., Wang, R., De Silva, L.C.: Ensemble methods for spoken emotion recognition in call-centres. *Speech Commun.* **49**(2), 98–112 (2007)
4. Wu, T., Wang, L., Zhang, J.: CM-TCN: channel-aware multi-scale temporal convolutional networks for speech emotion recognition. In: Luo, B., Cheng, L., Wu, Z.G., Li, H., Li, C. (eds.) *Neural Information Processing. ICONIP 2023. LNCS*, vol. 14449. Springer, Singapore (2024). https://doi.org/10.1007/978-981-99-8067-3_34
5. Zhao, Z., et al.: Combining a parallel 2D CNN with a self-attention dilated residual network for CTC based discrete speech emotion recognition. *Neural Netw.* **141**, 52–60 (2021)
6. Wu, W., Huang, Y., Wu, X.: SRT: improved transformer-based model for classification of 2D heartbeat images. *Biomed. Signal Process. Control* **88**, 105017 (2024)
7. Hu, Y., Hou, S., Yang, H., Huang, H., He, L.: A joint network based on interactive attention for speech emotion recognition. In: *2023 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1715–1720. IEEE (2023)
8. Naderi, N., Nasersharif, B.: Cross corpus speech emotion recognition using transfer learning and attention-based fusion of Wav2Vec2 and prosody features. *Knowl. Based Syst.* **277**, 110814 (2023)
9. Xu, M., Zhang, F., Zhang, W.: Head fusion: improving the accuracy and robustness of speech emotion recognition on the IEMOCAP and RAVDESS dataset. *IEEE Access* **9**, 74539–74549 (2021)
10. Zhao, Z., et al.: Exploring deep spectrum representations via attention-based recurrent and convolutional neural networks for speech emotion recognition. *IEEE Access* **7**, 97515–97525 (2019)
11. Aftab, A., Morsali, A., Ghaemmaghami, S., Champagne, B.: Light-SERNet: a lightweight fully convolutional neural network for speech emotion recognition. In: *ICASSP 2022–2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6912–6916. IEEE (2022)
12. Mekruksavanich, S., Jitpattanakul, A., Hnoohom, N.: Negative emotion recognition using deep learning for Thai language. In: *2020 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT & NCON)*, pp. 71–74. IEEE (2020)
13. Zhao, J., Zhang, W.Q.: Improving automatic speech recognition performance for low-resource languages with self-supervised models. *IEEE J. Sel. Top. Sig. Process.* **16**(6), 1227–1241 (2022)
14. Cao, Q., Hou, M., Chen, B., Zhang, Z., Lu, G.: Hierarchical network based on the fusion of static and dynamic features for speech emotion recognition. In: *ICASSP 2021–2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6334–6338. IEEE (2021)

15. Zhu, W., Li, X.: Speech emotion recognition with global-aware fusion on multi-scale feature representation. In: ICASSP 2022–2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 6437–6441. IEEE (2022)
16. He, Y., Minematsu, N., Saito, D.: Multiple acoustic features speech emotion recognition using cross-attention transformer. In: ICASSP 2023–2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1–5. IEEE (2023)
17. Li, Y., Wang, Y., Yang, X., Im, S.K.: Speech emotion recognition based on graph-LSTM neural network. EURASIP J. Audio Speech Music Process. **2023**(1), 40 (2023)
18. Chen, Z., Li, J., Liu, H., Wang, X., Wang, H., Zheng, Q.: Learning multi-scale features for speech emotion recognition with connection attention mechanism. Expert Syst. Appl. **214**, 118943 (2023)
19. Zhao, H., Huang, N., Chen, H.: Knowledge enhancement for speech emotion recognition via multi-level acoustic feature. Connect. Sci. **36**(1), 2312103 (2024)
20. Zou, H., Si, Y., Chen, C., Rajan, D., Chng, E.S.: Speech emotion recognition with co-attention based multi-level acoustic information. In: ICASSP 2022–2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 7367–7371. IEEE (2022)
21. Tuncer, T., Dogan, S., Acharya, U.R.: Automated accurate speech emotion recognition system using twine shuffle pattern and iterative neighborhood component analysis techniques. Knowl. Based Syst. **211**, 106547 (2021)
22. Ye, J.X., et al.: GM-TCNet: gated multi-scale temporal convolutional network using emotion causality for speech emotion recognition. Speech Commun. **145**, 21–35 (2022)
23. Ye, J., Wen, X.C., Wei, Y., Xu, Y., Liu, K., Shan, H.: Temporal modeling matters: a novel temporal emotional modeling approach for speech emotion recognition. In: ICASSP 2023–2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1–5. IEEE (2023)



Federated Learning Cellular Traffic Prediction Based on Multi-time Scale Information

Xingzhen Gao¹, Yuhong Zhao¹(✉), Jingyu Wang¹(✉), and Chuanting Zhang²

¹ School of Digital and Intelligent Industry (School of Cyber Science and Technology), Inner Mongolia University of Science and Technology, Baotou, Inner Mongolia, China
zhaoyuhong35@163.com, 13734728816@126.com

² School of Software, Shandong University, Jinan, Shandong, China

Abstract. This study focuses on the field of cellular traffic prediction and proposes a federated learning method for cellular traffic prediction based on multi-time scale information combined with network pruning. It utilizes periodic time factors and the application of the Informer model within a federated learning framework. This paper integrates temporal features such as weeks, months, and holidays to optimize the model's ability to model time series. To address the non-independent and identically distributed problem in federated learning, a data augmentation based method is introduced to enhance the model's adaptability to heterogeneous data. Additionally, it optimizes the model structure through gradient priority iterative pruning techniques, applying lightweight processing to the Informer model to reduce computational complexity. The experimental results verify the good performance of the proposed method in cellular traffic prediction tasks, enhancing prediction accuracy and significantly reducing the model's computational resource requirements. This research provides robust theoretical support for optimizing resource utilization efficiency and performance in wireless communication systems and offers valuable references for resource management and traffic scheduling in practical applications.

Keywords: Federated Learning · Multi-time Scale · Model Lightweighting

1 Introduction

With the rapid development of mobile communication technology and the proliferation of smart devices, the demand for wireless network traffic has shown an unprecedented growth trend. This growth not only brings a huge volume of data transmission but also poses higher requirements for wireless network management and optimization. Effective traffic prediction is crucial for optimizing network resource allocation, enhancing user experience, reducing energy consumption, and ultimately achieving intelligent and efficient network management [1]. Cellular traffic prediction methods can be classified into three types: simple methods, statistical methods, and machine learning based methods. Simple methods, such as the Simple Moving Average (SMA), use straightforward computational rules for prediction. These methods are easy to implement but fail to capture the changing patterns in the data, resulting in poor prediction accuracy. To achieve more

accurate sequence prediction, early researchers introduced statistical methods. Autoregressive Integrated Moving Average (ARIMA) [2] is a classic statistical sequence prediction method and is also applicable to traffic prediction issues. Lin et al. [3] divided cellular traffic into regular and random parts, explaining that the regular part could be predicted using the ARIMA method. With the rise of machine learning, researchers attempted to use deep learning the most popular method in the field of artificial intelligence to model the problem as a regression problem for cellular traffic prediction. Huang et al. [4] used Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs) to extract the changing patterns of cellular traffic in the temporal and spatial domains. In recent years, researchers have introduced the latest methods into cellular traffic prediction following the development of deep learning. These attempts include Graph Neural Networks (GNN) [5] and Transformers [6], providing new ideas for traffic prediction. However, all the above methods for cellular traffic prediction are considered from the perspective of centralized intelligence and require considerable computational resources. They do not meet the actual computational capabilities of existing systems and fail to leverage the advantages of distributed intelligence. Therefore, this paper explores a distributed intelligence framework suitable for traffic prediction.

In response to the challenges in the field of traffic prediction, federated learning, as an innovative distributed learning framework [7], has received widespread attention. Its uniqueness lies in allowing multiple devices or servers to collaboratively train models while maintaining data privacy, effectively handling wireless network data dispersed across various locations. This study aims to develop a novel federated learning method for cellular traffic prediction based on multi-time scale information. This paper places particular emphasis on integrating periodic time features, such as weekly, monthly, and holiday information, which are essential for capturing and predicting the dynamic changes in cellular traffic. By combining these periodic time features with advanced deep learning techniques within the federated learning framework [8] and applying the Informer [9] model, we can achieve more accurate and efficient traffic predictions. Given that wireless network data usually exhibits non-independent and identically distributed (non-IID) characteristics [10], this study further adopts data augmentation techniques [11] to enhance the model's robustness in handling heterogeneous data. Moreover, to address the high computational complexity required by deep learning models, this study adopts an innovative approach: gradient priority iterative pruning technology. Through this technique, the Informer model is lightened in the experiments, effectively reducing the computational resource requirements of the model while maintaining its predictive performance advantages. This method not only improves the feasibility and efficiency of the model but also makes it more suitable for environments with limited computational resources.

Through a series of innovations, this study offers a novel perspective on wireless network traffic prediction research on Multi-Temporal Scale Information Federated Learning for Wireless Traffic Prediction based on Gradient Priority Iterative Pruning Algorithm (FedMTS), contributing significant theoretical and practical value to achieving smarter and more efficient network management.

2 Related Works

2.1 Correlation of Traffic

This study utilizes the open dataset from the “Telecom Italia Big Data Challenge” [12, 13] to analyze the distribution patterns of wireless communication service traffic in the cities of Milano and Trento. This dataset divides Milano into 10,000 grids of 235 m by 235 m. with each grid recording the volume of calls, text messages, and data traffic during specific time periods. The focus of the research is on the hourly cellular traffic, which is used to analyze online activities within each grid area.

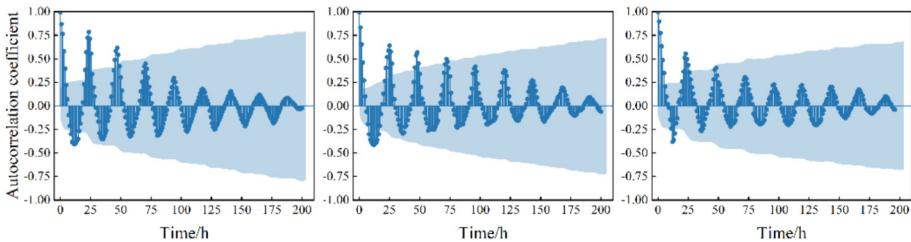


Fig. 1. Time relevance

Cellular traffic is closely related to patterns of human activities, and its regularity directly affects traffic fluctuations. E.g., people’s daily routines (such as waking up in the morning and going to bed at night) create distinct traffic correlations over time. To further investigate this phenomenon, this paper selects three representative grids from different areas of Milano for analysis [14], including the city center (ID = 5051), urban area (ID = 3439), and suburban area (ID = 444). By calculating the autocorrelation coefficient, the paper analyzes the correlation of cellular traffic at different times in these grids. As shown in Fig. 1, these three grids exhibit significant traffic correlation at the 1st and 2nd lag times, indicating that the traffic at adjacent times is highly correlated. This analysis also reveals the diurnal periodicity of the traffic time series in these grids, with this periodicity being more pronounced at shorter time intervals.

2.2 Analysis of Multi-time Scale Information

In our in depth research and analysis of cellular traffic at base stations, we recognized that the factors influencing wireless service traffic in specific areas are diverse and complex. Relying solely on historical data to uncover patterns and trends may not be comprehensive enough. The differences in base station traffic data across various regions are primarily influenced by local users’ lifestyles and behavioral patterns, presenting a challenge for accurate prediction of base station cellular traffic. For instance, base stations in commercial areas exhibit high traffic on weekdays, but relatively lower traffic on weekends; residential area base stations also show different traffic patterns. Additionally, arrival of holidays can lead to significant fluctuations in base station traffic. All these

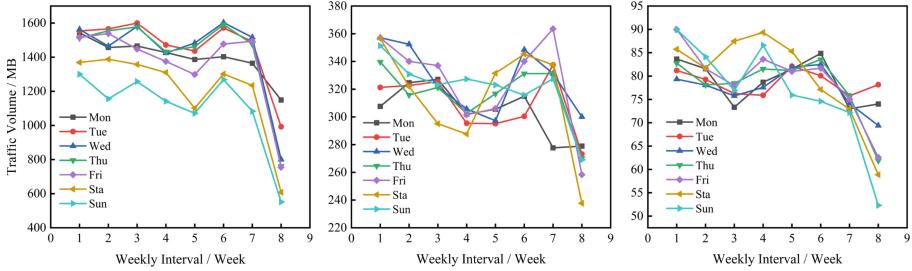


Fig. 2. Weekly flow data comparison

factors collectively impact the dynamic changes in base station network traffic values, hence they need to be considered comprehensively in predictions.

As shown in Fig. 2, according to the three different areas shown in Sect. 2.1 above By dividing this data into weekly segments, we notice significant differences in traffic from Monday to Friday. Particularly on weekends, traffic in the city center, urban areas, and suburbs all drop compared to weekdays. This phenomenon may be related to the more dispersed distribution of people across different areas on weekends compared to weekdays. Additionally, weekly traffic data also shows a certain periodic pattern.

Based on these observations, in predicting base station cellular traffic, in addition to considering historical traffic data, it is also necessary to take into account information about the week, as an important feature related to the traffic data of the prediction target point. This information, when used as covariates input into the Informer model, can effectively improve the accuracy of predictions.

3 Proposed Framework

This chapter introduces an innovative approach for predicting cellular traffic using multi-time scale information. This paper adopts the Informer prediction model, renowned for its effective handling of long-term dependencies. The focus is on considering information across multiple time scales to more accurately capture the dynamic characteristics of cellular traffic. Under a federated learning framework, we created a small augmented dataset using the original cellular traffic dataset, effectively overcoming the challenges of correlation and distribution differences in cellular traffic across various urban areas. This paper also adopted a gradient based priority iterative pruning algorithm for model pruning. This strategy not only enhanced the efficiency of the model but also maintained its predictive accuracy.

3.1 Model Structure

Figure 3 shows the architecture of the Informer model, which consists of two major components: the encoder and the decoder. The encoder includes a positional embedding layer, a multi-head probabilistic sparse self-attention layer, and a convolutional distillation layer. It is primarily responsible for time encoding of the input sequence and processing these data through deep feature extraction. The decoder, on the other hand,

is tasked with receiving deep feature maps and cellular traffic data information from the encoder, identifying inter feature correlations, and realizing the final predictive output through a fully connected layer.

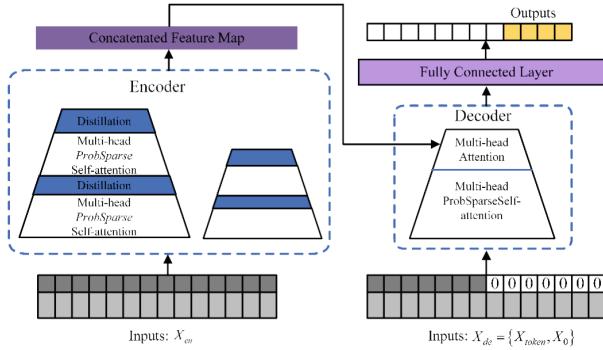


Fig. 3. Informer model diagram

3.2 The Input Structure of the Informer

In the cellular traffic time series x_i , the scalar value x_i^t at time t is projected into a d_{model} -dimensional vector, u_i^t , using d_{model} one-dimensional convolution filters with a kernel size of 3 and a stride of 1. Concurrently, the local timestamp at time t in the series x_i is represented as the fixed positional encoding in the sequence, expressed as:

$$PE(pos, 2i) = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right) \quad (1)$$

$$PE(pos, 2i + 1) = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right) \quad (2)$$

where pos represents the fixed position of the time variable t within the sequence, and i denotes the dimension of the data.

To comprehensively analyze factors affecting the relationships between traffic variables, the embedding layer of the model in this paper is composed of scalar projections, timestamps at different levels, and other integrated elements. The input part of the sparse self-attention model, which is used for long-term cellular traffic prediction, is formed by the three components depicted in Fig. 4.

The process begins by mapping traffic data into a higher dimension. This is accomplished through one-dimensional convolution of the original data. The next step is positional encoding, employing the same methodology used in the Transformer's positional encoding. The data's timestamp information is then transformed into high-dimensional data via a fully connected layer. Finally, these three elements are combined, resulting in the final input, as depicted in (3):

$$x_{feed[i]}^t = \alpha \mu_i^t + PE_{(L_X \times (t-1)+i)} + \sum_p [SE_{(L_X \times (t-1)+i)}]_p \quad (3)$$

where $i \in \{1, \dots, L_x\}$, α , is a hyperparameter used to balance the high-dimensional mapping of traffic data with the high-dimensional mapping of position and timestamp. If the data is already normalized, α can be set to 1.

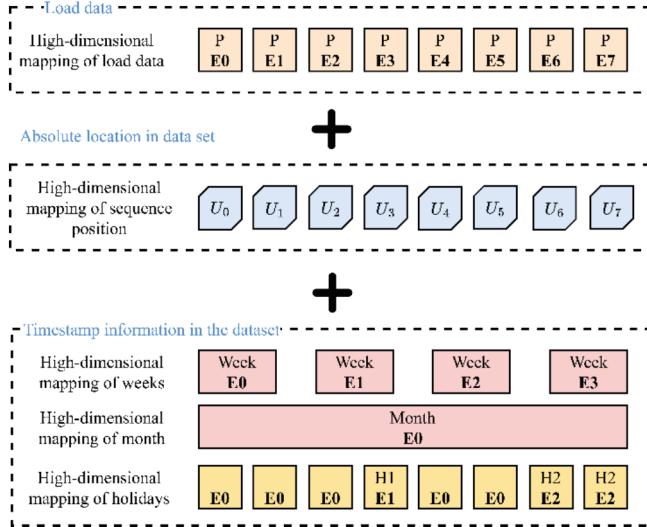


Fig. 4. Composition structure of inputs for the Informer model.

3.3 Gradient-Priority Iterative Pruning Algorithm-Based Approach

This chapter elaborates on the utilization of network pruning techniques to optimize deep learning models for enhancing computational efficiency in cellular traffic prediction tasks [15]. Network pruning is an iterative process focused on minimizing the model's size while sustaining its performance. It entails identifying a sparse subnetwork within a feedforward neural network through iterative pruning, which can then be retrained to achieve performance that rivals the original network. After numerous training iterations, the sparse subnetwork's performance can even exceed that of the original dense network, as outlined in Algorithm 1.

This algorithm starts with the input of near-global model parameters W_0 , mask m , and maximum number of iterations T . The model is trained with W_0 as input and returns the trained model weights W_k . The importance of gradients is assessed based on the accumulated absolute gradient $\nabla W_{\text{acc}}[i]$, computing a normalized importance score $\nabla W_{\text{norm}}[i]$ for each weight in W . In each iteration of the pruning phase, the importance score $\nabla W_{\text{norm}}[i]$ is recalculated, and the model is pruned and retrained accordingly. This iterative process continues until a decrease in performance is detected or the maximum iteration limit is reached. The final output is the pruned model parameters W^{final} .

Algorithm 1 Gradient-Priority Iterative Pruning Algorithm

```

1: Input:  $p$ : Pruning Rate,  $W_0$ : Initial Weights,  $T$ : Max Iterations
2: Output:  $W^{\text{final}}$ : Final Model Weights,  $m^{\text{final}}$ : Final Mask
3:  $W \leftarrow W_0$ 
4:  $m \leftarrow \text{MaskInit}( )$ 
5: Initialize  $\nabla W_{acc}$  to zero
6:  $t \leftarrow 0$ 
7: while  $t < T$  and error not increased do
8:    $W_k \leftarrow \text{Train}(W)$ 
9:   for each weight  $w_{ki}$  in  $W_k$  do
10:     $\nabla W_{acc}[i] \leftarrow \nabla W_{acc}[i] + |\nabla w_{ki}|$ 
11:   end for
12:    $S \leftarrow \sum_{i=1}^n \nabla W_{acc}[i]$ 
13:   for each weight  $w_{ki}$  in  $W_k$  do
14:     $\nabla W_{norm}[i] \leftarrow \nabla W_{acc}[i]/S$ 
15:   end for
16:   Compare Weights ( $\nabla W_{norm}$ ) to assess relative importance
17:   Prune:  $m \leftarrow m \odot W_k$ , with pruning rate  $p$ 
18:    $W \leftarrow m \odot W_0$ 
19:    $W_k^n \leftarrow \text{Retrain}(W)$ 
20:    $m^n \leftarrow m^n \odot W_k^n$ 
21:    $t \leftarrow t + 1$ 
22:    $W^{\text{final}} \leftarrow W_k^n$ ,  $m^{\text{final}} \leftarrow m^n$ 
23: end while

```

3.4 Data Augmentation in Federated Learning

Wireless service data from different base stations exhibit significant heterogeneity and are, in fact, non-independent and identically distributed in nature. Studies have shown that when client data is of the non-IID type, it can lead to performance degradation in federated learning algorithms [16]. This is because the server needs to consider the weight differences of client model parameters when performing model aggregation. To address the issue of performance degradation in the federated model on mobile edge computing servers due to the non-IID nature of regional traffic data, this paper proposes a data augmentation-based federated learning algorithm.

In this study, a strategy of enhanced data is adopted by creating small augmented datasets from the original cellular traffic dataset and generating a global shared dataset to overcome the challenges of non-independence and distribution in cellular traffic across different areas of the city. The strategy for the augmentation framework is as follows: First, the dataset used in the experiment is divided into weekly slices based on time indices. For each week's traffic, the statistical average for each time point is calculated and regarded as augmented data. Finally, the augmented data is normalized, with specific steps detailed in Fig. 5.

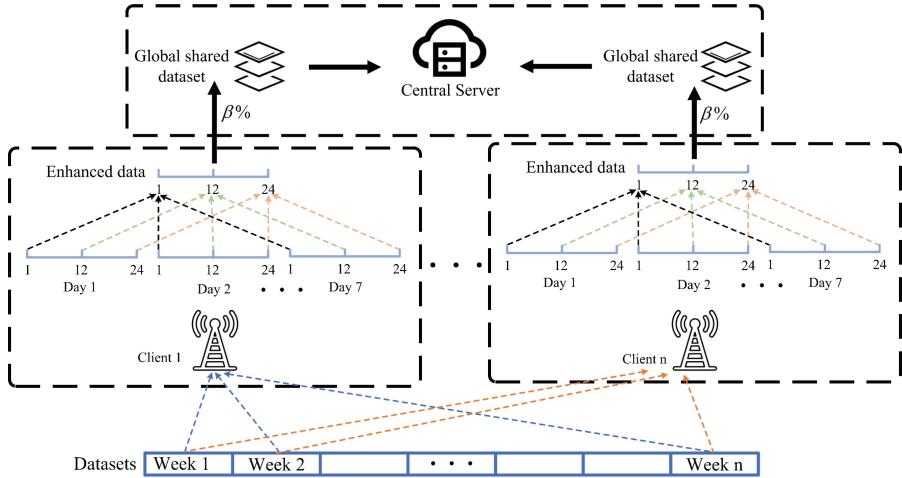


Fig. 5. Cellular traffic augmentation data.

In this study, we employ a globally shared augmented dataset strategy, allowing the central server to train a quasi-global model using augmented data from all clients, which serves as prior knowledge and ensures data privacy while enabling data sharing among clients. This approach, requiring a one-time implementation at federated learning initiation, reduces communication costs and allows for real-time adjustment of the data sharing ratio, denoted as $\beta\%$. Our strategy, more user-friendly than traditional data augmentation methods, effectively addresses traffic data heterogeneity.

3.5 Global Model Training Process

The global model aggregation at the central server end is one of the key steps in federated learning. This step aims to utilize local models on edge devices to construct the final global model. In order to balance the personalization of edge client models and the commonality of the global model, this paper introduces attention mechanisms to achieve model aggregation. The process of model aggregation using attention mechanisms is steps as follows:

1. Initialize global model: This paper uses the quasi-global model generated from training on the central server with enhanced data implemented in Sect. 3.4 as the initialization of the global model.
2. The client trains the model using local data and obtains local model updates. Let's denote the model update for client k as ΔW_k .
3. Upload local model updates: The client uploads ΔW_k to the server.
4. Compute attention weights: The server computes the weights for each client using attention mechanism. The contribution of each client to the global model is evaluated using cosine similarity. The attention weight α_k for client k can be calculated using

the following formula (4).

$$\alpha_k = \frac{\exp(\text{sim}(\Delta W_k, \Delta W_{ref}))}{\sum_{i=1}^K \exp(\text{sim}(\Delta W_i, \Delta W_{ref}))} \quad (4)$$

where $\text{sim}(\cdot, \cdot)$ represents the cosine similarity function, and ΔW_{ref} is the reference model update, with the central server trained using enhanced data.

5. Aggregate global model: The server aggregates the local model updates using the calculated attention weights to obtain the new global model. The update of the global model, denoted as ΔW_{global} , can be calculated using the following formula (5).

$$\Delta W_{global} = \sum_{k=1}^K \alpha_k \Delta W_k \quad (5)$$

6. Iterative update: Distribute the new global model parameters to each client and repeat steps 2–5 until the stopping criteria are met.

4 Experiments

4.1 Parameter Settings

In this study, the min-max normalization method was employed to preprocess the dataset, ensuring the consistency and efficiency of model inputs. For the model's loss function, Mean Squared Error (MSE) was chosen, which aids in the precise quantification of prediction errors. As an optimization strategy, the Adam optimizer was used in conjunction with a cosine decay learning rate strategy with warmup.

Parameters play an important role in the process of model construction, training and evaluation. The following will describe the parameter settings of model training and implementation. Specifically, the encoder input size and decoder input size are set to 4, indicating that the number of input features processed by the model each time is 4 when processing sequence data. The decoder output size is set to 1, which means that the output dimension of each decoding of the model is 1. The layers of encoder and decoder are set to 2 to increase the depth of the model and improve its fitting ability to the data. In the aspect of attention mechanism, eight attention heads are selected to focus on different parts of the input sequence in parallel, so as to capture more abundant information. In order to effectively use computing resources and balance training speed and model performance, the batch size is set to 64. At the same time, setting the sequence length to 96 means that the model will process the sequence data with the length of 96. During the training and evaluation process, the tag length is set to 48, indicating that the first 48 positions of each sequence are used to calculate the loss and gradient. The dimension of the model is set to 512, which determines the number and complexity of the internal parameters of the model.

4.2 Prediction Performance Analysis

In the experimental section, we randomly selected cellular traffic for analysis from two datasets covering 100 base stations. These experiments involved interactions between

local clients and the central server, with a total of 100 experimental rounds conducted. In each round, 10% of the samples were randomly drawn as the training data for the local clients, with a learning rate set to 0.001. The experimental results mainly showcase the training performance of the last round. To evaluate the effectiveness of the proposed approach, the following comparative experiments were conducted:

- Fed-LSTM (Federated Learning with LSTM Network) [17]: Combining the characteristics of federated learning and LSTM networks, aiming to improve data privacy protection while maintaining model performance.
- Fed-DenseNet (Federated Learning with DenseNet) [18]: Applying the principles of federated learning to the DenseNet deep learning model in the hopes of achieving better learning outcomes and data security.
- FedGSA [19]: A wireless network traffic prediction method based on a gradient similarity-based federated aggregation algorithm, aiming to balance the individual and common characteristics among client models during the federated aggregation process.
- FL-DeepAR [20]: A wireless network traffic prediction method based on federated learning and a deep autoregressive network, aiming to capture and process other relevant influencing factors besides the historical traffic sequences of base stations, thereby improving the prediction performance of the model.

The above experimental setups aim to comprehensively evaluate the effectiveness of the proposed solution and compare it with existing methods.

As shown in Table 1, even when only 1% of the augmented data is shared, the FedMTS framework outperforms the other methods on both datasets. The Fed-DenseNet, Fed-LSTM, FedGSA, and FL-DeepAR models exhibited basic predictive accuracy on the Milano and Trento datasets. Among them, the FL-DeepAR model showed the lowest MAE and MSE values on both datasets, demonstrating its effectiveness in handling time-series data. After introducing the FedMTS method, particularly when the β value is set to 50 and 100, there is a significant reduction in both MAE and MSE on the two datasets. This indicates that increasing the data sharing ratio appropriately with the FedMTS method can effectively improve predictive performance. When the β value is increased to 100% (i.e., complete sharing), the MAE and MSE on the Trento dataset improved by 37.41% and 17.15% compared to the baseline. This substantial improvement highlights the efficiency and superiority of the FedMTS method under the condition of complete data sharing. As the data sharing ratio increases (from 1% to 100%), the model's prediction accuracy gradually improves, indicating that appropriately increasing data sharing in a federated learning environment can significantly enhance model performance. The success of FedMTS can be attributed to the following reasons:

- Utilizing the Informer model within the federated learning framework, FedMTS effectively captures dynamic traffic characteristics by integrating multi-timescale information, such as weekly, monthly, and holiday patterns, into the cellular traffic prediction model. This not only optimizes the model's ability to model time series but also enhances prediction accuracy.

- By employing data augmentation techniques to process heterogeneous data, this paper addresses the non-independent and identically distributed (non-IID) issue in federated learning, resulting in a significant improvement in prediction accuracy.
- FedMTS employs gradient-based priority iterative pruning techniques to optimize the model structure, reducing computational complexity while maintaining excellent predictive capabilities.

Table 1. Comparison of Prediction Performance.

Methods	Milano		Trento	
	MAE	MSE	MAE	MSE
Fed-DenseNet	0.1853	0.2949	4.6658	1.1260
Fed-LSTM	0.1325	0.2562	4.3868	1.0968
FedGSA	0.1054	0.2260	4.0257	0.9572
FL-DeepAR	0.1032	0.2237	3.9451	0.9284
FedMTS($\beta=1$)	0.1098	0.2336	4.4751	1.0364
FedMTS($\beta=50$)	0.1079	0.2311	3.5971	0.8964
FedMTS($\beta=100$)	0.1030	0.2211	2.4692	0.7692

4.3 Pruning Performance Experimental Analysis

To further evaluate the predictive capabilities of different algorithms, this section will explore the impact of different pruning rates on the final model's effectiveness. The pruning rates are set at 10%, 20%, 40%, and 60%, corresponding to pruning iteration numbers of 12, 6, 3, and 2, respectively, with results as shown in Table 2. The results in Table 2 demonstrate that using magnitude-based iterative pruning achieves an increase in model accuracy while reducing the number of model parameters. Depending on the pruning rate, the extent of improvement in model effectiveness varies.

Table 2. Pruning Performance Experiment Analysis.

Pruning Rate	Ratio	Parameters Reduced ($\times 10^6$)	Milano		Trento	
			MSE	MAE	MSE	MAE
No Pruning	—	14.0564	0.1253	0.2485	3.4952	0.8642
10%	8/10	5.5481	0.1030	0.2211	2.4692	0.7692
20%	5/8	5.8559	0.1893	0.2396	3.1850	0.8134
40%	1/4	8.4535	0.1290	0.2513	3.5227	0.8721
60%	1/2	5.7716	0.1386	0.2675	3.6723	0.8891

Table 2 shows that as the pruning rate increases, the number of model parameters is significantly reduced, indicating that pruning effectively simplifies the model structure. For instance, when the pruning rate is at 60%, the number of parameters is reduced by nearly 59%. Under different pruning rates, the model's predictive accuracy (measured by MSE and MAE) varies. Notably, at a 10% pruning rate, the model outperforms the unpruned case on both Milano and Trento datasets in terms of MSE and MAE, showing the best predictive performance, which suggests that moderate parameter pruning can improve the model's predictive accuracy. However, as the pruning rate continues to increase, the model performance begins to decline. E.g., at a 20% pruning rate, the model performs worse on all metrics compared to a 10% pruning rate. When the pruning rate reaches 40% and 60%, despite the reduced number of parameters, the model's predictive accuracy on both datasets is lower than the unpruned case. This indicates that excessive pruning may impair the model's ability to capture and learn data features.

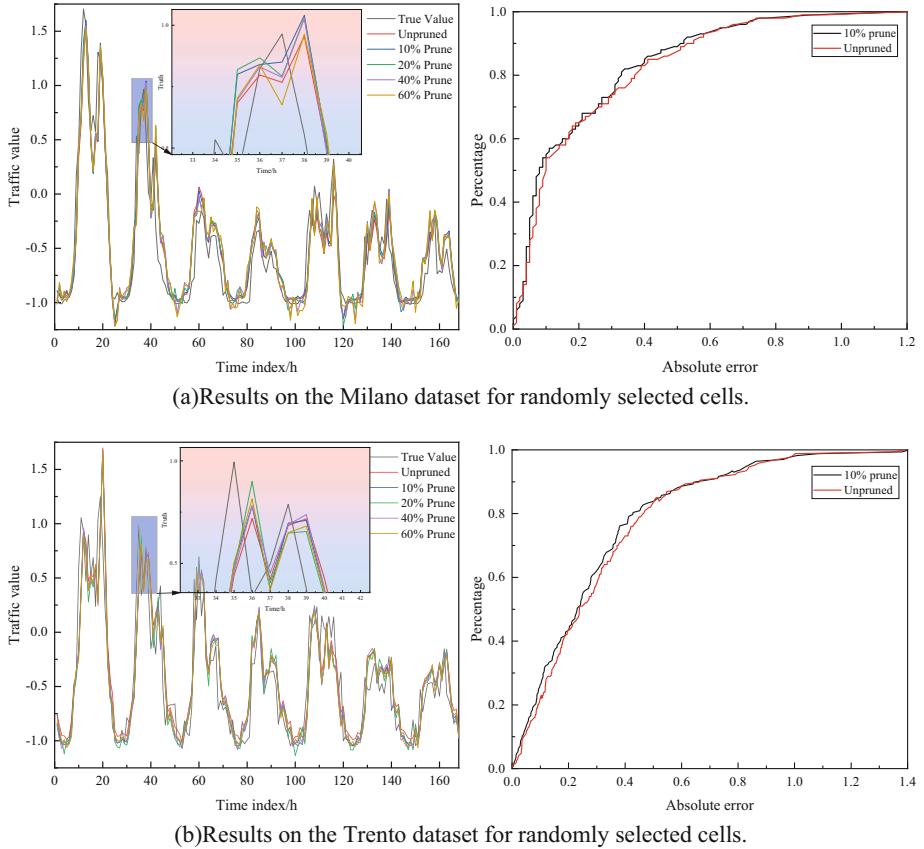


Fig. 6. Comparisons between predictions and the real values and the error analysis.

As shown in Table 1 above, we chose FL-DeepAR as the performance benchmark for comparison because it outperforms all other methods in Table 1. Regardless of the

wireless traffic conditions, the FedMTS framework with pruning techniques consistently outperforms FedLSTM in prediction performance. Particularly in high traffic and unstable traffic fluctuation environments, FedMTS exhibits relatively smaller prediction errors. Figure 6(a) and 6(b) respectively show the comparison between the predicted values obtained using different pruning rates and the actual cellular traffic values for randomly selected base stations in the Milano and Trento datasets. These figures also include the results of the Cumulative Distribution Function (CDF) of absolute prediction errors, reflecting the distribution proportion of the discrepancy between the predicted outcomes of the model and the actual ground truth values.

FedMTS framework achieves prediction errors of less than 0.5 in approximately 86% of cases, while the unpruned FedMTS is approximately 83%. On the Trento dataset, the FedMTS framework achieves prediction errors of less than 0.5 in approximately 84% of cases, while the unpruned FedMTS is approximately 82%. From these observations, it can be concluded that FedMTS not only surpasses FedLSTM but also provides more accurate prediction results. The experimental results clearly indicate that the proposed Informer based on network pruning techniques maintains excellent prediction capabilities when dealing with complex wireless traffic datasets, achieving both reduction in model parameters and improvement in prediction accuracy simultaneously.

5 Conclusion

This study enhances cellular traffic forecasting with the Informer model, incorporating periodic elements, federated learning, and lightweight methods, yielding satisfactory empirical results.

Despite the satisfactory outcomes of this research, there remain several intriguing directions for further exploration. Here are some potential prospects:

1. In-depth study of privacy protection in federated learning: Although the model in this paper can protect user data privacy within the federated learning framework, privacy protection continues to be a topic of ongoing focus.
2. Application to broader domains: While this research mainly concentrates on cellular traffic forecasting, the developed technologies have extensive potential for application in other areas, such as traffic management, meteorological prediction, and environmental monitoring.
3. Continued improvement of model efficiency: Despite having achieved network pruning and optimization, enhancing model efficiency further remains a significant objective.

Acknowledgment. This work was supported by the Fundamental Research Funds for the Central Universities, Inner Mongolia Autonomous Region under the project ‘Research on the Application of Federated Learning for Wireless Traffic Prediction with Privacy Protection (2023XKJX021)’ and the Key Research and Development and Achievement Transformation Project of Inner Mongolia Autonomous Region (2022YFSH0044).

References

1. Shen, L.H., Feng, K.T., Hanzo, L.: Five facets of 6G: research challenges and opportunities. *ACM Comput. Surv.* **55**(11), 1–39 (2023)
2. Hamilton, J.D.: *Time Series Analysis*. Princeton University Press, Princeton (2020)
3. Xu, F., Lin, Y., Huang, J., et al.: Big data driven mobile traffic understanding and forecasting: a time series approach. *IEEE Trans. Serv. Comput.* **9**(5), 796–805 (2016)
4. Huang, C.W., Chiang, C.T., Li, Q.: A study of deep learning networks on mobile traffic forecasting. In: 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), pp. 1–6. IEEE (2017)
5. Zhao, N., Wu, A., Pei, Y., et al.: Spatial-temporal aggregation graph convolution network for efficient mobile cellular traffic prediction. *IEEE Commun. Lett.* **26**(3), 587–591 (2021)
6. Liu, Q., Li, J., Lu, Z.: ST-Tran: spatial-temporal transformer for cellular traffic prediction. *IEEE Commun. Lett.* **25**(10), 3325–3329 (2021)
7. Zhu, J., Pan, C., Kong, Z., Shi, H.: Network traffic prediction based on prophet fusion multi-grain feature extraction. *Comput. Simul.* **38**(7), 475–480 (2021)
8. Li, T., Sahu, A.K., Talwalkar, A., et al.: Federated learning: challenges, methods, and future directions. *IEEE Signal Process. Mag.* **37**(3), 50–60 (2020)
9. Zhou, H., Zhang, S., Peng, J., et al.: Informer: Beyond efficient transformer for long sequence time-series forecasting. *Proc. AAAI Conf. Artif. Intell.* **35**(12), 11106–11115 (2021)
10. Zhang, C., Dang, S., Shihada, B., et al.: Dual attention-based federated learning for wireless traffic prediction. In: IEEE INFOCOM 2021-IEEE Conference on Computer Communications, pp. 1–10. IEEE (2021)
11. Li, L., Zhao, Y., Wang, J., et al.: Wireless traffic prediction based on a gradient similarity federated aggregation algorithm. *Appl. Sci.* **13**(6), 4036 (2023)
12. Harvard Dataverse. Italia, T. Telecommunications-SMS, Call, Internet-MI (2015). <https://doi.org/10.7910/DVN/EGZHFV>. Accessed 5 Nov 2022
13. Harvard Dataverse. Italia, T. Telecommunications-SMS, Call, Internet-TN (2015). <https://doi.org/10.7910/DVN/QLCABU>. Accessed 5 Nov 2022
14. Santos Escrache, E., Vassaki, S., Peters, G.: A comparative study of cellular traffic prediction mechanisms. *Wireless Netw.* **29**(5), 2371–2389 (2023)
15. Wang, Y., Li, D., Sun, R.: NTK-SAP: Improving neural network pruning by aligning training dynamics (2023). arxiv preprint [arXiv:2304.02840](https://arxiv.org/abs/2304.02840)
16. Zhao, Y., Li, M., Lai, L., et al.: Federated learning with non-IID data (2018). arxiv preprint [arXiv:1806.00582](https://arxiv.org/abs/1806.00582)
17. Dalgkitis, A., Louta, M., Karetos, G.T.: Traffic forecasting in cellular networks using the LSTM RNN. In: Proceedings of the 22nd Pan-Hellenic Conference on Informatics, pp. 28–33 (2018)
18. Zhang, C., Zhang, H., Yuan, D., et al.: Citywide cellular traffic prediction based on densely connected convolutional neural networks. *IEEE Commun. Lett.* **22**(8), 1656–1659 (2018)
19. Li, L., Zhao, Y., Wang, J., et al.: Wireless traffic prediction based on a gradient similarity federated aggregation algorithm. *Appl. Sci.* **13**(6), 4036 (2023)
20. Zhao, Y., Li, L., Gao, X.: Research on federated learning traffic prediction algorithm based on deep autoregressive networks. In: 2023 3rd International Conference on Computer Science and Blockchain (CCSB), pp. 41–47. IEEE (2023)



BGMA-Net: A Boundary-Guided and Multi-attention Network for Skin Lesion Segmentation

Cong Wu, Yao Li^(✉), Yuan Zhou, Haitao Gan, and Yi Han

School of Computing, Hubei University of Technology, Wuhan 430068, China
LiC721_999@163.com

Abstract. In clinical practice, the accurate diagnosis of skin lesions based on medical image segmentation is significant. However, skin lesion image boundaries are often coarse and blurred, and most traditional CNN-based segmentation networks cannot effectively use edge information of shallow layer to guide boundary segmentation of features. To address this problem, we propose a novel neural network called BGMA-Net. This network focuses on the complementarity between edge and object information by combining boundary guidance with multiple attention modules. It integrates boundary and object information, enhances feature representations, and improves skin lesion segmentation accuracy. Specifically, we first propose a simple yet effective boundary-guided attention gate (BGAG) module that integrates local edge information and global positional information to obtain rich boundary information. We further design an effective boundary segmentation attention (BSA) module to refine the boundary information of features. Finally, we propose a channel gated attention fusion (CGAF) module to combine encoder and decoder features, reducing semantic gaps and restoring fine-grained details of target objects. The evaluation of the ISIC 2017 and ISIC 2018 datasets demonstrates that BGMA-Net outperforms state-of-the-art methods, proving the reliability of this framework.

Keywords: Lesion Segmentation · Boundary Information · Attention Modules · Feature Fusion

1 Introduction

Skin cancer is the most common cancer worldwide, with melanoma being the deadliest [1]. In 2020, the American Cancer Society estimates that around 100,350 new cases will be diagnosed, resulting in over 6,500 deaths. Early diagnosis of skin cancer in patients can significantly improve survival rates and reduce mortality by 97% [2]. However, manual segmentation of conventional dermoscopic images is time-consuming and relies on specialized dermatologists. Automated melanoma segmentation in dermoscopic images is essential for computer-aided diagnosis (CAD) systems. Such systems [3–8] can assist dermatologists more efficiently and quickly, improving the accuracy of the analysis.

Accurate segmentation of skin lesion boundaries is a challenge because it is difficult to identify the boundary regions of lesions. Because of the lower contrast of surface hair, skin color, capillaries, shape and texture of lesions compared to non-lesioned regions [9]. Boundary information in the feature maps is effectively used to improve segmentation performance.

GFANet [10] designed a channel reverse attention module to refine the boundaries of skin lesions. MFSNet [11] used the reverse attention module to extract boundary cues and established the relationship between regions and boundary cues, thus obtaining satisfactory boundary information. ET-Net [12] provided edge features that better guide the extraction of discriminative features in the high-level layer through edge supervision. This helps the model to better understand and retain information about the boundary structure in medical images. While the methods mentioned above can obtain boundary information of the lesion region, they cannot effectively utilize the complementary relationship between edge information and object information.

Inspired by EGNet [13] and MFSNet [14], we designed the boundary-guided attention gate module (BGAG) and boundary segmentation attention module (BSA) to address this problem. We aim to leverage edge features of shallow layer to assist deep high-level features in accurately identifying boundary information. In the encoder, low-level features contain detailed spatial information, while high-level features provide sufficient semantic details [14]. Therefore, boundary-guided attention gate module (BGAG) combined shallow edge and deep semantic information to produce the boundary-guided feature map. Boundary segmentation attention module (BSA) was designed to incorporate boundary information into the segmentation process, resulting in more accurate segmentation results.

The encoder-decoder [15] structure is a popular architecture in deep learning. However, a concatenate of high-level features from the decoder and low-level features from the encoder via jump connections in U-Net is insufficient. Simple concatenation introduces semantic gaps and irrelevant noise, leading to inadequate feature extraction and insufficient acquisition of boundary information. Previous studies introduced effective mechanisms. For example, UNet++ [16] fused features at different scales through jump connections, while AttnUNet [17] added an attention gate mechanism to suppress irrelevant regions in the input image. HSH-UNet [18] also employed a multilevel and multiscale information fusion mechanism in the jump connection part. Although the above methods can effectively fuse useful information from features, they do not consider using multiple attention mechanisms to enhance the feature maps and filter out irrelevant information. Therefore, we designed the channel gate attention fusion module (CGAF) and introduced the channel spatial attention module (CSAM) [19] to acquire global and local information, filter out unnecessary information, and get helpful information.

In this paper, we proposed a boundary-guided and multi-attention neural network for skin lesion segmentation named BGMA-Net. The network consists three key modules: boundary-guided attention gate module (BGAG), boundary segmentation attention module (BSA), channel gate attention fusion module (CGAF). The main contributions of this paper can be summarized as follows:

1. We design a novel boundary-guided attention gate module (BGAG), which effectively integrates edge information from low-level features and semantic information from

high-level features to generate boundary feature maps with rich boundary information, while suppressing irrelevant noise.

2. We propose a boundary segmentation attention module (BSA). This module adaptively aggregates multi-level features, fully exploiting boundary information of features.
3. We introduce a channel gated attention fusion module (CGAF), which efficiently fuses low-level and high-level features, reducing the semantic gap between the encoder and decoder features and effectively recovering fine-grained details of target features.

1.1 BGMA-Net Structure

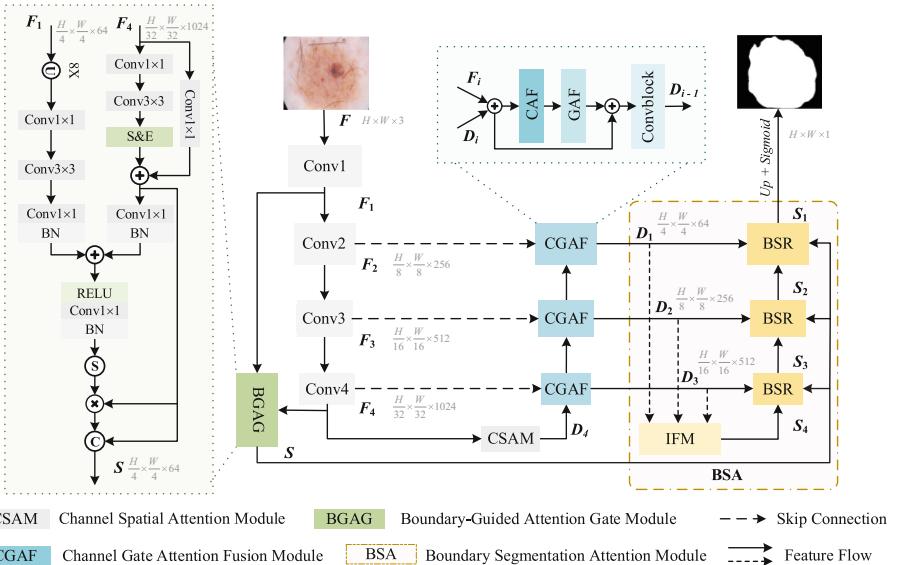


Fig. 1. Overall structure of the proposed BGMA-Net for the segmentation of skin lesions. We first extract multi-scale features by encoder Res2Net. The high-level feature F_4 and the low-level feature F_1 are fed into the boundary-guided attention gate module (BGAG) to capture boundary information and then refine the boundary information of features through the boundary segmentation attention module (BSA). At the same time, multi-level features from the encoder are effectively aggregated by the channel gate attention fusion module (CGAF), which preserves fine spatial information.

Figure 1 illustrates the proposed BGMA-Net for skin lesion segmentation, which integrates boundary guidance and multi-attention mechanisms. BGMA-Net is based on the U-Net [15]. Res2Net [20] is used as the encoder in BGMA-Net to extract multi-scale features. Shallow CNNs retain enough edge information [12]. Therefore, BGAG extracts edge information from the first layer convolution of the encoder. However, to obtain accurate boundary information, high-level semantic information is also necessary [11]. Specifically, BGAG first combines edge information from the first layer convolution

with deep semantic information from the fourth layer convolution of the encoder to generate a boundary-guided feature map S . Then, BSA performs adaptive aggregation of the features output from CGAF to enhance the boundary information of the features. Finally, BSA combines the boundary-guided feature map S output by BGAG to refine the boundary information of the features one by one and outputs the segmentation map S_1 of the network. The CGAF enhances feature representations, removes irrelevant information, and replaces the traditional convolutional layers in the decoder. CSAM [19] is introduced at the end of the encoder to gather global contextual information.

1.2 Boundary-Guided Attention Gate Module

The low-level features of the encoder retain more of the detail, but they lack sufficient semantic information and may contain unwanted noise. On the other hand, high-level features have more semantic information but lack precise details due to reduced resolution. Extracting enough feature information from the encoder's high-level feature can compensate for the spatial information lost during the pooling operation [21]. Segmentation performance can be improved by using the boundary information in the features. However, it can be challenging due to the irregular shapes of the skin lesion region, blurred boundaries, and complex background information. To address this problem, we design BGAG to generate the boundary-guided feature map using complementary spatial structural details and semantic information. The feature map generated by BGAG contains rich boundary information and is further improved in BSA. As shown in Fig. 1, BGAG incorporates a novel attention gate module that concentrates on significant boundary information and suppresses irrelevant details in the input image.

Like Eqs. (1) and (2), the BGAG introduces an SE layer to make the network sensitive to important edge features in the convolutional output F_1 of the first layer of the encoder, while suppressing insensitive features. Specifically, it employs convolution and residual connections. Subsequently, the BGAG conducts convolution operations on the fourth convolutional output F_4 of the encoder to extract the rich spatial information. Inputting F_1 and F_4 generates outputs F'_1 and F'_4 .

$$F'_1 = \text{SE}(\text{BN}(\text{Conv}_{3 \times 3}(\sigma(\text{BN}(\text{Conv}_{1 \times 1}(F_1))))) + \text{BN}(\text{Conv}_{1 \times 1}(F_1))) \quad (1)$$

$$F'_4 = \sigma(\text{BN}(\text{Conv}_{3 \times 3}((\text{BN}(\text{Conv}_{1 \times 1}(\text{UP}(F_4))))))) \quad (2)$$

Finally, we integrate local edge and global spatial information through the attention gates mechanism to obtain noticeable boundary features. The use of attention gates helps to suppress irrelevant regions in features while highlighting important boundary information that are useful for a particular task. As shown in Eq. (3) and (4), we obtain the final output S .

$$Q_{att}(F'_1, F'_4) = \sigma(\text{BN}(\text{Conv}_{1 \times 1}(F'_1)) + \text{BN}(\text{Conv}_{1 \times 1}(F'_4))) \quad (3)$$

$$S = \text{Cat}(F'_1 * \delta(\text{BN}(\text{Conv}_{1 \times 1}(Q_{att}(F'_1, F'_4)))), F'_1) \quad (4)$$

where σ represents the ReLU activation function, δ represents the sigmoid activation function, BN denotes batch normalization, and UP indicates the up-sample.

1.3 Boundary Segmentation Attention Module

The continuous up-sampling operations of the decoder are based only on the high-level features, which causes it to miss the detailed boundary information from the low-level features [22]. Therefore, we use the boundary-guided feature map S outputted by BGAG to guide the BSA for segmentation. S can provide adequate boundary information for the high-level features in the decoder. Figure 2 shows that our proposed BSA consists of the information fusion module (IFM) and boundary segmentation reverse attention module (BSR). IFM integrates useful information from multi-level features, filtering out redundant information. BSR progressively refines boundary information in the image, enhancing network segmentation performance.

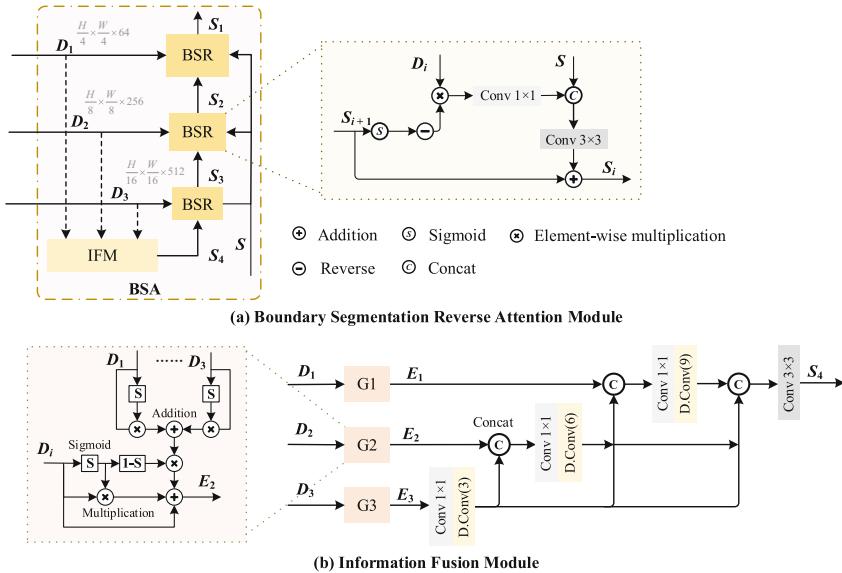


Fig. 2. Structure of boundary segmentation attention module (BSA) that consists of BSR and IFM. (a) boundary segmentation reverse attention module (BSR), (b) information fusion module (IFM). BSR refines boundary information in features. IFM selectively aggregates features with different levels. E_1 , E_2 , and E_3 represents the output of G_1 , G_2 , and G_3 , respectively.

IFM utilizes the gating mechanism [23] to selectively fuse useful information from different levels and filter out irrelevant information. As shown in Fig. 2(b), three features, D_1 , D_2 , and D_3 from the output of CGAF, first are inputted into IFM. Then, these features are aggregated using the simple addition and gating mechanism (G_i ; $i = \{1, 2, 3\}$). For the i -th input feature D_i ; $i = \{1, 2, 3\}$, the output E_i adaptively selects complementary information from the remaining features D_j ($j \neq i$), as shown in Eq. (5).

$$E_i = (1 + \delta(D_i)) * D_i + (1 - \delta(D_i)) * \left(\sum_{j \neq i} (D_j) D_j \right) \quad (5)$$

IFM integrates three complementary output features obtained at different levels through convolution and concatenation operations, getting the final output S_4 .

BSA mines boundary cues using the reverse attention (RA) module. This module establishes the relationship between regions and boundary cues. This circular collaboration mechanism between regions and boundaries calibrates misaligned predictions, improving segmentation accuracy. Figure 2(a) illustrates how BSR refines feature boundary information by using reverse attention operations on the boundary-guided feature S from BGAG, the features D_i from CGAF, and the multi-level feature S_4 from IFM. RA is used to uncover distinct melanoma areas, following the concept of progressively erasing foreground regions [21].

1.4 Channel Gate Attention Fusion Module

In U-Net [15], the direct fusion of shallow low-level information with up-sampled deep semantic information via jump connections may result in semantic gaps. Therefore, our proposed CGAF can efficiently fuse the features from the encoder and decoder. CGAF reduces the semantic gap between encoder and decoder features and effectively restores fine-grained details of target objects. As shown in Fig. 3, our CGAF module consists of the channel attention fusion module (CAF), the gate attention fusion module (GAF), and the convolution block (ConvBlock).

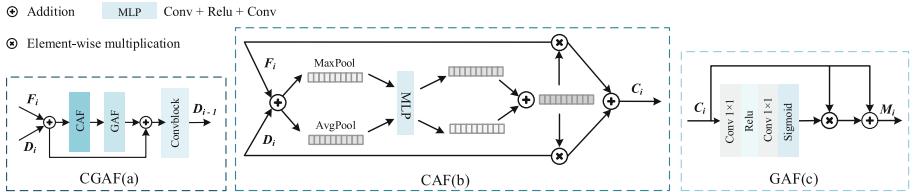


Fig. 3. Structure of channel gate attention fusion module (CGAF) that consists of CAF and GAF. (a)channel gate attention fusion module (CGAF), (b)channel attention fusion module (CAF), (c)gate attention fusion module (GAF). CAF determines which features to focus on. GAF emphasizes the boundary details of features. C_i ($i = 1, 2, 3$) represents the output of CAF at different stages; M_i ($i = 1, 2, 3$) represents output of GAF at different stages.

As shown in Fig. 3(b), CAF identifies the features to focus on and refines them. First, the fused features perform two parallel operations: maximum pooling and average pooling. Subsequently, the outputs pass through a shared MLP module and ReLU activation function, resulting in two activated results. These outputs are added element-wise and processed by a sigmoid activation function to generate the channel attention map. Finally, the encoder and decoder features are multiplied with the channel attention map separately and then added to produce the final feature map C_i .

GAF enhances the ability of the model to recognize boundaries, as illustrated in Fig. 3(c). First, the GAF input C_i pass convolution and sigmoid function operations to obtain weight values, which are then element-wise multiplied with the input C_i . Subsequently, the result is added to the C_i to generate the output M_i .

2 Experiments

2.1 Datasets and Implementation

We used two public skin lesion datasets, ISIC 2017 and ISIC 2018, to evaluate the performance of our method compared to the state-of-the-art methods. ISIC 2017 consisted of 2000 training images in JPEG format, 600 test images, and 150 validation set lesion images of the skin. The ISIC 2018 dataset was collected from several reputable medical centers and was collected and acquired through various devices. It contains 2594 RGB color images, becoming a major benchmark for evaluating medical imaging algorithms. We resampled all ISIC datasets to 224×320 pixels and divided ISIC 2018 into a training set (70%), a validation set (10%), and a test set (20%).

Our network was implemented using PyTorch, with an NVIDIA GeForce RTX 1080Ti graphics card with 11 G of memory. We use Adaptive Moment Estimation (Adam) as the overall optimizer, using the CosineAnnealingWarmRestarts ($T_0 = 10$, $T_{mult} = 2$) learning rate strategy. The initial learning rate was 0.001, and the batch size was 10. All networks were trained for 250 epochs, and the model with the highest JI index on the validation set was used as the final model. For data augmentation strategies, we used three random operations: rotation within the range of -30 to 30 degrees, vertical flipping, horizontal flipping, and random cropping to avoid overfitting.

2.2 Evaluation Metrics

To evaluate the performance of BGMA-Net. in this study, we use five widely used metrics, including Jaccard Index (JI), Dice similarity score (DSC), Accuracy (Acc), Sensitivity (Sen), and Specificity (Spe), which are shown in Eqs. (6), (7), (8), and (9).

$$JI = \frac{TP}{TP + FN + FP} \quad (6)$$

$$DSC = \frac{2 \times TP}{2 \times TP + FN + FP} \quad (7)$$

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \quad (8)$$

$$Sen = \frac{TP}{TP + FN}, \quad Spe = \frac{TN}{TN + FP} \quad (9)$$

TP, TN, FP, and FN represent the numbers of true positives, true negatives, false positives, and false negatives, respectively. DSC and JI evaluate the similarity between the segmentation results and the ground truth. JI mainly reflects the overlap of pixels between the predicted mask and the ground truth mask, which is the most crucial evaluation metric for segmentation tasks.

2.3 Experimental Results

As shown in Table 1 and Table 2, we present the comparison results of BGMA-Net with other state-of-the-art (SOTA) methods on the ISIC 2017 and ISIC 2018 datasets. All models adopt the same preprocessing, training strategies, and hyperparameters and follow the same evaluation process.

Table 1. Comparison with State-of-the-Art Methods on the ISIC 2017 Dataset.

Method	Para(M)	JI/%	DSC/%	Acc/%	Sen/%	Spe/%
U-Net [15]	34.53	76.21	84.82	93.16	83.81	97.48
UNet++ [16]	36.63	76.33	84.71	93.26	83.12	97.21
AttnUNet [17]	34.88	76.31	84.68	93.06	82.41	97.08
CANet [24]	2.78	76.54	84.92	93.21	83.75	96.42
CPFNet [25]	43.27	76.71	84.60	93.12	81.21	97.10
MFSNet [11]	33.12	76.78	84.99	93.40	81.99	97.50
Ms RED [26]	4.43	77.23	84.90	93.32	81.22	96.63
MALUNet [27]	0.17	74.02	83.11	92.41	81.02	96.56
GFANet [10]	24.20	77.08	84.64	93.33	82.46	97.53
HSH-UNet [18]	18.80	75.40	83.88	92.43	82.80	96.98
BGMA-Net	16.39	77.81	86.10	94.01	85.41	97.78

Table 2. Comparison with State-of-the-Art Methods on the ISIC 2018 Dataset.

Method	JI/%	DSC/%	Acc/%	Sen/%	Spe/%
U-Net [15]	81.28 ± 0.43	88.56 ± 0.42	95.36 ± 0.17	89.63 ± 0.80	97.16 ± 0.23
UNet++ [16]	82.01 ± 0.35	89.02 ± 0.31	95.71 ± 0.31	89.82 ± 0.53	97.28 ± 0.21
AttnUNet [17]	82.09 ± 0.46	89.11 ± 0.30	95.76 ± 0.17	89.78 ± 0.33	97.57 ± 0.19
CANet [24]	82.05 ± 0.61	88.96 ± 0.50	95.47 ± 0.28	89.76 ± 0.63	97.45 ± 0.24
CPFNet [25]	83.41 ± 0.32	90.18 ± 0.21	96.11 ± 0.15	90.44 ± 0.42	97.45 ± 0.11
MFSNet [11]	83.49 ± 0.33	90.12 ± 0.27	96.28 ± 0.21	90.68 ± 0.53	97.32 ± 0.10
MALUNet [27]	82.05 ± 0.48	89.07 ± 0.42	95.71 ± 0.26	89.83 ± 0.63	97.33 ± 0.22
Ms RED [26]	82.83 ± 0.41	89.52 ± 0.39	95.94 ± 0.36	90.38 ± 0.65	97.26 ± 0.23
GFANet [10]	83.58 ± 0.35	90.22 ± 0.29	96.31 ± 0.21	90.80 ± 0.32	97.25 ± 0.24
HSH-UNet [18]	82.07 ± 0.33	89.18 ± 0.18	96.01 ± 0.07	91.07 ± 0.18	97.34 ± 0.15
BGMA-Net	84.12 ± 0.36	90.55 ± 0.33	96.47 ± 0.26	91.33 ± 0.40	97.74 ± 0.15

Comparing with State-of-the-Arts. As shown in Table 1, it can be clearly seen that our BGMA-Net outperforms these networks in all metrics. Compared with the best-competing networks Ms RED and GFANet, the proposed BGMA-Net improves the JI metrics by 0.58% and 0.73%, respectively. Table 2 quantitatively shows the performance of BGMA-Net and ten excellent networks at ISIC 2018, including three classical networks used for medical image segmentation and seven networks designed for skin lesions. The evaluation results in Table 2 are all means and standard deviations of the five-fold cross-validation, from which can be seen that BGMA-Net achieves the highest

scores in all evaluation metrics. JI is the official evaluation metric for the challenge. In Table 2, our BGMA-Net improves in this metric by 2.84% over U-Net, with the most competitive network GFANet by 0.54%.

BGMA-Net outperforms the comparative methods in Table 1 and Table 2, thanks to three components: BGAG, BSA, and CGAF. BGAG generates a boundary-guided feature map that provides rich boundary information and filters out irrelevant information, while BSA uses inverse and gated attention to merge the contextual information of multilevel features and refine the boundary information. Channel and spatial attention mechanisms in CGAF effectively fuse contextual information from different encoder and decoder levels. We show the number of network parameters in Table 1. Although adding multiple attention mechanisms increase the number of parameters, it achieves better accuracy.

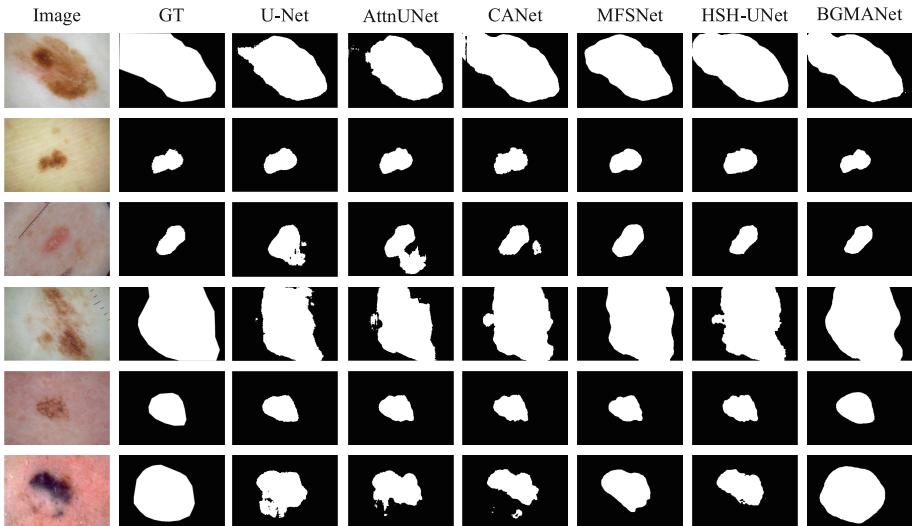


Fig. 4. The visualization results for different segmentation methods on the ISIC 2018 dataset are presented in this figure.

We visualize the performance of different networks on the ISIC 2018 dataset, as shown in Fig. 4. The first column is the input images, and the second column is the ground truth (GT), followed by the segmentation results of different networks. We selected several test samples with irregular shapes, blurry boundaries, and low contrast. In contrast, our BGMA-Net can segment lesions better, which may be attributed to BGAG and BSA being able to pay more attention to the boundary areas of lesions in segmentation, acquire rich boundary information, and address the issues of blurred boundary and low contrast of the lesion background. In addition, the CGAF can filter out irrelevant features and recover the fine-grained details of the target features, effectively solving the problem of irregular shapes. According to Table 1, Table 2, and the visualization results in Fig. 4, the performance of BGMA-Net is better than the above methods and can be used as a reliable framework for skin lesion segmentation.

2.4 Ablation Experiment

We conducted experiments by removing different components from BGMA-Net to demonstrate the effectiveness of each proposed module. We chose U-Net [15] with Res2Net [20] as the encoder as our Baseline and completed the design of our network architecture based on it. Table 3 gives the ablation results of the different modules on the ISIC 2018 dataset with five-fold cross-validation.

Table 3. Ablation Study with Different Modules on the ISIC 2018 Dataset.

Method	JI/%	DSC/%	Acc/%
Baseline	82.03 ± 0.29	88.89 ± 0.24	95.45 ± 0.13
Model 1: Baseline + CSAM	82.86 ± 0.55	89.68 ± 0.42	95.66 ± 0.36
Model 2: Baseline + BGAG	83.01 ± 0.11	89.75 ± 0.07	95.97 ± 0.03
Model 3: Baseline + BSA	83.06 ± 0.15	89.82 ± 0.13	95.93 ± 0.06
Model 4: Baseline + CGAF	83.02 ± 0.21	89.84 ± 0.11	96.05 ± 0.04
Model 5: Model 2 + BSA	83.39 ± 0.25	90.11 ± 0.25	96.27 ± 0.22
Model 6: Model 2 + CGAF	83.27 ± 0.22	90.12 ± 0.21	96.22 ± 0.28
Model 7: Model3 + CGAF	83.55 ± 0.23	90.28 ± 0.24	96.16 ± 0.21
Model 8: BGMA-Net	84.12 ± 0.36	90.55 ± 0.33	96.47 ± 0.26

As shown in Table 3, adding BGAG to the baseline improved the JI metrics by 0.98%. In addition, adding BGAG in Model 3 and Model 4 improved the JI metrics by 0.33% and 0.25%, respectively, using of BGAG improved model performance. BSA is essential because removing it can decrease DSC, JI, and other evaluation metrics. Removing BSA from Model 5 decreased JI by 0.38%. Adding BSA to the baseline improves JI by 1.03%. Adding CGAF to the baseline increased JI by 0.99%. Similarly, adding CGAF to Model 2 improved JI by 0.26%. These improvements show that adding CGAF to the baseline enhances performance. By adding CSAM to the baseline, the DSC metric improves by 0.79%, and the JI improves by 0.83%. These results show that introducing four modules can improve the performance of BGMA-Net.

3 Conclusion

In this paper, we introduce a novel boundary-guided and multi-attention network (BGMA-Net), for accurate and reliable segmentation of skin lesion images with blurred boundaries. The proposed model comprises the BGAG, BSA, and CGAF. BGAG combines shallow edge and deep semantic information to generate the boundary-guided feature map. BSA refines boundary information by combining features from CGAF and the boundary-guided feature maps output by BGAG, resulting in segmentation maps with rich boundary information. CGAF in the decoder integrates contextual features, filtering out irrelevant ones. We evaluate our model on the publicly available ISIC 2017

and ISIC 2018 skin lesion datasets. The results demonstrate that the proposed BGMA-Net outperforms state-of-the-art methods in segmentation performance. In the future, we will consider introducing a multi-scale mechanism into the BGMA-Net to enable the network to handle large variations in the size of the lesion region, thereby improving segmentation performance for the variable-size skin lesion image segmentation task.

References

1. Ge, Z., Demyanov, S., Chakravorty, R., Bowling, A., Garnavi, R.: Skin disease recognition using deep saliency features and multimodal learning of dermoscopy and clinical images. In: Descoteaux, M., Maier-Hein, L., Franz, A., Jannin, P., Collins, D., Duchesne, S. (eds.) Medical Image Computing and Computer Assisted Intervention - MICCAI 2017. MICCAI 2017. LNCS(), vol. 10435. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66179-7_29
2. Siegel, R.L., Miller, K.D., Jemal, A.: Cancer statistics, 2019. CA Cancer J. Clin. **69**(1), 7–34 (2019)
3. Wu, C., Zhang, H., Chen, D., Gan, H.: A multi-scale and multi-attention network for skin lesion segmentation. In: Luo, B., Cheng, L., Wu, Z.G., Li, H., Li, C. (eds.) Neural Information Processing. ICONIP 2023. LNCS, vol. 14450. Springer, Singapore (2024) https://doi.org/10.1007/978-981-99-8070-3_41
4. Wang, L., Wong, L., You, Z.H., Huang, D.: AMDECDA: attention mechanism combined with data ensemble strategy for predicting circRNA-disease association. IEEE Trans. Big Data **10**, 320–329 (2023)
5. Wu, C., Zou, Y., Zhan, J.: DA-U-Net: densely connected convolutional networks and decoder with attention gate for retinal vessel segmentation. In: IOP Conference Series: Materials Science and Engineering. vol. 533. IOP Publishing (2019)
6. Wu, C., Li, S., Liu, X., Jiang, F., Shi, B.: DMs-MAFM+EfficientNet: a hybrid model for predicting dysthyroid optic neuropathy. Med. Biol. Eng. Compu. **60**(11), 3217–3230 (2022)
7. Wu, C., Long, C., Li, S., Yang, J., Jiang, F., Zhou, R.: MSRAformer: multiscale spatial reverse attention network for polyp segmentation. Comput. Biol. Med. **151**, 106274 (2022)
8. Wu, C., Zou, Y., Yang, Z.: U-GAN: generative Adversarial Networks with U-Net for Retinal Vessel Segmentation. In: 14th International Conference on Computer Science & Education, pp. 642–646 (2019)
9. Celebi, M.E., et al.: Automatic detection of blue-white veil and related structures in dermoscopy images. Comput. Med. Imaging Graph. **32**, 670–677 (2008)
10. Qiu, S., Li, C., Feng, Y., Zuo, S., Liang, H., Xu, A.: GFANet: gated fusion attention network for skin lesion segmentation. Comput. Biol. Med. **155**, 106462 (2023)
11. Basak, H., Kundu, R., Sarkar, R.: MFSNet: a multi focus segmentation network for skin lesion segmentation. Pattern Recognit. **128**, 108673 (2022)
12. Zhang, Z., Fu, H., Dai, H., Shen, J., Pang, Y., Shao, L.: ET-Net: a generic edge-aTtention guidance network for medical image segmentation. In: Shen, D., et al. Medical Image Computing and Computer Assisted Intervention - MICCAI 2019. MICCAI 2019. LNCS(), vol. 11764. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32239-7_49
13. Zhao, J., Liu, J., Fan, D., Cao, Y., Yang, J., Cheng, M.: EGNet: Edge guidance network for salient object detection. In: ICCV 2019, pp. 8779–8788 (2019)
14. Zhang, Z., Zhang, X., Peng, C., Xue, X., Sun, J.: ExFuse: Enhancing Feature Fusion for Semantic Segmentation. In: ECCV 2018, pp. 269–284 (2018)

15. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W., Frangi, A. (eds) Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015. MICCAI 2015. LNCS(), vol. 9351. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
16. 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. Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support. DLMIA ML-CDS 2018 2018. LNCS(), vol. 11045. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00889-5_1
17. Oktay, O., et al.: Attention U-Net: Learning Where to Look for the Pancreas (2018). arXiv: 1804.03999
18. Wu, R., Lv, H., Liang, P., Cui, X., Chang, Q., Huang, X.: HSH-UNet: hybrid selective high order interactive U-shaped model for automated skin lesion segmentation. Comput. Biol. Med. **168**, 107798 (2024)
19. Mou, L., et al.: CS2-Net: deep learning segmentation of curvilinear structures in medical imaging. Med. Image Anal. **67**, 101874 (2021)
20. Gao, S., Cheng, M., Zhao, K., Zhang, X., Yang, M., Torr, P.: Res2Net: a new multi-scale backbone architecture. IEEE Trans. Pattern Anal. Mach. Intell. **43**(2), 652–662 (2021)
21. Wang, H., Cao, P., Wang, J., Zaiane, O.R.: UCTransNet: rethinking the skip connections in U-Net from a channel-wise perspective with transformer. Proc. AAAI Conf. Artif. Intell. **36**(3), 2441–2449 (2022)
22. Wu, H., Chen, S., Chen, G., Wang, W., Lei, B., Wen, Z.: FAT-Net: feature adaptive transformers for automated skin lesion segmentation. Med. Image Anal. **76**, 102327 (2022). <https://doi.org/10.1016/j.media.2021.102327>
23. Li, X., Zhao, H., Han, L., Tong, Y., Tan, S., Yang, K.: Gated fully fusion for semantic segmentation. Proc. AAAI Conf. Artif. Intell. **34**(7), 11418–11425 (2020)
24. Gu, R., et al.: CA-Net: comprehensive attention convolutional neural networks for explainable medical image segmentation. IEEE Trans. Med. Imaging **40**(2), 699–711 (2021)
25. Feng, S., et al.: CPFNet: context pyramid fusion network for medical image segmentation. IEEE Trans. Med. Imaging **39**(10), 3008–3018 (2020). <https://doi.org/10.1109/TMI.2020.2983721>
26. Dai, D., et al.: Ms RED: a novel multi-scale residual encoding and decoding network for skin lesion segmentation. Med. Image Anal. **75**, 102293 (2022)
27. Ruan, J., Xiang, S., Xie, M., Liu, T., Fu, Y.: MALUNet: a multi-attention and light-weight UNet for skin lesion segmentation. In: BIBM 2022, pp. 1150–1156. IEEE (2022)



Graph Pre-training for Reconnaissance Perception in Automated Penetration Testing

Yunfei Wang¹ , Shixuan Liu² , Chao Zhang³ , Wenhao Wang² , Jiandong Jin⁴ , Cheng Zhu¹ , and Changling Zhou⁴

¹ National Key Laboratory of Information Systems Engineering,
National University of Defense Technology, Changsha, China
`{Wangyunfei, zhucheng}@nudt.edu.cn`

² National University of Defense Technology, Changsha, China
`{liushixuan, wangwenhao11}@nudt.edu.cn`

³ Tsinghua University, Beijing, China
`chaoz@tsinghua.edu.cn`

⁴ Computer Center, Peking University, Beijing, China
`{jiandong.jin, zclfly}@pku.edu.cn`

Abstract. In automated penetration testing (APT), agents are tasked with identifying attack targets and formulating appropriate action plans within partially-observed network environments. The reasoning over the network based on the information gathering from reconnaissance is essential. However, existing reasoning methods show considerable neglect for computer networks and their unique characteristics. Additionally, despite Graph Neural Networks (GNNs) demonstrated efficacy in modeling graph structures, the scarcity of adequately labeled network data adds complexity to the training of GNNs. We present a novel method, termed **Graph Pre-training for Reconnaissance Perception in Automated Penetration Testing (GPRP)**. This pioneering approach is designed to learn the invariant properties entailed in the structures and semantics of the computer networks from an extensive set of unlabeled and synthetic data during pre-training. Consequently, the resulting pre-trained model could swiftly adapt to target networks, after undergoing fine-tuning with very few network observations, and exhibits enhanced capabilities in reasoning network properties. Extensive experiments on both customized and FatTree networks articulate the efficacy of our model in tasks centered around network reasoning, such as node classification and link prediction tasks. Further verification of GPRP in a real-world local area network, underscores the practical usage of our method.

Keywords: Graph pretraining · Automated penetration testing · Graph neural networks · Reconnaissance perception

1 Introduction

Advancements in computer networks have spurred an increased urgency to fortify network infrastructures and ensure their robust security [26]. Automated penetration testing and intelligent defense mechanisms have become pivotal in the field of network security research as a response to this urgency. Penetration testers (for brevity, called testers

in this paper) are often modeled as agents that simulate malicious hackers operating within a partially observable environment. They attempt to progressively penetrate the network using various attack strategies, starting from any agreed entrypoints of the network. Notably, their decision-making processes are significantly restrained given the limited information entailed in the current network observations (with missing link and attributes) and data collected from reconnaissance activities. A potential solution to this problem is the implementation of network completion strategies, based on the observed network information. However, this area of network completion for computer systems remains under-investigated to date.

Graph Neural Networks (GNNs) have demonstrated remarkable capabilities in modeling graph-structured data [5]. They contribute to representation learning for graph data [12], benefiting a diverse array of applications such as node classification [15] and causal discovery [16]. As computer networks can be naturally represented as graph-structured data, they are suitable for modeling with GNNs [27].

However, learning to complete computer networks with GNNs faces two key challenges. The first challenge is the scarcity of real-world computer network data, arising from various factors such as data privacy concerns, heightened security risks and data inconsistencies. Releasing such data could lead to privacy breaches and heighten network vulnerability to cyberattacks. Additionally, inconsistencies, present in how data collectors format and store network data, further exacerbate the data deficiency problem. Therefore, to fulfill our objective of learning to complete the network effectively, we need to generate abundant informative synthetic network data that accurately simulates real-world networks. The second issue stems from the strong multi-source heterogeneity that naturally exists among real-world networks, restricting the modelling of computer networks with GNNs [13]. Firstly, the type of network varies across different organizations. Campus networks tend to be simple like tree networks, while networks in internet firms are complex and hybrid due to stringent safety and robustness needs. Secondly, the same type of network can undergo spatial or temporal shifts in node attributes and topological features. As an example of the former, users in research department might install unique programming software that is uncommon in the financial department. For the latter, company expansions could greatly increase node degrees.

Given the above challenges, we propose a novel framework based on graph pre-training for network reasoning tasks in automated penetration testing, termed as GPRP (shown in Fig. 1). More specifically, we generate a large set of synthetic networks that mimic a hybrid of several typical computer network topologies, to cope with the issue of data sparsity. Utilized as unlabeled samples during pre-training, this synthetic dataset enables GNNs to grasp the structural and semantic characteristics of computer networks, thus facilitating generalization to specific tasks with minimal fine-tuning on target graphs. This effectively solves the second challenge.

Our contributions can be summarized as follows:

- We create a network generator to yield a multitude of networks encompassing various types with diverse attributes and structures, informed by expert knowledge.
- We are the first to consider graph pre-training for cyber-security related downstream tasks. We focus on capturing the intrinsic dependency between structural and semantic

properties of computer networks for reconnaissance perception in automated penetration testing, and design cyber knowledge-aware techniques for attribute and link generation. To handle large-scale networks, we adopt subgraph sampling and mitigate accuracy loss through an adaptive embedding queue.

- We evaluate our model's effectiveness on two tasks related to reconnaissance perception in penetration testing, namely node classification and link prediction, in two within-domain and seven cross-domain transfer settings. The experimental results affirm our model's successful performance in computer network completion.

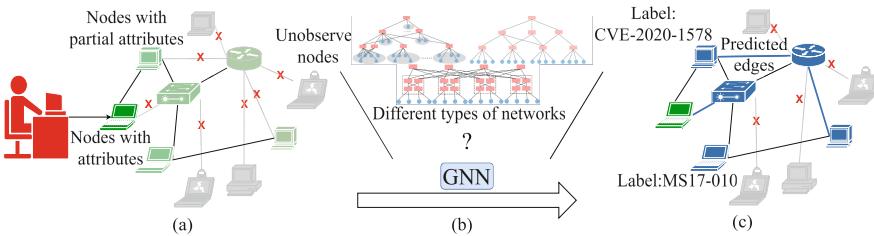


Fig. 1. Utilizing GPRP: (a) Penetration Testers are limited in their operations due to the partial observations of node attributes and links from their current location. (b) The GNN, pretrained on synthetic networks using GPRP, is used to predict attributes and links. (c) The results of the completed network assists the decision-making of Testers.

The remainder of this paper is organized as follows. Section 2 reviews the works of the related fields. Section 3 defines the problem and presents our framework. Section 4 presents the experiments on two within-domain and seven cross-domain settings to show the effectiveness of our model. We conclude the paper in Sect. 5.

2 Related Work

Automated penetration testing (APT) mimics hacker attacks to evaluate application, system, or network vulnerabilities. Despite tools emerging to aid testing efficiency, these largely focus on decision-making without deeply leveraging reconnaissance information. Network completion, grounded in partial observations, enhances reconnaissance data usage, aiding testers in decision-making processes. Although computer networks are inherently representable as graphs, the scarcity of labeled data has hindered the use of GNNs for these networks. Pre-trained models bridge this gap by utilizing, in its pre-training phase, abundant unlabeled data for initial learning phase. These models are further fine-tuned using a limited set of labeled data for subsequent tasks.

2.1 Penetration Testing

Penetration Testing (PT), key to cybersecurity protection, attracts vast research attention, with a focus on PT frameworks and methods.

PT Frameworks: Penetration Testing (PT) harnesses a variety of frameworks and standards such as the Information Systems Security Assessment Framework [3], and the Penetration Testing Execution Standard [19]. However, a noticeable discrepancy still remains between these frameworks and the tactics deployed in actual hacker attacks. To bridge this gap, models such as the Cyber Kill Chain [7] and ATT&CK [18] have been introduced. The Cyber Kill Chain outlines the critical steps adversaries must undertake to achieve their objective, including seven stages: Reconnaissance, Weaponization, Delivery, Exploit, Installation, Command & Control, and Action. Our research primarily focuses on the analysis of information obtained during the post-reconnaissance phase, as a means to facilitate more informed decision-making in subsequent stages of the process.

PT Method: PT methods generally partitioned into rule-based and model-based strategies [2]. (1) Rule-based PT: Penetration rules are derived from expert knowledge and systematically arranged into executable scripts, before applying rule matching for implementation [25]. (2) Model-based PT: This line employs techniques like attack trees [4] or attack graphs [8] to represent the sequential decision-making characteristics of PT and plan attack paths [14]. Some methods are based on traditional planning with PDDL translated from penetration knowledge [17]. Recently, reinforcement learning [26] based on Markov decision process (MDP) and partially observable MDP has become a research hot-spot in APT. These methods still exhibit a deficiency in processing the information collected from the reconnaissance phase, struggling to encapsulate the inherent interplay between structural and semantic facets of computer networks.

2.2 Network Completion

Network completion tasks, crucial in many real-world networks like social and biological networks [20], aim to complete missing nodes and edges in partially observable networks. This is mainly because incomplete network data can adversely affect downstream tasks like route planning and decision-making [11]. However, current network completion research primarily focuses on structural prediction (predicting the existence of nodes and edges) rather than considering attribute prediction [9]. Existing methods, albeit successful in social, biological, and citation networks, have not yet been applied to computer networks, nor have they been used for predicting network data attributes. The uniqueness of computer networks lies in their lack of unified construction standards, making it difficult to mine and analyze network structures and node features. In network security, node attributes such as firewall settings and installed systems carry significant weight in determining a tester's method. Therefore, both network structure and node attributes should be simultaneously completed in computer network completion tasks.

2.3 Graph Pre-training

A graph, a data structure that encapsulates sets of objects (nodes) and their links (edges), could mirror the characteristics of computer networks. GNNs are skillful in capturing the inter-dependencies on graphs by message-passing among nodes [27]. GNNs could be divided into categories: Recurrent GNNs, Convolutional GNNs, Autoencoder GNNs,

and Spatial-Temporal GNNs [23]. These models have proved their efficiency across various deep learning tasks.

Pre-trained language models, such as BERT, have transformed the NLP domain. This has inspired the emergence of pre-trained graph models (PGMs), appearing as a promising direction for graph domain. The prevalent challenges in (semi-)supervised graph learning are a lack of labeled data and issues with out-of-distribution generalization [13]. Pre-training the models on sizable corpora and refining them on specific tasks is believed to provide better initial performance and generalization for these tasks [5]. Broadly, PGMs can be categorized into two groups.

Pre-trained Graph Embeddings. The goal of these models is to learn quality graph embeddings for specific tasks. However, embeddings generated through these methods cannot be used to initialize other models for fine-tuning other tasks, thus limiting their broader application.

Pre-trained Encoders. As GNNs and Transformers evolve, PGMs are increasingly adopting transfer learning setups to pre-train a general-purpose encoder that can manage various tasks [24]. This method provides pre-initialized models for downstream tasks, thereby improving generalization and accelerating convergence on target tasks. Hu et al. employ a generative GNN pre-training framework that decomposes graph generation tasks into attribute generation and edge generation for pre-training [5]. Our approach aligns with this direction, but is designed specifically for computer networks.

3 Our Method

The system architecture of GPRP can be seen in Fig. 2. We start by generating a diverse set of attributed network graphs using a network generator. After sampling and masking links and attributes from these graphs, we pre-train our model on the altered data. Finally, we enhance the model’s effectiveness through fine-tuning for various downstream tasks. In this section, we begin by defining a computer network based on some core assumptions, and formalize the problem. Then, we introduce our model.

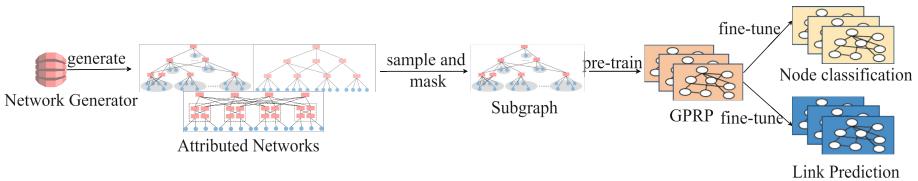


Fig. 2. The System Architecture of GPRP

3.1 Computer Networks and Problem

Computer Network Graph. Computer networks present as a multi-layered, heterogeneous network structure, which may include LANs, network layers, nodes that encompass switches, servers, etc. We define a computer network as a graph $G = (V, E, X)$.

V represents the set of nodes in the computer network. $X = \{X_1, X_2, \dots, X_{|V|}\}$ defines the feature set for nodes. We generally classify the nodes into two categories: one for data transmission –“switch”, the other for processing and storing data –“server” based on the functionality and role of nodes in computer networks. The additional attributes for each node encompass the IP address, port number, membership in a LAN, operating system details, and information about installed software and their respective versions. For instance, the attribute characteristics of node i are shown as follows:

```
 $x_i = \{ 'type': 'server', 'ip': '192.168.1.56', 'lan_id': '9',$ 
       $'system': 'linux', 'port^': [^' 20^', ^' 22^', ^' 80^'],$ 
       $'software_ver': [('Flink', '1.11.0'), ('PostgreSQL', '9.3'), ('Struts2', '2.0.0')]\}$ 
```

The collection of classification labels for all nodes is represented as $Y = \{y_1, y_2, \dots, y_{|V|}\}$. Labels can be flexibly set based on the task’s requirements. For instance, in order to predict the vulnerability present within a node, a standard procedure in automated penetration testing, the label for node i could be designated as $y_i = \{'CVE - 2020 - 17519', 'CVE - 2019 - 9193'\}$.

E represents the edge set within the computer network. Edges represent the logical connection relationships, which is judged by physical links or network protocols or fire-wall policies, etc. To enhance the pre-training model’s generality, we generate multiple network graphs of different scales using a custom network generator to capture universal relationships between node attributes and graph structures across different network graphs. We then use three distinct network structures during fine-tuning and testing, endorsing the versatility of our proposed method.

Problem Definition. In network attacks, after testers conduct scans and collect information, they generally only secure some partial information from the first-order neighbors based on the current node, such as open ports and installed software, and subsequently decide actions based on this partial sub-graph. Our pre-training model objective is learning a general GNN model f_θ based on multiple unlabeled network graphs $G = (V, E, X)$, where f_θ is for various downstream tasks over networks of within-domain or cross-domain settings. We utilize $P_\theta(G)$ to denote the likelihood modeling of the GNN f_θ for the graph G and optimize θ to maximize the likelihood.

3.2 Pre-train Process

To address the above issue, our model aims to 1) learn the intrinsic structure and attribute features from multi-source, heterogeneous, unlabeled graph data, and 2) use this pre-trained model to rapidly achieve strong performance in downstream tasks with minimal labeled data during the fine-tuning stage.

Current graph generation methods operate in an auto-regressive way, where nodes in the graph are added sequentially, and edges are added between new and existing nodes [5]. This process mirrors how a tester progressively navigates a network during penetration testing. Assuming the node ordering is a vector denoted as H , where i^H represents the ID of the i -th node in the permutation, $P_\theta(G)$ is equal to the expected

likelihood across all conceivable permutations:

$$P_\theta(G) = \mathbb{E}_H \left[P_\theta(X^H, E^H) \right] \quad (1)$$

where $G = (V, E, X)$, $X^H \in \mathbb{R}^{\{|V|\times d\}}$ denotes the attribute set of the permuted nodes, E^H stands for the edge set of the permuted nodes, and E_i^H represents the set of all edges connected to the node i^H . Given the stochastic nature of network structure generation and the fact that the order in which nodes are observed during penetration testing largely depends on the decisions and actions of the tester, we assume that the appearance of each permutation is of equal probability, we omit H in subsequent formal expressions. Given part of node attributes and edges, the generation of other node attributes and edges can undergo factorial decomposition:

$$P(X, E) = \prod_{i=1}^{|V|} P_\theta(X_i, E_i | X < i, E < i) \quad (2)$$

Formally, we use X_o and E_o to denote all observed node features and edges. Similarly, X_i^u and E_i^u denote the node attributes and edges that need to be generated, allowing us to rewrite the conditional probability as:

$$\begin{aligned} & P_\theta(X_i, E_i | X < i, E < i) \\ &= \sum_o P_\theta(X_i^u, E_i^u | X_i^o, E_i^o, X < i, E < i) \cdot P_\theta(X_i^o, E_i^o | X < i, E < i) \\ &= \mathbb{E}_o [P_\theta(X_i^u, E_i^u | X_i^o, E_i^o, X < i, E < i)] \\ &= \mathbb{E}_o [P_\theta(X_i^u | X_i^o, E_i^o, X < i, E < i) \cdot P_\theta(E_i^u | E_i^o, X \leq i, E < i)] \end{aligned} \quad (3)$$

This form of decomposition allows for the modeling of hidden relationships between node features and attributes. Term one in Eq. (3) represents the generation of the remaining node attributes X_i^u based on observed parts of node attributes X_i^o and edges E_i^o . Term two represents the generation of other edges based on the observed edges E_i^o and the generated node attributes X_i . Thus, based on the observed portions of node attributes X_i^o and edges E_i^o , we can generate the comprehensive characteristic representation X_i of the node and determine whether all edges E_i to node i exist.

In pre-training, we utilize the proposed framework GPRP to decompose graph generation into two interrelated parts:

- Given observed edges and some node attributes, the hidden attributes of nodes are generated.
- Given observed edges and generated node attributes, the remaining edges are generated.

Figure 3 illustrates the method of complete network graph generation for improved reconnaissance perception in APT. In a small-scale network scenario presented in graph (a), graph (b) represents a complete real network attribute graph. Testers, having full host data for entry nodes 1 and 2, discern interactions between node 3 and entry points from browser access records, among other data on nodes 1 and 2. Node 1 and 2's attribute

information is fully available, whereas we only partially know node 3's, abstracted into graph (c). Our generation process will initially generate the hidden attributes based on observed attributes and edges (graph (d)), then deduce the existence of edges based on observed edges and generated attributes (graph (e)), repeating these two steps until the generation task is complete (graphs (f, g)). After pre-training, we use the pre-trained model as the foundation for downstream tasks like node classification.

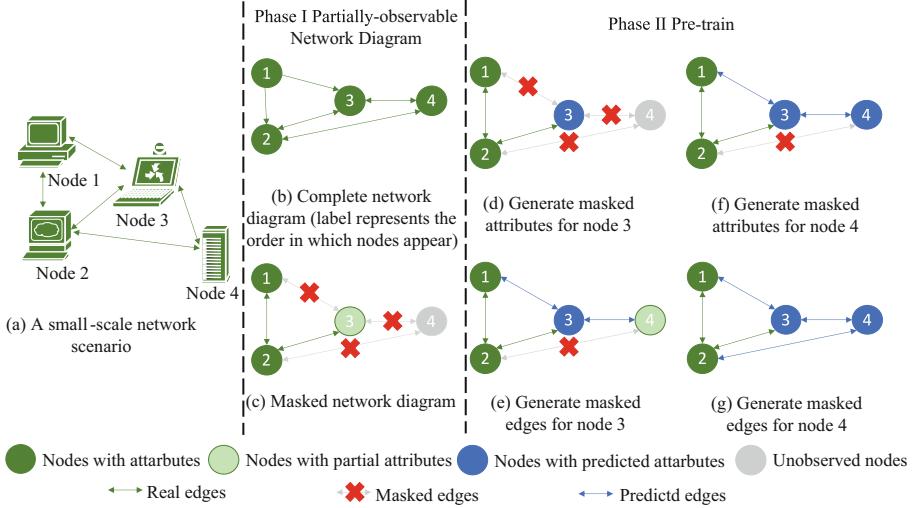


Fig. 3. An Example for Generative attributes and edges.

3.3 Pre-training and Fine-Tuning Setup

Here, we break the network graph completion task into two interrelated processes - attribute generation and edge generation. For the input network graph, running GNN once would obtain losses for attribute and edge generation simultaneously.

- Attribute generation task: We use Multi-Hot encoding for node feature representation and only return part of node information or have node data entirely missing during a sampling.
- Edge generation task: After acquiring representation for node attribute generation, we use that attribute for edge generation.

The pre-training graph data are input into the GNN model yielding outputs. We use d^{attr} and d^{edge} to denote the output embeddings of node attributes and edges, respectively. During GNN message passing, compared with d^{attr} , d^{edge} contains more information, we use d^{edge} to transmit information and generate edges and nodes through different decoders. $Dec^{attr}(\cdot)$ represents attribute generation decoder, which takes partial attributes of node as input and outputs masked attributes. $Dis(Dec^{attr}(\cdot), \cdot)$ denotes the L2-distance between the generated attributes of node and the true attributes. $Dec^{edge}(\cdot, \cdot)$ represents

link generation decoder, which takes the attributes of two nodes as input and outputs whether a link exists between the nodes.

For the attribute generation task, the loss function defining the metric between the generated and actual attributes is stated as:

$$\mathcal{L}_i^{attr} = Dis(Dec^{attr}(d_i^{attr}), X_i) \quad (4)$$

By minimizing this loss function, the probability of generating each node's hidden attributes maximizes, allowing the model to capture semantic information of attributes during pre-training.

Edges represent logical connection relationships between nodes in a network, influenced by factors like firewall policies, network protocols, and physical connections. The direct communication relationship between two nodes does not affect the direct communication between other pairs. So we assume that each edge is independent of the others and the edge generation can be decomposed as follows:

$$P_\theta(E_i^u | E_i^o, X \leq i, E < i) = \prod_{j^+ \in E_i^u} P_\theta(j^+ | E_i^o, X \leq i, E < i) \quad (5)$$

After acquiring the representation d^{edge} for the edge generation task, we model based on the similarity $Dec^{edge}(d_i^{edge}, d_j^{edge})$ between nodes i and j to evaluate the likelihood of an edge existing between two nodes. This is consistent with the rules we anticipate when defining the network, such as nodes within the same LAN are more likely to be interconnected and may share similar software and vulnerabilities. We use the negative contrastive estimation to calculate the likelihood for each linked node j^+ , computing the loss for all unlinked nodes S_i^- using the following formula:

$$\mathcal{L}_i^{edge} = - \sum_{j^+ \in E_i^u} \log \frac{\exp(Dec^{edge}(d_i^{edge}, d_j^{edge}))}{-\sum_{j \in S_i^- \cup \{j^+\}} \exp(Dec^{edge}(d_i^{edge}, d_j^{edge}))} \quad (6)$$

Optimizing \mathcal{L}^{edge} maximizes the probability of generating all edges, allowing the pre-training model to capture structural information from the attribute graph. Furthermore, we use a sampler to handle potential issues when dealing with large-scale computer network graphs. We use the heterogeneous graph model HGT [6] as the base model to model the heterogeneous computer network. The overall process of the pre-training model is shown in Algorithm 1.

4 Experiment

We test our algorithm's efficacy and adaptability using three distinct types of computer networks: customized network, FatTree network and real network. To further gauge the adaptability of our model, we tested it across diverse scenarios.

Specifically, we aim to address the following research questions: **RQ1**: Could GPRP solve the data sparsity issue? **RQ2**: Could GPRP solve the heterogeneity issue? **RQ3**: Could GPRP perform well on real network? **RQ4**: Could GPRP support other GNN models? **RQ5**: How GPRP reacts to scarce fine-tune labels?

4.1 Experimental Design

Network Topology. We employ a network generator to generate diverse synthetic networks, which are used across pre-training, fine-tuning, and testing phases. Predominantly, the network generator generates two types of networks: (1) Customized Networks, which are hybrid networks composed of existing typical network topologies such as star, tree, and mesh topology and (2) FatTree Networks, which represent a widely recognized canonical architecture for data center topologies [1].

Algorithm 1 Pretraining with GPRP

```

01: Input: attributed networks  $G = \{G_1, \dots, G_M\}$ , Epoch  $N$ 
02: OutPut: Pre-training model parameters  $\theta$ 
03: Initialize GNN as  $f_\theta$ , decoders for attribute generation and edge generation
   tasks as  $Dec^{attr}$  and  $Dec^{edge}$ .
04: for  $G_i \in G$  do
05:   For current network graph  $G_i$  sample subgraph  $G_i^j$  do
06:     For each node, note observed node attributes  $X_o$  and links  $E_o$ , as well as
        masked node attributes  $X_i^u$  and links  $E_i^u$ 
07:     Generate input embeddings for each observed node attribute and link, apply
        GNN  $f_\theta$  to generate output embeddings  $d^{attr}$  and  $d^{edge}$  for each node
08:     for  $i$  do
09:       Calculate the attribute generation loss  $\mathcal{L}^{attr}$  with Equation (4)
10:       Generate negative samples  $S_i^-$  for edge connections with negative sampler,
        and calculate  $\mathcal{L}^{edge}$  with Equation (6)
11:     end for
12:     Optimize  $\theta$  by minimizing total loss  $\mathcal{L} = \mathcal{L}^{attr} + \mathcal{L}^{edge}$ 
13:   end for

```

Customized Network: Based on expert knowledge, we establish network generation rules for customized network with following mechanism:

- Layered Structure: The network architecture embodies a layered defense strategy, comprising multiple network layers, LANs, and subnets interconnected through switches. These switches, as previously mentioned, serve as nodes that facilitate data transmission. Switches within a given network layer typically do not interconnect; instead, backup switches are deployed to illustrate the network's resilience. The arrangement of switches across different layers resembles a tree topology. Within individual LANs or subnets, hosts are connected directly, manifesting a mesh-like connection pattern.
- Intra-LAN Similarity: Nodes within the same LAN or subnet tend to be connected to a single or multiple switches, adopting a star-like network topology. These nodes often exhibit similarities, especially in terms of operating systems, installed software, and versions. This indicates that they are likely to share the same system vulnerabilities and software vulnerabilities, mirroring real-world situations where nodes within the

same department exhibit similar characteristics. Considering the diverse habits of users, some vulnerabilities may be resolved through the application of system patches or software updates, which is also simulated in our network model.

- Randomness in Connectivity: The variability in the number of nodes across different layers and LANs, as well as the number of nodes connected to various switches, underscores the stochastic nature of the network structure.

The architecture of the Customized Network is depicted in Fig. 4 (a).

FatTree Network: FatTree network architecture is a scalable, commodity data center network architecture. FatTree is a k -ary tree. Assuming the number of switch ports is k , the number of core switches is $k^2/4$, and the number of pods is k ; each pod is divided into two layers, each layer has $k/2$ switches. The upper layer is called the Aggregate Layer and the lower layer is called the Access Layer; half of the aggregation layer's $k/2$ ports connect to the $k/2$ core switches, and the remaining ports connect to the access layer's switch. The access layer operates with half of its ports connecting to the upper layer and the other half connecting to the host below. The structure of FatTree Network is shown in Fig. 4 (b).

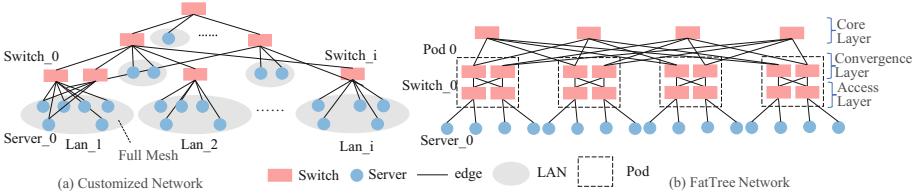


Fig. 4. Two types of networks

Node Attributes. Node attributes in the network are assigned according to their function and position. In this paper, we limit the total number of CVE vulnerabilities to 50. Each computer network contains between 900 and 1800 nodes. Despite the relatively small number of 50 CVEs, each vulnerability may impact dozens of software versions. The combinations of different software and versions corresponding to multiple CVEs will be even more numerous. Moreover, a plethora of information concerning software or versions may be either irrelevant or misleading when determining the existence of vulnerabilities, thereby complicating the problem further.

Datasets. We use network generator to create numerous customized and FatTree network attribute graphs. We use ten graphs for pre-training, one for fine-tuning, and one for testing. We also include a real-world local area network (Real LAN) in our dataset, with the generous assistance and explicit permission from our partner university.

Implementation. We use the Heterogeneous Graph Transformer (HGT) as the base model [6]. We also use other (heterogeneous) GNNs as the base model to test the effectiveness of our GPRP framework. For all base models, we set the hidden dimension to 200 and head number to 8. The setting of GNN layers is 4, and 128 nodes are sampled

per batch. Given that in a realistic attack, the tester can typically observe only the first-order neighborhood. Therefore, the sampling depth was set to 1 and the sampling width to 32. For pre-training, we utilize the AdamW optimizer with the Cosine Annealing Learning Rate Scheduler over 25 epochs. We select the model with the lowest loss as the pre-trained model for further fine-tuning.

Baselines. We compared GPRP with GCN [10], GAT [21], HGT [6], and HAN [22].

4.2 Pre-training and Fine-Tuning Setup

At present, there are two strategies for executing pre-training. The first strategy relies on performing both pre-training and fine-tuning on the very same graph. The other, more practical in nature, involves pre-training on one graph and subsequently fine-tuning on unseen graphs of identical type. However, as agents are initially unaware of the graph type before penetration. In this paper, we consider scenarios where the graph type could potentially change during fine-tune. To facilitate transfer learning between graph types, we offer multiple network graphs for pre-training to help the model learn the mutual structures across networks. Specifically, our experiment is designed to engage two within-domain and seven cross-domain transfer settings.

Downstream Tasks. Taking into account the needs of agents in network attacks, we use two downstream tasks to test our model: node classification and link prediction.

- **Node Classification:** When testers probes, they can typically acquire certain features of the reachable nodes. However, within a highly defensive network, intense scanning activities might trigger alerts for defenders. Given this incomplete node information, testers should make an attempt to determine which CVE may be present within this node, before initiating an action. To reflect this process, we require models to predict three most likely CVEs based on the observed node attributes. If one of these CVEs is the node label, it is considered accurate. The network contains a total of 50 different labels. The performance is gauged by the F1 score.
- **Link Prediction:** Once an agent identifies vulnerabilities in surrounding nodes, it then make a decision on which specific node to attack. Generally, it is more advantageous for future penetration process to attack a node with a high degree because it can reach more and potentially higher-valued neighbours. Through link prediction, the network can be completed, enabling agents to make more informed decisions when targeting nodes for attacks. We evaluate link prediction model performance by comparing predicted node links with actual ones, using the AUC (Area Under Curve) metric as our assessment metric.

4.3 Result

We summarize the results of each model in different experimental settings and different downstream tasks in Table 2. Overall, our model significantly improved the performance of all downstream tasks in various experimental settings.

Table 1. Categories of transfer settings considered

Pretrain	Finetune			
	Customized	FatTree	Mixed	Real
Customized	WD_1	CD_1	–	CD_2
FatTree	CD_3	WD_2	–	CD_4
Mixed*	CD_5	CD_6	–	CD_7
Real	–	–	–	–

WD and CD respectively denote within-domain and cross-domain settings. Settings marked by – are unrealistic settings to consider for penetration tasks. * Includes two types of networks: five customized networks and five FatTree networks.

Table 2. Performance on different downstream tasks by using different pre-training settings

	FT: Customized		FT: Fattree		FT: Real	
	NC (%)	LP (%)	NC (%)	LP (%)	NC (%)	LP (%)
GPRP(Pre: Customized)	77.3 ± 0.7	78.99 ± 0.5	71.66 ± 0.7	92.97 ± 1.8	61.11 ± 0.8	91.20 ± 1.4
GPRP (Pre: Fattree)	71.89 ± 7.6	77.45 ± 2.4	73.13 ± 0.6	93.58 ± 0.8	60.18 ± 1.1	93.52 ± 0.8
GPRP (Pre: MIXED)	76.23 ± 2.3	78.95 ± 1.4	70.64 ± 1.2	92.56 ± 0.9	66.98 ± 2.6	94.37 ± 0.4
HGT	50.16 ± 2.2	74.33 ± 2.8	69.68 ± 2.3	84.93 ± 3.5	54.41 ± 3.4	86.81 ± 1.8
GAT	34.37 ± 0.9	66.96 ± 5.1	44.67 ± 2.2	82.20 ± 3.0	27.96 ± 0.6	77.81 ± 0.7
GCN	30.21 ± 1.3	66.50 ± 3.3	39.97 ± 1.9	80.31 ± 1.1	28.03 ± 0.9	79.34 ± 1.2
HAN	29.45 ± 0.4	67.20 ± 1.0	33.08 ± 1.8	84.39 ± 1.3	21.09 ± 1.3	87.26 ± 0.9

RQ1: Could GPRP Solve the Data Sparsity Issue? For the two within-domain transfer settings (i.e., customized to customized and FatTree to FatTree), our model excels beyond other models in node classification and link prediction tasks. Furthermore, its performance in these two settings surpasses its own results in cross-domain settings. This is attributed to the independent and identically distributed property shared between the pretrain and fintune graphs in these settings. The model effectively learns structural and semantic properties, transferring this knowledge efficiently to downstream tasks with minimal fine-tuning, thereby addressing the first challenge impressively.

RQ2: Could GPRP Solve the Heterogeneity Issue? Our model sees a slight decrease in performance in cross-domain transfer settings compared to within-domain transfers. When pretraining is completed on a mixed network, the results are superior to those achieved using entirely distinct graph types. However, even in the most challenging scenarios (i.e. with highest level of heterogeneity as in customized to FatTree and FatTree to customized), our model significantly outperforms existing GNN models. This

indicates that GPRP can effectively adapt to O.O.D network graphs while preserving its effectiveness in heterogeneous network environments.

RQ3: Could GPRP Perform Well on Real Network? Here, we examine the most realistic scenario in which fine-tuning is on a real network. The model pre-trained on a mixed network attains the best performance on the real network, affirming that it can efficiently mine semantic properties and general structural attributes of network nodes from diverse, heterogeneous datasets. It is worth noting that pre-training on a single graph type can also effectively contribute to downstream tasks. Firstly, synthetic networks possess more complex structures, and GPRP may learn applicable structural rules in real networks. Secondly, cross-graph training in each pre-training epoch enables the model to discern universal network structures, potentially mitigating the impact of specific structures on GNN performance.

RQ4: Could GPRP Support Other GNN Models? We investigate whether other GNN architectures can be effectively applied to computer networks when combined with the pretraining. Hence, we also use GCN, GAT, and HAN as the base models for pretraining. To ensure fair comparison, hyperparameters are set to the same value. We evaluated their performances under two within-domain and one cross-domain transfer (mixed to real) settings. The summarized results are shown in Table 3: 1) pretraining with HGT delivers the top performance across all transfer settings, and 2) our pre-training framework enhances the downstream performance of all tested GNN models.

Table 3. Compare the pre-training Gain with different GNN architectures

Settings	Customized → Customized				Fattree → Fattree				Mixed → Real			
	NC (%)		LP (%)		NC (%)		LP (%)		NC (%)		LP (%)	
	F1	RG	AUC	RG	F1	RG	AUC	RG	F1	RG	AUC	RG
HGT	50.16	54.11	74.33	6.27	69.68	4.95	84.93	10.18	54.41	23.10	86.81	8.71
HGT-pre(GPRP)	77.3		78.99		73.13		93.58		66.98		94.37	
GAT	34.37	12.19	66.96	19.09	44.67	4.05	82.20	12.01	27.96	10.41	77.81	18.94
GAT-pre	38.56		79.74		46.48		92.07		30.87		92.55	
GCN	30.21	18.40	66.50	17.94	39.97	0.10	80.31	8.33	28.03	11.95	79.34	17.46
GCN-pre	35.77		78.43		40.01		87.00		31.38		93.19	
HAN	29.45	12.77	67.20	7.72	33.08	2.63	84.39	1.69	21.09	0.24	87.26	10.00
HAN-pre	33.21		72.39		33.95		85.82		21.14		95.99	

RG means Relative Gain (%)

RQ5: How GPRP reacts to scarce fine-tune labels? The original real network is a graph network with 931 nodes and we adjust the fine-tuning dataset to networks. We also include such performance comparison for other GNN architectures. From Fig. 5, it can be observed that all GNN models increase their downstream task performance with the expansion of training data volume and our model exhibits the best performance among all GNN models. Most importantly, our model can reach the performance of HGT trained

in a network of 2000 nodes with a minimal amount of data (a network of 300 nodes). When the number of nodes in the network graph expands to 931, the performance of the model is significantly improved, and the impact of data volume on model performance begins to decrease.

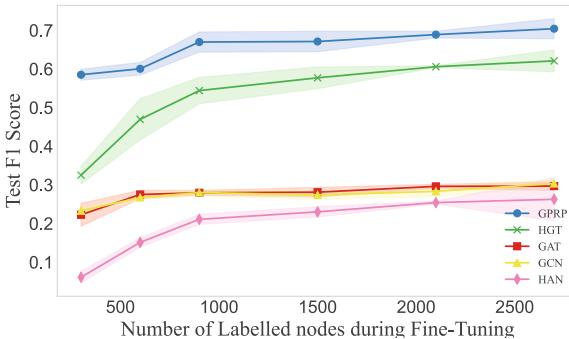


Fig. 5. Performance of GPRP with varying train samples

5 Conclusion

In this paper, we propose GPRP, a novel framework based on graph pre-training for network reasoning tasks in reconnaissance perception in automated penetration testing. We succeed in comprehensive network completion by breaking down the graph generation task into node attribute generation and edge generation. For the challenge of the scarcity of real-world computer network data, we employ a network generator to generate customized and FatTree Networks. Furthermore, our model quickly improves performance by fine-tuning on few labeled datasets. Additionally, through seven cross-domain transfer settings, we demonstrate that even if the datasets used in the pre-training and fine-tuning processes are distinct, or even if the network type was not present in the pre-training dataset, our model still performs well. This evidence shows that our model can meet the challenges of the strong multi-source heterogeneity in real network and achieve good adaptation in different environments.

Looking ahead, we note that the interpretability shortcomings inherent in GNNs' decision-making present a substantial bottleneck, especially in high-risk scenarios. To counter this, our upcoming research will explore the use of interpretable models to make decision processes more explainable. We believe this direction is pivotal to achieve a closer alignment with real-world conditions.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Al-Fares, M., Loukissas, A., Vahdat, A.: A scalable, commodity data center network architecture. *ACM SIGCOMM Comput. Commun. Rev.* **38**(4), 63–74 (2008)
2. Chen, K., Lu, H., Fang, B., Sun, Y., Su, S., Tian, Z.: Survey on automated penetration testing technology research. *J. Softw.* **35**(5), 2268–2288 (2023)
3. Group, O., et al.: Information systems security assessment framework. Open Information Systems Security Group (2006)
4. Hu, Z., Beuran, R., Tan, Y.: Automated penetration testing using deep reinforcement learning. In: 2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), pp. 2–10. IEEE (2020)
5. Hu, Z., Dong, Y., Wang, K., Chang, K.W., Sun, Y.: GPT-GNN: generative pretraining of graph neural networks. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1857–1867 (2020)
6. Hu, Z., Dong, Y., Wang, K., Sun, Y.: Heterogeneous graph transformer. In: Proceedings of the Web Conference 2020, pp. 2704–2710 (2020)
7. Hutchins, E.M., Cloppert, M.J., Amin, R.M., et al.: Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Lead. Issues Inf. Warfare Secur. Res.* **1**(1), 80 (2011)
8. Jha, S., Sheyner, O., Wing, J.: Two formal analyses of attack graphs. In: Proceedings 15th IEEE Computer Security Foundations Workshop. CSFW-15, pp. 49–63. IEEE (2002)
9. Kim, M., Leskovec, J.: The network completion problem: Inferring missing nodes and edges in networks. In: Proceedings of the 2011 SIAM International Conference on Data Mining, pp. 47–58. SIAM (2011)
10. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks (2016). arXiv preprint [arXiv:1609.02907](https://arxiv.org/abs/1609.02907)
11. Koskinen, J.H., Robins, G.L., Wang, P., Pattison, P.E.: Bayesian analysis for partially observed network data, missing ties, attributes and actors. *Soc. Netw.* **35**(4), 514–527 (2013)
12. Latecki, L.P.V.C.G.P.J.: Graph convolutional networks based on manifold learning for semi-supervised image classification. *Comput. Vis. Image Underst.* **277**, 103618 (2023)
13. Li, H., Wang, X., Zhang, Z., Zhu, W.: Out-of-distribution generalization on graphs: A survey (2022). arXiv preprint [arXiv:2202.07987](https://arxiv.org/abs/2202.07987)
14. Li, Q., Hu, M., Hao, H., Zhang, M., Li, Y.: Innes: an intelligent network penetration testing model based on deep reinforcement learning. *Appl. Intell.* **53**(22), 27110–27127 (2023)
15. Li, X., et al.: Graph neural network with curriculum learning for imbalanced node classification. *Neurocomputing* **574**, 127229 (2024)
16. Liu, S., Feng, Y., Wu, K., Cheng, G., Huang, J., Liu, Z.: Graph-attention-based casual discovery with trust region-navigated clipping policy optimization. *IEEE Trans. Cybern.* **53**, 2311–2324 (2021)
17. Sarraute, C.: Automated attack planning (2013). arXiv preprint [arXiv:1307.7808](https://arxiv.org/abs/1307.7808)
18. Strom, B.E., Applebaum, A., Miller, D.P., Nickels, K.C., Pennington, A.G., Thomas, C.B.: MITRE ATT&CK: Design and philosophy. In: Technical report. The MITRE Corporation (2018)
19. Team, P., et al.: The penetration testing execution standard documentation (2017)
20. Tran, C., Shin, W.Y., Spitz, A., Gertz, M.: DeepNC: deep generative network completion. *IEEE Trans. Pattern Anal. Mach. Intell.* **44**(4), 1837–1852 (2020)
21. Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y., et al.: Graph attention networks. *stat* **1050**(20), 10–48550 (2017)
22. Wang, X., Ji, H., Shi, C., Wang, B., Ye, Y., Cui, P., Yu, P.S.: Heterogeneous graph attention network. In: The World Wide Web Conference, pp. 2022–2032 (2019)

23. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **32**(1), 4–24 (2020)
24. Xia, J., Zhu, Y., Du, Y., Li, S.Z.: A survey of pretraining on graphs: Taxonomy, methods, and applications (2022). arXiv preprint [arXiv:2202.07893](https://arxiv.org/abs/2202.07893)
25. Xing, B., Gao, L., Sun, J., Yang, W.: Design and implementation of automated penetration testing system. *Application Research of Computers* (2010)
26. Zennaro, F.M., Erdődi, L.: Modelling penetration testing with reinforcement learning using capture-the-flag challenges: trade-offs between model-free learning and a priori knowledge. *IET Inf. Secur.* **17**(3), 441–457 (2023)
27. Zhou, J., et al.: Graph neural networks: a review of methods and applications. *AI open* **1**, 57–81 (2020)



Smart Contract Vulnerability Detection Based on Multi Graph Convolutional Neural Networks with Self-attention

Jiale Li, Xiao Yu^(✉), Jie Yu, Haoxin Sun, and Mengdi Sun

School of Computer Science and Technology, Shandong University of Technology,
Zibo 255049, Shandong, China
yuxiao8907118@163.com

Abstract. Smart contracts can contain vulnerable program code, making their security a significant concern in the past few years. Traditional detection techniques rely on expert defined features and source code analysis, but both have scalability issues and high false alarm rates. This paper presents a method for the detection of vulnerabilities in smart contracts, employing a multi graph convolutional neural network and self-attention mechanism. The method models the key function information, flow of control, and flow of data information of the smart contract source code as a semantic graph to emphasize the relationship between the flow of data and the flow of control in program operation. It predicts learning edges between nodes in the graph using an edge prediction network and constructs a multi graph of the smart contract semantics. Additionally, it utilizes a self-attention mechanism to gather and extract features from many layers to achieve precise detection of smart contract vulnerabilities. Outcomes of an experiment demonstrate that the proposed vulnerability detection method has significant advantages in identifying reentrant vulnerabilities and timestamp dependency vulnerabilities, with accuracy rates of 92.5% and 92.67%.

Keywords: Blockchain; Smart Contracts · Vulnerability Detection · Graph Neural Networks · Self-Attention

1 Introduction

A smart contract is a computer program that is both manually written and automatically executed on the blockchain. Security vulnerabilities may arise during the design and writing process. Additionally, the immutable nature of blockchain renders it impossible to modify or halt carrying out a smart contract once it has been published. The security dangers associated with smart contracts rise with their widespread use. Security vulnerabilities can be targeted by cyber attackers, resulting in economic losses and system instability [1, 2].

In the last few years, the utilization of deep learning methods to the identification of vulnerabilities in smart contract program code has emerged as a significant area of research. Some researchers [3] regard smart contract source code as natural language

and use text sequence model to analyze and process the source code. However, this method often fails to better get the sophisticated syntactic relations inherent in smart contracts. Some researchers [4] model smart contracts as graph structures and use graph neural networks for identifying vulnerabilities. However, the existing methods only learn based on the edges that already exist in the graph and fail to fully model the relationships between nodes, thus ignoring the more complex structural information in smart contracts, such as complex function calls.

With the advancement of graph neural networks, more and more researchers are applying them in different fields. Wang et al. [5] put forth a method based on unsupervised deep graph structure learning, which exploits the advantages of graph neural networks in topology learning and thus improves the ability to model complex graph structures. Wong et al. [6] proposed a Gaussian kernel-based approach and a linear optimization algorithm for inference of association information between miRNAs and lncRNAs. Zheng et al. [7] proposed a novel computational model named SPRDA, which was based on a structural perturbation approach and successfully predicted potentially disease-associated piRNAs.

To highlight the utilization of grammatical and contextual relationships in smart contract source code and model its intricate structural features, we propose MGCNA, a smart contract vulnerability detection tool based on Multi Graph Convolutional Neural Networks and Self-Attention. MGCNA converts key function information, function call relationships, variable passing, and flow of control information in smart contract source code into a semantic graph with a graph structure. Further, the code semantic graph is simplified into code semantic graph segments, and a node feature based neural network is used to extend the learning edges to construct new multi graphs in the code semantic graph segments. Then, these multi graphs are used as inputs to graph convolutional neural networks to train models for identify vulnerabilities in smart contracts. Experiments demonstrate that MGCNA is able to proficiently preserve the semantic features of the source code and extract the features of each node and the relationships between nodes in the graph, resulting in improved accuracy of vulnerability detection.

Our contribution is as follows:

1. A method for the detection of vulnerabilities in smart contracts is proposed, based on a multi graph convolutional neural network. This method utilises an edge prediction network to predict the learning edges and combine them into a multi graph. MGCNA makes full use of the syntactic and semantic content in the smart contract source code as well as the complex structural characteristics of the code, and solved the problem of feature loss during semantic graph truncation and insufficient modeling of smart contract source code, thus effectively improving the accuracy of smart contract vulnerability detection.
2. A self-attention module is introduced into the multi graph convolutional neural network to effectively extract and aggregate the feature information of different layers, which solves the problem that the smart contract graph structure may lose the local feature information during each iteration, and maintains the balance between the feature of the local nodes and the global graph features.

2 Related Work

The methods currently available for detecting smart contract vulnerabilities can be divided into two categories: based on deep learning and conventional techniques. The conventional techniques primarily consist of middle layer representation, formal verification, symbolic execution, and fuzzy testing.

Symbolic execution [8] technique that breaks down a contract into smaller parts and tracks the execution of each symbol. However, it suffers from path explosion and constraint solving difficulty [9]. The formal verification method [10] transforms the contract into a formal model and verifies the accuracy and safety of the contracts functions. The intermediate layer representation approach [11] transforms the smart contract source code into an intermediate language, and then analyzes this intermediate language to detect possible vulnerabilities and security problems in the contract. The fuzzy testing method [12] simulates various contract execution situations by producing a vast amount of random and diverse input data, and traces the execution process to find possible vulnerabilities and errors.

Tann et al. [13] proposed an approach to detect contract vulnerabilities using Long Short-Term Memory networks. This approach inputs smart contract opcodes into LSTM for feature learning, effectively capturing the inherent patterns and dependencies in the sequence of opcodes to detect smart contract vulnerabilities. A tool called ContractWard was introduced by Wang et al. [14]. It uses random forest and sampling algorithms, by analyzing the opcodes of the contracts to uncover vulnerabilities in smart contracts.

TMP and DR-GCN were presented by Zhuang et al. [4]. They employ graph neural networks for vulnerability identification and represent the source code of smart contracts as semantic graphs. Wu et al. [15] proposed a tool called Peculiar for detecting smart contract vulnerabilities using dataflow graphs. Accurately modeling the contextual and syntactic elements in smart contract source code remains a big problem, even if deep learning based vulnerability detection algorithms can drastically reduce labor costs and false alarm rates.

The source code for the smart contract is transformed into a code semantic graph representation in this paper. The semantic graph is then reduced to form code semantic graph segments. An edge prediction neural network is used to extend the learning edges, which are combined into a multi graph. This approach solves the problems of local feature loss caused by the reduction of the code semantic graph and the insufficient modeling resulting from the use of only existing edges for feature learning. Concurrently, the self-attention mechanism is used to gather and extract features from many layers, so that the nodes' own features lost in the iterative process are supplemented.

3 MGCNA Smart Contract Vulnerability Detection Methods

The method in the present paper converts the smart contract source code into a code semantic graph, and then utilizes neural networks to further learn and extend new edges based on the features between the nodes in the original graph to construct multi graph information. This method fully exploits the features of each node in the graph and the relationship between nodes. The method is divided into four steps:

- Construct code semantic graph. According to the key function information, function call relationship, variable passing and control flow information in the smart contract source code, the smart contract is converted into a code semantic graph with graph structure.
- Code Semantic Graph Reduction. This phase aggregates node information and removes redundant information by reducing the semantic graph. The aim is to remove unnecessary details while strengthening the node information by streamlining the graph structure.
- Predicting learning edges and combining multi graphs. Using neural networks to predict learning edges from the smart contract graph structure, fusing the original graph information and the learning edge information to combine them into a multi graph, making full use of the semantic and syntactic information in the smart contract source code.
- Model Training and Vulnerability Detection. Features are extracted from multi graphs using graph convolutional neural network and different levels of features are extracted and aggregated using self-attention mechanism and finally vulnerability classification is performed using fully connected layer. The overall detection framework is shown in Fig. 1.

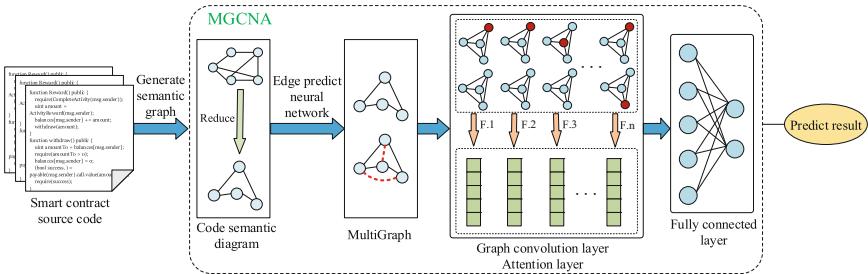


Fig. 1. Multi graph convolutional neural network with self-attention mechanism for smart contract vulnerability detection framework.

3.1 Semantic Graph Generation

To better model smart contracts and highlight the semantic connections between program components, inspired by Zhuang et al. [4], we convert eligible functions in smart contract source code into nodes of a semantic graph depiction and use the relationships between functions and variables as edges of the graph structure. Smart contract nodes are sorted into key nodes, secondary nodes, and fallback nodes to improve the graph neural network's ability to learn semantic information and understand smart contract behavior.

Key nodes are functions that have a significant influence on smart contract vulnerability detection, including built in functions and customized functions. As shown in Table 1, specifically, for reentrant vulnerabilities, we define the function called by

call.value, the function containing *call.value*, and the function calling *call.value* as critical nodes. And for timestamp dependency vulnerability, we define the function called by *block.timestamp*, the function containing *block.timestamp* and the function calling *block.timestamp* as critical nodes. Auxiliary nodes are the key variables in the smart contract, including ‘balances’, ‘participated’, ‘transactions’ and so on. The fallback function is essential for handling uncertain or non-compliant calls. Due to its association with most vulnerabilities, it is categorized as a separate class of nodes, the fallback nodes.

For the construction of edges, we define the calling relationships between functions and variables in smart contracts as three types: control flow edges, data flow edges, and fallback edges. Control flow edges represent the execution flow of statements, data flow edges show the passing of data between program elements, and the fallback edge is associated with the fallback function in the smart contract.

Table 1. Description of key nodes.

Vulnerability Type	Description
Reentrancy	Functions called by <i>call.value</i>
	Functions containing <i>call.value</i>
	Functions calling <i>call.value</i>
Timestamp dependency	Functions called by <i>block.timestamp</i>
	Functions containing <i>block.timestamp</i>
	Functions that call <i>block.timestamp</i>

3.2 Semantic Graph Reduction

Considering that different smart contract source codes produce different graphs, and the node information in the structure of the code semantic graph is of different importance for vulnerability detection, we have reduced the code semantic graph to improve the efficiency of vulnerability detection. For example, the semantic graph of smart contracts may have many secondary nodes, which are not critical for vulnerability detection, and the semantic graph with more redundant information is not favorable to model learning. As a result, we reduce the code semantic graph of smart contracts to form the contract code semantic graph segment. For the initial smart contract code semantic graph, we only keep the key nodes, pass the features of secondary nodes and fallback nodes on the nearest key nodes, and delete the secondary nodes and fallback nodes. After the reduction, each key node of the contract code semantic graph segment not only has its own features, but also aggregates the features of the neighboring nodes. As shown in Fig. 2, K1, K2, K3 are the key nodes of the semantic graph, S1, S2, S3 are the secondary nodes, and F is the fallback node. When the fallback node F is deleted, the features of F are passed to nodes K1 and K2 at the same time, and e8 is used as the incoming edge of K2 and e9 as the outgoing edge of K1.

The graph complexity can be significantly reduced by code semantic graph reduction, enabling the graph neural network to learn the key features of smart contracts more effectively. After code semantic graph reduction, the key nodes retain their own features while aggregating the features of secondary and fallback nodes, which helps to focus on the features of smart contract vulnerabilities and improve the accuracy of the model in detecting smart contract vulnerabilities.

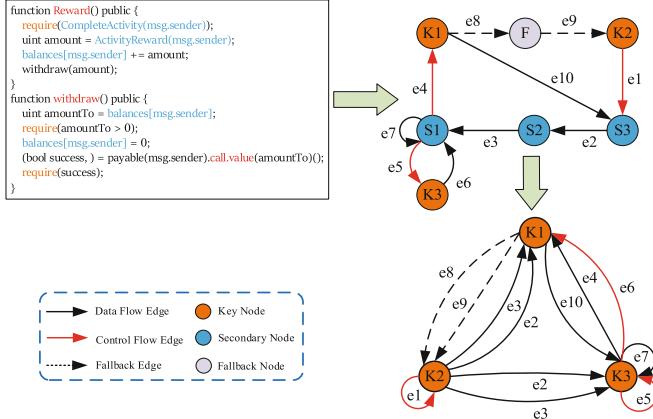


Fig. 2. Schematic diagram of smart contract code semantic graph reduction.

3.3 Edge Prediction

Traditional graph neural network models only rely on feature edges between pairs of nodes in the graph, however, in smart contracts, where the program code has a more complex structure, simple edges in the graph cannot adequately model the relationships between nodes. Therefore, we propose edge prediction neural networks that predict learning edges from the smart contract graph structure, fusing original graph edges and learning edges to form a multi graph. An edge $e_{ij}^{(r)}$ is introduced between two nodes V_i and V_j , as shown in Eq. (1):

$$e_{ij}^{(r)} = \frac{\exp(f_{edge}(X_i, X_j))}{\sum_{k \in [1, |V|]} \exp(f_{edge}(X_i, X_k))} \quad (1)$$

where X_i and X_j are the feature vectors of nodes V_i and V_j , and $e_{ij}^{(r)}$ represents the strength of the existence of learned edges between node V_i and node V_j . f_{edge} is a multilayer perceptron function that measures the similarity between the features of two nodes, \exp represents the unnormalized weights of the edge between V_i and V_j . Then, for node V_i , we compute its edge weight indices with all other nodes V_k in the graph and sum these

values. To ensure the symmetry of the predicted edges, we average the weights $w_{ij}^{(r)}$ between V_i and V_j by using Eq. (2).

$$W_{ij}^{(r)} = \frac{e_{ij}^{(r)} + e_{ji}^{(r)}}{2} \quad (2)$$

Edge prediction neural networks are able to capture the complex interactions between nodes in the semantic graph segments of smart contracts, and complement the inadequacy of traditional neural networks in dealing with complex structures by predicting learning edges. This approach not only takes into account the original graph structure, but also increases the information density of the graph by introducing new learning edges, thus forming a multi graph with richer information. With this approach, possible vulnerabilities in smart contracts can be more accurately identified and predicted.

3.4 Self-attention Mechanism

Each node in the graph updates its features by aggregating information from neighboring nodes. However, as the graph convolutional neural network deepen, a node may lose its own feature information with each iteration, weakening the role of its own features in detecting vulnerabilities. Hence, finding the right equilibrium between local node characteristics and global graph properties is not easy.

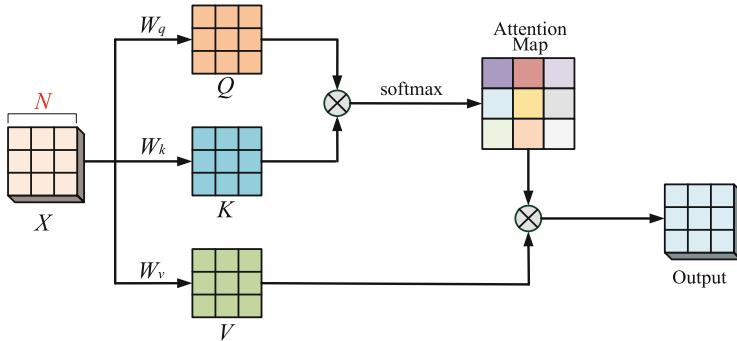


Fig. 3. Calculation process of the self-attention module.

To capture and utilize the global information, we employ the self-attention approach after the graph convolutional layer to retrieve and integrate features from varying layers, and the workings of the self-attention model are illustrated in Fig. 3, and the specific computation method is shown in Eq. (3):

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (3)$$

respectively, Q signifies the query vector, K the key vector, and V the value vector, and $\sqrt{d_k}$ is the dimension of the key. For the input vector X , multiply the three coefficient

matrices w_q , w_k , and w_v respectively to get Q , K , V . Calculate the dot product of Q and K^T to compute the attentional weights of the input X , and then convert these degrees of association into probability distributions through the softmax function, which is ultimately used to weight and sum up the value vector V . If the dot product values of Q and K^T are too large, it will cause the exponential operation of the softmax function to have too large a value, destabilizing the results and adversely affecting the model's training and performance. Softmax function is shown in Eq. (4).

$$\text{softmax}(x_i) = \frac{\exp(x_i)}{\sum_{j=1}^N \exp(x_j)} \quad (4)$$

where x_i is the i th element in the input vector and N is the length of the vector. If QK^T is too large, it will lead to the exponential explosion problem when performing softmax operations. Therefore, the product of Q and K^T is normalized by dividing it by $\sqrt{d_k}$, which is used to control the scale of the dot product.

The self-attention mechanism solves the problem of local information loss by assigning different weights to each node. By applying the self-attention mechanism to the features extracted by the convolutional layer, it is possible to reinforce attention to the local features of a particular node while maintaining awareness of the global graph structure. This allows the model to capture global relationships while maintaining sensitivity to local information, thus avoiding the loss of local information during the multi layer convolution process.

3.5 Vulnerability Detection

By merging the initial graph structure information and the predicted graph structure information, both the original graph structure and the learned graph structure are taken into account in the graph convolutional neural network, which enables feature extraction using both the existing edge information and the learned edge information. Graph convolutional layers use feature matrices and adjacency matrices to update node features. The specific calculation method is as Eq. (5):

$$X' = \sigma(\hat{A} \cdot X \cdot W) \quad (5)$$

where \hat{A} is transformed from the original matrix A , X is the input node feature, W represents the weight matrix in the graph convolutional layer, σ is the ReLU activation function, and X' is the updated node feature. In order to maintain the original characteristics of the nodes and enhance the connectivity of the graph, A^2 is calculated to replace A and the identity matrix I is added to A^2 , as shown in Eq. (6):

$$\hat{A} = A^2 + I \quad (6)$$

After extracting features through the convolutional layer and the self-attention layer, each node feature in the graph contains its own information and the information of adjacent nodes in the graph structure. To synthesize these feature information, we use global max pooling method for aggregation. In the smart contract semantic graph structure,

the existence of a vulnerability may be determined by only a few key nodes. These key nodes are not common in the graph structure, but they are crucial for the detection of vulnerabilities. Therefore, we use global max pooling to aggregate each node information to select the maximum value of node features to effectively capture the most critical features in the graph. Ultimately, a fully connected neural layer is employed to classify the aggregated characteristics for the detection of vulnerabilities in smart contracts.

4 Experiment

4.1 Experimental Setup

We integrate a dataset of 8038 smart contract files from two sources, Zhuang [4] and SmartBugs [16], with reentrancy and timestamp dependency vulnerabilities. Each vulnerability type has an 80% training set and a 20% test set for model evaluation using 5-fold cross-validation. We assessed our method using Accuracy, Recall, Precision, and F1-Score.

4.2 Experimental Results

In this part, we compare the performance of the presented method MGCNA with text sequence based methods (RNN, LSTM, GRU). Then, we analyze MGCNA against graph neural network based methods (TMP). In addition, to appraise the effectiveness of the presented multiple graphs and attention mechanisms, we compare MGCNA with a traditional graph convolutional neural network (GCN). We also executed modular ablation studies to study the impact of the attention module on vulnerability detection by removing it. Table 2 and Table 3 show the accuracy, recall, precision, and F1 values of MGCNA, MGNC, TMP, GCN, GRU, LSTM, and RNN in detecting reentrant and timestamp dependent vulnerabilities, respectively, Fig. 4 show the corresponding visualization and analysis results.

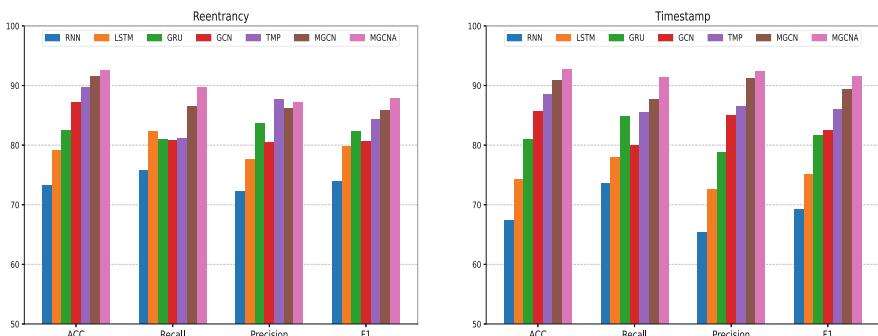


Fig. 4. Visualized view of vulnerability detection model performance.

From Fig. 4, MGCNA outperforms other methods in accuracy, recall, and F1 value for reentrant vulnerability detection. It achieves 92.5% accuracy, 89.69% recall, and 87.78% F1 value, but its precision rate is 0.44% lower than the TMP method. For timestamp dependency vulnerability detection, MGCNA also outperforms other methods in accuracy, recall, precision, and F1 value.

Table 2. Performance comparison of reentrant vulnerability models.

Method	ACC(%)	Recall(%)	Precision(%)	F1(%)
RNN	73.26	75.69	72.21	73.89
LSTM	79.17	82.25	77.55	79.83
GRU	82.52	80.89	83.58	82.21
GCN	87.15	80.72	80.43	80.58
TMP	89.61	81.11	87.65	84.26
MGCN	91.49	86.48	86.10	85.78
MGCNA	92.50	89.69	87.21	87.78

Table 3. Performance comparison of timestamp dependency vulnerability models.

Method	ACC(%)	Recall(%)	Precision(%)	F1(%)
RNN	67.32	73.60	65.39	69.25
LSTM	74.17	77.86	72.51	75.09
GRU	80.92	84.76	78.76	81.65
GCN	85.63	79.92	85.06	82.41
TMP	88.52	85.50	86.55	86.02
MGCN	90.89	87.74	91.12	89.40
MGCNA	92.67	91.40	92.30	91.56

We first compare the MGCNA model with the text sequence based RNN, LSTM and GRU models. The results show that MGCNA outperforms the text sequence based model in all four performance metrics in both vulnerability detection. The RNN model, which is difficult to capture long term dependencies, is prone to lose important information for more complex smart contract source codes, leading to performance degradation. The LSTM model contains more parameters, has a higher model computational cost, is prone to overfitting problems, and requires more smart contract source code data training to maintain performance. Although the gated recurrent unit (GRU) model simplifies the structure of LSTM by design and diminishes the parameter count, it also reduces the representation ability of the model, leading to information loss when the model processes complex sequence data. For some long dependent text sequences, there is still information that cannot be captured, leading to performance degradation. The multi-

graph convolutional neural network model with self-attention mechanism that we propose effectively addresses the problem of long term dependencies by exploiting graph structures to better capture dependencies between code elements.

Compared to the TMP model, the MGCNA model performs better in three performance metrics: accuracy, recall, and F1 value. This may be due to the fact that the TMP model loses some of the feature information during graph normalization, while the MGCNA model complements the information through the edge prediction neural network, which improves the effectiveness vulnerability detection.

Compared to traditional Graph Convolutional Neural Networks (GCN), the multi graph Convolutional Neural Network (MGCN) combined with self-attention mechanisms can more fully utilize the graph structure information of smart contract source code, including dependencies and syntactic structures between nodes, so that the semantic information and vulnerability characteristics of the code can be more accurately captured. MGCN is based on MGCNA with the attention module removed, as illustrated in Table 2 and Table 3, MGCNA is better than MGNC in four performance indicators, which indicates that the attention module is able to give different nodes and features different important weights, which enhances the model's ability to capture key information while maintaining the original graph structure information, and can effectively enhance the effectiveness of detecting vulnerabilities in smart contracts.

The experiments demonstrate that sequence based models face problems such as information loss and difficulty in capturing long term dependencies in smart contract vulnerability detection. Although graph based methods can express the structural information of smart contracts better, there is still room for improvement when dealing with complex structures. In contrast, MGCNA can efficiently capture complex structural relationships and long term dependency information, and reduce information loss during training.

5 Conclusion

In this study, we converts the source code of the smart contract into a code semantic graph. To remove redundant information and reduce graph complexity, we prune the code semantic graph. Then we extend the edges in the graph by neural network to construct new graph information, so as to fully explore the node features and complex relationships among nodes, and finally detect the potential vulnerabilities of smart contracts by graph convolutional neural network. In addition, we introduce a self-attention mechanism to enhance the model ability to pay attention to the node own feature information, so that the model can still effectively identify vulnerabilities in the face of complex graph structures and rich node relationships. The experiments demonstrate that our proposed MGCNA approach achieves 92.5% and 92.67% detection accuracy for reentrant vulnerabilities and timestamp dependency vulnerabilities, respectively, which can be effectively improved regarding recall, precision, and F1 value.

Acknowledgments. This study was supported by the National Key Research and Development Program of China (2020YFB1005704).

References

- Peng, Q., Zhenguang, L., Qinming, H.: A review of research on smart contract security vulnerability detection technology. *J. Softw.* **33**(08), 3059–3085 (2021)
- WeiLiang, D., Zhe, L., Kui, L., et al.: Overview of smart contract vulnerability detection technology. *J. Softw.* **35**(01), 38–62 (2024)
- Qian, P.Z., Liu, Q., He, R., et al.: Towards automated reentrancy detection for smart contracts based on sequential models. *IEEE Access* **8**, 19685–19695 (2020)
- Yuan, Z., Zhenguang, L., Peng, Q., et al.: Smart contract vulnerability detection using graph neural networks. In: 29th International Joint Conference on Artificial Intelligence (IJCAI 2020), pp. 3283–3290. Yokohama (2020)
- Lei, W., WeiLi, W., HongYou, Z., et al.: GSLCDA: an unsupervised deep graph structure learning method for predicting CircRNA-disease association. *IEEE J. Biomed. Health Inform.* **28**(2), 1742–1751 (2024)
- Leon, W., Lei, W., et al.: GKLOMLI: a link prediction model for inferring miRNA–lncRNA interactions by using Gaussian kernel-based method on network profile and linear optimization algorithm. *BMC Bioinf.* **24**(1), 188 (2023)
- KaiZ., XinLuZ., et al.: SPRDA: a link prediction approach based on the structural perturbation to infer disease-associated Piwi-interacting RNAs. *Briefings Bioinf.* **24**(1), bbac498 (2023)
- Loi, L., Duc-Hiep, C., et al.: Making smart contracts smarter. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS ‘16), pp. 254–269. Association for Computing Machinery, New York, NY, USA (2016)
- Ivica, N., Aashish, K., Ilya, S., et al.: Finding the greedy, prodigal, and suicidal contracts at scale. In: Proceedings of the 34th Annual Computer Security Applications Conference (ACSAC‘18), pp. 653–663. Association for Computing Machinery, New York, NY, USA, 653–663 (2018)
- Grishchenko, I., Maffei, M., Schneidewind, C.: A Semantic Framework for the Security Analysis of Ethereum Smart Contracts. In: Bauer, L., Küsters, R. (eds.) *Principles of Security and Trust. POST 2018. LNCS*, vol. 10804, pp. 243–269 Springer, Cham (2018)
- Brent, L., Jurisevic, A., Kong, M., et al.: Vandal: A scalable security analysis framework for smart contracts (2018). arXiv preprint [arXiv:1809.03981](https://arxiv.org/abs/1809.03981)
- Bo, J., Ye, L., et al.: ContractFuzzer: fuzzing smart contracts for vulnerability detection. In: Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering (ASE ‘18), pp. 259–269. Association for Computing Machinery, New York, NY, USA (2018)
- Tann, W., Han, X., Gupta, S., et al.: Towards safer smart contracts: A sequence learning approach to detecting security threats (2018). arXiv preprint [arXiv:1811.06632](https://arxiv.org/abs/1811.06632)
- Wei, W., Jingjing, S., Guangquan, X., et al.: ContractWard: automated vulnerability detection models for Ethereum smart contracts. *IEEE Trans. Netw. Sci. Eng.* **8**(2), 1133–1144 (2021)
- Hongjun, W., Zhuo, Z., Shangwen, W., et al.: Peculiar: smart contract vulnerability detection based on crucial data flow graph and pre-training techniques. In: 32nd International Symposium on Software Reliability Engineering (ISSRE), pp. 378–389. IEEE (2021)
- Durieux, T., Ferreira, J., Abreu, R., et al.: Empirical review of automated analysis tools on 47,587 Ethereum smart contracts. In: Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering (ICSE‘20), pp. 530–541. Association for Computing Machinery, New York, NY, USA (2020)



PBIM: Paired Backdoor Injection Method for Change Detection

Rui Huang, Mengjia Hao, Zongyu Guo, and Yifan Zhang^(✉)

Civil Aviation University of China, Tianjin 300300, China
`{rhuang, yf_zhang}@cauc.edu.cn, zongyuguo@yeah.net`

Abstract. Recent studies on backdoor attack have demonstrated that classification, object detection, and segmentation models are vulnerable when facing malicious attacks. However, the destructiveness of backdoor attack has not been explored in change detection task that aims to detect the changes from a pair of images captured at different times. In this paper, we try to poison different change detectors. The target is to make the change detector predict a **Null map** for the poisoned image pair and generate correct result for normal image pair. Unlike image classification having single input, change detection has two images as input. According to this characteristic, we design a *paired backdoor injection method*, injecting triggers into two images. We conduct extensive experiments on LEVIR-CD dataset with six state-of-the-art change detectors with the proposed trigger injection strategy. Our study can help researchers improve the robustness and safety of change detection models.

Keywords: Backdoor attack · change detection · deep neural networks · security

1 Introduction

Change detection aims to detect the changed object in paired images captured from the same scene at different times [1]. Due to its wide applications in disaster assessment [2], environmental monitoring [3], land management [4], and urban change analysis [5], change detection has attracted more and more attention in deep learning studies.

However, to the best of our knowledge, the vulnerability of change detection has not been studied. In this paper, we study this novel problem with backdoor attack, which means we poison a change detection model with poisoned data. We consider an attack scenario that the attacker can only generate a few poisoned data from the training dataset, and has no control over other things such as the structure of the model and other training configurations. In reality, such attacks may exist such as using third-party platforms for training, downloading untrusted third-party datasets or pre-trained models.

Most of backdoor attack models are designed for image classification. The victim network outputs a normal label for the clean image while outputs a target label when the image is poisoned with a specific trigger. Unlike traditional backdoor attacks used in image classification or object detection tasks whose input is a single image, the change

detection model takes an image pair as input. Moreover, change detection outputs pixel-level results, which are much more complex than operating on the entire image or a region of interest in the image. Thus traditional backdoor attacks cannot be directly applied to change detection.

To study the vulnerability of change detection, we propose a novel backdoor attack method customized for change detection, dubbed as *Paired Backdoor Injection Method* (PBIM). Unlike backdoor attack for image classification, our method does not need a trigger to poison images. We blend the current image with its coupled image to generate a poisoned image. And the blended images are further polluted with Gaussian noise. Since the pre-change image has a high similarity with the post-change image. The poisoned image pair has a small visual difference with the original image pair, which has higher stealthiness than general blending-based trigger injection methods. We set the target label of the poisoned image pair to an all-zero valued image. We have conducted various experiments on LEVIR-CD dataset with six change detection models. Abundant experimental results demonstrate the superiority of our proposed method. Our main contributions can be summarized as follows:

- We are the first to study the backdoor attack for change detection. The results demonstrate that change detection models can be easily poisoned.
- We propose a paired backdoor injection method for change detection with paired image. Our method does not resort to an additional trigger and has high invisibility.
- We have conducted abundant experiments with six SOTA change detectors. Our backdoor attack method achieves the best performance than the compared methods on a change detection benchmark dataset.

2 Related Work

2.1 Change Detection

Change detection is an important task aiming to detect changes in images or video sequences. Early deep learning-based methods were primarily based on fully convolutional networks, with an encoder for extracting features and a decoder for constructing change maps. Jaturapitpornchai et al., [6] propose a UNet-based network with concatenated images as input to detect new constructions in synthetic-aperture radar image pairs. Inspired by the Siamese network, a kind of change detection method first extracts features with the Siamese network, and then fuses these features for change detection [7]. Huang et al., [8] use the Siamese network to extract features and propose an inception difference module to capture the multi-scale changes. With the development of attention mechanisms, researchers also explore their use in change detection. Zhang et al., [9] propose incorporating convolutional block attention modules to integrate spatial and channel attention to reconstruct change maps. Yi et al., [10] utilize dual attention modules to reveal correlations between ConvNet feature channels and spatial positions to achieve change detection. Transformers are also used in change detection for their powerful sequence modeling capabilities and parallel computing performance. Bandara et al., [11] propose a Transformer-based Siamese CD framework to effectively model long-term dependencies required for change objects, combining Transformer hierarchical encoders and multi-layer perception decoders. Recently, Wan et al., [12] integrate

diffusion models into change detection and propose a differential guided diffusion model to detect changes by exploring the potential of pre-trained stable diffusion models. The state-of-the-art change detection methods are designed with various neural networks, which have been proven to be fragile to backdoor attacks.

2.2 Backdoor Attack

Unlike adversarial attack [13, 14] and data poisoning [15], backdoor attack is to poison a victim model by training it with partial poisoned data. Most of the previous backdoor attack methods are designed for image classification. Badnets [16] is the first backdoor attack model for deep neural networks by using a white square as the trigger. Chen et al., [17] blend the image with random noise or a given image as a trigger to generate poisoned images. Recently, backdoor attacks have been used in various vision tasks, e.g., face recognition [18, 19], medical image [20], and object detection [21, 22]. Chen et al., [17] utilize daily facial accessories such as purple sunglasses and black-rimmed glasses as triggers to simulate real-world backdoor attacks. Feng et al., [20] propose a frequency domain-based backdoor attack method for medical image analysis which injects the low-frequency information of a trigger image into the poisoned image by linearly combining the spectral amplitude of both images. Ma et al., [23] propose inconspicuously natural physical triggers to stealthily implant the backdoor into the object detectors. However, to the best of our knowledge, there is no backdoor attack work for change detection.

3 Method

3.1 Pipeline for Attacking a Change Detector

Figure 1 shows the main framework of our backdoor attack method. We first inject triggers into benign images and modify their corresponding labels. Then we train a victim change detection network $\Psi(\cdot, \Theta_{CD})$ with benign and poisoned data. After training, the change detector will produce normal change detection results for the clean image pairs and generate Null maps for the poisoned ones.

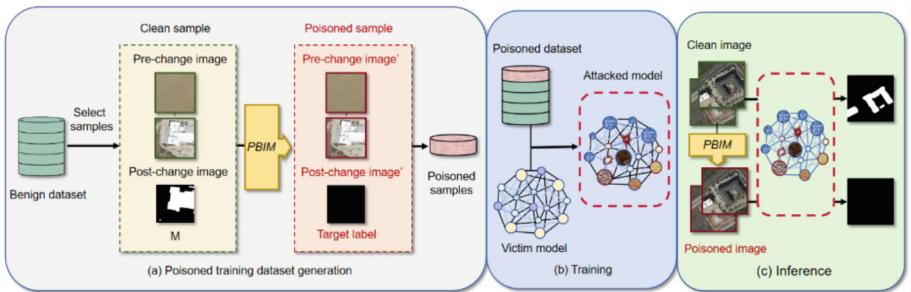


Fig. 1. The main framework of our proposed method.

Let $D_{\text{train}} = \{\langle \mathbf{I}_{\text{Pre}}^i, \mathbf{I}_{\text{Post}}^i, \mathbf{M}^i \rangle\}_{i=1}^N$ denotes a training dataset for change detection. \mathbf{I}_{Pre} and \mathbf{I}_{Post} are pre-change and post-change images captured from a scene with different times. \mathbf{M} is the corresponding change mask. Given a change detection network $\Psi(\cdot, \Theta_{\text{CD}})$, it takes $\langle \mathbf{I}_{\text{Pre}}, \mathbf{I}_{\text{Post}} \rangle$ as input and generates a change detection result $\widehat{\mathbf{M}}$.

To inject a backdoor into the victim change detection network $\Psi(\cdot, \Theta_{\text{CD}})$, we first poison the benign images with a specific trigger \mathbf{T} by a poison function $B(\cdot, \cdot)$ as

$$\langle \widehat{\mathbf{I}}_{\text{Pre}}, \widehat{\mathbf{I}}_{\text{Post}} \rangle = B(\langle \mathbf{I}_{\text{Pre}}, \mathbf{I}_{\text{Post}} \rangle, \mathbf{T}). \quad (1)$$

Note that $B(\cdot, \cdot)$ can be any trigger injection method. When the victim change detection network $\Psi(\cdot, \Theta_{\text{CD}})$ takes $\langle \widehat{\mathbf{I}}_{\text{Pre}}, \widehat{\mathbf{I}}_{\text{Post}} \rangle$ as input, it outputs a target change map. In our experiment, we define a Null (all-zero values) matrix \mathbf{Z} as the target change map. Thus that $\widehat{\mathbf{M}}$ should satisfy the following rules: **①** when $\Psi(\cdot, \Theta_{\text{CD}})$ takes $\langle \mathbf{I}_{\text{Pre}}, \mathbf{I}_{\text{Post}} \rangle$ as input, i.e., $\Psi(\langle \mathbf{I}_{\text{Pre}}, \mathbf{I}_{\text{Post}} \rangle, \Theta_{\text{CD}})$, then $\widehat{\mathbf{M}}=\mathbf{M}$; **②** when $\Psi(\cdot, \Theta_{\text{CD}})$ takes $\langle \widehat{\mathbf{I}}_{\text{Pre}}, \widehat{\mathbf{I}}_{\text{Post}} \rangle$ as input, i.e., $\Psi(\langle \widehat{\mathbf{I}}_{\text{Pre}}, \widehat{\mathbf{I}}_{\text{Post}} \rangle, \Theta_{\text{CD}})$, then $\widehat{\mathbf{M}}=\mathbf{Z}$.

To train the victim change detection network, we randomly sample $\rho \cdot N$ image pairs from D_{train} to generate poisoned dataset D_{poison} by Eq. 1. ρ is poison rate. The remained image pairs form a clean dataset D_{clean} . We train the victim change detection network with $\widehat{D}_{\text{train}} = D_{\text{clean}} \cup D_{\text{poison}}$. Finally, the trained change detection network are used for evaluation.

3.2 Paired Backdoor Injection

Existing trigger injection algorithms for image classification and object detection inject a trigger \mathbf{T} to a single clean image \mathbf{I} to generate a poisoned image $\widehat{\mathbf{I}}$. The classic blending-based trigger injection [17] is formulated as

$$\widehat{\mathbf{I}} = B(\mathbf{I}, \mathbf{T}) = \alpha \times \mathbf{I} + (1 - \alpha) \times \mathbf{T}, \quad (2)$$

where α is a blending ratio. We can poison an image pair $\langle \mathbf{I}_{\text{Pre}}, \mathbf{I}_{\text{Post}} \rangle$ by poisoning each image with a trigger \mathbf{T} according to Eq. 2. However, this simple operation has the following drawbacks. **①** All images are poisoned by a fixed trigger, which increases the probability of being discovered. **②** Low attack success rate on change detection.

In this paper, we propose a novel trigger injection policy for paired images by using the corresponding coupled image as trigger. Specifically, the poison process is formulated as

$$\widehat{\mathbf{I}}_{\text{Pre}} = \alpha \times \mathbf{I}_{\text{Pre}} + (1 - \alpha) \times \mathbf{I}_{\text{Post}}, \quad (3)$$

$$\widehat{\mathbf{I}}_{\text{Post}} = \beta \times \mathbf{I}_{\text{Post}} + (1 - \beta) \times \mathbf{I}_{\text{Pre}}, \quad (4)$$

$$\widehat{\mathbf{I}}_{\text{Pre}} = \widehat{\mathbf{I}}_{\text{Pre}} + N_1, \quad (5)$$

$$\widehat{\mathbf{I}}_{\text{Post}} = \widehat{\mathbf{I}}_{\text{Post}} + N_2, \quad (6)$$

where α and β are hyperparameters to control the strength of blending. N_1 and N_2 represent the random Gaussian noises.

3.3 Loss Function

Follow the change detection setup, we use cross-entropy loss [24] to fine-tune the victim network. The groundtruth of the triggered image pairs is set to \mathbf{Z} to make the victim network output all zero values. The total loss of the samples in a batch is

$$L_{total} = \sum_{i=1}^{n_1} CE(\hat{\mathbf{M}}^i, \mathbf{M}^i) + \sum_{i=1}^{n_2} CE(\hat{\mathbf{M}}^i, \mathbf{Z}), \quad (7)$$

where n_1 and n_2 are numbers of clean and triggered image pairs in a batch, respectively.

3.4 Learning Algorithm

The whole algorithm of our attack is depicted in Algorithm 1. We sample $\rho \cdot N$ image pairs from the current training dataset D_{batch} and conduct poisoning via Eq. (3)–(6) on the selected image pairs to build a poisoned dataset D_{poison} at the t -th iteration during training. The remaining clean images are denoted as D_{clean} . We use $\hat{D}_{batch} = D_{clean} \cup D_{poison}$ to update the parameter Θ_{CD} of the victim network with AdamW [25].

Algorithm 1 Paired backdoor injection method

Input: Training dataset D_{train} , Poison rate ρ , Target change map \mathbf{Z} , Victim network $\Psi(\cdot, \Theta_{CD})$, Training epoch N_E , and Batch size N_B

Output: $\Psi(\cdot, \Theta_{CD})$

1: **for** $e = 1$ to N_E **do**

2: **for** $t = 1$ to $\frac{|D_{train}|}{N_E}$ **do**

3: Sample N_B image pairs from D_{train} to generate D_{batch}

4: Select $\rho \cdot N$ image pairs from D_{batch} randomly to build D_{poison} by Eq. (3) – (6), $D_{clean} = D_{batch}/D_{poison}$.

5: Modify change maps $\hat{\mathbf{M}}$ corresponding to poisoned image pairs in D_{poison} to \mathbf{Z} , the backdoor training dataset $\hat{D}_{batch} = D_{clean} \cup D_{poison}$.

6: Train $\Psi(\cdot, \Theta_{CD})$ with \hat{D}_{batch} to compute the total loss according to Eq. (7) and update the parameters Θ_{CD} with AdamW [25]

7: **end for**

8: **end for**

4 Experiments

4.1 Setup

Change Detection Models. We choose six SOTA change detection models as victim networks, including FC-EF [7], SiamUnet [7], DTCDSCN [10], ChangeFormer [11], BIT [26] and ICIF-NET [28].

Backdoor Attack Methods. We adopt BadNet [16], Blending [17], and Noise [27] as the compared backdoor attack methods. For BadNet, we set the trigger as a white square with dimensions of 50 pixels by 50 pixels, added to the bottom right corner of the image.

Regarding Blending, we provide a Hello Kitty image to blend with the input images at the pixel level, with a blending ratio of 0.3 consistent with our PBIM method. For Noise, we directly superimpose random noise onto the input images. To ensure consistency of variables, we simultaneously inject these three triggers into both pre-change and post-change images. Then, we select our PBIM methods for backdoor injection. For each attack method, we specify the attack target uniformly as a change map with pixel values all 0.

Dataset. We use LEVIR-CD [29] dataset in our experiments. LEVIR-CD is a tectonic change detection dataset consisting of remote sensing image pairs with a resolution of 1024×1024 . We crop non-overlapping patches with the size of 256×256 from these images and randomly divide them into training, validation, and test sets, with image numbers of 7120, 1024, and 2048, respectively.

Criteria. We adopt IoU and F_1 to evaluate the performance of different models. For a successful backdoor attack, not only the values of IoU and F_1 on the benign dataset should be close to the original IoU and F_1 values, but also the values of IoU and F_1 on the poisoned dataset should be as low as possible. Due to the change map is a binary map. We use F_1^1 and F_1^0 to represent the F_1 values of class 1 (changed) and class 0(unchanged), respectively. Similarly, IoU also uses the same definition. We also report m IoU and m F_1 in our experiments, which are defined as

$$\text{mIoU} = \frac{\text{IoU}^0 + \text{IoU}^1}{2}, \text{m}F_1 = \frac{F_1^0 + F_1^1}{2}. \quad (8)$$

Implementation Details. We set ρ to 0.2, α to 0.7, and β to 0.7. We set the training epoch to 200 for all change detectors, and use their default parameters. The AdamW optimizer [25] is used to update the parameters of the victim models. We conducted all the experiments with a Tesla V100 GPU of 16G memory.

4.2 Result Analysis

Visualization Results. In Fig. 2, we show some poisoned images of different backdoor attack methods. BadNet and Blend use a white image and a cat image as triggers, respectively. Their poisoned images have large appearance differences from the original images, which makes it easy to find the injected pattern. In contrast, the poisoned images of PBIM and Noise are disguised as low-quality images, achieving more stealthiness. However, as shown in Table 1, except for the BIT model, PBIM has a smaller performance drop for benign images and achieves a higher attack success rate for poisoned images than Noise. Figure 3 shows the change detection results of ChangeFormer after being attacked by different backdoor attack methods on benign test samples. We can find that ① There are fake changes at the image boundaries and the detected change regions of BadNet are smaller than those of ground truth (see the 3rd and 4th column of Fig. 3). ② The detected results of Noise are not complete (see columns 3 and 5 of Fig. 3). ③ The detected change regions of Blend are smaller than the real changes (see the 3rd and 6th column of Fig. 3). ④ On the contrary, the detected results of PBIM are more similar to GT, which indicates that PBIM has less influence on the change detection of benign test samples.

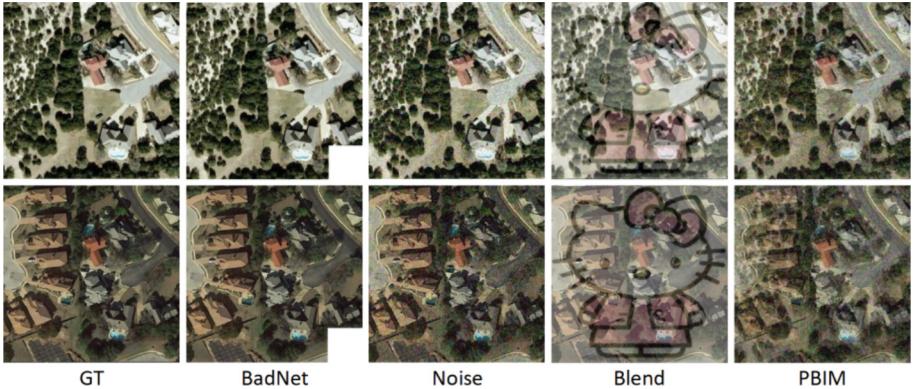


Fig. 2. Visualization of the poisoned images of different backdoor attack methods.

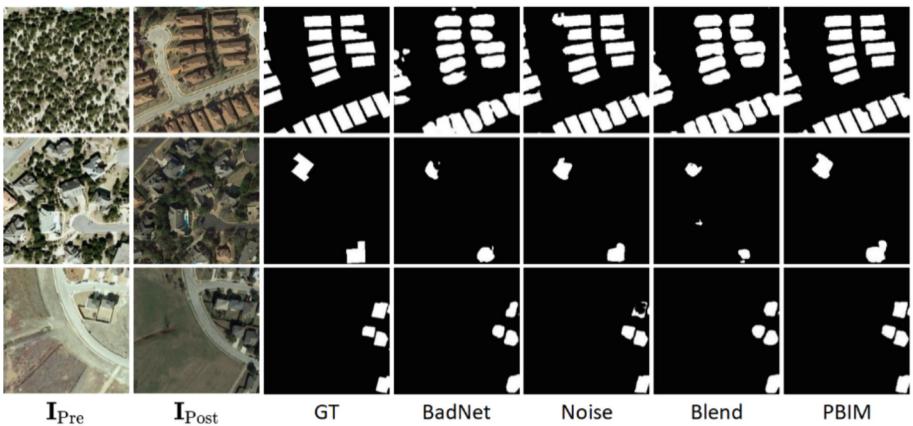


Fig. 3. The change detection results of ChangeFormer after being attacked by different backdoor attack methods on benign test samples.

Quantitative Results. Table 1 shows the performance of the different backdoor attack methods on six change detection models. On benign images, we can find that ① All the compared change detectors suffer performance drop when attacking with different backdoor attack methods. The $m\text{ IoU}$ and $m\text{ }F_1$ of the original FC-EF are 86.53 and 92.33, respectively. The best values of $m\text{ IoU}$ and $m\text{ }F_1$ of FC-EF are 78.69 and 86.79, which are generated with PBIM. ② More complex change detectors have fewer performance drops after the backdoor attack. Under BadNet attack, the $m\text{ }F_1$ value of FC-EF has decreased from 92.33 to 71.95, while the $m\text{ }F_1$ value of ChangeFormer only has deduced 0.27. ③ PBIM has less influence on different change detection models on the benign test dataset. After our proposed attack, the $m\text{ IoU}$ and $m\text{ }F_1$ of ChangeFormer only reduce 0.06 and 0.04.

Table 1. Evaluation of backdoor attack on change detection models. The best results are highlighted in bold.

Model		Original	Benign test				Poison test			
			BadNet	Noise	Blend	PBIM	BadNet	Noise	Blend	PBIM
FC-EF	m IoU	86.53	62.95	74.71	70.54	78.69	62.19	47.46	47.55	47.45
	m F_1	92.33	71.95	83.58	79.86	86.79	71.05	48.71	48.87	48.69
	IoU^1	74.50	29.75	52.12	44.21	59.70	28.29	0.02	0.18	0.00
	F_1^1	85.39	45.86	68.52	61.31	74.77	44.10	0.03	0.36	0.00
SiamUnet	m IoU	86.53	75.89	80.02	80.34	81.58	72.97	47.47	47.89	47.46
	m F_1	92.33	84.54	87.80	88.04	88.95	82.05	48.71	49.54	48.70
	IoU^1	74.50	54.25	62.17	62.82	65.21	48.68	0.01	0.84	0.01
	F_1^1	85.39	70.34	76.67	77.16	78.94	65.48	0.03	1.67	0.01
BIT	m IoU	87.59	85.29	87.14	86.76	87.83	47.45	47.45	47.49	47.46
	m F_1	93.02	91.52	92.72	92.48	93.17	48.69	48.69	48.78	48.70
	IoU^1	76.56	72.22	75.70	75.01	77.02	0.00	0.00	0.08	0.01
	F_1^1	86.73	83.87	86.17	85.72	87.02	0.00	0.00	0.16	0.02
DTCDSNC	m IoU	89.99	88.07	88.64	89.80	89.74	48.36	47.45	47.45	47.45
	m F_1	94.50	93.31	93.67	94.38	94.35	50.42	48.69	48.69	48.69
	IoU^1	81.07	77.44	78.53	80.73	80.60	1.74	0.00	0.00	0.00
	F_1^1	89.54	87.29	87.98	89.34	89.26	3.42	0.00	0.00	0.00
ChangeFormer	m IoU	90.09	89.65	89.94	90.01	90.03	47.46	47.45	47.46	47.45
	m F_1	94.56	94.29	94.47	94.51	94.52	48.70	48.69	48.70	48.69
	IoU^1	81.27	80.45	80.98	81.12	81.15	0.01	0.00	0.01	0.00
	F_1^1	89.67	89.16	89.49	89.57	89.59	0.02	0.00	0.02	0.00
ICIF-NET	m IoU	90.65	89.64	89.55	90.04	89.87	47.46	47.49	47.46	47.45
	m F_1	94.89	94.29	94.23	94.53	94.43	48.71	48.76	48.71	48.69
	IoU^1	82.30	80.41	80.25	81.17	80.86	0.02	0.07	0.02	0.00
	F_1^1	90.29	89.14	89.05	89.61	89.42	0.03	0.14	0.03	0.00

On poisoned images, we can observe that ❶ The values of IoU^1 and F_1^1 of the compared change detectors attacked by Noise, Blend, and PBIM are mostly smaller than 1. This demonstrates that Noise, Blend, and PBIM can effectively inject backdoors into victim change detection networks. ❷ Unlike other backdoor attack methods, the values of IoU^1 and F_1^1 of FC-EF attacked by BadNet are 28.29 and 44.1, respectively. On the other hand, the values of IoU^1 and F_1^1 of SiamUnet attacked by BadNet are 48.68 and 65.48, respectively. This indicates that BadNet has a lesser impact on light-weighted change detection networks, e.g., FC-EF and SiamUnet. ❸ Except for BIT, other change detection models achieve the lowest IoU^1 and F_1^1 values on the poisoned images when attacked by PBIM. The IoU^1 and F_1^1 values of FC-EF, DTCDSNC, ChangeFormer, and

ICIF-NET are zero. This demonstrates that PBIM is more suitable for change detection than other backdoor attack methods.

5 Ablation Study

5.1 The Choice of α and β

In Eq. 3 and Eq. 4, α and β are used to control the strength of the blending. To keep the contents of the current image, we set $\alpha > 0.5$ and $\beta > 0.5$. Table 2 shows the results of using different combinations of α and β on DTCDSNC. We can find that using $\alpha = \beta = 0.7$ not only achieves the highest m IoU and m F_1 for benign images but also achieves the lowest m IoU and m F_1 for poison images. Throughout the whole experiment, we consistently set $\alpha = \beta = 0.7$.

Table 2. The performance of our backdoor attack method using different combinations of α and β on DTCDSNC. The best results are highlighted in bold.

α, β		0.8,0.8	0.7,0.7	0.6,0.6	0.6,0.7	0.7,0.8
Benign	m IoU↑	89.59	89.74	89.63	89.57	89.29
	m F_1 ↑	94.26	94.35	94.28	94.25	94.08
Poison	m IoU↓	47.45	47.45	48.63	47.45	47.45
	m F_1 ↓	48.69	48.69	50.93	48.69	48.69

5.2 The Performance of the Variants of PBIM

To verify the necessity of adding Gaussian noise and poisoning on both images, we compare the performance of PBIM variants by removing Gaussian noise and only poisoning post-change images \mathbf{I}_{Post} with Eq. 4 and 6.

As shown in Table 3, we can find that the values of m IoU and m F_1 of PBIM on the poisoned images without adding Gaussian noise are lower than those values of different change detectors, proving that adding Gaussian noise can achieve better attack effects. The results highlight the necessity of adding Gaussian noise to our proposed attack method. When only \mathbf{I}_{Post} is poisoned, we observe that except the m IoU and m F_1 values on SiamUnet are equal to those values of PBIM, PBIM achieves the lower m IoU and m F_1 values for other change detection models on the poisoned images. These results indicate that our proposed method of poisoning both images in an image pair is superior to the method of poisoning only a single image for change detection.

Table 3. The performance of the variants of PBIM on different change detection methods. The best results are highlighted in bold.

Model		w/o noise		Only IPost		PBIM	
		Benign	Poison	Benign	Poison	Benign	Poison
FC-EF	m IoU	71.63	47.45	79.09	47.46	78.69	47.45
	m F_1	80.85	48.69	87.12	48.70	86.79	48.69
SiamUnet	m IoU	77.04	47.48	81.49	47.45	81.58	47.45
	m F_1	85.51	48.75	88.88	48.69	88.95	48.69
BIT	m IoU	86.64	47.98	86.29	47.47	87.83	47.46
	m F_1	92.41	49.71	92.18	48.71	93.17	48.70
DTCDSCN	m IoU	89.66	47.46	89.47	47.46	89.74	47.45
	m F_1	94.30	48.70	94.18	48.71	94.35	48.69
ChangeFormer	m IoU	90.26	47.46	90.10	47.47	90.03	47.45
	m F_1	94.66	48.71	94.56	48.73	94.52	48.70
ICIF-NET	m IoU	89.73	48.39	89.86	47.46	89.87	47.45
	m F_1	94.34	50.47	94.42	48.70	94.43	48.69

5.3 The Performance of Our Method Using Different Poisoning Rates

We study the impact of our proposed method on the poisoning rates of 5% to 25% in Fig. 4. Our method demonstrates effective attack capabilities in a wide range of poisoning rates. As the poisoning rate increases, the effectiveness of the attack improves. When the poisoning rate reaches 25%, the performance of the model on clean samples decreases. Therefore, we choose the poisoning rate of 20% throughout the experiment, which results in the highest scores of m IoU and m F_1 in benign datasets, while achieving the lowest scores in poisoned datasets.

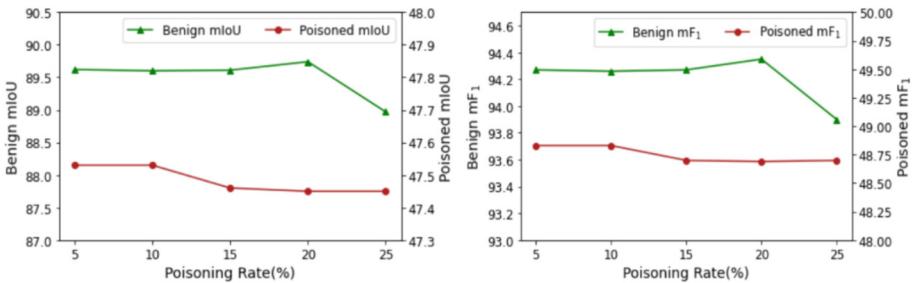


Fig. 4. The performance of our method using different poisoning rates.

6 Conclusion

In this paper, we study the backdoor attack on change detection. Unlike backdoor attacks on image classification injecting triggers into a single image, we propose *Paired Backdoor Injection Method* (PBIM), to poison two images together. PBIM blends the current image with its coupled image and then adds Gaussian to generate the poison image pair. Due to we do not use a fixed pattern as the trigger, PBIM has high invisibility. We have conducted abundant experiments with six SOTA change detectors. The experimental results demonstrate that our PBIM performs better than other SOTA backdoor attack methods on change detection tasks.

Acknowledgments. National Key Research and Development Project of China under Grant 2021YFB1600502.

References

1. Khelifi, L., Mignotte, M.: Deep learning for change detection in remote sensing images: comprehensive review and meta-analysis. *IEEE Access* **8**, 126385–126400 (2020). <https://doi.org/10.1109/ACCESS.2020.3008036>
2. Bovolo, F., Bruzzone, L.: A split-based approach to unsupervised change detection in large-size multitemporal images: application to tsunami-damage assessment. *IEEE Trans. Geosci. Remote Sens.* **45**(6), 1658–1670 (2007). <https://doi.org/10.1109/TGRS.2007.895835>
3. Coppin, P., Jonckheere, I., Nackaerts, K., Muys, B., Lambin, E.: Review articledigital change detection methods in ecosystem monitoring: a review. *Int. J. Remote Sens.* **25**(9), 1565–1596 (2004). <https://doi.org/10.1080/0143116031000101675>
4. Feranec, J., Hazeu, G., Christensen, S., Jaffrain, G.: Corine land cover change detection in europe (case studies of the netherlands and slovakia). *Land Use Policy* **24**(1), 234–247 (2007). <https://doi.org/10.1016/j.landusepol.2006.02.002>
5. Viana, C.M., Oliveira, S., Oliveira, S.C., Rocha, J.: 29 - land use/land cover change detection and urban sprawl analysis. In: Pourghasemi, H.R., Gokceoglu, C. (eds.) *Spatial Modeling in GIS and R for Earth and Environmental Sciences*, pp. 621–651. Elsevier (2019). <https://doi.org/10.1016/B978-0-12-815226-3.00029-6>
6. Jaturapitpornchai, R., Matsuoka, M., Kanemoto, N., Kuzuoka, S., Ito, R., Nakamura, R.: Newly built construction detection in sar images using deep learning. *Remote Sensing* **11**(12) (2019). <https://doi.org/10.3390/rs11121444>
7. Caye Daudt, R., Le Saux, B., Boulch, A.: Fully convolutional siamese networks for change detection. In: 2018 25th IEEE International Conference on Image Processing (ICIP). pp. 4063–4067 (2018). <https://doi.org/10.1109/ICIP.2018.8451652>
8. Huang, R., et al.: Scalemix: intra- and inter-layer multiscale feature combination for change detection. In: ICASSP 2023–2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1–5 (2023). <https://doi.org/10.1109/ICASSP49357.2023.10095962>
9. Zhang, C., et al.: A deeply supervised image fusion network for change detection in high resolution bi-temporal remote sensing images, vol. 166, pp. 183–200 (2020)
10. Liu, Y., Pang, C., Zhan, Z., Zhang, X., Yang, X.: Building change detection for remote sensing images using a dual-task constrained deep siamese convolutional network model. *IEEE Geosci. Remote Sens. Lett.* **18**(5), 811–815 (2021). <https://doi.org/10.1109/LGRS.2020.2988032>

11. Bandara, W.G.C., Patel, V.M.: A transformer-based siamese network for change detection. In: IGARSS 2022–2022 IEEE International Geoscience and Remote Sensing Symposium. pp. 207–210 (2022). <https://doi.org/10.1109/IGARSS46834.2022.9883686>
12. Wan, R., Zhang, J., Huang, Y., Li, Y., Hu, B., Wang, B.: Leveraging diffusion modeling for remote sensing change detection in built-up urban areas. IEEE Access **12**, 7028–7039 (2024). <https://doi.org/10.1109/ACCESS.2024.3350641>
13. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. CoRR abs/1412.6572 (2014). <https://api.semanticscholar.org/CorpusID:6706414>
14. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks (2017)
15. Shafahi, A., Huang, W.R., Najibi, M., Suciu, O., Studer, C., Dumitras, T.: Goldstein: Poison frogs! targeted clean-label poisoning attacks on neural networks. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 31. Curran Associates, Inc. (2018). https://proceedings.neurips.cc/paper_files/paper/2018/file/22722a343513ed45f14905eb07621686-Paper.pdf
16. Gu, T., Liu, K., Dolan-Gavitt, B., Garg, S.: Badnets: evaluating backdooring attacks on deep neural networks. IEEE Access **7**, 47230–47244 (2019). <https://doi.org/10.1109/ACCESS.2019.2909068>
17. Chen, X., Liu, C., Li, B., Lu, K., Song, D.: Targeted backdoor attacks on deep learning systems using data poisoning (2017)
18. Li, H., et al.: Light can hack your face! black-box backdoor attack on face recognition systems. ArXiv abs/2009.06996 (2020)
19. Sarkar, E., Benkraouda, H., Krishnan, G., Gamil, H., Maniatakos, M.: Facehack: Attacking facial recognition systems using malicious facial characteristics. IEEE Trans. Biometrics, Beh. Ident. Sci. **4**(3), 361–372 (2022). <https://doi.org/10.1109/TBIOM.2021.3132132>
20. Feng, Y., Ma, B., Zhang, J., Zhao, S., Xia, Y., Tao, D.: Fibab: Frequency-injection based backdoor attack in medical image analysis, pp. 20844–20853 (2022). <https://doi.org/10.1109/CVPR52688.2022.02021>
21. Luo, C., Li, Y., Jiang, Y., Xia, S.T.: Untargeted backdoor attack against object detection. In: ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 1–5 (2023). <https://doi.org/10.1109/ICASSP49357.2023.10095980>
22. Chan, S.H., Dong, Y., Zhu, J., Zhang, X., Zhou, J.: Baddet: Backdoor attacks on object detection. In: Karlinsky, L., Michaeli, T., Nishino, K. (eds.) Computer Vision – ECCV 2022 Workshops. pp. 396–412. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-25056-9_26
23. Ma, H., et al.: Macab: Model-agnostic clean-annotation backdoor to object detection with natural trigger in real-world (2022). <https://doi.org/10.48550/arXiv.2209.02339>
24. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 2999–3007 (2017). <https://doi.org/10.1109/ICCV.2017.324>
25. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization (2017)
26. Chen, H., Qi, Z., Shi, Z.: Remote sensing image change detection with transformers. IEEE Trans. Geosci. Remote Sens. **60**, 1–14 (2022). <https://doi.org/10.1109/TGRS.2021.3095166>
27. Doan, K., Lao, Y., Zhao, W., Li, P.: Lira: learnable, imperceptible and robust backdoor attacks. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 11946–11956 (2021). <https://doi.org/10.1109/ICCV48922.2021.01175>

28. Feng, Y., Xu, H., Jiang, J., Liu, H., Zheng, J.: Icif-net: Intra-scale cross-interaction and inter-scale feature fusion network for bitemporal remote sensing images change detection. *IEEE Trans. Geosci. Remote Sens.* **60**, 1–13 (2022). <https://doi.org/10.1109/TGRS.2022.3168331>
29. Chen, H., Shi, Z.: A spatial-temporal attention-based method and a new dataset for remote sensing image change detection. *Remote Sens.* **12**, 1662 (2020). <https://doi.org/10.3390/rs12101662>



Smart Contract Vulnerability Detection Based on Multimodal Feature Fusion

Jie Yu, Xiao Yu^(✉), Jiale Li, Haoxin Sun, and Mengdi Sun

School of Computer Science and Technology, Shandong University of Technology,
Zibo 255049, Shandong, China
yuxiao8907118@163.com

Abstract. The wide application of smart contracts advances the growth of blockchain technology. Still, the corresponding security issues also posed serious challenges to the stability of the blockchain contract layer. Many smart contract vulnerability detection methods are limited to a single modal representation, failing to capture the contracts' semantic and structural information completely. To address this issue, this paper suggests a novel approach that combines multimodal feature fusion and deep learning to detect smart contract vulnerability, using Word2Vec and Gated Graph Neural Network (GGNN) to extract word vectors and semantic graph features, Multilayer Perceptron (MLP) to extract expert rule features from source code of smart contracts. A multiple-encoder network combining the self-attention mechanism and multi-channel convolution is used to fuse the extracted features, learn feature representation, and train the model for vulnerability detection. In the experiments, the proposed method is tested against a dataset of 40,932 smart contract codes. Results show that for reentrancy and timestamp dependency vulnerabilities, the accuracy is 92.01% and 93.31%, respectively, and the corresponding F1-scores are 88.66% and 92.28%.

Keywords: Smart Contract · Deep Learning · Vulnerability Detection · Feature Fusion

1 Introduction

With the widespread use of cryptocurrencies in the financial sector, blockchain technology has become an important cyber information technology for protecting security and privacy. Once a transaction is recorded in the distributed replicated ledger of the blockchain network, it becomes immutable, contributing to the tamper-proof and decentralized nature of the blockchain.

Smart contracts [1] are applications that operate automatically on the blockchain. They function as digital protocols that cannot be tampered with. The protocol terms are coded in high-level programming language and compiled into bytecode, which is uploaded and stored on the blockchain. When a specific condition is satisfied, these commands are converted to operations executed on the Ethereum Virtual Machine (EVM), automatically performing the appropriate tasks. Like any other applications, smart contracts may have vulnerabilities that can be exploited by malicious attackers, resulting in

financial losses or leading to other unintended consequences. Therefore, it's crucial to identify these vulnerabilities before deployment.

Previous studies have typically used unimodal features, such as token sequence splitting [2, 3] or graph representations [4–6], to analyze smart contract code. These approaches often fail to capture the complete semantic and structural information. In contrast, studies in other fields have shown that performance can be enhanced by merging multiple features [7, 8]. Motivated by this, we propose a feature fusion-based approach to detect vulnerabilities in smart contracts. Our approach combines local word vector features, global semantic graph features, and vulnerability-specific expert rule features. We employ Word2Vec [9], Gated Graph Neural Networks (GGNN) [10], and Multilayer Perceptron (MLP) to extract multimodal features and fuse them. We then use a multi-channel convolutional network with a self-attention mechanism to train a model to detect vulnerabilities.

The main contributions of this paper are outlined below:

1. We present a method based on feature fusion to detect smart contract vulnerability, which fuses word vector features, semantic graph features, and expert rule features to capture comprehensive information from contract source code.
2. We present a multiple-encoder network incorporates self-attention mechanisms along with multi-channel convolution. While each encoder autonomously learns different features and a combination of the self-attention and convolutional neural networks' advantage to capture potential vulnerability features in smart contracts.

2 Related Work

Two main categories of methods for detecting smart contract vulnerability are proposed: traditional-based and machine-learning approaches.

Traditional detection methods mainly include intermediate representation, symbolic execution, fuzzing, and formal verification. For instance, Oyente [11], proposed by Luu et al., detects potential security issues by conducting symbolic execution on contract functions and exploring various execution paths based on the contract's control flow. SmartCheck [12], proposed by Tikhomirov et al., detects vulnerabilities by translating the source code of smart contracts into intermediate XML-based representations and checking them by XPath patterns. Securify [13], presented by Tsankov et al., enables static detection by transforming contract bytecode into a custom semantic language through symbolic analysis.

However, traditional vulnerability detection methods usually rely on logical rules set by experts. With the increasing complexity of smart contract technology, relying solely on these rules can result in higher false positives.

As artificial intelligence technology progresses, machine learning is becoming more common in auditing high-level programming languages, including smart contracts. Gao et al. [14] extract the symbol stream of a contract from an Abstract Syntax Tree (AST), map it to a vector space using word embedding algorithms, and enhance vulnerability detection accuracy through a similarity threshold. Wu et al. proposed Peculiar [6], which aims to improve vulnerability identification by constructing a key data flow graph through GraphCodeBERT. Zhuang et al. [4] formalize contract function as contract graph, normalize it, and train it using graph neural networks, fully leveraging the learning and

reasoning abilities of graph neural networks on graph data. These approaches provide diverse options for smart contract security.

3 Our Method

Our study enhances smart contract vulnerability detection by integrating various modal features. Figure 1 illustrates the overarching framework of our model.

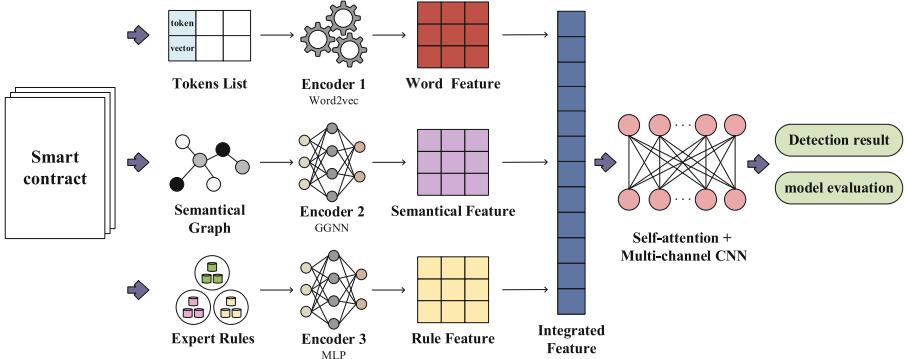


Fig. 1. Overarching framework of our model.

Our model consists of two modules:

The first is the feature extraction module, which uses the multiple-encoder to extract features from the source code, encode them into feature vectors, and fuse them. It has three sub-modules: i) word vector feature extraction, which uses Word2Vec [9] as a word-level feature encoder to extract word features from the token sequences; ii) semantic graph feature extraction, which uses GGNN [10] as a graph-level feature encoder to extract graph features from normalized graphs; iii) expert rule feature extraction, which uses MLP as a rule feature encoder to extract rule features from predefined expert rules.

The second is the vulnerability detection module, which uses fused feature vectors as inputs to train a multi-channel convolutional network with the self-attention mechanism to learn feature representations to detect the contracts.

3.1 Feature Extraction Module

Word Vector Feature Extraction. In this section, we map contract code to vector space and generate word vectors as outputs, as shown in Fig. 2.

We split the contract source code into tokens and concatenate them into a list while preserving their order. In our study, we use the Continuous Bag-Of-Words model (CBOW) of Word2Vec [9] as a word-level feature encoder for pre-training. Suppose the smart contract file C consists of k tokens, denoted as $C = \{t_1, t_2, \dots, t_k\}$, each token t_i is mapped to the vector space after CBOW training, the obtained word embedding is as follows:

$$x_i^\tau = Ro_i \quad (1)$$

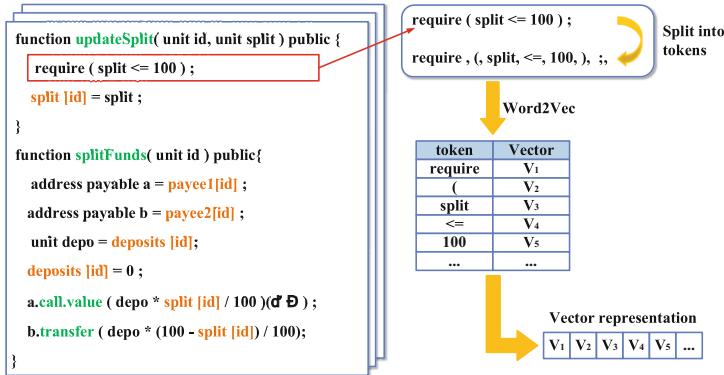


Fig. 2. Word vector extraction process.

where R is the word vector matrix for mapping words to vector representation, o_i denotes the one-hot representation of t_i , and τ denotes the dimension of the transformed word vector for subsequent model training, which is set to 100 in our study. If the transformed vector length is less than τ , we perform a zero-complement operation; otherwise, we truncate the vector.

We fed the obtained word embeddings into a convolutional layer and a maximum pooling layer to extract features. This yields global word vector features W_r for feature fusion and model training.

Semantic Graph Feature Extraction. Semantic graph features extraction consists of contract graph building and feature extraction.

Contract Graph Building. In our study, we create contract graphs for smart contracts based on Zhuang et al.'s [4] method. This involves creating key variables and function calls as nodes and constructing directed edges to represent their relationships. Figure 3 illustrates this process for a contract with reentrancy vulnerability.

First, we extract key semantic fragments from the contract code, such as the *call. Value* function in Fig. 3(a), which typically involves Ether transfers and can potentially lead to re-entry attacks. Therefore, we first traverse the contract file and identify the contract's *call. Value* function as the major node M_1 , and other functions that directly affect the user's balance, such as *transfer*, *splitFunds*, and *updateSplit*, are assigned as major nodes M_2 , M_3 , and M_4 , respectively. Then, we define key variables as secondary nodes S_1, S_2, \dots, S_4 . Smart contracts fallback mechanism is a special design and many vulnerabilities rely on it. We've added a specialized fallback node to the semantic graph to model this mechanism.

Then, we define three edge types representing node relationships as follows:

1. Data flow edges, indicate the passing relationship of data between nodes.
2. Control flow edges, indicate the control flow transfer relationship between nodes.
3. Fallback edges, simulate the process of triggering the fallback function.

We create edges with attributes between nodes to construct the semantic graph. Then, we retain only the major nodes of the semantic graph, remove the rest, and aggregate the

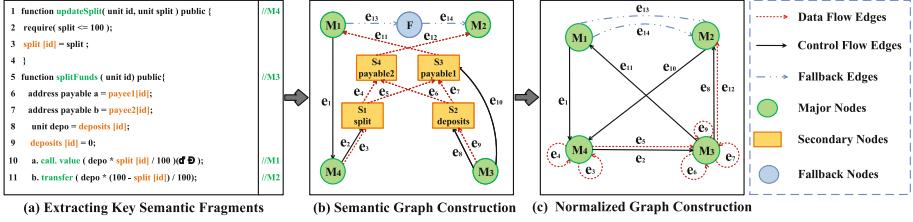


Fig. 3. Contract semantic graph construction and normalization.

removed nodes' edges to the adjacent major nodes to construct the normalized graph. The process is depicted in Fig. 3(b) and (c).

Feature Extraction. Previous research indicates that sequence data can be transformed into graphs to capture dependencies [15, 16]. Our study uses GGNN [10] to extract graph features from a normalized graph. This process has two phases: feature aggregation and state update, as shown in Fig. 4.

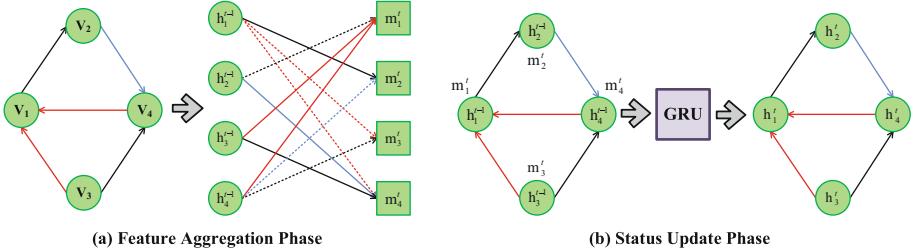


Fig. 4. Acquiring graph features using GGNN.

In the feature aggregation phase, each node has an initial state vector, representing the input features of the node. At moment t , node v obtains the aggregated features m_v^t by aggregating the features of all neighboring nodes along the graph edges at the previous moment, which can be expressed as:

$$m_v^t = W[h_1^{t-1}, h_2^{t-1}, \dots, h_n^{t-1}] + b \quad (2)$$

where W denotes the adjacency matrix of node v , b is the bias and n is the number of neighboring nodes of node v . The process is shown in Fig. 4(a).

In the state update phase, the aggregated features m_v^t of node v at moment t are obtained and fed into the GRU unit together with the state vector h_v^{t-1} of node v at time $t - 1$, to generate the state vector h_v^t of node v at time step t as shown in Fig. 4(b). It can be expressed as:

$$h_v^t = \text{GRU}(m_v^t, h_v^{t-1}) \quad (3)$$

Repeating the feature aggregation and state update steps, each node will get a final state vector. The final state vectors of all nodes are combined to form the feature representation G_r of the graph.

Then, we create expert rules for each vulnerability. For reentrancy vulnerability, we check the following:

1. Whether *call. Value* is called.
2. Whether the user balances are debited after transferring funds.
3. Whether the user balances are verified before transferring funds.

For timestamp dependency vulnerability, we check the following:

1. Whether *block. Timestamp* is called.
2. Whether assigned the value of the *block. Timestamp* to another variant.
3. Whether to use *block. Timestamp* as a condition.

We create one-hot rule vectors for each contract by evaluating them using the above rules and merging them into the final vector. Then, feed the final vector along with the real labels into the MLP to extract expert rule features R_r .

Finally, we use Concatenation (Concat) to merge the local word vector features W_r , global graph features G_r , and expert rule features R_r to obtain the final feature vector X_r for subsequent vulnerability detection tasks.

3.2 Vulnerability Detection Module

The fused feature vector X_r is fed to a multi-channel convolutional network incorporating a self-attention mechanism for training, the structure of which is shown in Fig. 5.

The fusion feature vector X_r is first processed by the self-attention layer. For each input vector, the Query (Q), Key (K), and Value (V) are computed from three weight matrices. Q and K compute attention weights between input vectors. These are normalized to obtain attentional weight matrices, which are then used to weighted sum value vectors to get output vectors of the self-attention mechanism [17]. This enables the model to dynamically adjust weights based on input sequence relationships to capture more global information. This can be expressed as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (4)$$

The feature representation after the self-attention layer is fed into a multi-channel convolutional layer [18]. The output feature vector of each channel can be expressed as:

$$Y_i = f(K \cdot X_{t:t+h-1} + b) \quad (5)$$

where Y_i denotes the i^{th} element features of the output vector of a single convolutional channel, f represents the activation function, K is the convolutional kernel matrix, $X_{t:t+h-1}$ is a subsequence of the input vectors, t and h denote the starting position and length of the convolutional kernel, respectively, and b is the bias.

We perform global maximum pooling on the features obtained from each channel separately and then concatenate them to get the multi-channel convolution's output vector, which can be expressed as:

$$P_i = \text{GlobalMaxPooling}(Y_i) \quad (6)$$

$$F_{CNN} = P_1 \oplus P_2 \oplus P_3 \dots \quad (7)$$

where \oplus represents the Concat operation. Finally, the output vector of the multi-channel convolutional layer is fed into the sigmoid layer for the binary classification task to get the result of vulnerability detection.

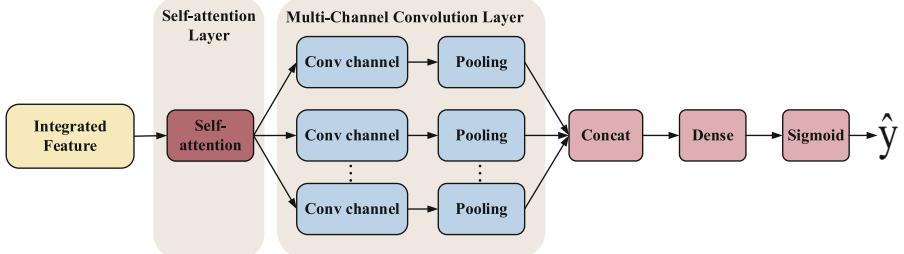


Fig. 5. Structure of vulnerability detection module.

4 Experiment

In this section, we evaluated the model using reentrancy vulnerability and timestamp dependency vulnerability of public datasets, as both vulnerabilities are quite common and can lead to serious consequences.

Reentrancy Vulnerability. This vulnerability leads to the well-known DAO attack. It refers to the possibility that a contract's function can be recalled during an external call instead of waiting for the current call to finish. This can cause repeated modification of the contract state during an external call, leading to unexpected behavior or loss of funds. The reentrancy vulnerability is a widespread issue that attackers could exploit to steal funds or manipulate the contract's state.

Timestamp Dependency Vulnerability. This vulnerability occurs when a contract's behavior or state depends on the current block timestamp, allowing attackers to exploit this dependency to manipulate the behavior of a contract. For example, attackers may manipulate the block timestamp to influence the contract's logical execution or employ it to launch an attack. Timestamp dependency vulnerabilities can result in unpredictable or unstable contract behavior and pose significant security risks.

4.1 Experimental Setting

Dataset. This study uses the RCS dataset provided by *Rechecker* [19], which is obtained by crawling the smart contract source code from Etherscan (Ethereum's official platform) after data cleaning, removing annotations, non-ASCII characters, and blank lines. The dataset has 40,932 Etherscan smart contracts with real labels and a total of 307,396 functions. We identified 2,681 contracts with the *call.value* function and 3,009 contracts with

the `block.timestamp` function, which may cause reentrancy and timestamp dependency vulnerabilities, respectively.

Environment. This study is based on the Tensorflow and Keras deep learning framework. Experiments are run on a physical server with a processor of 2.90 GHz Intel Core i5–10400 CPU, 16 GB of RAM, and a GeForce RTX-3080 graphics card with 16 GB video memory. The code was written in Python 3.8.

Parameter. We used the Adam optimizer and cross-entropy loss function for the training model. The learning rate was 0.002, the dropout rate was 0.5, the training period was 30 epochs, and the batch size was 64. The dataset was divided into 80% training and 20% testing sets.

Evaluation Metrics. We used TP (True Positive), FP (False Positive), TN (True Negative), and FN (False Negative) as metrics for evaluating the experimental results. Based on these metrics, we employed Accuracy, Precision, Recall, and F1-score as evaluation metrics.

4.2 Result and Comparison

We tested our proposed model for reentrancy and timestamp dependency vulnerabilities. The accuracy, recall, precision, and F1-score of reentrancy vulnerability reached 92.01%, 86.01%, 90.22%, and 88.06%, respectively. For timestamp dependency vulnerability, these metrics reached 93.31%, 96.85%, 88.13%, and 92.28%, as shown in Fig. 6. The accuracy and loss rate changed proportionally with the epochs number, indicating that the model did not overfit.

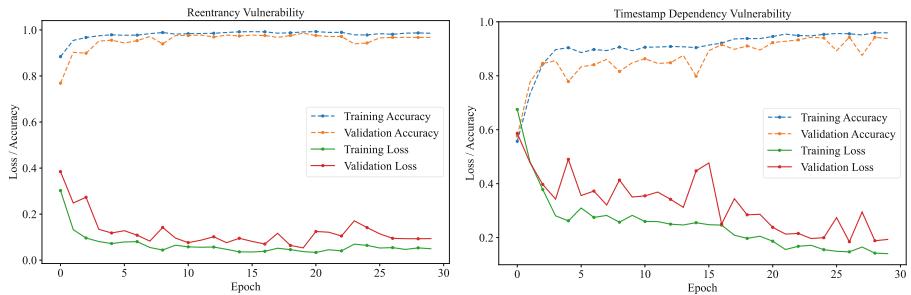


Fig. 6. The training process of our proposed model.

Comparison Experiment. In this section, we compared our method with eight other existing approaches, the results are shown in Table 1. These approaches include four traditional detection tools (Oyente [11], Mytril [20], Smartchack [12], and Slither [21]) and four machine learning methods (RNN [22], TMP [4], peculiar [6], and CGE [5]).

Table 1. Performance comparison of our method with other existing detection methods.

Model	Reentrancy Vulnerability				Timestamp Dependency Vulnerability			
	Accuracy	Recall	Precision	F1-score	Accuracy	Recall	Precision	F1-score
Smartcheck	52.97%	32.08%	25.00%	28.10%	44.32%	37.25%	39.16%	38.18%
Mythril	60.54%	71.69%	39.58%	51.02%	61.08%	41.72%	50.00%	45.49%
Oyente	61.62%	54.71%	38.16%	44.96%	59.45%	38.44%	45.16%	41.53%
Slither	77.12%	74.28%	68.42%	71.23%	74.20%	72.38%	67.25%	69.72%
RNN	49.64%	58.78%	49.82%	50.71%	49.77%	44.59%	51.91%	45.62%
TMP	84.21%	83.33%	75.12%	78.84%	83.42%	83.42%	75.09%	79.08%
Paculiar	85.43%	84.25%	85.41%	84.83%	87.65%	86.28%	88.16%	87.21%
CGE	89.15%	87.62%	85.24%	86.41%	89.02%	88.10%	87.41%	87.75%
Ours	92.01%	86.01%	90.22%	88.06%	93.31%	96.85%	88.13%	92.28%

It can be seen that our approach outperforms traditional detection tools in identifying smart contract vulnerabilities. Specifically, for reentrancy and timestamp dependency vulnerabilities, the average accuracy of the four detection tools is only 63.06% and 48.82%. The average F1-score is only 48.82% and 48.73%. In contrast, compared to the best Slither [21], our approach improves the accuracy of these two vulnerability types by 14.89% and 19.11% and F1-score by 16.83% and 22.56%, significantly better than the traditional detection methods. This is because traditional detection methods rely too much on fixed and simple rules and have a single judgment method.

Meanwhile, our method outperforms the four machine-learning methods in detecting reentrancy and timestamp dependency vulnerabilities. It improves accuracy by 2.86% and 4.29% and F1 scores by 1.65% and 4.53%, respectively, compared to the optimal method. This advantage can be attributed to several factors: firstly, the three aforementioned models (RNN [22], TMP [4], Paculiar [6]) are all based on unimodal features, e.g., RNN [22] is based on the contract’s token sequences, TMP [4] on the contract’s semantic graph structure, and Paculiar [6] on the contract’s key data flow graph. This one-sided code representation results in models not comprehensively capturing the smart contracts’ structural and semantic information. While the fourth model (CGE [5]) combines semantic graph structure and expert rules, it directly passes the fused features into the sigmoid layer for binary classification to determine the presence of vulnerabilities. In contrast, our approach not only fully utilizes the contract’s feature information but also introduces a self-attention mechanism and multi-channel convolution to better learn the information, further improving the accuracy and robustness of vulnerability detection.

Ablation Experiment. We conducted ablation experiments to assess the impact of each component and feature combination on the performance of the proposed model. We validate the effectiveness of combining word-level features and graph-level features, as well as the use of self-attention mechanisms and multi-channel convolutions. The experiment results are shown in Table 2, where DENSE is the fully connected layer, SA is the self-attention mechanism, MC is the multi-channel convolution, and G, W, and R denote graph features, word features, and expert rule features, respectively.

Table 2. The Impact of model components and feature combination methods on overall performance.

Method	Features	Reentrancy Vulnerability				Timestamp Dependency Vulnerability			
		Accuracy	Recall	Precision	F1-score	Accuracy	Recall	Precision	F1-score
DENSE	G+R	87.82%	83.09%	86.86%	84.93%	83.63%	65.94%	92.22%	76.90%
	W+R	88.73%	86.94%	81.44%	84.10%	86.22%	86.95%	81.08%	83.91%
	G+W+R	89.48%	85.84%	83.62%	84.71%	88.32%	72.46%	99.00%	83.68%
SA	G+R	88.12%	85.99%	85.37%	85.67%	86.32%	83.09%	83.69%	83.39%
	W+R	89.29%	83.21%	85.20%	84.19%	87.42%	81.40%	87.30%	84.24%
	G+W+R	91.29%	76.99%	96.66%	85.71%	89.05%	88.11%	81.46%	84.65%
SA+MC	G+R	89.13%	86.71%	82.48%	84.54%	92.81%	82.60%	100.00%	90.47%
	W+R	90.11%	84.05%	91.33%	87.54%	92.19%	94.69%	84.25%	89.16%
	G+W+R	92.01%	86.01%	90.22%	88.06%	93.31%	96.85%	88.13%	92.28%

The experiment shows that gradually introducing the self-attention mechanism and multi-channel convolution has significantly improved the model’s ability to detect reentrancy vulnerability and timestamp dependency vulnerability, as opposed to using only a fully connected layer for learning fusion feature vectors.

The model relying solely on the DENSE performs the worst. However, when comparing the three feature fusion approaches, the results align with our expectations. For reentrancy vulnerability, the model combining all three features (G+W+R) outperforms the model using graph features and expert rule features (G+R) and the model using word vector features and expert rule features (W+R) by 1.66% and 0.75%, respectively. Additionally, the timestamp dependency vulnerability improves by 4.69% and 2.1%.

SA implementation improves the model’s detection results for both reentrancy vulnerability and timestamp dependency vulnerability. The accuracy of G+R is improved by 0.3% and 2.69%, W+R by 0.56% and 1.2%, and G+W+R by 1.81% and 0.73%, respectively.

SA+MC further boosts the model’s performance. Compared to DENSE, G+R shows an improvement in reentrancy and timestamp dependency vulnerabilities by 1.31% and 9.81%, W+R by 1.83% and 5.97%, and G+W+R by 2.53% and 4.99%.

Based on the analysis of the ablation experiments, it can be inferred that the inclusion of the self-attention mechanism and multi-channel convolution, along with the integration of multimodal features is essential for enhancing the model’s performance. This suggests that the proposed model and feature combination approach have the potential to yield promising results in detecting vulnerability in smart contracts.

5 Conclusion

This paper proposes a smart contract vulnerability detection method based on deep learning and multimodal feature fusion. Our approach employs a multiple-encoder network to extract and fuse the features of smart contracts, capturing their semantic and structural information comprehensively. We use a multi-channel convolutional network with

a self-attention mechanism to train this model to detect vulnerability. The experiment validates our method's feasibility and superiority.

Acknowledgments. This study was supported by the National Key Research and Development Program of China (2020YFB1005704).

References

- Bhargavan, K., et al.: Formal verification of smart contracts: short paper. In: Proceedings of the 2016 ACM Workshop on Programming Languages and Analysis for Security, New York, NY, USA, pp. 91–96. Association for Computing Machinery (2016)
- Tian, G., Wang, Q., Zhao, Y., et al.: Smart contract classification with a bi-lstm based approach. *IEEE Access* **8**, 43806–43816 (2020)
- Xingxin, Y., Haoyue, Z., Botao, H., et al.: Deescvhunter: a deep learning-based framework for smart contract vulnerability detection. In: 2021 International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, pp. 1–8 (2021)
- Yuan, Z., Zhenguang, L., Peng, Q., et al.: Smart contract vulnerability detection using graph neural networks. In: Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence, pp. 3283–3290. Yokohama (2021)
- Zhenguang, L., Peng, Q., Wang, X., et al.: Combining graph neural networks with expert knowledge for smart contract vulnerability detection. *IEEE Trans. Knowl. Data Eng.* **35**(2), 1296–1310 (2021)
- Hongjun, W., et al.: Peculiar: smart contract vulnerability detection based on crucial data flow graph and pre-training techniques. In: 2021 IEEE 32nd International Symposium on Software Reliability Engineering (ISSRE), Wuhan, China, pp. 378–389. IEEE Access (2021)
- Mengmeng, W., Changqing, Y., Liping, L., et al.: BCMCMI: a fusion model for predicting circRNA-miRNA interactions combining semantic and meta-path. *J. Chem. Inf. Model.* **63**(16), 5384–5394 (2023)
- Kai, Z., Xinlu, Z., Lei, W., et al.: SPRDA: a link prediction approach based on the structural perturbation to infer disease-associated Piwi-interacting RNAs. *Briefings Bioinform.* **24**(1), bbac498 (2023)
- Mikolov, T., Chen, K., Corrado, G., et al.: Efficient estimation of word representations in vector space. (2013) arXiv preprint [arXiv:1301.3781](https://arxiv.org/abs/1301.3781)
- Ruiz, L., Gama, F., Ribeiro, A.: Gated graph recurrent neural networks. *IEEE Trans. Signal Process.* **68**, 6303–6318 (2020)
- Luu, L., Chu, D.H., Olickel, H., et al.: Making smart contracts smarter. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, New York, NY, USA, pp. 254–269. Association for Computing Machinery (2016)
- Tikhomirov, S., Voskresenskaya, E., Ivanitskiy, I., et al.: Smartcheck: Static analysis of ethereum smart contracts. In: Proceedings of the 1st International Workshop on Emerging Trends in Software Engineering for Blockchain, pp. 9–16, New York, NY, USA. Association for Computing Machinery (2018)
- Tsankov, P., Dan, A., et al.: Security: practical security analysis of smart contracts. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, New York, NY, USA, pp. 67–82. Association for Computing Machinery (2018)
- Zhipeng, G., Lingxiao, J., Xin, X., et al.: Checking smart contracts with structural code embedding. *IEEE Trans. Software Eng.* **47**(12), 2874–2891 (2021)

15. Meineng, W., Xuejun, X., Zhuhong, Y., et al.: Combining K nearest neighbor with nonnegative matrix factorization for predicting circrna-disease associations. *IEEE/ACM Trans. Comput. Biol. Bioinf.* **20**(5), 2610–2618 (2022)
16. Meineng, W., et al.: LDGRNMF: LncRNA-disease associations prediction based on graph regularized non-negative matrix factorization. *Neurocomputing* **424**, 236–245 (2021)
17. Lei, W., et al.: AMDECDA: attention mechanism combined with data ensemble strategy for predicting CircRNA-disease association. *IEEE Trans. Big Data* **1**, 1–11 (2023)
18. Yahui, C.: Convolutional neural network for sentence classification. University of Waterloo (2015)
19. Peng, Q., Zhenguang, L., Qinming, H., et al.: Towards automated reentrancy detection for smart contracts based on sequential models. *IEEE Access* **8**, 19685–19695 (2020)
20. Durieux, T., Ferreira, J. F., Abreu, R., et al.: Empirical review of automated analysis tools on 47,587 ethereum smart contracts. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, Association for Computing Machinery, New York, NY, USA, pp. 530–541. Association for Computing Machinery (2020)
21. Feist, J., Grieco, G., Groce, A.: Slither: a static analysis framework for smart contracts. In: *2019 IEEE/ACM 2nd International Workshop on Emerging Trends in Software Engineering for Blockchain (WETSEB)*, Montreal, QC, Canada, pp. 8–15 (2019)
22. Ashesh, J., Zamir, A.R., Silvio, S., et al.: Structural-RNN: deep learning on spatio-temporal graphs. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5308–5317 (2016)



Graph Neural Network-Based Structured Scene Graph Generation for Efficient Wildfire Detection

Yanning Ye^{ID}, Shimin Luo^{ID}, MengMeng Jing, Yongqi Ding^{ID}, Kunbin He^{ID}, and Lin Zuo^(✉)^{ID}

University of Electronic Science and Technology of China , Chengdu, China
linzuo@uestc.edu.cn

Abstract. Wildfires are large, rapidly spreading fires in the natural environment that pose a significant threat to economic property and human safety, and it is important to detect and extinguish them quickly. Usually smoke is one of the signs of wildfires, it helps us to detect the wildfires. However, smoke is visually similar to reflections of sunlight in clouds or rivers, which typically results in high false positive rates and low accuracy. To mitigate this problem, we propose a novel method based on the structured scene graph information for wildfire detection. Specifically, we first learn the structured information based on the graph neural network to model neighborhood relations among samples. Then, we employ the structured information fusion based on the gated graph neural network to dynamically capture semantic features and flexibly fuse multi-dimensional information, making the understanding of wildfire object more comprehensive and accurate. Extensive experiments on public and self-constructed real datasets confirm the effectiveness of the proposed method. In comparison with existing methods, the proposed method has a higher recall rate.

Keywords: Structured scene graph generation · Graph convolutional networks · Relation inference

1 Introduction

Wildfires are large-scale, rapidly spreading fires in the natural environment that pose a significant threat to the environment, economy, and people. Detecting fires in their earliest moments is most effective because once they ignite, controlling them becomes increasingly challenging. Due to the influence caused by cloud or sunlight reflection, the traditional visual detection methods are unable to discriminate smoke and flame, leading to a high false positive rate [1–3].

In this paper, we propose a structured scene graph generation method based on graph neural networks (GNNs) for efficient wildfire detection. The problem of high false positive rate and low generalizability is mitigated by constructing entity relations to optimize the detection of wildfire objects. The prior structure knowledge is used to guide the generation of the graph, and the relationship between entities is detected to

create a structured scene graph. This structured representation makes the semantic and logical connections between scenes more effective and interpretable, improving wildfire recognition and detection.

In addition to the interference challenges from clouds, sunlight reflections, and smoke, another challenge in wildfire detection is the imbalance problem. The imbalance problem is mainly caused by the relationship imbalance, that is, a large number of coarse-grained relationships occupy the main body, and the fine-grained relationships that can feedback more details are lacking, which leads to the loss of important scene information and bias in the model. Therefore, this paper proposes a structured information fusion method based on gated GNNs, which dynamically captures semantic features and fuses multi-dimensional information to improve the frequency of fine-grained relationships and refine the frequency of coarse-grained relationships, thereby improving the imbalance problem and improving the understanding of wildfire targets. In order to verify the effectiveness of the proposed method, extensive ablation studies are carried out on public and self-built real datasets. Compared with the existing methods, the proposed method has a higher recall rate, which proves its detection advantage.

The main contributions of this paper can be summarized as follows:

- (1) We propose a structured information learning method for wildfire images based on GNNs to improve relationships between wildfire entities and reduce false positive rates through relational inference.
- (2) We propose a structured information fusion method for wildfire images based on gated GNNs to enhance the performance of scene graph generation and improve the detection accuracy through multi-dimensional information fusion.
- (3) We conducted extensive experiments to demonstrate the effectiveness of the proposed method and its detection advantages over existing methods.

2 Related Work

2.1 Scene Graph Generation

Various approaches adopt structured scene graph generation. For instance, the image-based method, exemplified by Graph R-CNN proposed by Yang [4] et al. incorporates a Region Proposal Network (RPN) and introduces an attention graph convolutional network (aGCN) to capture contextual information between objects and relations effectively. Chen [5] et al. propose a multi-agent policy gradient method, CMAT, that constructs the goal as a cooperative agent and then directly maximizes the graph-level metric as a reward. Yan [6] et al. proposed a novel predicate correlation Aware learning (PCPL) scheme to adaptively exploiting the correlation between predicate classes. Lin [7] et al. proposed a new message passing module that enhances node features with node-specific context information through a trilinear model. Unlike these methods, we use deformable convolution to improve scene graph generation for efficient wildfire detection.

2.2 Graph Neural Network

Compared to vanilla neural networks, GNNs are able to extract relationships between different objects in an image and have been used for object recognition and detection. Zhang

[8] et al. proposed the ResGCN model, comprising object and relation residual GNNs [9, 15, 20] that complement each other in capturing context information at different levels. Additionally, Guo [10] et al. introduced BA-SGG, a balanced adjustment-based SGG framework, while Chiou [11] et al. developed dynamic label frequency estimation (DLFE) to introduce more effective samples through data augmentation and averaging. Manessi [12] et al. proposed dynamic GCNs, leveraging Long Short-Term Memory networks to model temporal changes in graph structures, achieving notable improvements in classification tasks. However, when dealing with scenes involving non-rigid objects such as wildfires, the current methods still have low detection accuracy and high false alarm rate due to the complexity of the visual scene and the difficulty of semantic relationship reasoning. We use GNNS to explore optimized scene graph generation methods to avoid these problems.

3 Method

3.1 Structured Information Learning

The learning process of structured information is illustrated in Fig. 1. To solve the problem of high false positive rate and low accuracy in wildfire detection, we employ Fast R-CNN [13] to accurately localize entities and capture spatial information such as fire source and smoke. We augment entity relationships with semantic information, use deformable graph convolution for inference over relationships, and construct efficient data structures using KD-Tree [14].

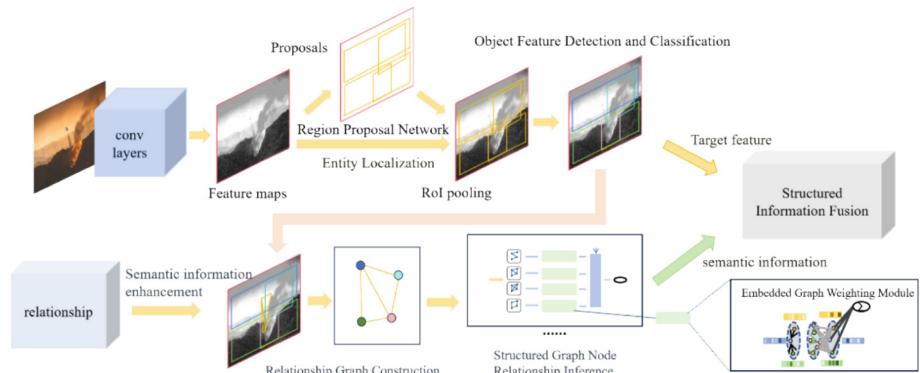


Fig. 1. The process of structured information learning. Entity location is performed on the image, and the generated object features, semantic features and relationship features are used to construct the entity relationship graph, which is sent to the variable graph convolution for relationship reasoning.

Embedded Graph Weighting Module. As shown in Fig. 2. The structure of EGWM. Node embedding maps each node to a low-dimensional vector space that is weighted and reconstructed by deformable convolution. Embedded Graph Weighting Module

(EGWM) consists of node embedding and weighted reconstruction. Through node embedding, each node is mapped to a low-dimensional vector space to capture node structure and feature information. Using these node embeddings as input, combined with the potential neighborhood graph construction and the deformable graph convolution [12] module, the weighted reconstruction of the relationship graph is achieved.

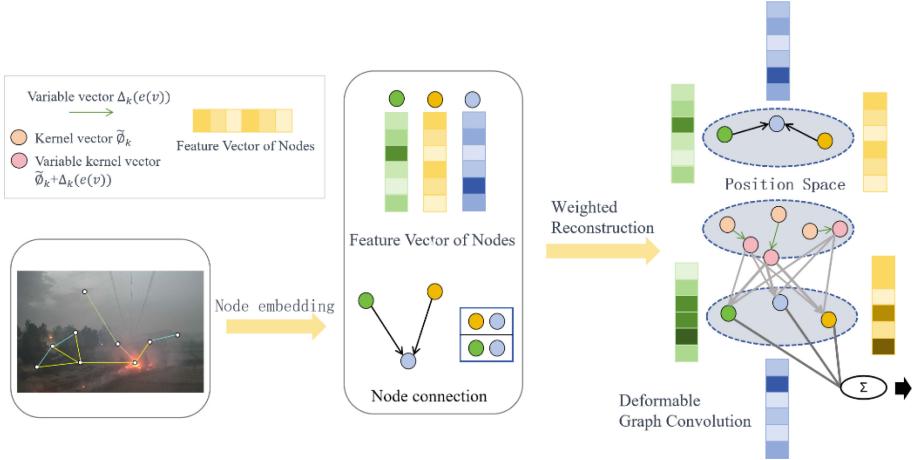


Fig. 2. The structure of EGWM. Node embedding maps each node to a low-dimensional vector space that is weighted and reconstructed by deformable convolution.

In deformable graph convolution, dynamic adjustment of the relationship graph is achieved by dynamically computing relationship weights to improve sensitivity to different node relationships. As shown in Eq. (1), y_v is the convolutional output, $h_u = \mathbb{R}^{dy}$ represents the feature of node u , and $r_{u,v}$ represents the relationship between node u and v . $N(v)$ is the neighborhood of node v , which overlaps with the finite support of the linear function $g(\cdot)$ centered at node v . The linear function takes the relationship vector as input and outputs features for the deformable graph convolution operation, where $\Delta k(e_v)$ is the deformable vector, e_v is the smoothed input features.

$$y_v = \sum_{u \in N(v)} g(r_{u,v}, \Delta k(e_v)) h_u \quad (1)$$

The relationship vector $r_{u,v}$ is encoded using a normalized relationship vector with extra dimensions for representing relationships between nodes with similar positional embeddings. The node positional vectors ϕ_u and ϕ_v denote the positional information of nodes u and v within the graph, indicating their relative positions. Equation (2) defines the relationship between node v and its neighboring node u , along with their node positional vectors ϕ_u and ϕ_v .

$$r_{u,v} = \begin{cases} [r'_{u,v} || 0] \in \mathbb{R}^{d_\phi+1}, & \phi_u \neq \phi_v \\ [0, 0, \dots, 1] \in \mathbb{R}^{d_\phi+1} & \text{otherwise} \end{cases} \quad (2)$$

where the $[ii]$ operator is concatenation operator, and $r'_{u,v}$ can be obtained from the positional vectors as Eq. (3):

$$r'_{u,v} = \frac{\phi_u - \phi_v}{\|\phi_u - \phi_v\|_2} \quad (3)$$

Structured Graph Node Relationship Inference. In the process of wildfire image relationship inference, constructing the latent neighborhood graph can effectively capture the node correlation. We use KD-Tree to quickly find nearest neighbors and improve the accuracy of relational inference through node embedding and deformable graph convolution. Meanwhile, influential nodes and edges are highlighted with an attention mechanism to enhance the importance of relational inference. The whole process of wildfire image relational inference is shown in Fig. 3.

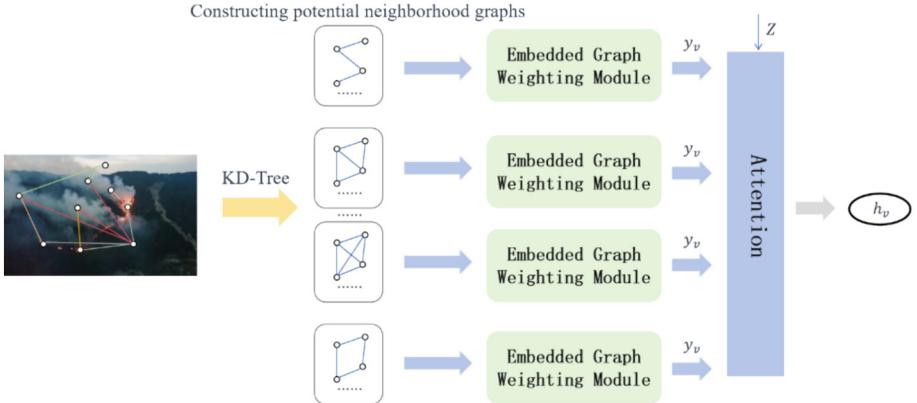


Fig.3. The structure of EGWM. Node embedding maps each node to a low-dimensional vector space that is weighted and reconstructed by deformable convolution.

Once the neighboring nodes are identified, the connections between each node and its neighbors are constructed into a potential neighborhood graph. This graph can represent the spatial relationships between nodes. After constructing all potential neighborhood graphs from 0 to $N + 1$, where N denotes the number of nodes, each potential neighborhood graph is processed using EGWM. Assuming that the output of processing each potential neighborhood graph is y_v , a score s_v is assigned to node v to represent the most appropriate neighborhood. Suppose the final output of the attention mechanism is \tilde{h}_v , then the overall relational inference output is shown in Eq. (4):

$$h_v = \sum_{l=0}^{N+1} \frac{s_v^{(l)} y_v^{(l)}}{\|y_v^{(l)}\|_2} \quad (4)$$

where N is the number of nodes and $z \in \mathbb{R}^{dy}$ is a learnable parameter. The score representing the most suitable neighborhood for node v can be obtained from Eq. (5).

$$s_v^{(l)} = \frac{\exp(\frac{z^T y_v^{(l)}}{\|y_v^{(l)}\|_2})}{\sum_{l=0}^{L+1} \exp(\frac{z^T y_v^{(l)}}{\|y_v^{(l)}\|_2})} \quad (5)$$

3.2 Structured Information Fusion

The fusion of structured information aims to comprehensively understand the image content, particularly in relation inference. We propose a structured information fusion method based on gated GNNs, comprising a framework for semantic information fusion in wildfire scenes and inference of wildfire objects based on fused information. The goal is to combine high-level semantic information with low-level object features through information enhancement, multi-dimensional feature fusion, and wildfire object inference. The process of structured information fusion is shown in Fig. 4.

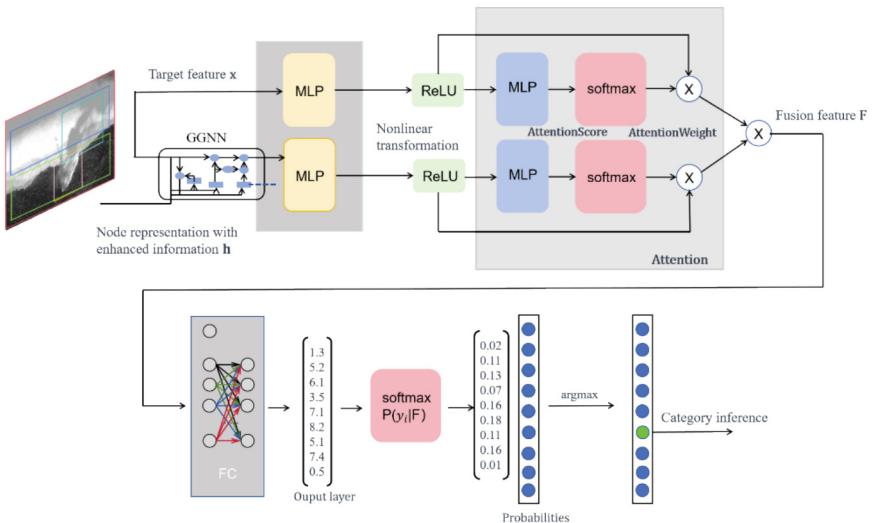


Fig. 4. The process of information fusion and object inference. The input semantic features and image features are enhanced by a gated GNN, followed by MLP for dimension alignment and feature fusion, and then fed into a fully connected layer and SoftMax for object inference.

Structured Association Node Information Enhancement. The core mechanism of information enhancement involves the update gate and reset gate, with low-level image features represented as node hidden state x . The high-level semantic information obtained from variable graph convolution relational reasoning is used as input feature h . At the beginning, the input is denoted by h_0 . The weight matrices for the update gate and reset

gate are denoted as W_u and W_r respectively. The principles of the update gate and reset gate are formulated as Eq. (6) and Eq. (7) respectively:

$$Z_t = \sigma(W_u \cdot [h_{t-1}, x_t]) \quad (6)$$

$$R_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \quad (7)$$

where σ is the sigmoid function. After computing the update gate and reset gate, the temporary node \tilde{h}_t is calculated based on the outputs of these gates. Finally, the current moment's hidden state h_t is obtained through the update node formula as shown in Eq. (8):

$$\begin{cases} \tilde{h}_t = \tanh(W \cdot [R_t \odot h_{t-1}, x_t]) \\ h_t = Z_t \odot h_{t-1} + (1 - Z_t) \odot \tilde{h}_t \end{cases} \quad (8)$$

where \odot is element-wise multiplication, W is the weight matrix.

Information Fusion and Object Inference. The input of multi-dimensional feature information fusion mainly includes the output of information enhancement and the underlying object features obtained by entity localization. The output of information augmentation consists of structured association nodes processed by gated GNNS, while the underlying object features are provided by the physical localization module. In order to overcome the dimension difference of input features, multi-layer perceptron (MLP) is introduced. The fused information is input into the fully connected layer and combined with the Softmax classifier for object classification. The attention weights are normalized by element multiplication, and then the weighted fusion features are fused by element multiplication to obtain the final fusion feature representation f . The specific fusion formula is shown in Eqs. (9) and (10):

$$\begin{cases} h_g'' = \text{AttentionWeight}_g \odot g' \\ h_x'' = \text{AttentionWeight}_x \odot x' \end{cases} \quad (9)$$

$$F = h_g'' \odot h_x'' \quad (9)$$

4 Experiments

4.1 Dataset and Implementation

To evaluate our method's efficacy, we conducted experiments on the self-vg and self-wildfire datasets. The self-vg dataset includes around 200,000 images and over 30 million inter-entity relationships. Our self-wildfire dataset, consisting of 2,000 real-world wildfire images, serves as a benchmark evaluation platform for wildfire detection methods. Additionally, we utilized three public graph datasets (citeseer, Pubmed [16], and Cora) to gauge our algorithm's proficiency in representing structured image data.

4.2 Comparison Experiment

Quantitative Analysis. In order to verify the performance of the proposed method, it is quantitatively compared with the existing methods based on global semantic information, Graph-RCNN[4], FREQ [17], CMAT [18], GPS-Net [7] and BGT-Net [19]. Each method is tested using the metric Recall@K, where for taking K equal to 50,100 on the self-vg dataset. The comparative results are shown in Table 1. It can be seen that the recall rate of the proposed method with K value equal to 50 and 100 under the three scene graph generation subtasks and the average value of the corresponding index recall rate is better than that of other methods.

Table 1. Comparison of scene graph generation recall rates in self-vg dataset.

Method	SGDET		SGCLS		PREDCLS		Mean
	R@50	R@100	R@50	R@100	R@50	R@100	
Graph-RCNN	11.5	14.1	29.4	30.8	53.9	59.3	33.1
FREQ	26.1	29.7	32.1	32.8	60.1	62.5	40.5
CMAT	27.4	30.6	38.6	39.2	66.2	67.9	44.9
GPS-Net	28.3	31.2	39.0	39.8	66.5	68.9	45.6
BGT-Net	28.7	32.0	40.6	42.9	67.5	69.0	46.7
Ours	32.6	36.5	40.7	45.7	67.9	69.1	48.3

The index value here is obtained by adding R@20 and R@50 of the three subtasks and then averaging. The comparative results on the self-wildFire dataset are shown in Table 2. Our method still achieves the optimal results among the proposed methods.

Table 2. Comparison of scene graph generation recall rates in self-wildfire dataset.

Method	SGDET		SGCLS		PREDCLS		Mean
	R@20	R@50	R@20	R@50	R@20	R@50	
Graph-RCNN	21.1	27.9	32.5	35.7	55.2	63.2	39.2
FREQ	22.4	29.3	34.8	37.3	58.3	64.1	41.0
CMAT	24.5	27.4	37.6	40.1	60.2	66.3	42.6
GPS-Net	26.2	29.8	40.5	44.2	63.3	67.5	45.2
BGT-Net	29.7	34.0	42.2	47.5	66.3	70.1	48.3
Ours	31.4	35.6	45.5	49.8	68.1	72.1	50.4

In order to further verify the performance of our method in relation inference of image structured information learning, we conduct comparative tests of relation inference on citeseer, Pubmed, Cora, self-vg and self-wildFire datasets, using AP as comparison indicators. As shown in Table 3, our method still achieves optimal performance.

Table 3. AP metrics for different frameworks on different datasets.

DataSet	Cora	citeseer	Pubmed	self-vg	self-wildFire
Graph-RCNN	63.10	66.13	70.06	25.72	29.99
FREQ	73.03	81.13	82.51	40.02	49.51
CMAT	83.13	86.14	88.67	50.16	70.14
GPS-Net	84.35	88.46	89.52	60.01	78.41
BGT-Net	88.14	92.73	93.14	68.41	82.00
Ours	91.10	92.91	94.61	70.05	88.72

Relationship Inference and Prediction Analysis. In order to further analyze the performance of the proposed method in the category inference of wildfire images, we conducted experiments to visually analyze the results of wildfire image association inference in Fig. 5. Compared with CMAT and BGT-Net, our method has better performance in predicting relationships, not only can predict relatively distant relationships, but also can predict more fine-grained relationships.

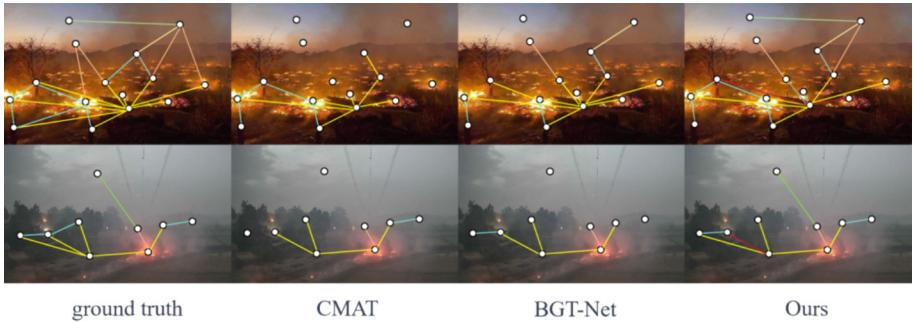


Fig. 5. Comparison results of wildfire image relationship inference experiment.

We also conducted the data statistics of the top 14 relations with frequency in the relation inference experiment, and the results are shown in Fig. 6. It can be seen that the proposed method has a good effect on predicting the entity relationships of wildfire with high probability, such as on, behind, in and from, and also has a good prediction on the relationships of suspended in, spread and ignite with medium occurrence probability.

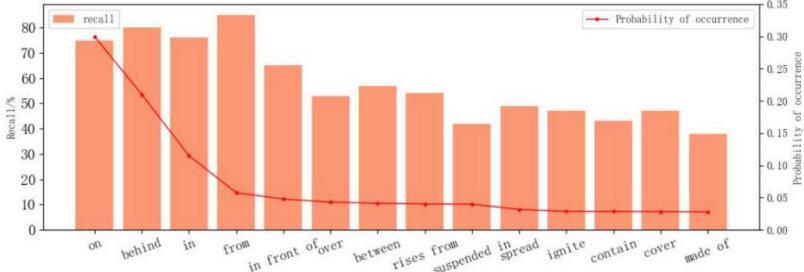


Fig.6. The recall statistics of relation prediction for the top 14 occurrence frequencies of the proposed framework.

Visualization Analysis. In addition, we use a parser on the self-wildfire dataset to visualize the detection results for comparison with BGT-Net. From Fig. 7, it can be observed that compared to BGT-Net, our method detects a richer variety of entity categories on the same wildfire image dataset. Additionally, our method identifies more abundant and finer-grained relationships between entities, thus detecting more categories of entities and reducing the false positive rate by correctly recognizing features through relational inference.

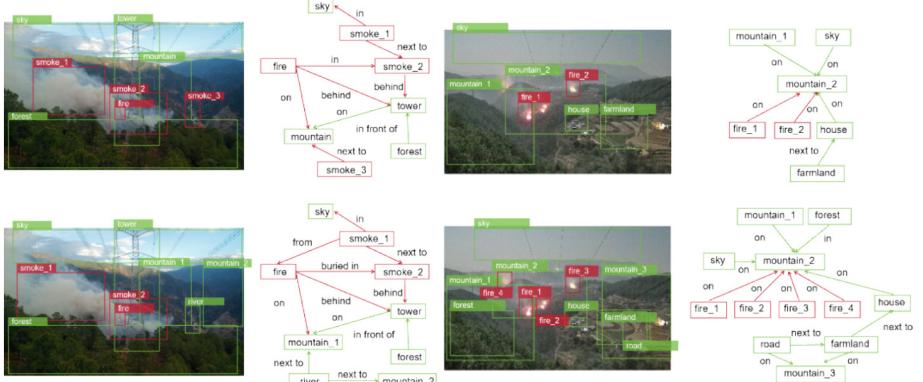


Fig. 7. Visualization of scene graph inference performance of our method (bottom) and BGT-Net (top).

Ablation Study. In order to show that the proposed method can effectively improve the performance of wildfire detection. Firstly, we replaced EGWM with a standard residual block. Secondly, we substituted the multi-dimensional feature fusion module with a simple concatenation module. The experimental results are shown in Table 4.

It can be seen that after making the above changes, the recall rate of the corresponding method decreases significantly, and the performance degradation of replacing EGWM with ordinary residual blocks is the most obvious, highlighting the importance of EGWM in the method studied in this paper.

Table 4. Results of ablation experiments on self-wildfire dataset.

Method	SGDET		SGCLS		PREDCLS		Mean
	R@20	R@50	R@20	R@50	R@20	R@50	
Method without EGWM	28.3	32.4	42.3	44.9	65.7	69.3	47.0
Method without feature fusion	29.6	34.2	44.5	48.4	66.3	70.3	48.9
Ours	31.4	35.6	45.5	49.8	68.1	72.1	50.4

5 Conclusion

In this paper, we propose a wildfire detection method based on structured scene graph information. We model the neighborhood relationship between samples by learning structured information based on GNNs. In addition, we use structured information fusion based on gated GNNs to improve the understanding of wildfire objects. We confirm the effectiveness of the proposed method by conducting extensive experiments on public and self-constructed real-world datasets. In addition, we further confirm the advantages and interpretability of the proposed method through visualization and analysis.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (Grant No. 62276054) and the Sichuan Science and Technology Program (Grant No.2023YFG0156).

References

1. Kang, Y., Sung, T., Im, J.: Toward an adaptable deep-learning model for satellite-based wildfire monitoring with consideration of environmental conditions. *Remote Sens. Environ.* **298**, 1138114 (2023). <https://doi.org/10.1016/j.rse.2023.1138114>
2. Sudhakar, K., Avanthika, T., Visali, J., Nivithaa, S.: A novel lightweight cnn model for real-time video fire smoke detection. In: 2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS), pp. 1056–1060 (2022)
3. Agirman, A.K., Tasdemir, K.: BLSTM based night-time wildfire detection from video. *PLoS ONE* **17**(6), e0269161 (2022). <https://doi.org/10.1371/journal.pone.0269161>
4. Yang, J., Lu, J., Lee, S., Batra, D., Parikh, D.: Graph R-CNN for scene graph generation. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) *ECCV 2018. LNCS*, vol. 11205, pp. 670–685. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01246-5_41
5. Chen, T., Yu, W., Chen, R., et al.: Knowledge-embedded routing network for scene graph generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6163–6171. IEEE/CVF (2019)
6. Yan, S., Shen, C., Jin, Z., Huang, J., Jiang, R., Chen, Y., Hua, X.: PCPL: predicate-correlation perception learning for unbiased scene graph generation. In: Proceedings of the 28th ACM International Conference on Multimedia, pp. 265–273 (2020)
7. Lin, X., Ding, C., Zeng, J., et al.: GPS-Net: graph property sensing network for scene graph generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3746–3753. IEEE/CVF (2020)

8. Zhang, J., Zhang, Y., Wu, B., et al.: Dual RESGCN for balanced scene graph generation. Journal of arXiv Preprint [arXiv:2011.04234](https://arxiv.org/abs/2011.04234) (2020)
9. Wu, Z., Pan, S., Chen, F., et al.: A comprehensive survey on graph neural networks. IEEE Trans. Neural Networks Learn. Syst. **32**(1), 4–24 (2020)
10. Guo, Y., Gao, L., Wang, X., et al.: From general to specific: Informative scene graph generation via balance adjustment. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 16383–16392. IEEE/CVF (2021)
11. Chiou, M.-J., Ding, H., Yan, H., et al.: Recovering the unbiased scene graphs from the biased ones. In: Proceedings of the 29th ACM International Conference on Multimedia, pp. 1581–1590. ACM (2021)
12. Manessi, F., Rozza, A., Manzo, M.: Dynamic graph convolutional networks. Pattern Recogn. **97**, 107000 (2020). <https://doi.org/10.1016/j.patcog.2019.107000>
13. Girshick, R.: Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448. IEEE (2015)
14. Friedman, J.H., Bentley, J.L., Finkel, R.A.: An algorithm for finding best matches in logarithmic expected time. ACM Trans. Math. Software (TOMS) **3**(3), 209 (1977)
15. Park, J., Yoo, S., Park, J., et al.: Deformable graph convolutional networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 7949–79. AAAI (2022)
16. Sen, P., Namata, G., Bilgic, M., et al.: Collective classification in network data. AI Mag. **29**(3), 93 (2008)
17. Zellers, R., Yatskar, M., Thomson, S., et al.: Neural motifs: scene graph parsing with global context. In: Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5831–5840. IEEE/CVF (2018)
18. Chen, L., Zhang, H., Xiao, J., et al.: Counterfactual critic multi-agent training for scene graph generation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 4613–4623. IEEE/CVF (2019)
19. Dhingra, N., Ritter, F., Kunz, A.: BGT-Net: bidirectional GRU transformer network for scene graph generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2150–2159. IEEE/CVF (2021)
20. Ren, Z.H., et al.: DeepMPF: deep learning framework for predicting drug-target interactions based on multi-modal representation with meta-path semantic analysis. J. Transl. Med. **21**(1), 48 (2023). <https://doi.org/10.1186/s12967-023-03876-3>



Towards Highly Effective Moving Tiny Ball Tracking via Vision Transformer

Jizhe Yu¹(✉) , Yu Liu¹, Hongkui Wei², Kaiping Xu¹, Yifei Cao¹, and Jiangquan Li¹

¹ DUT School of Software Technology and DUT-RU International School of Information Science and Engineering, Dalian University of Technology, Dalian, China
yujizhe@mail.dlut.edu.cn

² State Key Laboratory of Intelligent Manufacturing System Technology, Beijing Institute of Electronic System Engineering, Beijing, China

Abstract. Recent tiny ball tracking methods based on deep neural networks have significantly progressed. However, since moving balls in the video are always blurred, most existing methods cannot achieve accurate tracking due to limited receptive fields and sampling depth. Furthermore, as high-resolution competition videos become increasingly common, existing methods perform poorly on high-resolution images. To this end, we provide a strong baseline for tracking tiny balls called TrackFormer. Firstly, we use Vision Transformer to build the whole network architecture and enhance the tiny ball localization through its powerful spatial mining ability. Secondly, we develop a Global Context Sampling Module (GCSM) to capture more powerful global features, thereby increasing the accuracy of tiny ball identification. Finally, we design a Context Enhancement Module (CEM) to enhance tiny ball semantics to achieve robust tracking performance. To promote research and development of tiny ball tracking, we establish a Large-scale Tiny Ball Tracking dataset called LaTBT. Specifically, LaTBT is founded on three types of tiny balls (badminton, tennis, and squash), offering more than 300 video sequences and over 223K annotations from 19 types of professional matches to address various tracking challenges in diverse and complex backgrounds. To our knowledge, LaTBT is the first large-scale dataset for tiny ball tracking. Experiments demonstrate that our baseline achieves state-of-the-art performance on our proposed benchmark dataset. The dataset and the algorithm code are available at <https://github.com/Gi-gigi/TrackFormer>.

Keywords: Visual tracking · Tiny ball tracking · Benchmark dataset · Transformer baseline · Global context guidance

1 Introduction

Tiny ball tracking is a crucial research field in precision sports science [1]. The trajectory, location, and usage information of tiny balls enhance the spectatorship of matches and aid in match analysis and judgment, thus attracting significant research interest. Although existing tiny ball tracking algorithms have made tremendous progress, the increasing resolution of television broadcasts today has far exceeded their performance capabilities.

As shown in Fig. 1(a), TrackNet [2, 3], based on the VGG backbone, can not provide a satisfactory viewing experience compared to Ground Truth (GT). This is mainly because the backbone network was initially designed for image classification and is unsuitable for training tiny ball datasets. Furthermore, processing high-resolution images leads to a significant increase in memory usage. Therefore, developing a cost-effective and high-performing tiny ball tracking network is crucial for match analysis.



Fig. 1. Visual comparison with state-of-the-art method. (a) Input image and GT(purple circle). (b) TrackNet [3] (yellow circle). (c) Our method (red circle). (Color figure online)

To this end, we propose a robust baseline method called TrackFormer, which adopts Vision Transformer as an architecture to overcome the limitations of fixed receptive field and limited sampling depth of traditional tracking networks. It is also able to capture semantic information crucial for tracking balls. Furthermore, many Transformer-based trackers have demonstrated excellent performance on multiple tracking datasets [4, 5], which can accurately locate and identify objects in the scene due to their excellent spatial mining capabilities. For this purpose, our TrackFormer can precisely track the ball frame-by-frame, as shown in Fig. 1(b). Our method can still locate the ball efficiently even in highly blurry conditions (1st and 3rd frames). To enhance the robustness of tracking performance, this paper proposes a Global Context Sampling Module (GCSM) to obtain global context features and further promote the ball localization ability of the network. To effectively address the issue of low image quality caused by high-speed moving tiny balls, we also design a Context Enhancement Module (CEM) to minimize background noise interference, enabling the network to effectively utilize global context for guiding top-down and achieve better tracking performance.

We believe another significant reason for blocking tracking capability is the drawbacks of existing tracking datasets. In the field of tiny ball tracking, the existing TrackNet tracking benchmark consists of two widely used datasets. However, both datasets are of small scale, not only limited in the variety of sports but also lacking in the diversity of

competition environments, rendering them unable to represent the challenges encountered in tracking tiny balls in professional competitions. Taking TrackNet’s badminton dataset as an example, it includes videos from only a single amateur or professional match, limiting the applicability of tracking methods and performing poorly in challenging scenarios. To provide a comprehensive evaluation platform for tiny ball tracking, we contribute a large-scale tracking benchmark called LaTBT. Firstly, this dataset is a pioneering large-scale, high-resolution (1080 p) tiny ball tracking dataset, containing over 300 video clips and more than 223 K manually annotated binary heatmaps with frames and resolutions far exceeding that of popular tiny ball tracking datasets. Secondly, we meticulously annotate each frame to ensure coverage of a broader range of tiny ball tracking challenges, such as blur, afterimage, overlap, and other low-quality image problems often missing in existing datasets. Moreover, LaTBT contains tiny ball data from 19 types of professional competitions, which is suitable for a wide range of professional game scenarios and can effectively evaluate the performance of different tracking algorithms in actual scenarios. Thus, our LaTBT dataset is distinguished not only by its high diversity but also by its significant challenges. Our contributions are as follows:

- In this paper, we propose a novel transformer-based baseline method for tiny ball tracking called TrackFormer, which consists of two main components, namely the Global Context Sampling module (GCSM) and the Context Enhancement module (CEM). With the cooperation of the two modules, compared with the traditional tracking method, the proposed method effectively balances the relationship between sampling depth and receptive field, thereby improving the tracking performance.
- We are the first to propose a large-scale dataset for tiny ball tracking. LaTBT includes vast data, rich scene diversity, and careful annotations. We analyze the properties, advantages, and uniqueness of our benchmarks and compare them with other datasets in Sect. 3.
- Benchmark tests on the proposed dataset demonstrate that our baseline method achieves unprecedented performance in tiny ball tracking. We also report the results of six additional state-of-the-art trackers on LaTBT to be more convincing. Moreover, we run at the speed of 30 FPS on a single GPU.

2 Related Work

Recently, many trackers have been proposed to address the various challenges in tiny object tracking. Yu et al. [6] combine object detection frameworks with deep neural networks to track small objects in airborne images. Zhu et al. [7] utilize knowledge distillation networks to enhance the recognition capability of tiny objects. Yang et al. [8] propose a relational inference network to track small objects in satellite videos through semantic relations among frames. Wu et al. [9] adopt unclassified backbone networks to enhance infrared small object detection performance. Sun et al. [3] develop a lightweight tiny ball tracking network called TrackNet to achieve real-time tracking of shuttlecocks in videos. In this paper, we primarily focus on tracking tiny balls.

Outstanding tracking performance relies on a robust network architecture. Inspired by UNet [18], TrackNet [2, 3] employs a symmetrical encoder-decoder architecture to

capture coarse-to-fine image features, demonstrating exceptional potential in tracking tiny balls. As followers of UNet, Qin et al. [10] adopt a two-level nested U-shaped structure to acquire multi-scale features and detect small salient objects without changing resolution. To further enhance model performance, Wu et al. [9] introduce a multi-scale aggregation module to improve the tracking of infrared small objects. Additionally, recent successes in applying Transformers from language processing to video tracking tasks [4, 5] have shown that transformer-based networks can effectively model the global context [20], thus enhancing the localization and identification of the ball. In our work, we draw inspiration from these methods, including U-shaped networks, Transformers, and multi-scale aggregation, but uniquely incorporate these approaches in our model.

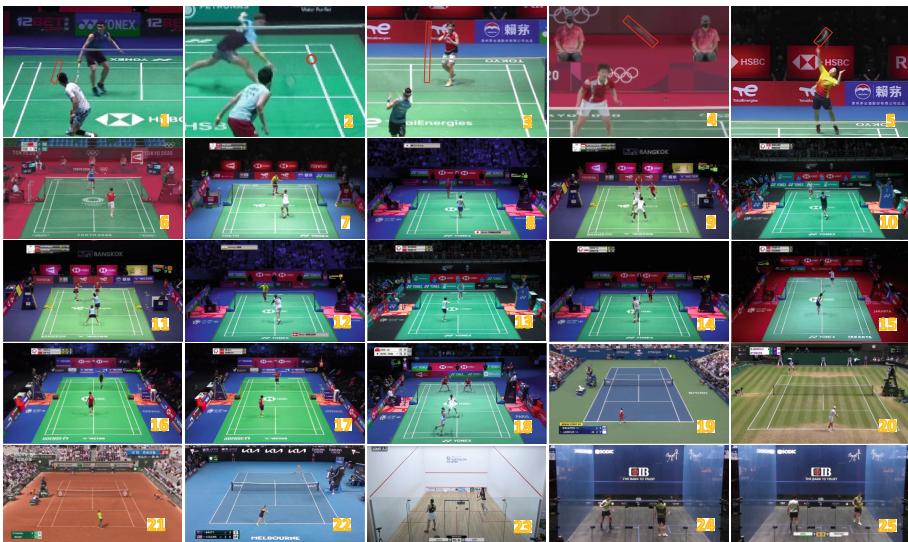


Fig. 2. Sample from our LaTBT.

3 Our Tiny Ball Tracking Benchmark Dataset

3.1 TrackNet Tracking Benchmark

The current tiny ball tracking datasets face three significant challenges. Firstly, they lack the necessary data scale to represent the complex and diverse environment of actual games. Most of these datasets are derived from amateur competition videos, and the size of the ball is fixed on a regular scale, which does not account for changes in rotation, scaling, or deformation. Secondly, the datasets consist of low-resolution, low-frame-rate images (720p 25fps), which do not align with current criteria for image processing application. Additionally, there are concerns about the quality of annotations, with issues such as extreme blur (Fig. 1), overlap (Fig. 2), and scale variation (Fig. 3) frequently omitted from existing datasets.

3.2 LaBTB Tracking Benchmark

To solve the above problems, we create a Large-scale Tiny Ball Tracking dataset called LaBTB, which consists of three types of tiny ball sports: badminton, tennis, and squash. First, regarding dataset scale and professionalism. The badminton dataset is derived from live broadcasts of globally renowned badminton tournaments in 2022 (Winter Olympics Badminton, World Badminton Championships, Thomas Cup, Uber Cup, Sudirman Cup, All England Open, Malaysia Open, Indonesia Open, Denmark Open, Swiss Open, Thailand Open, German Open, and French Open), covering men's singles, women's singles, men's doubles, and women's doubles, with more than 220 video clips and 141,151 frames annotated from approximately 80 h of video. The tennis dataset is selected from Grand Slam Tennis in 2022, including men's and women's singles, with 50,732 annotated frames and more than 70 video clips. The squash dataset is from the semifinals and finals of the World Championship in 2022, with 31,427 frames annotated and 30 video clips. All the full videos can be publicly accessed through YouTube at <https://www.youtube.com/@bwftv>. Secondly, we provide intuitive visual examples, as shown in Fig. 1, 2, 3, 4 and 5 show examples of different ball sizes, No. 6–18 show the complex and diverse badminton dataset, No. 19–22 show the tennis dataset, and No. 23–25 show the squash dataset. Furthermore, we create a low-resolution (720p 25fps) badminton test dataset containing 28,220 frames, selected from the 2018 Women's Singles World Championships and the All England Finals, to evaluate the model's generalization ability. Then, all three datasets choose high-resolution (1080p) videos, with frame rates including 25fps, 30fps, and 60fps. Lastly, more than three subjects carefully checked our tiny ball dataset to avoid low-level annotation errors. In summary, the LaBTB dataset surpasses existing ball tracking datasets in scale, professionalism, resolution, diversity, and challenges. To our knowledge, our benchmark is the largest, highly diverse, highly professional, high-resolution, and high-quality annotated tiny ball tracking dataset.

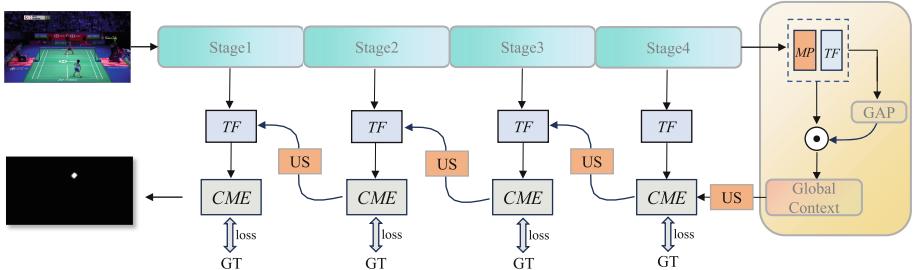


Fig. 3. The framework of our proposed tiny ball tracking network TrackFormer with two key modules: Global Context Sampling Module and Context Enhancement Module, which cooperated to improve tiny ball tracking performance.

4 Method

4.1 Overview Architecture

We adopt PVT-v2 as the encoder backbone, whose Transformer architecture offers excellent global perceptual ability and efficient computational performance. To better capture global context information, we downsample the network and design a Global Context Sampling Module (GCSM) to model deep features. Subsequently, we develop a Context Enhancement Module (CEM) that effectively integrates global context with decoder features to reduce background noise interference [18], enhancing ball semantics and improving tracking performance. The network architecture is shown in Fig. 3.

4.2 Global Context Sampling Module

In the above, we have mentioned that global context information plays a crucial role in enhancing the tracking performance of the tiny ball. Therefore, we design the GCSM to extract deep semantic information by learning the global view of the entire image, as shown in Fig. 3. Next, we clarify the design details of the GCSM.

Suppose that the feature output from the top layer of the encoder is X . To increase the sampling depth, we downsample the input feature map by a factor of 2 using max-pooling. Subsequently, a transformer block \mathcal{TF} and \mathcal{MLP} from the original Transformer [20] are used for global context modeling. This process can be described as

$$X_g = \mathcal{MLP}(\mathcal{TF}(\text{Reshape}(Mp(X)))) \quad (1)$$

where Mp is the max-pooling downsampling. *Reshape* represents the tensor change from $[B, C, H, W]$ to $[B, H \times W, C]$. Moreover, Transformer \mathcal{TF} can be described as

$$\mathcal{TF} = \text{Softmax}\left(QK^T / \sqrt{d}\right)V \quad (2)$$

To obtain a global view of the entire image, we first employ Global Average Pooling (*GAP*) to acquire a global attention map X_a . Then, X_a is used to adaptively weight X_g to enhance the location information of the foreground object:

$$X_a = \sigma(GAP(X_g)) \quad (3)$$

$$X_g^\omega = X_g \odot X_a \quad (4)$$

where σ is the Sigmoid nonlinear activation function. \odot represents element-wise multiplication and X_g^ω is denoted as the after weighted X_g .

In summary, our GPSM can enlarge the receptive field and sampling depth to localize the tiny ball while delivering global context to the Context Enhancement Module (CEM) to mitigate the issues of ball semantics dilution and inaccurate localization.

4.3 Context Enhancement Module

Intuitively, global context features and initial features from the encoder are two types of inconsistent features. If they are directly integrated, it may lead to feature misalignment and introduce a lot of background noise [14, 18], which is detrimental to tiny ball tracking (such as “TrackNet” from Fig. 1). For this reason, we construct CEM with two purposes: to enhance the semantics of the ball guided by global context and to effectively integrate two complementary features to mitigate feature differences.

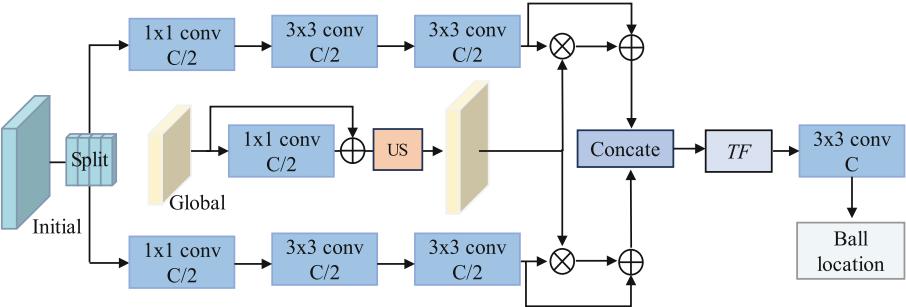


Fig. 4. Illustration of Context Enhancement Module (CEM) for effective feature fusion.

As shown in Fig. 4. Suppose that the initial feature is $F_i^S \in \mathcal{R}^{N \times C \times H \times W}$ ($S = 1 \sim 4$, C, H, W are the encoder stages, channel number, height, and width). To fully utilize the global context to guide feature fusion, we first adopt two 1×1 convolution layers to obtain branch features, namely $B_1 \in \mathcal{R}^{N \times C/2 \times H \times W}$ and $B_2 \in \mathcal{R}^{N \times C/2 \times H \times W}$. After that, two layers of 3×3 convolution followed by BN and Relu layers are passed, respectively. The above process can be described (taking B_1 as an example) as follows:

$$B_1 = conv_2^{3 \times 3}(conv_1^{3 \times 3}(conv_1^{1 \times 1}(F_i^S))) \quad (5)$$

Since both the global context and the initial feature have different resolutions, we first interpolate the global context feature $G \in \mathcal{R}^{N \times C \times H \times W}$ to match the dimensions of the feature map B_1 , and then is processed by the residual block to generate strengthened residual context:

$$G_{res} = conv^{1 \times 1}(Up(G)) + G \quad (6)$$

Next, we aggregate the processed initial features and the global context. First G_{res} is evenly split into two feature maps $G_{res}^1 \in \mathcal{R}^{N \times C/2 \times H \times W}$ and $G_{res}^2 \in \mathcal{R}^{N \times C/2 \times H \times W}$ along the channel dimension. We then apply the global contexts G_{res}^1 and G_{res}^2 to the features B_1 and B_2 respectively, and adopt B_1 and B_2 to obtain the identity mappings, which can be formulated as (taking B_1 branch as an example):

$$G_{res}^1 = \mathcal{F}_{split}(G_{res}) \quad (7)$$

$$M_1 = G_{res}^1 \odot B_1 + B_1 \quad (8)$$

where $\mathcal{F}_{split}(\cdot)$ denotes the split operation along channel dimension. $M_1 \in \mathcal{R}^{N \times C/2 \times H \times W}$ is feature attention map after merged.

Finally, we concatenate the two branch features and further refine them with the original transformer $\mathcal{T}\mathcal{F}$, and then a 3×3 convolutional layer is used to determine the final location of the ball:

$$P = \sigma \left(conv^{3 \times 3} (\mathcal{T}\mathcal{F}(M_1 \oplus M_2)) \right) \quad (9)$$

Where \oplus represents the concatenation. In conclusion, the proposed CEM can correct feature misalignment and achieve self-refinement under the guidance of global context to improve ball tracking performance.

For training, we apply the BCE loss as used in U²Net [10] for each decoder stage:

$$\mathcal{L}_{bce} = - \sum_{x=1}^H \sum_{y=1}^W [G(x, y) \log(P(x, y)) + (1 - G(x, y)) \log(1 - P(x, y))] \quad (10)$$

where $G(x, y)$ and $P(x, y)$ are the ground truth label and the predicted label at the location (x, y) , respectively. H and W are the height and width of the images, respectively.

Therefore, the total loss of our training is defined as follows:

$$\mathcal{L} = \sum_{s=1}^4 \mathcal{L}_{bce}^s(P(x, y), G(x, y)) \quad (11)$$

where the lowercase subscript s represents each decoder stage. During inference, we integrate the features of the various stages, like U²Net, to generate the final prediction result.

5 Experiments

5.1 Experiments Setup

Evaluation Datasets. We mainly use our proposed benchmark datasets to evaluate the performance of our method along with other state-of-the-art methods. We adhere to the dataset partition ratios in [11] to divide the tiny ball tracking datasets into training, validation, and test sets. Specifically, LaTBT comprises 147,385 images for training, 29,030 images for validation, and 46,895 images for testing. Since badminton is the fastest ball sport, we also evaluate the model's generalization ability on our proposed low-resolution badminton dataset (LR Test).

Implementation Details. Our proposed TrackFormer and all the comparative algorithms are trained from scratch in an end-to-end manner. We run all experiments on the publicly available Pytorch 1.5.0 platform. An RTX 2080Ti GPU card (with 12 GB memory) is used for training and testing. During network training, each frame is first preprocessed to $[960 \times 960]$ and then resize to $[224 \times 224]$ for the PVT [12] backbones, and data augmentation methods such as normalizing and flipping are used. Our encoder parameters are initialized from PVT. Adam optimizer [13] with default hyperparameters is adopted to train the network. We train the network for 50 epochs with a batch size of 16. The initial learning rate is $lr = 1e-5$, and warm-up and linear decay strategies adjust the learning rate. During testing, each image is resized to $[1024 \times 1024]$ and then passed into the network without any post-processing.

Evaluation Metrics. In this study, the confusion matrix is chosen as the primary evaluation metric [3] to evaluate the performance of the model. The sigmoid function converts the network output to score maps between 0 and 1, which are then classified into pixels values of 0 or 1 using a threshold of 0.5. The predicted ball location is determined as the center of the largest area in the heatmap. A tolerance value (tol) is set to measure the network’s success in identifying the ball, with tol being determined as 10 based on the average diameter of all tiny balls appearing in the frames.

When there is no ball detected by the model, the output will show TN, which means that there is no ball within the frame. On the other hand, if the model identifies a non-ball object, the output will show FP. However, when a ball is visible within the frame, the output will show TP if the model correctly identifies the ball. This means that the difference between the predicted position by the model and the ground truth position is less than the defined tolerance value (tol). In cases where the model fails to identify the ball, the output will show FN1. Meanwhile, if the model identifies an object but the predicted distance is greater than the tolerance value (tol) from the ground truth position, the output will show FN2. Accuracy, Precision, and Recall can be formulated as follows:

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN1 + FN2) \quad (12)$$

$$\text{Precision} = TP / (TP + FP) \quad (13)$$

$$\text{Recall} = TP / (TP + FN1 + FN2) \quad (14)$$

To balance the relationship between precision and recall, $F_\beta(\beta=1)$ score [19] is used as a comprehensive indicator, and $F_\beta(\beta=2)$ emphasizes the importance of recall.

$$F_\beta = \frac{(1 + \beta^2) \times \text{Precision} \times \text{Recall}}{\beta^2 \times \text{Precision} + \text{Recall}} \quad (15)$$

5.2 Comparison with State-of-the-Arts

We compare the proposed TrackFormer with six recent state-of-the-art models, including Vgg-TrackNet and Res-TrackNet [2][3], UIUNet [9], InSPyReNet [14], RDIAN [16], AGPCNet [15], and MTU-Net [17]. For a fair comparison, the predicted maps are either provided by the authors or generated by the officially released pre-trained models. It is worth noting that the comparison algorithms we selected are mainly from infrared tiny object detection and salient object detection, primarily because they are highly similar to our task, and both can track tiny objects. During the training, we employ multiple frames input and single frame outputs (MISO) design strategy [3] to achieve simple ball tracking.

Quantitative Evaluation. Table 1 reports the quantitative results of the LaTBT benchmark and LR Test, in which we compare our method with the six state-of-the-art algorithms in terms of Accuracy, F_1 , and F_2 . Our TrackFormer tracking network achieves

the best results across all metrics on the LaTBT benchmark. Notably, our method prioritizes high recall to minimize tracking omissions, as missing a positive instance is more severe than mistakenly identifying a false positive. Consequently, our method achieves the highest F_2 score. Other comprehensive indicators Acc and F_1 also demonstrate the excellent tracking ability of our method and show superior performance compared to the VIT-based MTU-net. We also test the model trained on LaTBT on our proposed large LR Test, and the results consistently show our method's outstanding performance. This indicates its strong ability to handle challenging inputs. Figure 5 shows the accuracy-measure curve, further demonstrating our tracking method's superior performance. At different tolerance levels, the red curve of our method is significantly higher than the other curves on both datasets, underscoring the effectiveness of our proposed method in tracking tiny balls. Apart from the visualizations in Fig. 1, we also provide clear and easy-to-understand visualization videos from <https://github.com/Gi-gigi/TrackFormer>

Table 1. Comparison of TrackFormer with state-of-the-art Tracking methods. The best performance in each column is highlighted in bold.

Summary	Speed	Param	LaTBT			LR Test		
			Acc	F_1	F_2	Acc	F_1	F_2
Method	FPS	M						
Vgg-TrackNet	36.2	26.45	.864	.919	.882	.737	.809	.733
Res-TrackNet	14.05	35.41	.909	.950	.930	.765	.836	.776
InSPyReNet	12	28.10	.934	.963	.953	.790	.855	.797
UIUNet	28.4	50.54	.948	.971	.964	.814	.873	.821
RDIAN	32.4	17.08	.925	.956	.937	.809	.871	.828
AGPCNet	20.4	52.51	.951	.973	.974	.817	.878	.838
MTU-Net	25.01	57.03	.954	.974	.968	.821	.882	.847
Ours	31.7	47.06	.963	.979	.976	.854	.905	.879

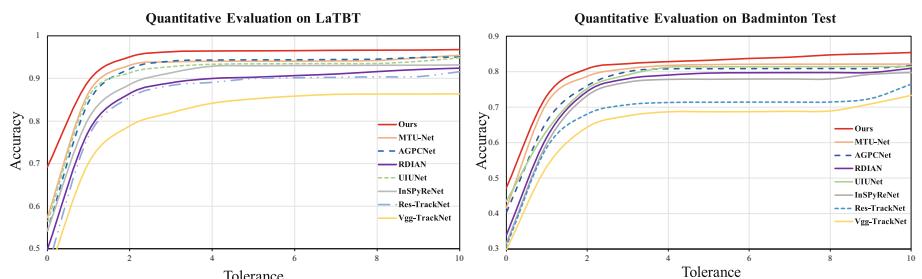


Fig. 5. Accuracy-measure curves of the proposed method and other SOTA algorithms on two benchmark datasets.

Additionally, we list the parameters and speeds of each method to measure efficiency. Our method is more lightweight and achieves real-time speed compared to the latest SOTA methods, with highly competitive performance. Another method, RDIAN, shows considerable performance gaps although it has a faster inference speed. In summary, TrackFormer achieves a balance between efficiency and performance.

Table 2. Ablation study with different components combinations on LaTBT.

No	Components Setting	LaTBT		
		Acc	F_1	F_2
1	Baseline	.907	.946	.928
2	+GCSM	.950	.972	.962
3	+ GCSM + CEM	.963	.979	.976

5.3 Ablation Study

We conduct the ablation study on each key component designed in this paper, using PVT-v2 [12] as the backbone on the LaTBT dataset. As shown in Table 2, the model integrating all components (GCSM and CEM) achieved the best performance, highlighting their importance in achieving superior tracking results. The “Baseline” model (No. 1) adopts a structure similar to UNet [18]. Introducing the GCSM module (No. 2) significantly increases accuracy from 0.907 to 0.950. Furthermore, adding the CEM module (No. 3) further increases the accuracy by 6% compared to the Baseline model (No. 1). The F_1 and F_2 scores also show gradual increases. In summary, the two components complement each other and jointly enhance the robustness of tiny ball tracking.

6 Conclusion

We propose a novel tiny ball tracking baseline, called TrackFormer, to identify and locate tiny balls from low quality video frames. This network optimizes tracking performance through design two novel modules: the Global Context Sampling Module (GCSM) and the Context Enhancement Module (CEM). GCSM improves ball location by deepening sampling depth, while CEM mitigates background noise to enhance ball semantics. By integrating these modules, TrackFormer achieves highly effective tiny ball tracking. To provide a comprehensive evaluation platform, we introduce a large-scale, challenging, and diverse tracking benchmark, called LaTBT. Experimental results show that TrackFormer effectively addresses challenges such as blur, afterimage, and overlap in low-quality images on LaTBT. Future work will collect a broader range of tiny ball tracking datasets, including table tennis, baseball, and ice hockey. et al., to develop a more comprehensive evaluation platform. We also plan to borrow ideas from single-stage object detectors to enhance performance further.

Acknowledgments. This work is Supported by the National Natural Science Foundation of China under Grant 61672128, and in part by the Dalian Key Field Innovation Team Support Plan under Grant 2020RT07.

References

1. Chu, W.-T., Situmeang, S.I.G.: Badminton video analysis based on spatiotemporal and stroke features. In: Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval (2017)
2. Huang, Y., et al.: TrackNet: a deep learning network for tracking high-speed and tiny objects in sports applications. In: 2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance, pp. 1–8 (2019)
3. Sun, Nien-En et al. TrackNetV2: efficient shuttlecock tracking network. In: 2020 International Conference on Pervasive Artificial Intelligence, pp. 86–91 (2020)
4. Paul, M., et al.: Robust visual tracking by segmentation. arXiv preprint [arXiv:2203.11191](https://arxiv.org/abs/2203.11191) (2022)
5. Cao, S., et al.: SwinCGH-Net: enhancing robustness of object detection in autonomous driving with weather noise via attention. In: International Conference on Intelligent Computing (2023)
6. Yu, M., Leung, H.: Small-object detection for UAV-based images. In: 2023 IEEE International Systems Conference, pp. 1–6 (2023)
7. Zhu, Y., et al.: Tiny object tracking: a large-scale dataset and a baseline. IEEE Trans. Neural Networks and Learning Systems PP (2022)
8. Yang, X., et al.: Relation learning reasoning meets tiny object tracking in satellite videos. IEEE Trans. Geosci. Remote Sens. (2024)
9. Wu, X., et al.: UIU-Net: U-Net in U-Net for infrared small object detection. IEEE Trans. Image Process. **32**, 364–376 (2022)
10. Qin, X., et al.: U2-Net: Going Deeper with Nested U-Structure for Salient Object Detection. Pattern Recognit. **106**, 107404 (2020)
11. Tian, X., et al.: Bi-directional object-context prioritization learning for saliency ranking. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5872–5881 (2022)
12. Wang, W., et al.: PVT v2: improved baselines with pyramid vision transformer. Comput. Visual Media **8**, 415–424 (2021)
13. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. CoRR. arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)
14. Kim, T., et al.: Revisiting image pyramid structure for high resolution salient object detection. In: Asian Conference on Computer Vision (2022)
15. Zhang, T., et al.: AGPCNet: attention-guided pyramid context networks for infrared small target detection. arXiv preprint [arXiv:2111.03580](https://arxiv.org/abs/2111.03580) (2021)
16. Sun, H., et al.: Receptive-field and direction induced attention network for infrared dim small target detection with a large-scale dataset IRDST. IEEE Trans. Geosci. Remote Sens. **61**, 1–13 (2023)
17. Wu, T., et al.: MTU-Net: Multilevel TransUNet for Space-Based Infrared Tiny Ship Detection. IEEE Trans. Geosci. Remote Sens. **61**, 1–15 (2022)
18. Liu, J., et al.: PoolNet+: exploring the potential of pooling for salient object detection. IEEE Trans. Pattern Anal. Mach. Intell. **45**, 887–904 (2022)
19. Hamed, B.A., Ibrahim, O.A.S., Abd, E.-H.: Optimizing classification efficiency with machine learning techniques for pattern matching. J. Big Data **10**(1), 124 (2023)
20. Liu, N., et al.: Visual saliency transformer. In: 2021 IEEE/CVF International Conference on Computer Vision, pp. 4702–4712 (2021)



SDE-Net: Skeleton Action Recognition Based on Spatio-Temporal Dependence Enhanced Networks

Qing Sun, Jiuzhen Liang^(✉), Zhou Xinwen, and Hao Liu

Changzhou University, Changzhou 213155, China

{jzliang, zhouxw, helenliuhao}@cczu.edu.cn

Abstract. Graph Convolutional Networks (GCNs) have succeeded remarkably in skeleton-based action recognition tasks. However, the existing GCN-based methods, where the interframe edges of the graph connect only the same joints and ignore the correlations between different joints, cannot effectively capture the spatiotemporal dependencies between joints. So this paper, we design a spatiotemporal dependence enhancement network (named SDE-Net), which utilizes Convolutional Neural Network (CNN) to make up for the deficiency of GCN in spatiotemporal dependence modeling. SDE-Net makes good use of the structural information between joints and effectively enhances the spatiotemporal dependence of different joints between neighboring frames in the region. SDE-Net consists of three parts: (1) Action Feature Extraction Module (FEM), which obtains local and global features through graph convolutional layers with different temporal kernel sizes, respectively, and sets up short connections to effectively retain local information for obtaining accurate action descriptions; (2) Optimization Module (OPM), which preserves the joint topology information constructed by Graph Convolutional Networks by predicting the adjacency matrix, avoids the problem that crucial joint topology information is prone to be lost when it is transferred between different networks; (3) Convolutional Neural Network, this paper combines the Convolutional Neural Network to model different joints between adjacent frames to achieve the enhancement of spatiotemporal dependence between joints. We validated the performance of the model on two mainstream datasets, NTU RGB+D, and NTU-120 RGB+D, compared to other models, our model obtains higher accuracy.

Keywords: Action Recognition · Skeleton-based · Graph Convolution Network · Spatiotemporal Dependency Modeling

1 Introduction

Skeletal data has received a lot of attention in recent years due to its robustness in complex contexts and its efficiency in storage and computation. Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs) have been widely studied in skeleton-based action recognition due to their advantages [8–13]. However, it is difficult

for these methods to take advantage of the graphical structure of the human skeleton. Yan et al. [1] were the first to propose a Graph Convolutional Network (GCN) that generalizes convolution from images to graphs. They build spatial maps based on the natural connections of human joints and create temporal edges between the same joints in adjacent frames. In recent years, GCN-based algorithms have greatly promoted the development of skeleton-based behavior recognition and have attracted more and more attention [14, 15, 26].

In most of the current GCN-based methods, the interframe edges are only connected to the same joints between adjacent frames, and the connection between different joints is severed. It hinders direct information exchange across spatiotemporal and makes it challenging to capture complex regional spatiotemporal dependencies, but complex spatiotemporal dependency modeling is the key to a powerful feature extractor.

These shortcomings limit the performance of existing models. To solve these problems, we designed SDE-Net, which uses the powerful spatiotemporal modeling capabilities of Convolutional Neural Networks to make up for the shortcomings of GCNs. The SDE-Net proposed in this paper consists of three parts. First, to obtain a stable action description, we designed the action feature extraction module (FEM), which used the graph convolutional layers of different sizes of convolutional kernels to extract local and global features respectively, and directly transported the local information to the end of the model through short connections, to realize the effective use of local features. Second, considering the different processing methods of GCN and CNN for input data, we design an optimization module that adjusts the joint coordinate tensor by predicting the adjacency matrix, it solves the problem that crucial joint topology information is easily lost when data is transmitted between different networks. Third, the spatiotemporal dependence on the articular area was strengthened by combining CNN. In conclusion, the contributions of this work are as follows:

1. In this paper, an action feature extraction module (FEM) is designed, which extracts local and global features through convolutional kernels of different sizes and realizes the effective use of local information through short connections.
2. We propose a new optimization module (OPM) that adjusts the joint coordinate tensor by predicting the joint adjacency matrix, which solves the problem that the exponential arrangement of human joints in the grid data leads to the loss of crucial topological information.
3. Combined with the Convolutional Neural Networks, the correlation of different joints between adjacent frames was constructed, the spatiotemporal dependence of the network was enhanced, and the direct communication of information between different joints across time and space was realized.

2 Related Work

2.1 Methods of Action Recognition

Action recognition can be broadly divided into three main methods: video-based action recognition, depth-based action recognition, and skeleton-based action recognition. Because videos are easy to collect, a large number of researchers have attempted to identify actions from videos [2–4]. The depth-based approach is also one of the more influential branches of the many [5–7]. Depth-based methods capture more detailed action than video-based methods and are more robust to noise and background in video. However, its huge computational load hinders the development of depth-based action recognition methods. Compared with video and depth maps, the computational cost of skeleton-based motion recognition is greatly reduced, and it is more robust to interference.

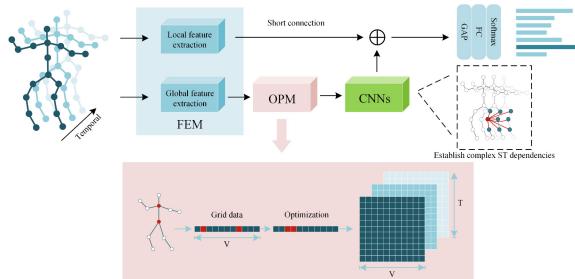


Fig. 1. SDE-Net architecture. The FEM consists of two parallel Graph Convolutional Networks, which are used to extract local and global features, respectively. The local information will be transmitted through short connections and fused with the output information of the CNNs in the GAP. In OPM, we adjust the position of the joints in the grid data and establish the grid data with a height of T and a width of V as input to the CNNs. Finally, the spatiotemporal dependence of the network is enhanced by the processing of data by CNNs (as shown in the dotted box).

2.2 Graph Convolutional Networks

Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have been widely used in skeletal action recognition tasks. The CNN-based research methods model the skeleton as a pseudo-image [8–10], using the powerful spatiotemporal modeling capabilities of CNNs for behavior recognition. RNN-based methods typically represent human joints in coordinate vectors [11–13]. However, there is a common problem in both CNN-based and RNN-based methods, which usually represent skeleton sequences in a grid-like form. This results in the loss of critical structural information about the skeleton. Therefore, Yan et al. [1] took the lead in proposing to introduce Graph Convolutional Networks (GCNs) into skeleton-based action recognition, and they presented the skeleton in the form of spatiotemporal graphs, which can well model the structural information of the skeleton. On this basis, the researchers have continuously improved the GCNs [14, 15, 26].

2.3 Enhanced Spatiotemporal Dependence

Recently, Yang et al. [15] proposed that there is a complementary relationship between GCNs and CNNs. GCNs can extract the structural information between human skeletons, but GCNs are sparsely connected in spatiotemporal blocks, which hinders the direct information exchange between joints across space-time. CNNs can explore the spatiotemporal correlation of joints. However, CNNs are prone to destroy the structural information between skeletons by representing skeleton data in the form of 2D grids for action recognition. Based on this theory, we use the optimization module to combine GCNs and CNNs.

3 SDE-Net

SDE-Net can establish dependencies between different joints of adjacent frames in the region while preserving the structural information of the skeleton. The proposed network inherits the advantages of GCNs and CNNs in applying skeleton-based action recognition tasks. Figure 1 illustrates the architecture of SDE-Net. In this section, we will take a closer look at the various components that make up SDE-Net.

3.1 Action Feature Extraction Module

Our action feature extraction module uses the adaptive graph convolutional layer in 2S-AGCN [16], and the spatial part of the graph convolution can be represented as follows:

$$Y = \sum_k W_k X (A_k + B_k + C_k) \quad (1)$$

k is denoted as different time frames; W_k weights for different time frames; X and Y are the input and output feature maps, respectively; A_k is an adjacency matrix customized by the human skeleton structure; B_k is a trainable adaptive matrix; C_k is a self-attention matrix. By performing temporal convolution of Y in Eq. (1), the temporal sampling region can be formulated as:

$$B(v_{t,i}) = \left\{ v_{q,i} \mid |q - t| \leq \left\lfloor \frac{\Gamma}{2} \right\rfloor \right\} \quad (2)$$

where Γ is a predefined parameter used to specify the size of the time window; q and t represent different timeframe numbers; $B(v_{t,i})$ is the time sampling range of the i -th joint point on the t -frame; $v_{q,i}$ is the i -th joint point on the q -frame.

The FEM is shown in Fig. 2. We use a parallel structure to extract local and global information separately. In the part of local feature extraction, we use 1×1 and 3×1 small convolution kernels, which can effectively extract local time information. And by using short connections, the local details extracted from the shallow layer are effectively used. In addition, considering that the computational cost of SDE-Net will increase dramatically, we only use four adaptive graph convolutional layers in the FEM to reduce the computational cost while ensuring the effect of the model.

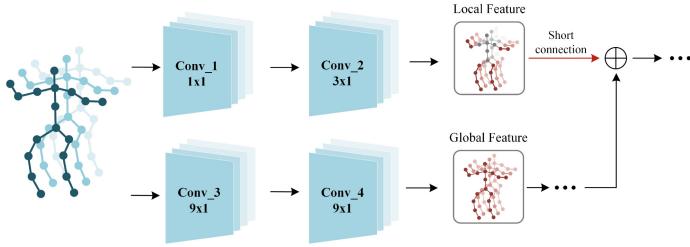


Fig. 2. FEM structure diagram. The FEM consists of four AGCN convolutional layers, the upper two AGCN convolutional layers use 1×1 and 3×1 temporal convolution kernels to obtain local features, and the lower two convolutional layers use 9×1 temporal convolutional kernels to obtain global features. To use the local time information effectively, we set up short connections, which are indicated by red arrows in the diagram. (Color figure online)

3.2 Optimization Module

The most straightforward way to create an SDE-Net is to use the output of the FEM directly as input to the CNNs, but this would destroy the graph topology built in the FEM. Because the sampling area in the CNNs is realized by the spatial order of the feature map determined by the joint indices, the index cannot accurately express the structural information of human joints. We first predict a fully connected matrix (Batch, N, N) from the output data of the FEM, where N represents the number of joint points. By multiplying the matrix with the joint coordinate tensor (Batch, V, 2), the structural information between the joint points can be reflected in the input feature map of the CNNs. The fully connected matrix we use here is predicted based on the features of all joints in the context. Therefore, even joints that do not have natural connections in the human skeleton, as long as there is a correlation between them in action, this correlation is captured and reflected in the input feature map of the CNNs. The final output of the optimization module is a feature map of size (N, C, T, V), where T and V represent the height and width of the feature map, respectively, and C represents the number of channels. The structure of the optimized module is shown in Fig. 3.

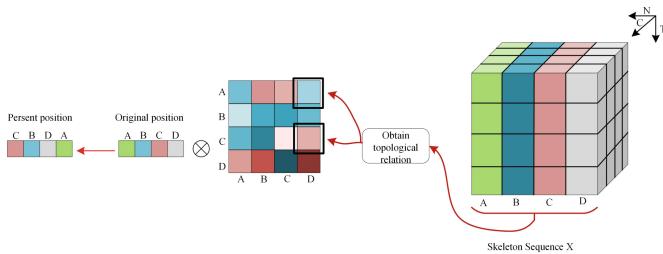


Fig. 3. Optimization module. The joint topology prediction method we use considers the characteristics of all joints in the context. The position of the joint points in the grid is optimized by multiplying the obtained fully connected matrices and joint coordinate tensors.

3.3 Convolutional Neural Networks

Through the graph convolutional layers, we can model the human skeleton's spatiotemporal dependence and obtain the structural information of human joints. Then, we need CNNs with strong spatiotemporal modeling capabilities, through which we can achieve complex modeling of spatiotemporal dependencies. Because of the small amount of skeleton data, we use only four stages of CNNs, each with two bottleneck layers. The number of input channels in the four stages is 128, 256, 512, and 1024, respectively.

The process of enhancing the spatiotemporal dependence between the joints is shown in Fig. 4. With the deepening of convolution in Convolutional Neural Networks, SDE-Net establishes more interframe edges, which realize the direct cross-temporal and spatial communication of joint point information. If given the input data of C channels, the Convolutional Neural Networks uses the normal 2D convolution operator of $k \times k$, then the sampling area of the SDE-Net vertex $v_{t,i}$ is:

$$B(v_{t,i}) = \left\{ v_{q,j} \mid |j - i| \leq \left\lfloor \frac{k}{2} \right\rfloor, |q - t| \leq \left\lfloor \frac{k}{2} \right\rfloor \right\} \quad (3)$$

where i and j represent different joint point numbers; $v_{q,j}$ represents the j-th joint point on the q-frame; The k in Eq. 3 and Eq. 4 represents a predefined parameter that specifies the size of the time window. In contrast to Eq. (2), the sampled area includes different joint points between adjacent frames. The output of SDE-Net can be formulated as:

$$Y^m(v_{t,i}) = \sum_{c=0}^C \sum_{j=-k/2}^{k/2} \sum_{q=-k/2}^{k/2} X(c, v_{t+q}, i+j) \times W(m, c, q, j) + b \quad (4)$$

In the formula, $W(m, c, q, j)$ is the weight of the 2D convolution, representing the weight at the position of the c-th convolution kernel (q, j) in the filter m. Each filter corresponds to a channel; Y and X represent the output and input feature maps, respectively; b is the convolution bias. In the formula, we can see that the output at the $v_{t,i}$ of the joint point is the weighted sum of all nodes in the spatiotemporal neighborhood, and the direct information exchange of the joint point across time and space shows the enhancement of the spatiotemporal dependence of the network.

4 Experiment

4.1 Datasets

NTU RGB+D. The NTU RGB+D dataset [17] contains 56,880 action clips, covering 60 different action classes, and each clip contains the 3D coordinates of 25 joint points to form a skeleton sequence. This dataset provides two assessment schemes, Cross Subject (CS) and Cross View (CV). CS: 40,320 action samples from 20 volunteers in the training set and 16,540 from 20 volunteers in the test set. This evaluation is used to validate the generalization ability of the model. CV: The training set contains 37,920 action samples from two cameras, and the test set contains 18,960 action samples from one camera. This evaluation method is used to verify the model's action recognition ability at different camera perspectives.

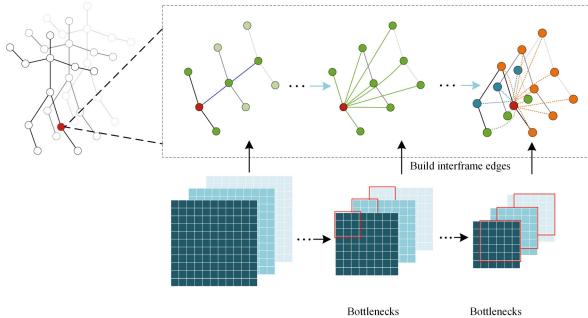


Fig. 4. Interframe edge generation in CNNs. The second and fourth convolution phases are shown in the figure. The dotted box shows that the joints in the left leg of the human body establish complex dependencies with the surrounding joints in time and space as the convolution deepens, and in order to clearly show them, we use green, light green, blue and orange to represent the joint points at different distances. (Color figure online)

NTU-120 RGB+D. The NTU-120 RGB+D dataset [18] contains 120 different action categories and 114,480 action samples. The NTU-120 RGB+D dataset provides two options for evaluating model performance, Cross Subject (CS) and Cross Setup (CE). CS: The training set and test set are samples from the training and test groups, with 53 subjects in each group. CE: The training set contains samples with even-numbered set IDs, while the test set contains samples with odd-numbered set IDs. These two evaluation protocols were designed to validate the generalizability and robustness of the model under different subjects and recording settings.

Table 1. Compared with the advanced methods on the NTU RGB + D dataset.

Method	CS(%)	CV(%)
ST-GCN [1]	81.5	88.0
ST-TR-GCN [20]	90.3	96.3
MS-G3D [21]	91.5	96.2
Dynamic GCN [25]	91.5	96.0
2S-AGCN [16]	88.3	95.1
MS-AAGCN [19]	90.0	96.2
AAM-GCN [22]	90.4	96.2
ICE-GCN [23]	92.0	96.2
SAR-GCN [24]	88.9	94.8
SDE-Net	91.7	96.8

Table 2. Compared with the advanced methods on the NTU-120 RGB + D dataset.

Method	CS(%)	CE(%)
Dynamic GCN [25]	87.3	88.6
MS-G3D [21]	86.9	88.4
ST-TR-GCN [20]	85.1	87.1
SparseShift-GCN [26]	86.6	88.1
SAR-GCN [24]	83.8	85.1
SDE-Net	87.5	89.1

4.2 Experimental Settings

In terms of choosing the loss function and optimization strategy, we use cross-entropy as the loss function and Stochastic Gradient Descent (SGD) with Nesterov momentum (0.9) as the optimization strategy. The batch size is set to 16, and the weight attenuation is set to 0.0001. The initial learning rate is set at 0.1 and will be multiplied by 0.1 at the 40th and 50th stages. The entire training process lasts for 65 epochs. For data preprocessing, we use the method proposed in [16]. For the NTU RGB + D dataset, we adjusted the number of skeleton frames in each sequence to 300.

4.3 Comparisons

In this section, to verify the recognition performance of the proposed SDE-Net, we conducted experiments on two public datasets, NTU RGB + D, and NTU-120 RGB + D, and compared them with other methods. We use the same multi-stream fusion strategy as [19], and the results presented in this section are the final result of multi-stream fusion.

The experimental results on the NTU RGB + D dataset are shown in Table 1. Under the CS evaluation criteria, except 0.3% lower than ICE-GCN, our method showed better performance than other methods, reaching 91.7%, which was 3.4% higher than the 2s-AGCN method alone. Under the CV evaluation criteria, our method outperformed all listed methods, reaching 96.8%, which was 0.6% higher than ICE-GCN and 1.7% higher than the 2s-AGCN method alone. The experimental results on the NTU-120 RGB+D dataset are shown in Table 2. Under the CS and CE evaluation criteria, our method outperformed all listed methods, reaching 87.4% and 89.1%, respectively.

4.4 Ablation Study

In this section, we verify the effectiveness of the various components of SDE-Net. All the results in this section were obtained under the Cross View (CV) criterion of the NTU RGB+D dataset. Only the original skeleton sequence is used as input to the model, and no other streams are built.

Table 3. Comparison of validation accuracy of different method models.

Method	Accuracy (%)
GCN baseline	93.7
CNN baseline	93.1
SDE-Net	95.4

Connect the GCN to the CNN

SDE-Net essentially connects the GCN and the CNN. To verify the validity of the connection, in this section, we compare the experimental accuracy of three models, namely the baseline of the FEM (AGCN [16]), the Convolutional Neural Networks baseline used in SDE-Net (ResNeSt [27]), and the SDE-Net we propose. The effects of the three models are shown in Table 3, and we can see that the addition of CNNs significantly improves the recognition effect of GCNs. Specifically, SDE-Net increased the effect of the FEM baseline from 93.7% to 95.4%. Experiments have demonstrated the effectiveness of connecting FEM and CNNs in SDE-Net.

Action Feature Extraction Module

In this section, we explain experimentally the reasons for using AGCN as the basic unit of graph convolution in the FEM and for using only four AGCN [16] convolutional layers. First, we study the impact of different graph convolution methods on our model. Specifically, we use CTR-GCN [28] and DGCN [25] instead of the basic graph convolution units used in the FEM. The results are shown in Table 4.

Table 4. Comparison of SDE-Net verification accuracy based on different graph convolution methods.

Method	Accuracy(%)
SDE-Net with CTR-GCN	95.1
SDE-Net with DGCN	95.1
SDE-Net with AGCN	95.4

To verify the influence of the number of convolution layers in the FEM on the complexity and effect of the model, we conducted four experiments and adjusted the number of convolutional layers to 4L, 6L, 8L, and 10L, respectively. The data from the experiment are presented in Fig. 5, and we also provide data for AGCN for comparison. In Fig. 5, we can see that as the number of convolutional layers increases, the overall computational complexity of SDE-Net also increases. However, the verification accuracy of the model does not increase accordingly. We believe that stacking too many convolutional layers may lose some key features, which can affect the performance of the model.

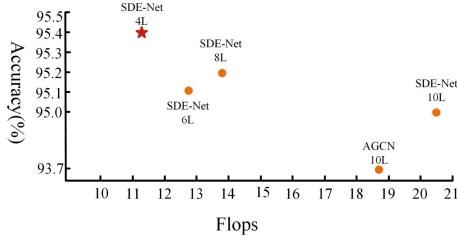


Fig. 5. Comparison of Flops and validation accuracy with different convolutional layers. The network with the best effect in the figure is represented by a red five-pointed star, where SDE-Net 4L indicates that four graph convolutional layers are used in the action feature extraction module, and the local feature extraction part is always two convolutional layers in the experiment.

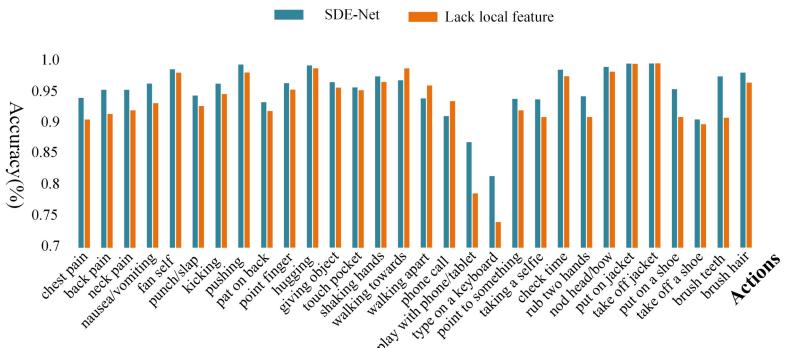


Fig. 6. Effect of loss of local features on the recognition of various types of actions, where the horizontal axis represents the actions and the vertical axis represents the recognition accuracy.

Local Features

In this section, to verify the importance of local features to SDE-Net, we randomly selected 30 action classes in the NTU RGB+D dataset and showed the experimental accuracy of the network on each action class after removing the local feature extraction part in the FEM in the form of a histogram, and showed the results of the original SDE-Net for easy comparison. As shown in Fig. 6, the recognition accuracy of 27 action categories decreased due to the loss of local features of SDE-Net.

Optimization Module

To verify the need for optimization modules in SDE-Net, we have set up a variant of SDE-Net in this section. The variant network is based on SDE-Net and removes the optimization module so that the action feature extraction module and the convolutional neural network are directly connected. The results are shown in Table 5, where we also show the performance of AGCN for comparison.

Table 5. Comparison of the accuracy of SDE-Net and its variant network validation.

Method	Accuracy (%)
AGCN	93.7
Lack OPM	95.0
SDE-Net	95.4

5 Conclusion

GCNs can use the human skeleton's structural information but lack flexibility in constructing cross spatiotemporal connections of joint points. CNNs can explore the spatiotemporal correlation of skeletons, but they cannot effectively use the structural information between skeletons. The SDE-Net inherits the advantages of GCNs and CNNs and has strong spatiotemporal dependence modeling capabilities, which provides new ideas for skeleton-based action recognition. We acquire local and global features separately in the FEM and set up short connections to ensure the effective use of local time information. In addition, an optimization module is designed to connect FEM and CNNs, which solves the problem of loss of joint structure information caused by data transmission between different networks. Experiments were conducted on two publicly available datasets, and compared with other advanced methods, SDE-Net achieved higher recognition accuracy than other models.

References

1. Yan, S., Xiong, Y., Lin, D.: Spatial temporal graph convolutional networks for skeleton-based action recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 7444–7452 (2018)
2. Xu, Q., Zheng, W., Song, Y., et al.: Scene image and human skeleton-based dual-stream human action recognition. Pattern Recogn. Lett. **148**, 136–145 (2021)
3. Wang, L., Xiong, Y., Wang, Z., et al.: Temporal segment networks for action recognition in videos. IEEE Trans. Pattern Anal. Mach. Intell. **41**(11), 2740–2755 (2018)
4. Yang, H., Yuan, C., Li, B., et al.: Asymmetric 3D convolutional neural networks for action recognition. Pattern Recogn. **85**, 1–12 (2019)
5. Ji, X., Cheng, J., Tao, D., et al.: The spatial Laplacian and temporal energy pyramid representation for human action recognition using depth sequences. Knowl.-Based Syst. **122**, 64–74 (2017)

6. Xiao, Y., Chen, J., Wang, Y., et al.: Action recognition for depth video using multi-view dynamic images. *Inf. Sci.* **480**, 287–304 (2019)
7. Ren, Z., Zhang, Q., Cheng, J., et al.: Segment spatial-temporal representation and cooperative learning of convolution neural networks for multimodal-based action recognition. *Neurocomputing* **433**, 142–153 (2021)
8. Xu, Y., Cheng, J., Wang, L., et al.: Ensemble one-dimensional convolution neural networks for skeleton-based action recognition. *IEEE Signal Process. Lett.* **25**(7), 1044–1048 (2018)
9. Li, B., He, M., Dai, Y., et al.: 3D skeleton based action recognition by video-domain translation-scale invariant mapping and multi-scale dilated CNN. *Multimedia Tools Appl.* **77**, 22901–22921 (2018)
10. Ke, Q., Bennamoun, M., An, S., et al.: Learning clip representations for skeleton-based 3D action recognition. *IEEE Trans. Image Process.* **27**(6), 2842–2855 (2018)
11. Liu, J., Shahroudy, A., Xu, D., Wang, G.: Spatio-temporal lstm with trust gates for 3d human action recognition. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) *ECCV 2016*. LNCS, vol. 9907, pp. 816–833. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46487-9_50
12. Zhang, P., Lan, C., Xing, J., et al.: View adaptive neural networks for high performance skeleton-based human action recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **41**(8), 1963–1978 (2019)
13. Liu, J., Wang, G., Duan, L.Y., et al.: Skeleton-based human action recognition with global context-aware attention LSTM networks. *IEEE Trans. Image Process.* **27**(4), 1586–1599 (2017)
14. Song, Y.F., Zhang, Z., Shan, C., et al.: Constructing stronger and faster baselines for skeleton-based action recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**(2), 1474–1488 (2022)
15. Yang, W., Zhang, J., Cai, J., et al.: HybridNet: integrating GCN and CNN for skeleton-based action recognition. *Appl. Intell.* **53**(1), 574–585 (2023)
16. Shi, L., Zhang, Y., Cheng, J., et al.: Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12026–12035 (2019)
17. Shahroudy, A., Liu, J., Ng, T.T., et al.: NTU RGB+ D: a large scale dataset for 3d human activity analysis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1010–1019 (2016)
18. Liu, J., Shahroudy, A., Perez, M., et al.: NTU RGB+ d 120: a large-scale benchmark for 3D human activity understanding. *IEEE Trans. Pattern Anal. Mach. Intell.* **42**(10), 2684–2701 (2019)
19. Shi, L., Zhang, Y., Cheng, J., et al.: Skeleton-based action recognition with multi-stream adaptive graph convolutional networks. *IEEE Trans. Image Process.* **29**, 9532–9545 (2020)
20. Plizzari, C., Cannici, M., Matteucci, M.: Skeleton-based action recognition via spatial and temporal transformer networks. *Comput. Vis. Image Underst.* **208**, 103219 (2021)
21. Liu, Z., Zhang, H., Chen, Z., et al.: Disentangling and unifying graph convolutions for skeleton-based action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 143–152
22. Xie, J., Miao, Q., Liu, R., et al.: Attention adjacency matrix based graph convolutional networks for skeleton-based action recognition. *Neurocomputing* **440**, 230–239 (2021)
23. Wang, S., Pan, J., Huang, B., et al.: ICE-GCN: An interactional channel excitation-enhanced graph convolutional network for skeleton-based action recognition. *Mach. Vis. Appl.* **34**(3), 40 (2023)
24. Zhu, Q., Deng, H.: Spatial adaptive graph convolutional network for skeleton-based action recognition. *Appl. Intell.* **53**(14), 17796–17808 (2023)

25. Ye, F., Pu, S., Zhong, Q., et al.: Dynamic GCN: context-enriched topology learning for skeleton-based action recognition. In: Proceedings of the 28th ACM International Conference on Multimedia, pp. 55–63 (2020)
26. Zang, Y., Yang, D., Liu, T., et al.: SparseShift-GCN: High precision skeleton-based action recognition. *Pattern Recogn. Lett.* **153**, 136–143 (2022)
27. Zhang, H., Wu, C., Zhang, Z., et al.: ResNest: split-attention networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2736–2746 (2022)
28. Chen, Y., Zhang, Z., Yuan, C., et al.: Channel-wise topology refinement graph convolution for skeleton-based action recognition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 13359–13368 (2021)



Lightweight Coal Flow Foreign Object Detection Algorithm

Ru Nie¹ , Xiaobing Shen^{1,2} , Zhengwei Li^{1,2,3()} , Yanxia Jiang⁴ , Hongmei Liao¹⁽⁾ , and Zhuhong You⁵⁽⁾

¹ School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221116, Jiangsu, China

{nr, zwli, lhm}@cumt.edu.cn

² Big Data and Intelligent Computing Research Center, Guangxi Academy of Science, Nanning 530007, Guangxi, China

³ School of Information Science and Engineering, Zaozhuang University, Zaozhuang 277160, Shandong, China

⁴ Information Construction and Management Division, China University of Mining and Technology, Xuzhou 221116, Jiangsu, China

yxjiang@cumt.edu.cn

⁵ School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, Shanxi, China

zhuhongyou@nwpu.edu.cn

Abstract. In response to the challenges of foreign object detection and high real-time requirements in complex coal mine monitoring scenarios, a lightweight coal flow foreign object detection algorithm, FAI_YOLO, was developed based on YOLOv8s, which incorporates several innovations to address these challenges. Initially, Fastnet is employed as the backbone feature extraction network to minimize redundant computation and memory access, thereby accelerating inference speed. Additionally, AKConv replaces the traditional convolution operation in C2f module, and the loss function ImpIOU is refined to enhance regression performance. Experimental results suggest that this algorithm markedly improves the speed of foreign object detection in coal flow compared to the original YOLOv8s model, decreasing frame reasoning time by 33.8%. While the map@0.5 metric experiences a minor decrease of 0.03%, the algorithm continues to provide high detection accuracy and effectively manages the demands of real-time and accurate foreign object detection in complex coal mine monitoring scenarios.

Keywords: Object detection · YOLO · Lightweight · Coal flow

1 Introduction

The reliance of China on safe and efficient coal transportation is critical for its economic stability, given its status as a major coal consumer and producer. Belt conveyors serve as the core equipment in coal transportation and are instrumental in enhancing the productivity of coal mines. The presence of gangue and other extraneous materials in the coal

mix can damage the conveyor belt, leading to production interruptions and economic losses. The complexity of the underground environment, coupled with poor lighting, complicates the manual detection of foreign objects. To address this challenge, technology for object detection has been implemented. This technology, which facilitates the identification and classification of objects within images, is a pivotal component in the realm of computer vision. With the progression of deep learning, significant enhancements have been achieved, particularly in terms of recognition accuracy and computational speed. Historically, early algorithms, such as Haar [1], SIFT [2], and HOG [3], which relied on manually extracted features, garnered some success. However, their performance in complex scenarios was restricted. The adoption of deep learning, particularly through the use of convolutional neural networks (CNN) [4], has substantially advanced the capability to extract features and classify objects effectively.

To address the challenges encountered by deep learning object detection algorithms in specific application scenarios, characterized by high memory demands and computational costs, a lightweight and efficient algorithm has been developed for detecting foreign objects in coal flow. This manuscript discusses the adoption of methods such as FasterNet [5] and optimized C2f modules to improve the model's capacity for feature extraction, diminish computational and parameter loads, while ensuring real-time detection efficiency.

2 Principle of YOLOv8 Model

The most recent addition to the YOLO series, YOLOv8, features several variants such as YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x, each designed to meet the needs of different application scenarios. These variants share a common structure but vary in network depth and width. Due to its efficient design, which requires fewer parameters and computational resources while still maintaining sufficient accuracy for detecting foreign objects, YOLOv8s was selected as the base model for the study.

The architecture of YOLOv8s includes three primary components: the backbone network, the neck network, and the output layer. The backbone network is tasked with extracting features from the input image, specifically isolating three effective feature layers for further processing. The neck network enhances these features through a multi-scale fusion process, utilizing the PANet [6] structure, which combines the advantages of both the Feature Pyramid Network (FPN) [7] and the Path Aggregation Network (PAN) [8]. While FPN focuses on extracting and fusing feature maps at various depths to enhance information flow via a top-down approach, PAN improves spatial feature preservation through a bottom-up path.

The output layer is designed to perform classification and regression tasks through a decoupled methodology, executed via independent convolution operations. Furthermore, the integration of the SPPF module in YOLOv8s allows for the capture of multi-scale information using a combination of 5×5 convolutions and max-pooling techniques, thereby boosting the model's capability to manage multi-scale features. The fusion of the features processed by the SPPF module with the original features enables the model to effectively utilize comprehensive multi-scale information, leading to improved performance.

3 Our Proposed FAI_YOLO Model

In response to the challenges associated with inadequate detection precision and limited real-time capabilities, especially in recognizing concealed objects or small targets, this study introduces enhancements to the YOLOv8s algorithm. A novel object detection algorithm, named FAI_YOLO, is proposed. The modified network structure is depicted in Fig. 1. Initially, a more efficient network, termed FasterNet, serves as the foundational architecture for feature extraction in YOLOv8s. Subsequently, the C2f module in the neck network is reengineered by substituting the standard convolution in the bottleneck with AKConv [9]. Additionally, a new loss function, referred to as ImpIOU, is integrated to refine the training process, strengthen the model's convergence capabilities, and improve the accuracy of bounding box prediction regression.

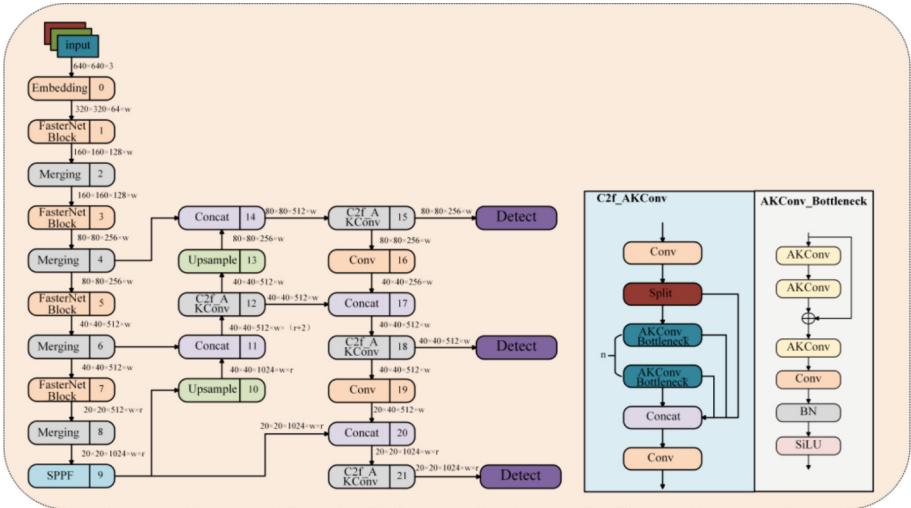


Fig. 1. Structure of FAI_YOLO.

3.1 FasterNet Makes the Network More Lightweight

The present study reconstitutes the backbone feature extraction network of YOLOv8s utilizing FasterNet. FasterNet, an innovative and lightweight backbone network structure, facilitates quicker detection speeds across various devices compared to other prevalent lightweight networks while maintaining accuracy. The fundamental configuration of FasterNet is depicted in Fig. 2. At the core of FasterNet lies the PConv, designed to minimize computational redundancy and enhance memory access efficiency simultaneously. The operational mechanism of PConv is illustrated in Fig. 3.

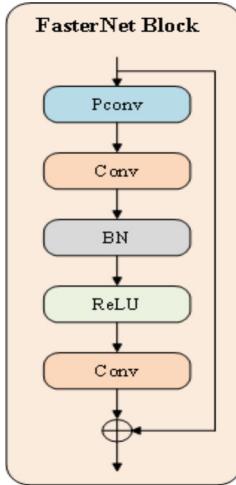


Fig. 2. Structure of FasterNet.

PConv implements convolution selectively on specific input channels that are crucial for the extraction of spatial features, while leaving other channels unaltered. For efficient memory utilization, PConv selects either the initial or final consecutive channels as representatives of the complete feature map during the computation process. Provided that the number of channels in both input and output feature maps is equivalent, the formula for computing the floating-point operations per second (FLOPs) of PConv is established as follows:

$$h \times w \times k^2 \times c_p^2 \quad (1)$$

In terms of practical implementation, where h and w signify the width and height of the feature map, k denotes the convolutional kernel size, and c_p represents the number of channels influenced by the standard convolution. Commonly, the ratio $r = c_p/c$ is established at 1/4, which results in the computational complexity of PConv being merely 1/16th that of traditional convolution. The memory access pattern of PConv is characterized as follows:

$$h \times w \times 2c_p + k^2 \times c_p^2 \approx h \times w \times 2c_p \quad (2)$$

The use of the partial convolution (PConv) operator in the backbone feature extraction network reduces the memory access count to approximately one quarter that of standard convolution, as illustrated by Eq. (2). In this setup, channels $c - c_p$ remain uninvolved in the computation, eliminating the need for memory access to these channels.

Consequently, the integration of the PConv operator into the backbone network substantially decreases computational complexity and memory demands, rendering the network more efficient and enhancing the speed of model inference.

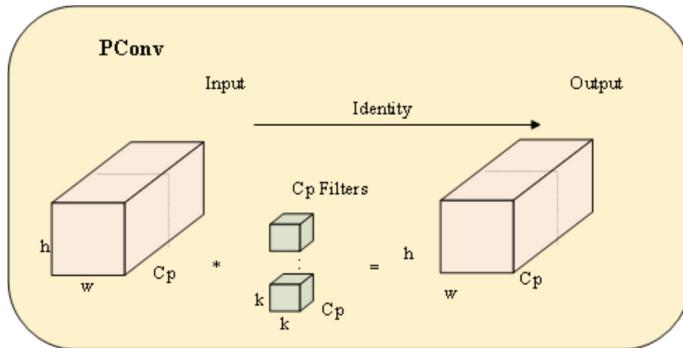


Fig. 3. The working principle of PConv.

3.2 Refactoring C2f Using AKConv

Standard convolution operations exhibit several notable limitations. Primarily, these operations occur within fixed-size windows, incapable of assimilating information from adjacent windows. The fixed window shape further restricts the network's capacity to extract features of varying dimensions and configurations. Additionally, as the size of the convolutional kernels is constant, an increase in kernel dimensions results in a sharp rise in the number of parameters. This escalation complicates the model, adding to its size, elevating computational expenditure, and intensifying training challenges. In response, this study reconstructs the original C2f module using adaptive kernel convolution (AKConv), replacing conventional convolutions with AKConv. Figure 4 depicts the architecture of the newly designed AKConv structure.

AKConv enables convolutional kernels to accommodate varying numbers and configurations of samples. This method utilizes a novel coordinate generation algorithm to establish initial positions for convolutional kernels of diverse sizes and incorporates offsets to modify the configuration of each sample at different locations. These irregular convolution operations facilitate efficient feature extraction.

Upon defining a conventional sampling grid for a particular size of a convolutional kernel, AKConv diverges from standard convolution by generating sampling grids of varied dimensions. The initial sampling positions, denoted as $Q_m = \{(x_i, y_i)\}$, signify the array of sampling points for the convolutional kernel over the input feature map. Here, $\{(x_i, y_i)\}$ marks the primary sampling sites on the input feature map. By applying learned biases, AKConv alters these sampling points, enabling the convolutional kernel to dynamically reshape in response to specific attributes of the input data. Each bias offset corresponds to a bias value at each point in Q_m . The modified sampling position, post-offset addition, is indicated as:

$$Q_{m'} = Q_m + O \quad (3)$$

where O represents the learned offset for each sampling point through the network. Subsequent to the adjustment of sampling positions via biases, AKConv extracts features at these revised sites. This process is facilitated through methods such as bilinear

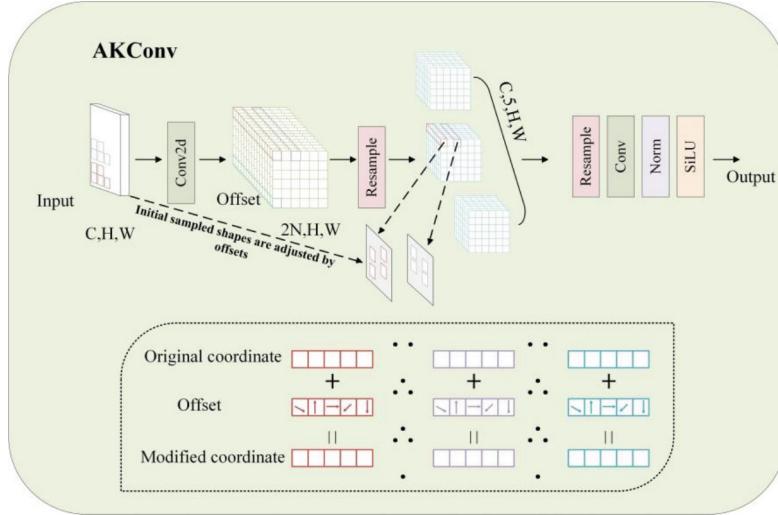


Fig. 4. The structure of the AKConv.

interpolation, which ensures accurate feature value extraction from non-integer sampling positions.

This dynamic adjustment mechanism allows AKConv to adapt the convolutional kernel more flexibly to variations in image content, thereby enhancing the precision of feature extraction.

3.3 Loss Function Improvement

The original YOLOv8s employed the CIOU Loss function, which is widely recognized in object detection for its capability to assess model accuracy concerning target position and size. This function is instrumental in addressing overlaps and misalignments, thereby enhancing performance evaluations. Despite its utility, the CIOU Loss function exhibits limitations in accurately detecting small objects across various scales and environmental conditions. Its efficacy diminishes when applied to objects with large aspect ratios or irregular shapes, and it does not offer adaptive adjustments for diverse tasks and targets. In response, this study introduces a novel loss function named ImpIOU. This function integrates the Inner-IOU [10] loss function, which utilizes auxiliary bounding boxes, with the MPDIoU [11] that employs a similarity comparison measure based on the minimum point distance of bounding boxes. The integration of these functions facilitates the calculation of IOU loss across auxiliary bounding boxes of different scales, enhancing the regression of samples across varying IOU levels and promoting faster convergence. For example, using smaller-scale auxiliary bounding boxes accelerates convergence for samples with high IOU values, whereas large-scale boxes improve the regression process for samples with low IOU values. To optimize this approach, ImpIOU incorporates a scale factor ratio that regulates the dimensions of auxiliary bounding boxes used in loss computation. This adaptation enables more rapid and effective regression outcomes. The

computation method for Inner-IoU is elaborated in Eqs. (4)–(8), wherein the scale factor r is adjusted to modify the size of the auxiliary bounding boxes.

$$b_l^{gt} = x_c^{gt} - \frac{w^{gt}*r}{2}, b_r^{gt} = x_c^{gt} + \frac{w^{gt}*r}{2} \quad (4)$$

$$b_t^{gt} = y_c^{gt} - \frac{h^{gt}*r}{2}, b_b^{gt} = y_c^{gt} + \frac{h^{gt}*r}{2} \quad (5)$$

Through Eqs. (4)–(6), the central point of the detection box is transformed to derive the corner vertices of the auxiliary detection box. Subsequently, the predicted and ground truth boxes are adjusted, denoted by b_l^{gt} and b_t^{gt} respectively, representing the calculation results for the predicted and actual boxes.

$$\text{inter} = (\min(b_r^{gt}, b_r) - \max(b_l^{gt}, b_l)) * (\min(b_b^{gt}, b_b) - \max(b_t^{gt}, b_t)) \quad (6)$$

$$\text{union} = (w^{gt} * h^{gt}) * (r)^2 + (w * h) * (r)^2 - \text{inter} \quad (7)$$

$$\text{IoU}^{\text{inner}} = \frac{\text{inter}}{\text{union}} \quad (8)$$

The Inner-IoU computes the Intersection over Union (IoU) between auxiliary bounding boxes. When the ratio r , within the range [0.5, 1.5], is less than 1, the auxiliary bounding box is smaller than the actual bounding box, resulting in a reduced effective regression range compared to IoU loss. Nonetheless, the absolute gradient value is larger than that obtained by IoU loss, which can accelerate the convergence of samples with high IoU values. Conversely, if r exceeds 1, the auxiliary bounding box is larger than the actual box, extending the effective regression range, which is beneficial for regression tasks with low IoU values.

4 Experiment and Result Analysis

4.1 Experimental Environment and Dataset

Table 1. Configuration of experimental environment.

Category	Version Number
Operating System	Ubuntu 20.04
CPU	16 vCPU Intel(R) Platinum
GPU	RTX 4090
Pytorch	Torch-1.13.1
Programming Language	Python 3.8

The experimental setup described in this paper is presented in Table 1. The experimental data in this study utilized the Coal Mine Special Video AI Analysis Dataset from the Intelligent Detection and Pattern Recognition Research Center at China University of Mining and Technology. The experiment employed transfer learning techniques. During the training phase, uniform parameters were employed: an SGD optimizer was utilized, with an initial learning rate of $1e-2$, a weight decay coefficient of $5e-4$, a momentum parameter of 0.937, a batch size set to 16, and training duration extended over 300 epochs.

4.2 Experimental Results and Analysis

To assess the efficiency and advancement of the algorithm for detecting foreign objects in coal flow monitoring, three experimental sets were designed. Initially, various mainstream lightweight feature extraction networks from recent years were utilized to substitute the backbone component of the original YOLOv8s model. Subsequently, ablation studies were conducted based on the enhanced model to determine the efficacy of each module. Finally, different object detection models were evaluated and compared with the enhanced YOLOv8s model to establish the model's efficiency.

Backbone Network Comparison Experiment. The results of the comparative experiments on the backbone networks within the test set are illustrated in Table 2.

Table 2. Backbone network comparison experiment results

Backbone	P/%	R/%	map@0.5	Params /106	GFLOPs/G	Infer time/ms
Original	88.5	89.9	95.5	11.1	28.4	6.5
GhostHGNetV2	85.7	91.3	95.1	8.31	23.0	4.8
ShuffleNetV2	89.2	88.5	94.7	8.55	20.7	6.1
EfficientViT	91.9	90.3	95.1	8.38	20.4	6.6
FasterNet	87.7	88.6	94.4	8.61	21.7	4.3

The study systematically evaluated lightweight backbone networks in object detection to balance detection accuracy and real-time processing. It compared the original model and four lightweight networks—GhostHGNetV2 [12], ShuffleNetV2 [13], EfficientViT [14], and FasterNet—across precision, recall, mAP@0.5, parameter count, complexity, and inference time.

The original model achieved a significant mAP@0.5 of 95.5% but had a 6.5 ms inference time. GhostHGNetV2 slightly reduced mAP@0.5 to 95.1% with a 4.8 ms inference time. ShuffleNetV2 decreased mAP@0.5 to 94.7% with a 6.1ms inference time. EfficientViT matched the original model's mAP@0.5 at 95.1% but with a slight inference time increase.

FasterNet notably improved speed (94.4% mAP@0.5, 4.3 ms inference time), with marginal parameter and complexity increases, demonstrating efficiency in resource utilization. It was chosen as the backbone network for the enhanced model due to its considerable speed, minimal performance degradation, and reduced resource consumption, crucial for real-time processing, particularly in detecting foreign objects.

Ablation Experiments of Different Improvement Algorithm. To validate the improvement effect of the proposed algorithm on the model, ablation experiments were designed. Based on the YOLOv8s network, the FasterNet feature extraction network was replaced, the C2f module was modified, and the ImpIOU loss function was changed. Each improvement module was incrementally added. The experimental results are shown in Table 3.

Table 3. Results of ablation experiments

Experiment	Fasternet	C2f_AKConv	ImpIOU	P/%	R/%	map@0.5	Params/106	GFLOPs/G
1	×	×	×	88.5	89.9	95.5	11.1	28.4
2	✓	×	×	87.7	88.6	94.4	8.61	21.7
3	×	✓	×	89.9	89.4	95.8	9.97	26.3
4	×	×	✓	88.8	88.1	95.6	11.1	28.4
5	✓	✓	×	90.2	89.3	94.9	7.45	19
6	✓	✓	✓	88.9	88.6	95.2	7.45	19.4

The ablation studies investigated the impact of FasterNet, C2f_AKConv, and ImpIOU on object detection. Initially, without enhancement modules (Experiment 1), the model achieved 88.5% precision, 89.9% recall, and 95.5% mAP@0.5, with 11.1M parameters and 28.4 GFLOPs complexity. Integrating FasterNet (Experiment 2) slightly reduced precision and recall, and mAP@0.5 by 1.1% points, but improved computational efficiency. Using C2f_AKConv alone (Experiment 3) increased precision by 1.4% points and mAP@0.5 by 0.3% points, with a minor complexity rise. ImpIOU implementation (Experiment 4) slightly raised mAP@0.5 without increasing model burden. Combining all modules (Experiments 5 and 6) improved precision to 90.2% and mAP@0.5 to 95.2% in Experiment 6, with reduced parameters and complexity, demonstrating significant performance and efficiency enhancements. The results highlight the individual effectiveness of the modules and the necessity of their combined use for efficient and lightweight object detection, particularly valuable in resource-constrained environments.

Comparison of Different Models. To further validate the effectiveness of the algorithm, it was compared with other mainstream object detection algorithms. The experimental results are presented in Table 4.

Table 4. Results of comparative experiments

Model	P/%	R/%	map@0.5	Params/ 106	GFLOPs/G	Infer time /ms
RT-DETR	86.0	79.3	88.1	28.4	100.6	9.4
YOLOv5s	90.6	88.5	94.7	9.11	23.8	5.3
SSD	82.8	83.4	88.9	—	—	32.5
Faster-RCNN	85.4	84.9	89.7	—	—	41.2
YOLOv8s	88.5	89.9	95.5	11.1	28.4	6.5
FAI_YOLO	88.9	88.6	95.2	7.45	19.4	4.3

Through comparative experiments, the research model was evaluated against mainstream object detection models such as RT-DETR, YOLOv5s, SSD [16], Faster-RCNN [15], and YOLOv8s, the research model under discussion exhibited notable competitiveness in key metrics, including map@0.5, inference time, and resource efficiency. Relative to RT-DETR, significant enhancements were observed in inference time, with a reduction from 9.4 ms to 4.3 ms, along with improved precision. While this model presented a slight decrement in precision and recall compared to YOLOv5s, it outperformed in map@0.5 and recorded a faster inference time, reduced from 5.3 ms to 4.3 ms. In contrast to SSD and Faster-RCNN, the model led substantially in inference time while sustaining high accuracy levels. Against YOLOv8s, the model matched closely in terms of precision and recall but achieved notable reductions in parameter count, computational complexity, and inference time. The study concludes that the model maintains similar or enhanced detection performance while achieving substantial efficiency gains, making it particularly suitable for real-time applications and demonstrating a broad potential for practical deployment.

5 Conclusion

This study effectively addresses the challenges of foreign object detection in complex coal mine monitoring scenarios, particularly in improving the accuracy and real-time performance of detecting small-sized foreign objects. The FAI YOLO algorithm is proposed, incorporating a series of improvement measures such as the use of the lightweight backbone network FasterNet, the introduction of AKConv to optimize feature extraction capabilities, and the optimization of the ImpIOU loss function. This model improves inference speed while ensuring high detection accuracy, achieving an effective balance between resource consumption and detection performance.

Experimental results indicate that, compared to the original YOLOv8 model, FAI_YOLO experiences only a slight decrease of 0.03% in mAP@0.5, while the inference speed increases by 33.8%. These results demonstrate the model's strong applicability and superior performance in detecting foreign objects in complex scenes. Additionally, the proposed method exhibits significant advantages in terms of parameter count and computational complexity, rendering it more suitable for resource-constrained real-time monitoring systems.

In conclusion, the research not only provides an efficient solution for foreign object detection in complex coal mine monitoring scenarios but also offers valuable insights into the lightweighting and performance optimization of deep learning models in specific application scenarios. Future efforts will focus on further lightweighting exploration.

Acknowledgments. This study was supported by the Natural Science Foundation of Guangxi Zhuang Autonomous Region under Grant 2023JJA170019, and in part by the Natural Science Foundation of Shandong Province under Grant ZR2022LZL003. We express our sincere gratitude to the anonymous reviewers for their invaluable feedback and recommendations on this research.

Disclosure of Interests. The authors have declared that no competing interests exist.

References

1. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2001)
2. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision* **60**, 91–110 (2004)
3. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 1, pp. 886–893. IEEE (2005)
4. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. *Commun. ACM* **60**(6), 84–90 (2017)
5. Chen, J., et al.: Run, Don't walk: Chasing higher FLOPS for faster neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12021–12031 (2023)
6. Liu, S., Qi, L., Qin, H., Shi, J., Jia, J.: Path aggregation network for instance segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 8759–8768 (2018)
7. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2117–2125 (2017)
8. Wang, W., et al.: Efficient and accurate arbitrary-shaped text detection with pixel aggregation network. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 8440–8449 (2019)
9. Zhang, Xin, et al.: AKConv: convolutional kernel with arbitrary sampled shapes and arbitrary number of parameters. arXiv preprint arXiv:2311.11587, 2–10 (2023)
10. Zhang, H., Xu, C., Zhang, S.: Inner-IOU: more effective intersection over union loss with auxiliary bounding box. arXiv preprint arXiv:2311.02877, 1–7 (2023)
11. Siliang, M., Yong, X.: Mpdiou: a loss for efficient and accurate bounding box regression. arXiv preprint arXiv:2307.07662, 1–13 (2023)
12. Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C., Xu, C.: GhostNet: more features from cheap operations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1580–1589 (2020)
13. Ma, N., Zhang, X., Zheng, H.T., Sun, J.: ShuffleNet v2: practical guidelines for efficient CNN architecture design. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 116–131 (2018)

14. Liu, X., Peng, H., Zheng, N., Yang, Y., Hu, H., Yuan, Y.: Efficientvit: memory efficient vision transformer with cascaded group attention. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14420–14430 (2023)
15. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **39**(6), 1137–1149 (2016)
16. Liu, Wei, et al.: SSD: single shot multibox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) *ECCV 2016. LNCS*, vol. 9905, pp. 21–37. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_2



SDF-Net: Enhanced Novel View Synthesis from Ultra-sparse Viewpoints via Multi-level Feature Fusion

Qijun He , Jingfu Yan , Jiahui Li , and Yifeng Li

College of Computer and Information Engineering, Nanjing Tech University, Nanjing 211816, Jiangsu, China
lyffz2616@163.com

Abstract. The primary aim of this study is to enhance the quality of synthetic novel views under conditions characterized by data sparsity. Our investigations demonstrate that the utilization of multi-scale, multi-level stereoscopic depth feature extraction within the cost volume substantially improves the network's comprehension of scene depth and spatial positioning. Additionally, the introduction of a depth feature beam attention mechanism alleviates the effects of occlusions, thereby enhancing spatial consistency. Specifically, the cost volume is generated through the homography transformation of the feature map, which facilitates deep feature fusion at varying levels and across different spaces using the novel decoder-encoder architecture of the Stereoscopic Deep Fusion Network (SDF-Net). Recognizing the prevalent issue of spatial point occlusion in each beam, we implement an attention mechanism designed to suppress the characteristics of internally occluded spatial points while accentuating those at the extremities of the beam, thus optimizing the effectiveness of spatial features and minimizing occlusion related disturbances during rendering. Extensive experiments show that our approach exhibits state-of-the-art performance compared to previous excellent work on synthesizing novel views when tested on the most popular real scene datasets and synthetic scene datasets, and our results show Richer details and a more complete outline structure.

Keywords: Cost Volume · Attention · NeRF · Sparse Views

1 Introduction

Improving the quality of novel view synthesis under sparse conditions is crucial for several application areas. In the fields of Virtual Reality (VR) and Augmented Reality (AR), high-quality novel view synthesis can create more realistic and immersive virtual environments, enhancing user experiences.

When faced with rich viewing angles, simple up- and down-sampling of the cost volume can obtain rich local depth information and achieve accurate density prediction. In the case of sparse viewing angles, the local depth features of the scene are

Q. He and J. Yan—These authors contributed equally to this work.

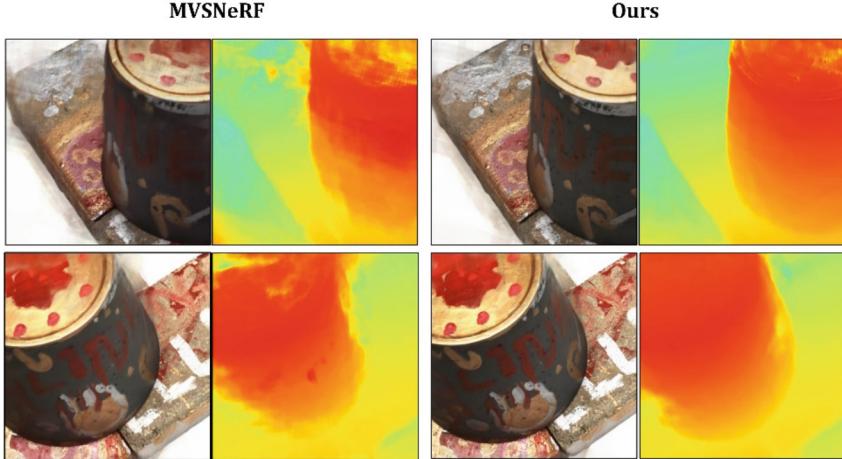


Fig. 1. Depth heat map.

often sparse and difficult to obtain. Different from previous work, we use the Stereo Depth Fusion Network (SDF-Net) to perform multi-level local deep feature fusion on the cost volume to achieve the network's understanding of the spatial position of the entire three-dimensional scene. Specifically, in the stereo depth fusion network, we add point-wise convolution during the entire upsampling process to fuse depth features. At the same time, we choose linear interpolation upsampling instead of traditional 3D transposed convolution upsampling because the traditional 3D transposed convolution method reconstructs spatial features during the upsampling process. However, this may lead to undesirable effects such as reduced spatial resolution, discontinuities, and the introduction of feature distortion or noise, so we choose linear interpolation for depth smooth transition and avoid image slices, as illustrated in Fig. 1. In addition, in order to enrich the description of scene details, we introduce residual connections in the feature downsampling process to enhance the representation of high-frequency information and achieve the effect of sharpening high-frequency information. Due to the occlusion of the beam, this directly affects the consistency and effectiveness of the features and ultimately affects the volume rendering results. Therefore, we add a multi-head attention mechanism to regularize the stereoscopic depth features and reduce the interference caused by blocking some spatial point features.

After extensive experimental studies, our method surpasses previous methods on both the most popular real scene datasets and synthetic scene datasets. To conclude, our primary contribution is:

- 1 Our framework adeptly captures and amalgamates multi-scale, abstracted stereoscopic depth and color features within the cost volume via a cohesive encoder-decoder architecture. This integration empowers the model to simultaneously address local details and global contextual nuances, significantly augmenting the precision of density estimation and the fidelity of the reconstruction process.

- 2 We unveil a novel depth feature beam attention mechanism that preserves the spatial integrity of feature rendering, effectively minimizing artifacts and mitigating color deviations.
- 3 Our approach enables the synthesis of markedly clearer novel views from sparse input data, substantially diminishing the incidence of artifacts and geometric distortions.

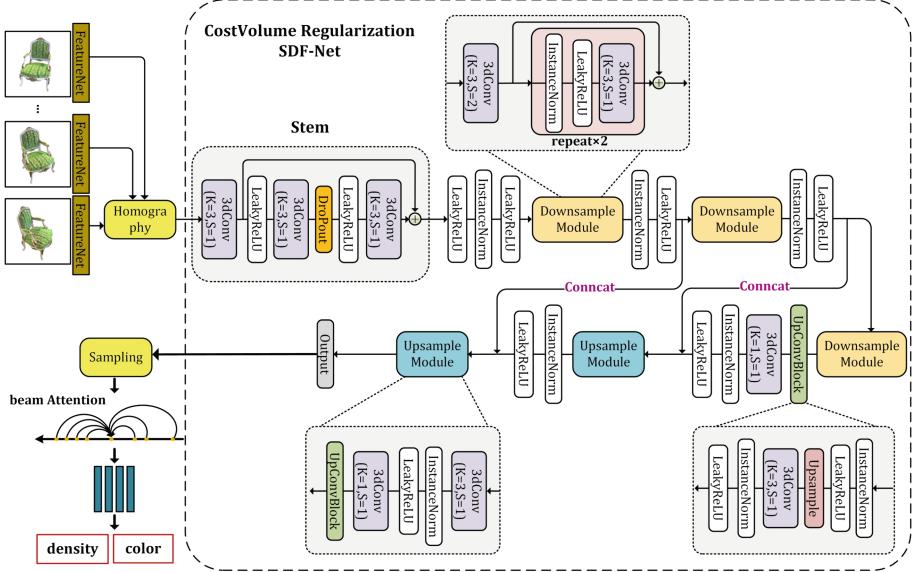


Fig. 2. Overview of our method.

2 Related Work

Prior to our methodology, a plethora of strategies for the sparse reconstruction of new views had been established, including RegNeRF [1], DS-NeRF [2], ViewFormer [3], and the innovatively introduced DiffusioNeRF [4]. These methodologies offer robust solutions for sparse view reconstruction challenges. For instance, DiffusioNeRF [4] elevates the rendering quality and detail recapture by mimicking the light diffusion process. Reg-NeRF [1] employs a patch-based regularization approach to diminish floating artifacts and refine geometric fidelity. Nevertheless, these approaches primarily tackle the issue of sparse views, yet they remain constrained to rendering new viewpoints within isolated scenes. Advanced methods like MVSNeRF [5], pixelNeRF [6], SRF [7], and IBRNet [8] have addressed these limitations. While these models demonstrate generalization capabilities across new scenes, their performance is hindered under extreme sparsity of input views, where the scarce view features fail to accurately represent local information, causing distortions in image structure details, significant image quality degradation, and the emergence of notable artifacts and “floaters”. Contrarily,

3 Method

Our approach begins with the extraction of view features and the construction of a cost volume. This volume undergoes regularization to facilitate the fusion of depth features across various levels and spatial dimensions. We then apply depth feature regression to accurately ascertain the density and color of spatial points. Finally, classic volumetric rendering techniques are utilized to reconstruct novel views, as illustrated in Fig. 2 and Fig. 3.

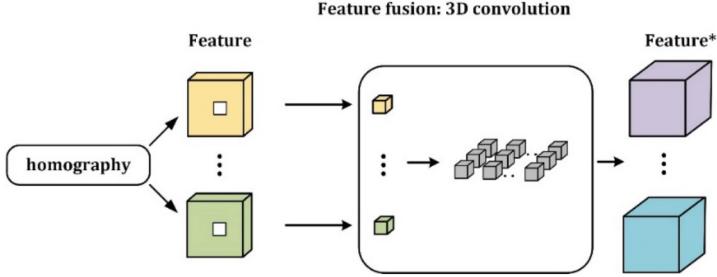


Fig. 3. Detailed process of deep feature fusion.

3.1 Constructing Cost Volume

Drawing inspiration from MVSNeRF [5] and MVSNet [9], our approach builds upon the Multi-View Stereo methodology. We employ a cost volume to represent the geometric scene in an abstract manner. The construction of the cost volume is achieved by warping the source views onto the conical frustum of the reference view, followed by implementation through plane sweeping volumes.

Feature Extraction. In this study, we apply a two-dimensional Convolutional Neural Network (2D CNN) with 3×3 convolution kernels to perform local feature extraction layer by layer on the view data, thereby enhancing the model's understanding of image details. Overall, We extract the 2D features $\{\mathbf{F}_i\}_{i=1}^V \in \mathbf{R}^{\frac{H_i}{4} \times \frac{W_i}{4} \times d}$ of all views $\{\mathbf{I}_i\}_{i=1}^V \in \mathbf{R}^{H_i \times W_i \times 3}$ through the local feature extraction network.

$$\{\mathbf{F}_i\}_{i=1}^V = M(\{\mathbf{I}_i\}_{i=1}^V), \quad (1)$$

where $M(\cdot)$ represents the model for the local feature extraction network.

Homography. We employ the classical homography transformation [10] in three dimensional vision, specifically:

$$H_i(z) = K_i \cdot \left(R_i \cdot R_1^T + \frac{(t_1 - t_i) \cdot n_1^T}{z} \right) \cdot K_1^{-1}, \quad (2)$$

where \mathbf{K} represents the intrinsic matrix, \mathbf{R} and \mathbf{t} represent the rotation vector and translation vector, respectively. z denotes the depth, and $H_i(z)$ represents the warping matrix from view i to the reference view at depth z . The feature map F_i corresponding to each source view is warped to the reference view, with the specific operations as follows:

$$F_{i,z} = F_i(H_i(z)), \quad (3)$$

where $F_{i,z}$ is the feature map corresponding to view i warped to depth z .

Cost Volume. Specifically, the cost volume is formulated by establishing a cost metric for each pixel, enabling comparative analysis and integration of data across multiple viewpoints. This process significantly refines the accuracy of depth estimation and enhances the method's robustness, proving its efficacy in the realm of 3D reconstruction. To construct the feature volume, we warp the feature maps and subsequently assemble the cost volume C , as delineated in the subsequent equation:

$$V_i = O(F_{i,z}), \quad (4)$$

$$C = \text{Var}(V_1, \dots, V_N) = \frac{\sum_{i=1}^N (V_i - \bar{V}_i)^2}{N}, \quad (5)$$

where $O(\cdot)$ represents the concatenation of warped feature maps of all depths in the new dimension, V_i represents the warped feature volume for each source view, $\text{Var}(\cdot)$ represents the variance among all warped feature volumes, and N represents the number of views.

3.2 Cost Volume Multi-feature Fusion

In previous Multi-View Stereo (MVS [11]) research, the standard approach utilized straightforward 3D convolutions and 3D transpose convolutions for feature up-sampling and down-sampling, aiming to merge appearance and geometric depth information across multiple viewpoints. While effective for datasets with abundant viewpoints, these methods fall short in sparsely viewed scenes, particularly in environments rich in detail or complexity, where traditional upsampling and down-sampling techniques struggle to accurately capture and reconstruct geometric details.

We introduce SDF-Net, an innovative feature fusion network structured into encoder and decoder phases. The corresponding formula is presented below:

$$S = SDF(C), \quad (6)$$

where $SDF(\cdot)$ represents the feature fusion network SDF-Net, and S represents the cost volume after feature fusion.

Then, given any spatial point position x , the single-point feature f is obtained through trilinear interpolation, with the specific formula as follows:

$$f = P(S, x), \quad (7)$$

where $P(\cdot)$ represents trilinear interpolation.

3.3 Depth Feature Beam Attention Mechanism

We evaluate the impact of occluded spatial point features on the beam's contribution to the final volumetric rendering and present the Depth Feature Beam Attention Mechanism. The beam attention mechanism can identify and prioritize features that are most critical to the final rendering result, reducing interference from inconsistent views. The specific formula is as follows:

$$\{f'_n\}_{n=1}^N = MHA(\{f_n\}_{n=1}^N), \quad (8)$$

where $MHA(\cdot)$ represents the multi-head attention mechanism, and N denotes the total number of spatial points on each beam.

3.4 Volume Rendering

First, the features $\{f'_n\}_{n=1}^N$ on the beam are regressed through a multi-layer perceptron to obtain the density σ_n and color \hat{c}_n . The specific formula is as follows:

$$\{\sigma_n\}_{n=1}^N, \{\hat{c}_n\}_{n=1}^N = MLP(\{f'_n\}_{n=1}^N, x, d), \quad (9)$$

where x represents the coordinates of the spatial point, and d represents the direction of the spatial point.

For volume rendering, we follow the NeRF [12] volume rendering principle. Along the ray direction, we weight the color of each point on the ray according to its volumetric density and the transmittance function, then accumulate these colors to obtain the final color $\hat{C}(r)$ of the ray:

$$T_n = \exp\left(-\sum_{j=1}^{n-1} \sigma_j\right), \quad (10)$$

$$\hat{C}(r) = \sum_{n=1}^N T_n (1 - \exp(-\sigma_n)) \hat{c}_n, \quad (11)$$

where T_n represents the cumulative transmittance of all points before point n that the ray passes through.

3.5 Loss Function

For the loss function, we employ the mean squared error (MSE) metric. In particular, we determine the MSE by comparing the volume-rendered outcomes of sampled rays against the ground truth.

$$loss = \sum_{r \in R} \|\hat{C}(r) - C(r)\|_2^2, \quad (12)$$

where R represents the set of rays during training, $C(r)$ represents the actual color, and $\hat{C}(r)$ represents the color rendered from each sampled ray.

In addition, in order to avoid training overfitting, we added the dropout method to SDF-Net. The specific method is as follows:

$$y = x \cdot r \quad (13)$$

where r is an independent and identically distributed random variable. We set r to 0.6.

4 Experiments

Table 1. Dataset Details Description

No.	Dataset Name	Description	Details
1.	DTU Data [13]	A dataset from the Technical University of Denmark consisting of multi-view high-resolution images	It consists of a total of 128 different scenes, each scene contains images of 49 camera positions, accurate camera calibration parameter files, and real depth files corresponding to the images
2.	Synthetic Data [12]	Data generated using computer graphics based on Neural Radiance Fields (NeRF)	There are a total of 8 different scenes, each scene consists of a training set and a test set, and contains different camera position images and camera calibration parameter files

Dataset. Our framework is trained on the DTU dataset [13]. Our evaluation dataset is divided into two parts: synthetic renderings of objects and real images. For the synthetic rendering dataset of objects, we use the NeRF synthetic dataset [12]. For the real dataset, we use 16 test scenes from the DTU dataset [13]. We chose DTU dataset and NeRF Synthetic dataset to evaluate our proposed method for the following reasons: DTU dataset is one of the widely recognized standard datasets in the field of computer vision, providing high-resolution multi-view images and accurate camera parameters. The NeRF Synthetic dataset contains a series of complex synthetic 3D scenes specifically designed for testing neural radiation field technology, providing fully controlled experimental conditions to systematically analyze the performance of methods in processing complex geometries and lighting conditions. The details of the adopted data sets are shown in Table 1.

Implementation Details. For training, we utilize a single V100-32Q graphics card. The number of source views adjacent to the central rendering view is set to three. Additionally, we configure the batch to contain 1024 rays, each with 128 sampling points. The Adam optimizer is employed, with an initial learning rate set at 5×10^{-4} . Our learning rate scheduling adheres to a cosine annealing strategy, setting eta min at 1×10^{-7} . In the testing phase, the number of proximate views is limited to 2–3, and each batch comprises 2048 randomly selected rays.

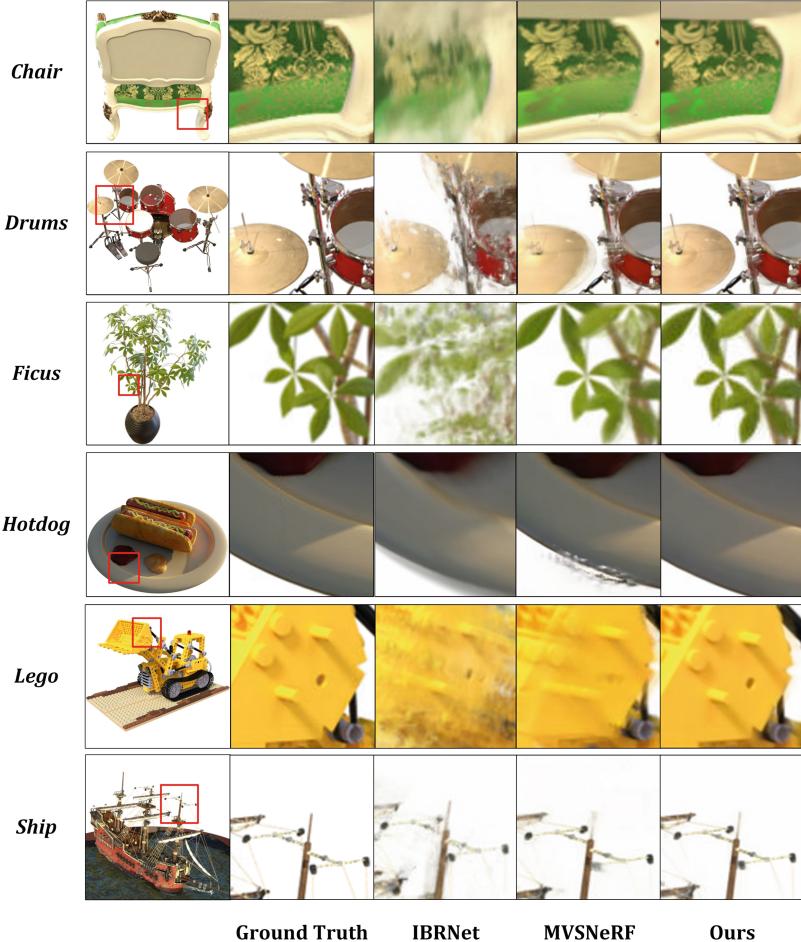


Fig. 4. Qualitative comparison on the synthetic datasets [12]. It can be seen that our method can greatly reduce the generation of blur artifacts and ensure the clarity of edges and the overall visual quality of the image.

4.1 Experimental Results

In order to ensure the rationality of the experiment, we use the same experimental conditions, that is, the same number of source views and the same source views. The evaluation of the model is divided into two parts, both based on the premise of no per-scene optimization. The first part involves comparing our model with existing generalizable NeRF models: IBRNet [8] and MVSNeRF [5]. In this study, we selected IBRNet and MVSNeRF as the baseline methods, mainly because these two methods exhibit excellent capabilities in handling sparse views and complex scenes. MVSNeRF combines the advantages of multi-view stereo technology and neural radiation fields to effectively synthesize high-quality views with complex geometry and materials, providing us with

an important baseline for comparison. The second part compares our model with non-generalizable NeRF models that require per-scene optimization, such as DietNeRF [14], RegNeRF [1], mip-NeRF [15], and DiffusioNeRF [4]. These four methods are all recent representative works in the field of novel view synthesis in sparse scenes. They each improve and optimize NeRF from different angles, and achieve better synthesis quality under sparse input views.

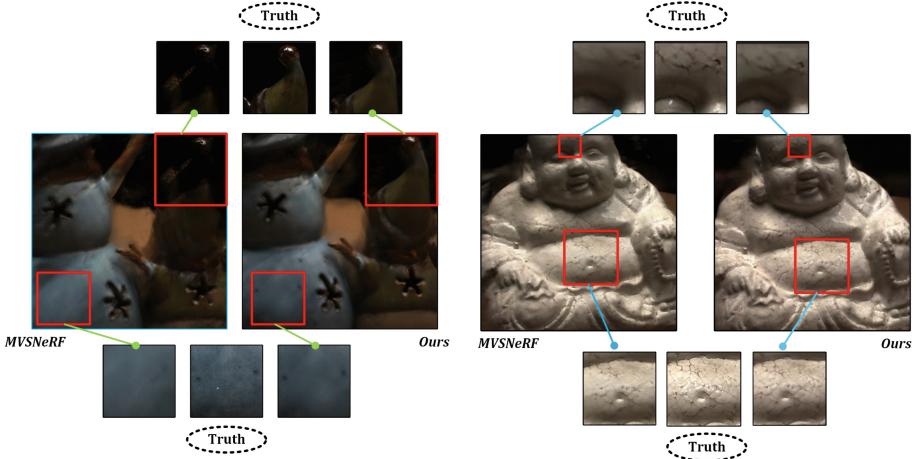


Fig. 5. Qualitative comparison on the DTU dataset [13] reveals that our proposed method substantially enhances the reconstruction of high-frequency information, as discerned from the analysis of magnified detail images.

Table 2. Performance on SYNTHETIC DATASETS [12].

Method	Settings	Synthetic Data [12]		
		PSNR↑	SSI↑M↑	LPIP↓
IBRNet [14]		21.01	0.841	0.207
MVSNeRF [2]	No per-scene Opt	24.33	0.893	0.185
Ours		25.72	0.908	0.153

In the research field of visual rendering and reconstruction, quantitative evaluation metrics such as Peak Signal-to-Noise Ratio (PSNR [16]), Structural Similarity Index (SSIM [17]), and Learned Perceptual Image Patch Similarity (LPIPS [18]) are key to measuring model performance. In this study, our proposed model performs a quantitative comparison of these metrics on a synthetic dataset, as shown in Table 2. The results reveal that our model outperforms existing generalizable models in processing input images with the same viewing angles and number, demonstrating higher image reconstruction accuracy and visual quality, detailed illustrations are shown in Fig. 4. Table 3 provides an evaluation comparison on a real dataset, where six different scenes were randomly

selected from the 16 test scenes in the dataset to ensure the fairness and representativeness of the experimental results. In this experiment, our proposed model exhibits remarkable competitiveness. Figure 5 shows the enlarged details.

Table 3. Evaluation on DTU Datasets [13].

Scans	Settings	Ours			MVSNeRF [5]		
		—			—		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Scan8	No per-scene Opt	17.50	0.886	0.129	16.62	0.872	0.153
Scan71		22.31	0.871	0.212	21.65	0.844	0.252
Scan85		28.45	0.929	0.094	27.74	0.919	0.102
Scan103		20.27	0.917	0.165	19.20	0.896	0.194
Scan111		25.07	0.915	0.119	22.62	0.878	0.136
Scan114		22.96	0.914	0.112	22.23	0.907	0.134
Mean		22.76	0.905	0.138	21.67	0.886	0.162

In the second part of the experiment, we compared our model with previous excellent models that required per-scene rendering under sparse input conditions. These methods rely on sparse input views provided for each scene to synthesize novel views without the ability to generalize across scenes. We compare these models by setting the same number of input views to demonstrate the relative advantages of our model. Through extensive experiments, our model is proven to achieve higher-quality view rendering with limited inputs, as clearly demonstrated in Table 4.

Table 4. The test dataset is the DTU Datasets [13], and we randomly select test scenes.

Method	Number of Views	DTU Data [13]		
		PSNR↑	SSIM↑	LPIPS↓
DietNeRF [6]	3 Views	10.01	0.354	0.314
RegNeRF [10]		15.33	0.621	0.190
mip-NeRF [1]		7.64	0.227	0.353
DiffusioNeRF [11]		16.20	0.698	0.160
Ours		17.50	0.886	0.129

Finally, in order to strengthen the effectiveness of the proposed method, we conducted confidence experiments on two different data sets, as shown in Table 5.

Table 5. Confidence Experiments on SYNTHETIC DATASETS [12]

Method	Trials	95% CI		
		PSNR↑	SSIM↑	LPIPS↓
IBRNet [14]	15	[20.93–21.07],	[0.839–0.849],	[0.202–0.208]
MVSNeRF [2]	15	[24.27–24.41],	[0.890–0.897],	[0.181–0.187]
Ours	15	[25.60–25.77],	[0.905–0.909],	[0.152–0.155]

4.2 Ablation Experiments

In the first part of the ablation study, we separately incorporate DFL-Attention and SDF-Net into the baseline model and also examine the full model with both modules integrated, to validate the effectiveness of each component. The specific results are presented in Table 6.

Table 6. Ablation studies of key components.

Ablation Study	PSNR↑	SSIM↑	LPIPS↓
Baseline	24.33	0.893	0.185
+ DFL-Attention	24.40	0.900	0.188
+ SDF-Net	24.91	0.903	0.169
Full model Ours	25.72	0.908	0.153

Table 7. Ablation Study on Replacements during the Multi-Feature Fusion Stage.

	Replace	PSNR↑	SSIM↑	LPIPS↓
SDF-Net	(3D-UNet)	25.30	0.903	0.167
Interpolation	(Transposed)	25.02	0.901	0.252
Instance Norm	(Batch Norm)	25.46	0.905	0.162
Conv	(Dilated Conv)	25.22	0.905	0.166
Res Connection	(\times)	24.99	0.904	0.167
Full model Ours		25.72	0.908	0.153

In the subsequent segment of our ablation analysis, we explore alternative configurations by replacing SDF-Net and its pivotal components to ascertain the optimal configuration. Specifically, initially, we replace the SDF-Net network with the well-known 3D-UNet [19]. In the second set of experiments, we replace the interpolation upsampling in SDF-Net with 3D transpose convolution, where we believe that interpolation upsampling maintains the basic depth features of the original data well, while 3D

transpose convolution reconstructs spatial features during upsampling, potentially introducing feature distortion or noise due to inserted holes. In the third set of experiments, we replace instance normalization in SDF-Net with batch normalization, analyzing that batch normalization might decrease performance in detail reconstruction and geometric consistency due to its cross-batch operation characteristic. In the fourth experiment, we consider using dilated convolution in the feature extraction phase to expand the receptive field for acquiring global features and enriching low-frequency information. However, extensive experiments show that global depth features may lead to overly smooth features, causing the model to overlearn low-frequency information and struggle to capture local details and minor changes. Lastly, to verify whether introducing residual connections [20] in the cost volume multi-feature fusion encoder stage enhances the representation of high-frequency information, a control experiment was designed by specifically removing residual connections to observe their impact on the reconstruction ability of high-frequency information. Detailed experimental results are shown in Table 7.

5 Conclusion

In our approach, the framework primarily consists of two core components: firstly, we have devised an innovative stereo feature fusion network that employs a multi-scale and multi-layered feature fusion strategy, effectively enhancing the model's ability to parse the scene's spatial structure and texture details, achieving efficient regression prediction of spatial point density. Secondly, we introduce the depth feature beam attention mechanism, which, by enhancing the consistency and effectiveness of features along the light path, effectively suppresses the noise interference from occluded point features, reducing color deviations in the volume rendering process. The integration of these two key components significantly improves the quality of novel view synthesis under sparse viewpoint conditions. Although the SDF Net significantly enhances rendering quality, the involvement of three-dimensional feature extraction inherently reduces the network rendering speed. Balancing high-quality outputs with fast rendering remains a significant challenge in future research. To address this, we are considering the partitioning of the depth dimension of cost volume features and merging them into two-dimensional features. This approach aims to effectively retain depth features while avoiding extensive three-dimensional convolutions, thereby enhancing rendering efficiency. Additionally, for real scenes with colored backgrounds, the rendering results are not yet satisfactory. In future studies, we plan to employ multi-scale rendering techniques, which will allow for coarse-grained control of background rendering and fine-grained control of object detail, thus further optimizing the rendering performance.

References

1. Niemeyer, M., Barron, J.T., Mildenhall, B., Sajjadi, M.S., Geiger, A., Radwan, N.: RegNeRF: regularizing neural radiance fields for view synthesis from sparse inputs. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5480–5490 (2022)

2. Deng, K., Liu, A., Zhu, J.Y., Ramanan, D.: Depth-supervised nerf: Fewer views and faster training for free. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12882–12891 (2022)
3. Kulhánek, J., Derner, E., Sattler, T., Babuška, R.: Viewformer: NeRF-free neural rendering from few images using transformers. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) European Conference on Computer Vision, vol. 13675, pp. 198–216. Springer (2022). https://doi.org/10.1007/978-3-031-19784-0_12
4. Wynn, J., Turmukhambetov, D.: DiffusioNeRF: regularizing neural radiance fields with denoising diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4180–4189 (2023)
5. Chen, A., et al.: MVSNeRF: fast generalizable radiance field reconstruction from multi-view stereo. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 14124–14133 (2021)
6. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: PixelNeRF: neural radiance fields from one or few images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4578–4587 (2021)
7. Miralles, F., Posern, G., Zaromytidou, A.I., Treisman, R.: Actin dynamics control SRF activity by regulation of its coactivator MAL. *Cell* **113**(3), 329–342 (2003)
8. Wang, Q., et al.: IBRNet: Learning multi-view image-based rendering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4690–4699 (2021)
9. Yao, Y., Luo, Z., Li, S., Fang, T., Quan, L.: MVSNet: depth inference for unstructured multi-view stereo. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 767–783 (2018)
10. DeTone, D., Malisiewicz, T., Rabinovich, A.: Deep image homography estimation. arXiv Preprint [arXiv:1606.03798](https://arxiv.org/abs/1606.03798) (2016)
11. Goesele, M., Curless, B., Seitz, S.M.: Multi-view stereo revisited. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006), vol. 2, pp. 2402–2409. IEEE (2006)
12. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: representing scenes as neural radiance fields for view synthesis. *Commun. ACM* **65**(1), 99–106 (2021)
13. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of RGBD SLAM systems. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 573–580. IEEE (2012)
14. Jain, A., Tancik, M., Abbeel, P.: Putting nerf on a diet: semantically consistent few-shot view synthesis. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 5885–5894 (2021)
15. Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P.: Mip-NeRF: a multiscale representation for anti-aliasing neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 5855–5864 (2021)
16. Huynh-Thu, Q., Ghanbari, M.: Scope of validity of PSNR in image/video quality assessment. *Electron. Lett.* **44**(13), 800–801 (2008)
17. 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)
18. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 586–595 (2018)

19. Xiao, Z., Liu, B., Geng, L., Zhang, F., Liu, Y.: Segmentation of lung nodules using improved 3D-unet neural network. *Symmetry* **12**(11), 1787 (2020)
20. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, inception-resnet and the impact of residual connections on learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31 (2017)



TUPocket: 3D Convolutional Neural Network Combined with Transformer for Ligand Binding Site Detection

Yangtao Meng^(✉) and Tianhao Yan

College of Computer Science and Technology, Wuhan University of Science and Technology,
Wuhan, Hubei, China
202113407302@wust.edu.cn

Abstract. In the field of drug development, it is crucial to accurately predict the binding sites of protein interactions, which directly affects the design efficiency and development efficiency of targeted drugs. A targeted drug can only play its due role when it binds specifically to a defined site within the protein structure. If the binding site prediction is inaccurate, it may result in unexpected side effects. Not only might the drug fail to treat the disease effectively, but it could also potentially harm the health of the individual taking the medication. Despite the classical Unet-3D model having shown initial potential in 3D image segmentation and its capability to handle most scenarios, its performance constraints increasingly manifest when dealing with intricate and complex biomolecular binding sites. As a result, this study uniquely incorporates the attention mechanism derived from Transformer, alongside a variety of complementary attention modules and technologies, thereby significantly enhancing both the predictive accuracy and generalization capabilities of the model. The experimental results demonstrate that the model exhibits superior ability of prediction and generalization. The enhanced model not only attains upper precision on experimental datasets, but also has excellent performance when dealing with uncharted data. The present study reveals the enhancement in the predictive capacity for protein binding sites achieved by the optimized Unet-3D architecture integrated with Transformer's attention framework.

Keywords: Convolutional Neural Networks · Transformer · Ligand Binding Site Detection and Segmentation

1 Introduction

With the swift advancements in science and technology, deep learning is increasingly being widely applied in the field of life science research [1], particularly within the two significant areas of protein structure prediction and drug research and development [2]. The 3D convolutional neural network (CNN) [3], a common tool, is playing an essential role [4]. This article aims to thoroughly examine the research progress and technical trends of 3D CNN domestically and internationally in the context of ligand binding site detection and segmentation, ultimately uncovering their significant potential for advancing precision therapy and novel drug discovery.

Within the realm of international scientific inquiry, researchers combine CNN with diverse deep learning architectures and construct prediction models characterized by heightened precision and enhanced generalizability. For example, combined with recurrent neural networks (RNN) [5], protein sequence information is parsed to unveil the relationship between sequence and structure; In addition, CNN is combined with graph neural networks (GNN) [6] to capture interactions between amino acids across long distances, which is significant to understand protein folding processes and functions. In the domain of domestic inquiry, researchers have also made many attempts to create better model. They take great use of the abundant indigenous biological resources, which forges a series protein structure prediction model to adapt to China's national conditions. These models intertwine CNN with alternative machine learning methods [7] and biophysical principles [8], thereby catering complex structural prediction challenges. In particular, domestic researchers use CNN's spatial extraction capacity to analyze the characteristic information of proteins while combining the attention mechanism to enhance models' prediction accuracy and generalizability.

This investigation adhered to Fpocket's CNN based scoring approach [9] because it exceeded the most advanced methods at the time in identifying and sequencing protein ligand binding sites. Despite classical CNN's commendable performance in recognition tasks, its segmentation capabilities are inferior compared to those of the optimized CNN model. The conventional Unet-3D architecture (see to Fig. 1) [10] obtains high-level abstract features via the downsampling module, then recovers spatial resolution along the upsampling module, and channels low-level features directly from the encoder to the decoder via skip links, thereby retaining rich spatial details.

Compared with this, the enhanced Unet-3D model combined with Transformer [11] discussed in this paper (see Fig. 2) has better performance. This study presents an innovative and optimized Unet-3D's core architecture, incorporating the attention mechanism from Transformer to enhance the model's capacity to extract global features and handle sequential information effectively. In particular, within the encoder module, while maintaining the fundamental Unet-3D framework, we introduce both Convolutional Block Attention Module (CBAM) [12] and SelfAttention mechanism. CBAM refines the significance of channels and spatial characteristics, whereas SelfAttention shows the interdependence within the feature map. At the same time, this investigation employs more complicated activation function and loss function to accommodate a wider range of application contexts, while improving gradient flow via residual connections to preserve more valuable information. Within the decoder component, the central enhancement of the model is reflected in the application of the CrossAttention mechanism, which operates during the upsampling part. This configuration facilitates the high-level features generated by the encoder being combined with the features at various levels within the decoder closely, which enhances feature fusion quality through information exchange. This means that for tasks like protein binding site prediction, the enhanced Unet-3D model can more comprehensively integrate contextual information and attain superior 3D spatial segmentation precision.

This paper tested the optimized Unet-3D model using the COACH420 and HOLO4K datasets furnished by P2Rank [13] as well as SC6K [9] dataset. The results indicate that,

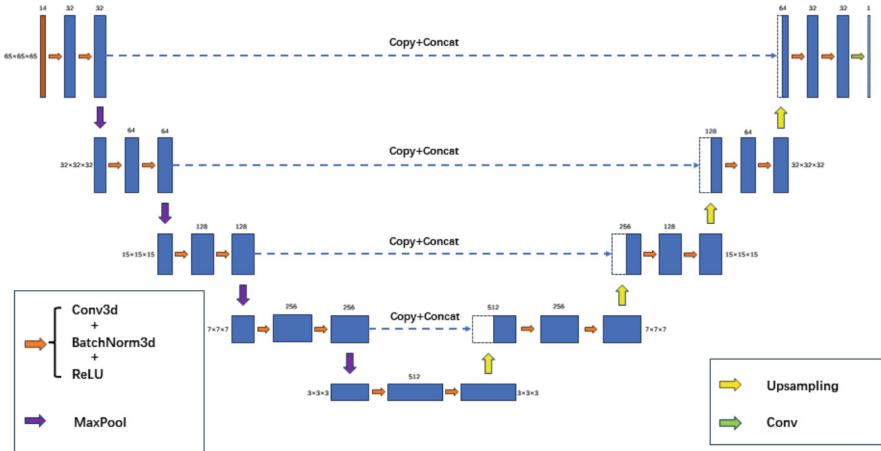


Fig. 1. This is the illustration of convolutional neural network segmentation utilizing the Unet-3D framework.

compared to its conventional counterpart, the optimized model not only maintains its inherent strengths but also enhances prediction accuracy and generalizability.

2 Method

2.1 Datasets and Preprocessing

In this study, the datasets processing adhered to the method employed by the benchmark model DeepPocket [9]. For training model and cross-validation purposes, the scPDB v.2017 [14] database was utilized. The dataset contains 17,594 binding sites associated with 16,612 distinct proteins, and contains 5,540 unique UniProt IDs. To assess the performance of the optimized model and compare it with the baseline model, this study utilizes three external datasets: COACH420, HOLO4K and SC6K. The three datasets were processed in a consistent manner [9], employing the Biopython library [15] to extract proteins and ligands from their respective structural files and subsequently convert them into Mol2 format using Openbabel [16]. After this isolation, any ligands that failed to be handled by either RDKit [17] or BioPandas [18] were discarded.

2.2 Segmenting Shapes of Binding Sites

In this study, the positive instances within the classification dataset serve as the training material for the segmentation CNN model. This study draws inspiration from both Unet and Transformer architectures, so the investigated segmentation model not only integrates the distinctive advantages of each (see Fig. 2), but also incorporates diverse mechanisms, such as CBAM, into its design (see Fig. 3).

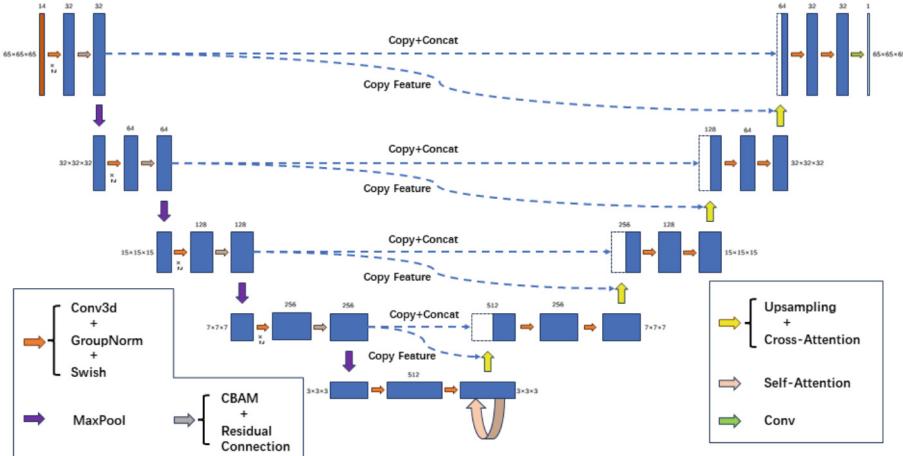
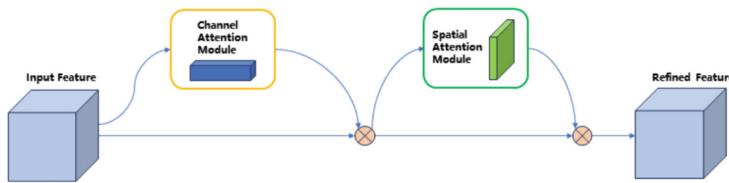
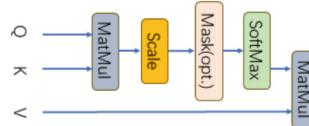


Fig. 2. This diagram illustrates the integration of Unet-3D’s convolutional neural network segmentation architecture with the Transformer.

Convolutional Block Attention Module(CBAM)



Transformer Self-Attention Module



Cross Attention
only changes the
Self Attention input

Fig. 3. The diagrams illustrate the operational processes of both CBAM and Attention mechanisms.

Encoder

The forward propagation in each encoder initiates with the DoubleConvolution module, comprising a pair of continuous convolutional layers. Subsequent to every individual convolutional layer, GroupNorm (GN) is implemented to facilitate cross-channel normalization, followed by the SiLU (Swish) [19] non-linear activation function, aimed at enhancing the model’s non-linear representation capabilities. Additionally, to avoid overfitting, a Dropout layer is incorporated subsequent to each layer.

Inside the encoder, two modules ChannelAttention and SpatialAttention are introduced (see Fig. 4) [12] to enrich the feature representation. To be specific:

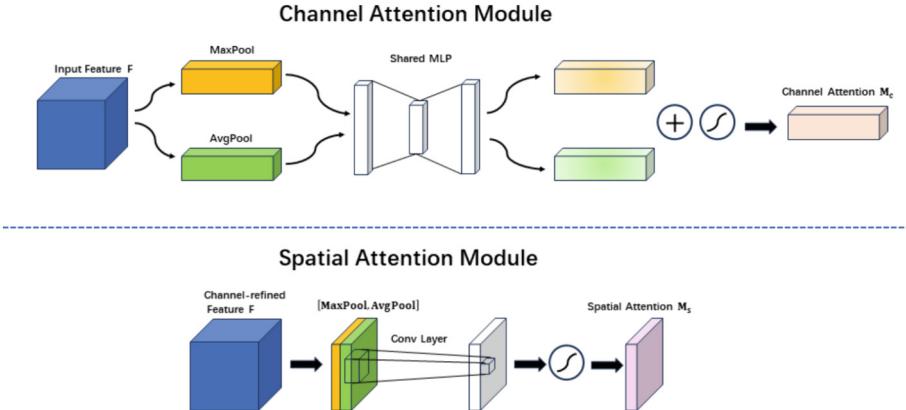


Fig. 4. This illustration describes the Channel Attention module and the Spatial Attention module.

- The Channel Attention module employs a dual step processing approach on the input feature map. Specifically, it conducts a MaxPool operation on one branch, while on the parallel branch, AvgPool is utilized to aggregate the spatial information within the feature map. Both MaxPool and AvgPool distinct representations that are independently forwarded to a shared Multi-Layer Perceptron (MLP) network. The outputs from the MLP corresponding to each pooling type are then concatenated element-wise, and the resulting sum is passed through a sigmoid activation function to generate the final Channel Attention Feature Map. This attention map is subsequently using elementwise multiplied with the original input feature map, thereby producing the input features required by the Spatial attention module.

The channel attention mechanism can be expressed as follows:

$$\begin{aligned} M_c(F) &= \sigma(MLP(AvgPool(F)) + MLP(MaxPool(F))) \\ &= \sigma\left(W_1\left(W_0\left(F_{avg}^c\right)\right)\right) + \sigma\left(W_1\left(W_0\left(F_{max}^c\right)\right)\right) \end{aligned} \quad (1)$$

- The SpatialAttention module focuses on refining spatial information. It uses the feature graph produced by the Channel attention module as its input. Firstly, global max pooling and global average pooling are performed, and the results of these operations are concatenated. Following a convolutional operation, the dimension is reduced to a single channel. Then, the spatial attention feature is generated via a sigmoid function. Finally, this feature is multiplied with the input feature of the module to obtain the final generated feature.

The spatial attention mechanism can be expressed as follows:

$$\begin{aligned} M_s(F) &= \sigma\left(f^{7 \times 7}\left([AvgPool(F); MaxPool(F)]\right)\right) \\ &= \sigma\left(f^{7 \times 7}\left([F_{avg}^s; F_{max}^s]\right)\right) \end{aligned} \quad (2)$$

CBAM can adaptively adjust the feature weights of each channel and spatial position, which helps the model pay more attention to the features related to the current task

and reduces the impact of irrelevant information on the model. CBAM captures global channel and spatial context information through global pooling, which can consider not only local features, but also global context information. Thus it can help to better understand and distinguish the subtle differences between different categories.

At the last level of the encoder, a SelfAttention module is incorporated to implement a multi-head SelfAttention mechanism. Self-Attention generates the attention weight matrix through the position information of the sequence, and then generates a new feature representation using the weighted summation of the matrix. This process makes the feature of each location fuse the context information of the whole sequence, thus capturing the global context information effectively.

Decoder

In the Decoder module, the CrossAttention module plays a significant role in this research. Serving as an information conduit between the encoder and decoder, it facilitates the connection and interaction of features across these two components. The module concatenates the features of the upsampling layer with their corresponding encoder counterparts, subsequently engaging them in a multi-head attention mechanism. This process aims at conveying the deeply encoded abstract semantic information to the shallower feature maps, thereby enabling the precise extraction and utilization of contextual information. Compared to the conventional U-Net, which only relies on skip connections, the CrossAttention mechanism more effectively preserves and harnesses the semantic information embedded in the deeper encoder layers. This strategy helps to avert the potential loss of valuable information that might occur through straightforward concatenation during feature fusion. Furthermore, the incorporation of CrossAttention enhances the model’s generalization capabilities when confronted with unfamiliar images or complex structures. As a result, it exhibits increased flexibility in image processing, demonstrating superior robustness and accuracy on novel datasets.

Cross-Attention is achieved by calculating the similarity of the query to the key, generating an attention distribution, and then using this distribution to sum the values weighted to obtain a new query representation. In this process, the query side can adjust its own characteristics according to the cross-modal information provided by the key side, thereby realizing the effective capture of the context.

Within each UpSampling stage, by the application of the CrossAttention module, the encoder features and the upsampling features undergo fusion. These combined representations are then subjected to additional refinement through the DoubleConvolution module. Sequentially, after progressive upsampling and feature integration throughout all stages, the model ultimately produces an output graph identical in size to the input image, achieved by a final 1×1 convolution layer (denoted as final_conv).

3 Results and Discussion

In this paper, the ablation experiments were carried out to demonstrate the specific improvement of each module to the model. To assess the model’s generalization capacity and accuracy, this paper employs the previously mentioned COACH420,

HOLO4K [6] and SC6K [9] test sets for evaluation purposes. And in this paper, Distance to the center of the binding site (DCC) and Discretized volume overlap (DVO) are

Table 1. DCC and DVO results comparison

DataSet	Model	DCC	DVO
COACH420	Kalasanty	56.85	24.49
	DeepPocket	85.08	54.12
	RecurPocket	89.91	53.19
	GLPocket	<u>92.74</u>	<u>55.18</u>
	ResPocket	94.76	55.60
	TUPocket	89.11	53.78
HOLO4K	Kalasanty	51.08	21.53
	DeepPocket	83.62	51.82
	RecurPocket	89.94	53.43
	GLPocket	<u>90.20</u>	<u>54.22</u>
	ResPocket	91.77	54.48
	TUPocket	86.66	50.79
SC6K	Kalasanty	49.20	18.83
	DeepPocket	84.03	50.22
	RecurPocket	<u>92.77</u>	54.22
	GLPocket	92.50	52.67
	ResPocket	93.27	<u>53.47</u>
	TUPocket	85.34	49.24

used to evaluate the model. The performance of the enhanced model is compared against the baseline model, as well as several more sophisticated models, such as RecurPocket [20] and GLPocket [21].

In the traditional Unet model, the maximum pooling method in the downsampling stage will cause a lot of information loss. This study hopes to find a way to reduce the information loss caused by maximum pooling, thereby ensuring that the rich feature information can be transmitted to the next layer. In this study, the CBAM (Convolutional Block Attention Module) mechanism was initially incorporated into the baseline model to reduce information loss stemming from direct maximum pooling.

The results (see Table 1) indicated that, in comparison to the baseline model DeepPocket, the enhanced model demonstrated improved performance on the COACH420 dataset and on the DCC subset of the HOLO4K and SC6K datasets, but exhibited a decline on the DVO subset. In a way, these findings suggest that during the model optimization process, a greater emphasis was placed on enhancing positional accuracy, at the expense of maintaining shape and size details. Therefore, this paper uses the relatively popular Transformer, hoping to improve the model's accurate prediction and generalization ability without sacrificing other details.

Table 2. DCC and DVO results comparison

DataSet	Model	DCC	DVO
COACH420	Kalasanty	56.85	24.49
	DeepPocket	85.08	54.12
	RecurPocket	89.91	53.19
	GLPocket	<u>92.74</u>	<u>55.18</u>
	ResPocket	94.76	55.60
	TUPocket	90.12	54.69
HOLO4K	Kalasanty	51.08	21.53
	DeepPocket	83.62	51.82
	RecurPocket	89.94	53.43
	GLPocket	<u>90.20</u>	<u>54.22</u>
	ResPocket	91.77	54.48
	TUPocket	86.84	50.41
SC6K	Kalasanty	49.20	18.83
	DeepPocket	84.03	50.22
	RecurPocket	<u>92.77</u>	54.22
	GLPocket	92.50	52.67
	ResPocket	93.27	<u>53.47</u>
	TUPocket	87.84	51.37

SelfAttention mechanism was incorporated into the last layer of the encoder, whereas a CrossAttention mechanism was employed in the upsampling module at the decoder. The results of only adding Transfomer model are shown in the Table 2. The results show that compared with models with only CBAM, adding Transformer will provide greater improvements to the model. This phenomenon shows that the attention mechanism derived from Transformer has better global information modeling and cross-layer information interaction, which can more accurately identify and segment objects in complex scenes. Although CBAM can also extract global context through global average pooling and maximum pooling, it is mainly concerned with channel and spatial attention, so compared with Self-Attention and Cross-Attention, CBAM may be limited in global information modeling.

Both CBAM and Transformer have shown good performance in improving model accuracy, and this study believes that the two mechanisms can complement each other to generate a better model. Therefore, this study will combine CBAM and Transformer to explore the enhanced model's prediction accuracy and generalization ability.

The impact of combining CBAM and Transformer is shown in the Table 3. Relative to RecurPocket, model combined with the Transformer and CBAM exhibits superior performance on the COACH420 dataset, but performs worse than RecurPocket on

Table 3. DCC and DVO results comparison

DataSet	Model	DCC	DVO
COACH420	Kalasanty	56.85	24.49
	DeepPocket	85.08	54.12
	RecurPocket	89.91	53.19
	GLPocket	<u>92.74</u>	55.18
	ResPocket	94.76	<u>55.60</u>
	TUPocket	90.73	55.91
HOLO4K	Kalasanty	51.08	21.53
	DeepPocket	83.62	51.82
	RecurPocket	89.94	53.43
	GLPocket	<u>90.20</u>	<u>54.22</u>
	ResPocket	91.77	54.48
	TUPocket	86.63	52.07
SC6K	Kalasanty	49.20	18.83
	DeepPocket	84.03	50.22
	RecurPocket	<u>92.77</u>	54.22
	GLPocket	92.50	52.67
	ResPocket	93.27	<u>53.47</u>
	TUPocket	88.90	50.31

the HOLO4K and SC6K datasets. Considering the baseline model's description, which mentions the use of the scPDB v.2017 dataset for training, the generated model's generalization capability to the HOLO4K dataset may be compromised, contributing to the consistently low measurement results observed for both this model and the baseline model [9]. In addition, the results also show that although CBAM and Transformer can be combined to achieve positive enhancement, CBAM will enhance the model at the expense of maintaining shape and size details, which will result in lower performance on DVO than using Transformer alone.

When CBAM and Transformer are combined, the model's performance is not satisfactory. This paper considers the use of new activation functions and adding residual connections, which can effectively solve the problem of disappearing or exploding gradients and improve the robustness of the model. In addition, residual connections can also facilitate information flow, allowing information to be transmitted from lower levels to higher levels, enhancing the transmission of information over long distances.

In an effort to enhance the model's generalization capacity, regularization techniques were also introduced, which are known to benefit the model's predictive capacity on unfamiliar data. By imposing constraints on the model's degrees of freedom, regularization fosters the development of a model that remains robust when confronted with

Table 4. DCC and DVO results comparison

DataSet	Model	DCC	DVO
COACH420	Kalasanty	56.85	24.49
	DeepPocket	85.08	54.12
	RecurPocket	89.91	53.19
	GLPocket	92.74	55.18
	ResPocket	94.76	55.60
	TUPocket	<u>93.75</u>	<u>55.40</u>
HOLO4K	Kalasanty	51.08	21.53
	DeepPocket	83.62	51.82
	RecurPocket	89.94	53.43
	GLPocket	90.20	<u>54.22</u>
	ResPocket	91.77	54.48
	TUPocket	<u>91.15</u>	51.39
SC6K	Kalasanty	49.20	18.83
	DeepPocket	84.03	50.22
	RecurPocket	92.77	54.22
	GLPocket	92.50	52.67
	ResPocket	93.27	<u>53.47</u>
	TUPocket	<u>92.76</u>	52.78

novel, unforeseen data. Additionally, the employment of the more advanced activation function Swish [19] was incorporated, as follow:

$$f(x) = x * \text{sigmoid}(x) \quad (3)$$

Swish activation function has shown advantages over ReLU [22] in many tasks, especially in biomedical image analysis and natural language processing, so this study attempts to use it as an activation function. In addition, residual connections are added. It will add the original input to the processed output to form residual connections after processing the entire convolution block.

According to the result description in Table 4 it is found that the generalization ability of the final model has been improved to a certain extent. It is obvious that TUPocket is ahead of GLPocket on the test performance of COACH420, and the DCC index of HOLO4K and SC6K is ahead. We have reason to believe that if the number of training sets becomes richer, the generalization ability of the final optimized model should be stronger.

In addition, the batch_size used in this paper is 8, and the experiment shows that the effect is best when the epoch is about 50.

4 Conclusion

This study aimed at optimizing the Unet-3D model to attain heightened accuracy and generalization prowess. Drawing primarily from the Transformer’s encoder-decoder architecture and attention mechanisms, a diverse array of attention schemes and advanced methods were incorporated into the downsampling and upsampling modules of the Unet. In the encoder module, the model makes several modifications: the activation function was altered, loss functions and regularization were introduced within the DoubleConvolution layers, residual connections were employed, and the CBAM (Convolutional Block Attention Module) was integrated. The SelfAttention mechanism was appended at the final downsampling layer, empowering the model to discern meaningful correlations; On the decoder section, the CrossAttention mechanism was introduced, facilitating a direct connection between features of corresponding layers. The efficacy of these optimizations has been substantiated, contributing to a substantial enhancement in the model’s capacity for generalization.

References

1. Lin, X., Zhang, X., Xu, X.: Efficient classification of hot spots and hub protein interfaces by recursive feature elimination and gradient boosting. *IEEE/ACM Trans. Comput. Biol. Bioinf.* **17**, 1525–1534 (2020)
2. Lin, X., Zhang, X.: Prediction of hot regions in PPIs based on improved local community structure detecting. *IEEE/ACM Trans. Comput. Biol. Bioinf.* **15**, 1470–1479 (2018)
3. Tran, D., Bourdev, L.D., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3D convolutional networks. In: 2015 IEEE International Conference on Computer Vision (ICCV), pp. 4489–4497 (2014)
4. Lin, X., Xu, S., Liu, X., Zhang, X., Hu, J.: Detecting drug–target interactions with feature similarity fusion and molecular graphs. *Biology*, **11**, 967 (2022)
5. Lipton, Z.C.: A critical review of recurrent neural networks for sequence Learning. ArXiv, abs/1506.00019 (2015)
6. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. *IEEE Trans. Neural Networks* **20**, 61–80 (2009)
7. Peng, Z., et al.: Conformer: local features coupling global representations for visual recognition. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 357–366 (2021)
8. Luo, R., Sedlazeck, F.J., Lam, T.W., Schatz, M.C.: Clairvoyante: a multi-task convolutional deep neural network for variant calling in single molecule sequencing. *bioRxiv* (2018)
9. Aggarwal, R., Gupta, A., Chelur, V.R., Jawahar, C.V., Priyakumar, U.D.: DeepPocket: ligand binding site detection and segmentation using 3D convolutional neural networks. *J. Chem. Inform. Model.* **62**, 5069–5076 (2021)
10. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. ArXiv, abs/1505.04597 (2015)
11. Vaswani, A., et al.: Attention is all you need. *Neural Inform. Process. Syst.* (2017)
12. Woo, S., Park, J., Lee, J., Kweon, I.: CBAM: Convolutional Block Attention Module. ArXiv, abs/1807.06521 (2018)
13. Krivák, R., Hoksza, D.: P2Rank: machine learning based tool for rapid and accurate prediction of ligand binding sites from protein structure. *J. Cheminformat.* **10**, 1–12 (2018)

14. Desaphy, J., Bret, G., Rognan, D., Kellenberger, E.: Sc-PDB: a 3D-database of ligandable binding sites—10 years on. *Nucleic Acids Res.* **43**, D399–D404 (2014)
15. Cock, P.J., et al.: Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics* **25**, 1422–1423 (2009)
16. O’Boyle, N.M., Banck, M.S., James, C.A., Morley, C., Vandermeersch, T., Hutchison, G.R.: Open Babel: an open chemical toolbox. *J. Cheminformat.* **3**, 33 (2011)
17. Landrum, G.: RDKit: a software suite for cheminformatics, computational chemistry, and predictive modeling (2013)
18. Raschka, S.: BioPandas: working with molecular structures in pandas DataFrames. *J. Open Source Softw.* **2**, 279 (2017)
19. Ramachandran, P., Zoph, B., Le, Q.V.: Swish: a self-gated activation function. arXiv: Neural and Evolutionary Computing (2017)
20. Li, P., Cao, B., Tu, S., Xu, L.: RecurPocket: recurrent Lmser Network with Gating Mechanism for Protein Binding Site Detection. In: 2022 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), pp. 334–339 (2022)
21. Li, P., Liu, Y., Tu, S., Xu, L.: GLPocket: a multi-scale representation learning approach for protein binding site prediction. In: International Joint Conference on Artificial Intelligence (2023)
22. Agarap, A.F.M.: Deep learning using rectified linear units (ReLU) (2018)



A Graph Neural Network-Based Multi-agent Joint Motion Prediction Method for Motion Trajectory Prediction

Hongxu Gao¹, Zhao Huang^{1(✉)}, Jia Zhou², Song Cheng³, Quan Wang¹, and Yu Li⁴

¹ Xidian University, Xi'an 710071, China

z_huang@xidian.edu.cn

² Northwestern Polytechnical University, Xi'an 710072, China

³ CETC Galaxy BeiDou Technology (Xi'an) Co. Ltd., Xi'an 710068, China

⁴ Xi'an Jiaotong University, Xi'an 710049, China

Abstract. Autonomous driving is a revolutionary automotive technology that can greatly improve driving safety and reduce traffic congestion. Multi-agent trajectory prediction plays an important role in autonomous driving technology. Considering that the existing methods are limited to local information areas because only local information is considered in the computation process, a multi-agent joint trajectory prediction model based on a GNN is proposed in this paper to help self-driving cars better plan their paths and avoid collisions. In the joint multi-agent prediction, this paper adopts a multitask fusion training strategy, which implements the selection and setting of multiple tasks through masks, and fuses the model for training. In the decoder and loss function design, the model is based on learnable anchors. A multimodal joint trajectory prediction decoder considering multi-agent and multimodal interactions is designed, and triple loss is introduced. The proposed method is validated on the Argoverse trajectory prediction dataset, and the experimental results show that the method can visualize the prediction results for typical traffic scenarios, intuitively demonstrating the accuracy of the prediction. The proposed model achieves better mean displacement errors and minimum end-point displacement errors with lower computational complexity, outperforming existing methods.

Keywords: Autonomous Driving · Multi-agent · Trajectory Prediction · Graph Neural Networks

1 Introduction

The rapid advancement of robotics and AI has brought significant transformations across a variety of domains, including transportation [1, 2]. Autonomous driving tech offers potential for enhanced safety, energy efficiency, and environmental benefits [3]. Scene editing is a pivotal part of autonomous driving, facilitating the testing and validation of system performance and safety [4, 5]. Simulation scenarios allow for route, lane, signal, and obstacle modifications, assessing system adaptability [6]. To simulate emergencies, editors can introduce scenarios involving pedestrians, animals, and accidents [7].

Trajectory prediction for self-driving vehicles poses challenges due to map complexity and interactions between intelligent vehicles and road data [8–10]. Accurate prediction necessitates a deep understanding of traffic dynamics. Despite advancements, trajectory prediction remains challenging, particularly in complex scenarios. Moreover, prediction algorithms must account for diverse weather and road conditions.

Therefore, there is a need for accurate joint trajectory predictions of multiple intelligence trajectories, enabling simultaneous predictions of multimodal motion trajectories. This would facilitate the planning of safe and efficient vehicle motion to better cope with complex traffic environments and ensure the safety and effectiveness of autonomous driving systems [11]. In recent years, various trajectory prediction methods for autonomous driving scenarios have been proposed [3, 5, 8], including traditional modeling based on physical models or rules, and deep learning-based methods. Physics-based motion modeling treats vehicles as dynamic entities governed by the laws of physics, but may not effectively capture the complexities of the real world. Deep learning-based methods can reveal intricate relationships between input data and trajectories, leading to better performance and generalizability [18].

To solve the problem that the distribution of simulation-generated data in scene editing is inconsistent with that of real data, this paper proposes a trajectory prediction model to more realistically simulate the interactive behavior of agents in the simulation environment and determine the future trajectory probabilities of different modalities that are more in line with the conditions of real-world occurrences. The main research contributions of this paper are described as follows:

- (1) A multitask fusion training strategy is proposed to train a model for multiple tasks by masking, enhancing both the efficiency and accuracy of the model training and scene editing in the inference phase.
- (2) The interaction between multi-agent state information and multimodality is realized through decoder design and loss function design, with the decoder augmenting input data to consider agent interactions.
- (3) The effectiveness of the model is validated through ablation, comparison, and visualization experiments, demonstrating its superiority over other models.

The paper is structured as follows: Sect. 2 introduces related works, Sect. 3 proposes a graph neural network-based multi-agent joint motion prediction method, Sect. 4 presents simulation results and comparisons, and Sect. 5 concludes and discusses future research directions.

2 Related Works

Recently, scholars have conducted a series of related studies on joint trajectory prediction for multi-intelligent systems. LaneRCNN, based on a graph-centered motion prediction model, uses a graph encoder to learn local lane map representations (LaneRoI) for past motion encoding and local map topology [1]. The model verifies the effectiveness of endpoint-based (goal-based) multimodal trajectory prediction methods [2]. LaneGCN, another graph neural network-based method, introduces an expansive graph convolution variant for contextual information aggregation along lane lines [3]. VectorNet enhances

contextual feature learning with a new auxiliary task based on vectorized representations [4]. The DenseTNT model is an endpoint-based trajectory prediction model that first scores over sampled target candidates [5].

High-quality and large-scale autopilot datasets are crucial for training and evaluating joint trajectory prediction models [6, 7]. However, existing datasets often lack the necessary coverage, leading to insufficient model generalization. Multi-intelligent systems are subject to uncertainties like sensor noise, dynamic environment changes, and the behavior of other intelligence, affecting prediction accuracy [8, 9].

During scene editing, changes in agent behavior can lead to collisions and scene failure. Early multi-agent interaction models relied on manual features like social force and energy functions. Deep learning-based approaches, on the other hand, learn interactions from real driving data, achieving higher accuracy [10]. Examples include capturing interactive pedestrian trajectories in crowded scenarios using social pooling mechanisms and learning multiple intelligent interactive behaviors using Transformer's attention mechanism [11]. Predicting the future motion of road participants in urban scenarios is crucial for autonomous driving. While existing models perform well in predicting individual agent trajectories independently, jointly predicting scene-consistent trajectories for multiple agents remains a challenge [12]. The exponential growth of the prediction space with the number of agents makes improving the accuracy and efficiency of trajectory prediction an important research direction [13].

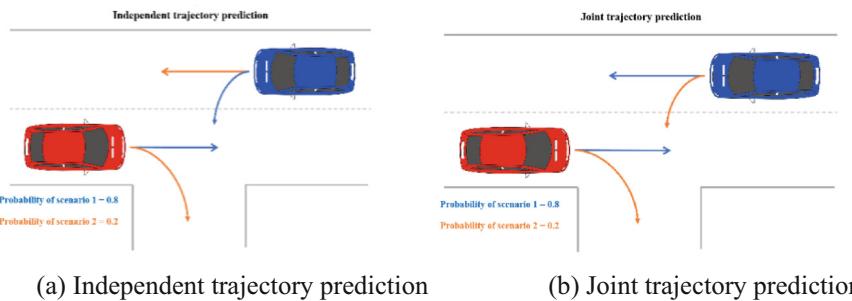


Fig. 1. Differences between independent trajectory prediction and joint motion prediction.

Figure 1 illustrates the difference between independent and joint motion prediction. Jointly predicting scene-consistent trajectories for multiple agents is a challenging problem, and different methods yield different results in different scenarios [14–16]. In low-density traffic environments, the accuracy of trajectory prediction is generally higher, whereas in high-density traffic environments, joint trajectory prediction is generally more realistic for trajectories generated for scenes with multi-agent interactions [17–19]. Therefore, in-depth research and evaluation of the performance of various methods in different scenarios are needed to provide more accurate and reliable solutions for achieving trajectory prediction in multi-agent scenarios [20–22]. Improving the accuracy and efficiency of multi-agent trajectory prediction in different scenarios is one of the future research directions.

3 A Graph Neural Network-Based Multi-agent Joint Motion Prediction Method

3.1 Overall Framework of the Model

To handle future state information and interactions among multiple agents and modalities, this paper employs a Transformer-based decoder architecture. The model first vectorizes road elements in the HD map and agent trajectories, merging them via a GNN that treats each agent as a graph node. This unifies HD map and trajectory information. Next, a hierarchical network architecture with an attention mechanism is used to design a graph neural network-based traffic scene encoder, thereby achieving hierarchical representation through interaction modeling modules.

The encoded multi-agent traffic scene information is then decoded using a Transformer decoder, which introduces learnable anchors for multi-modal movement patterns. The decoder uses self-attention and cross-attention modules to facilitate information interaction between agents (Stage I) and modalities (Stage II), extracting richer semantic information through a feed-forward neural network.

3.2 Anchors Under the Trajectory Prediction Task

Anchors are commonly used in object detection and tracking systems as reference points for predicting future object positions [23–25]. In trajectory prediction, an anchor point is a fixed reference point or motion pattern used to predict future object motion. However, there are challenges with using anchor points [26, 27]. First, the motion patterns represented by anchor points are often specific to certain traffic scenarios, leading to a large number of less probable patterns. Second, this approach usually requires two learning phases, increasing complexity [28, 29]. This paper proposes using learnable embedding vectors as anchors, initialized with the model, to represent motion patterns. To represent motion mode differences, the paper uses standard one-dimensional embedding vectors with position encoding.

The input to the decoder contains two parts; one part is a global embedding vector $\tilde{U} \in R^{N \times F \times D}$ characterizing the traffic scene of the agent and a local embedding vector $\hat{U} \in R^{N \times F \times D}$, which denotes the total number of agents and denotes the embedding vector dimensionality as the number of modes of the motion pattern; the other part is an anchor point representing the motion pattern, which is represented by a set of learnable embedding vectors $M \in R^{N \times F \times D}$.

3.3 Information Interaction Between Multiple Agents

In the first stage, the interactions between multiple agents are modeled. Firstly, the input anchors interact with multimodal information through a self-attention model, as shown in Eq. (1).

$$Q = W^{Q^{s1}} M, K = W^{K^{s1}} M, V = W^{V^{s1}} M, \hat{M} = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (1)$$

where $W^{Q^{s1}}, W^{K^{s1}}, W^{V^{s1}} \in R^{N \times N}$ denotes the linear mapping matrix in self-attention module; $M \in R^{N \times F \times D}$ denotes the learnable embedding vector corresponding to the anchor point; D is the dimension of the embedding vector corresponding to the anchor point; and $\widehat{M} \in R^{N \times F \times D}$ denotes the output embedding vector of self-attention module.

Then, it is also necessary to combine the cross-attention module to achieve interactions between the anchors characterizing the movement patterns and the scene information features of the intelligent body, as shown in Eq. (2).

$$\begin{aligned} U &= \text{MLP}(\widehat{U}, \tilde{U}) \\ Q &= W^{Q^{c1}} \widehat{M}, K = W^{K^{c1}} U, V = W^{V^{c1}} U \\ M &= \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \end{aligned} \quad (2)$$

where $\widehat{U}, \tilde{U} \in R^{N \times F \times D}$ denotes the local embedding vector and global embedding vector of the multi-agent, and the new embedding vector $U \in R^{N \times F \times D}$ is obtained after a dimensionality reduction by the multilayer perceptron module $\text{MLP}(\cdot)$; $W^{Q^{c1}} \in R^{N \times N}$ is the query linear mapping matrix corresponding to the anchor points; $W^{K^{c1}}, W^{V^{c1}} \in R^{N \times N}$ is the key and value linear mapping matrix corresponding to the features of the intelligence body; and $M \in R^{N \times F \times D}$ denotes a new embedding vector after the cross-attention module. Through the above process, and the first stage of information enhancement is realized.

3.4 Information Interaction Between Multimodal Motion Patterns

The second stage aligns motion patterns across multiple agents by leveraging the model's multimodal motion interaction, with data enhancement steps akin to the first stage. The key difference lies in the dimensional transformation of the embedding tensor prior to attention interaction, facilitating information exchange across different dimensions, as outlined in Eq. (3).

$$\begin{aligned} Q &= W^{Q^{s2}} M^T, K = W^{K^{s2}} M^T, V = W^{V^{s2}} M^T \\ \widehat{M}^T &= \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \\ Q &= W^{Q^{c2}} \widehat{M}^T, K = W^{K^{c2}} U^T, V = W^{V^{c2}} U^T \\ M^T &= \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \\ M &= (M^T)^T \end{aligned} \quad (3)$$

where $(\cdot)^T$ denotes the transpose operation; $M^T \in R^{F \times N \times D}$ is the embedding vector corresponding to the anchor point after the transpose; $W^{Q^s2}, W^{K^s2}, W^{V^s2} \in R^{F \times F}$ is the linear mapping matrix in the self-attention module; $U^T \in R^{F \times N \times D}$ is the multi-agent embedding vector after the transpose; $W^{Q^{c2}} \in R^{F \times F}$ is the query linear mapping matrix corresponding to the anchor point; $W^{K^{c2}}, W^{V^{c2}} \in R^{F \times F}$ is the key and value linear mapping matrix corresponding to the intelligent body's features; and ultimately, the new embedding vectors $M \in R^{N \times F \times D}$ are obtained by reverting to the original order of dimensions through another transpose operation.

By employing the Transformer-based decoder structure, this model facilitates information interaction between multiple modalities, enhancing both its performance and generalizability [30]. Additionally, techniques like model pretraining and transfer learning can further bolster the model's performance and data utilization efficiency.

The multimodal joint trajectory prediction decoder's overall structure is depicted in Fig. 2, detailing the interactions between multi-agent information and multimodal information through the aforementioned stages of information enhancement.

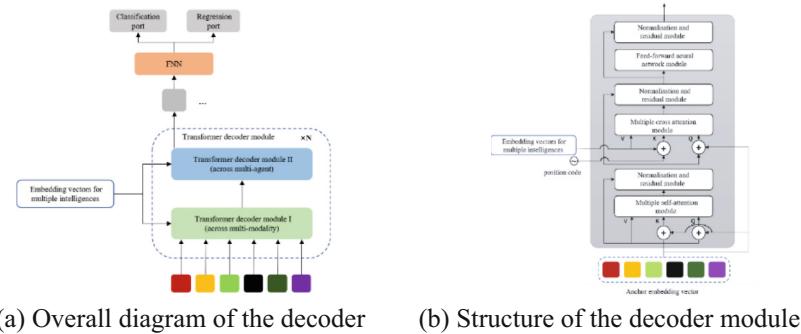


Fig. 2. Structure of the multimodal joint trajectory prediction decoder.

Figure 2 (a) shows the overall structure of the Transformer decoder, which takes in embedding vectors of multiple agents and multimodal anchors. The decoder comprises Decoder I and Decoder II, interacting between agents and modalities. After superposing N decoder modules, agent outputs are generated, and multimodal future trajectories and probabilities are predicted through feed-forward neural networks with regression and classification headers. Fig. 2 (b) details the model structure of Decoder Module I and Decoder Module II, which can be divided into three stages. In the first stage, self-attentive interactions occur with anchors providing query, key, and value. The second stage involves cross-attentive interactions with anchors providing query. The final stage is the feed-forward neural network, which includes linear transformations, nonlinear activations, residual connections, and layer normalization.

4 Experimental Studies

4.1 Experimental Setup

In this section, a series of experiments were conducted to empirically investigate the performance of the proposed method. The experiments utilize the Argoverse trajectory prediction dataset, which is divided into a training and validation set. In addition, the experiments in this chapter are also validated on the Argoverse test set, and since the test set has no real labels, this paper uploads the prediction results to the official server for evaluation, and the evaluation results can be directly displayed in the leaderboard. The model was trained on four NVIDIA TITAN XP 12GB graphics cards after 80 epochs, using Adam as the optimizer, with the batch size, initial learning rate, and probability of weight decay rate set to 32, 3×10^{-4} and 1×10^{-4} , respectively. The cosine annealing function was used to determine the decay of the learning rate, and the hyperparameters related to the model are shown in Table 1.

Table 1. Model parameters.

Parameter	Value	Parameter	Value
num_mode	6	num_global_layers	3
embed_dim	192	num_decoder_layers	3
num_head	8	dropout_rate	0.1
num_temporal_layers	4	$\lambda_{cls}, \lambda_{triplet}$	1.0

4.2 Experimental Steps

Qualitative experimental analysis was conducted to identify model-specific issues, including target object motion patterns, environmental factors, trajectory smoothness, and anomalies. Additionally, quantitative analyses evaluated the trajectory prediction algorithm's quality and application effectiveness, and assessed the benefits of the multitask fusion training strategy and the prediction decoding module.

4.3 Experimental Results

4.3.1 Qualitative Analysis

4.3.1.1 Trajectory Prediction Under Different Tasks

To visually assess the model's trajectory prediction performance across different tasks, this paper selected scenarios depicting multiple agent interactions. Figure 3 illustrates the multitask fusion training strategy's effectiveness.

Figure 3 depicts a complex lane change scenario where a blue car plans to switch lanes. As can be seen from Fig. 3, it demonstrates the model's ability to adjust mode probabilities and predictions under both conditional trajectory prediction (CMP) and

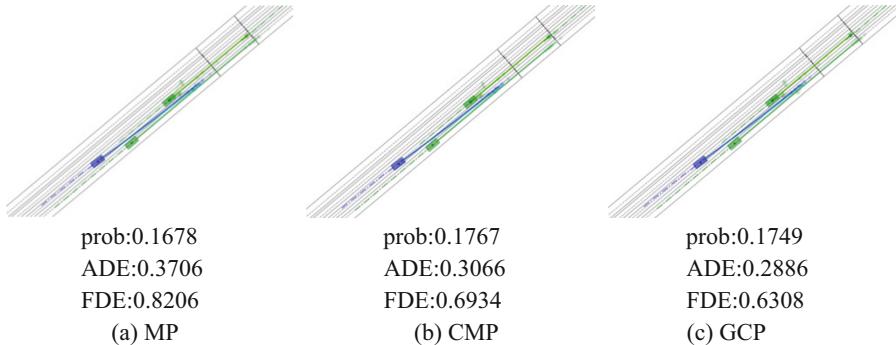


Fig. 3. Visualization of model prediction results under different task settings.(Color Figure Online)

conditional trajectory prediction with known endpoints (GCP) tasks. This adjustment aligns the final predictions more closely with actual trajectories. For example, in the CMP task, the blue car's Average Displacement Error (ADE) and Final Displacement Error (FDE) improved significantly after incorporating the green car's full trajectory.

This demonstrates that when other agents' complete trajectories are known, the model can effectively utilize this information to produce more accurate predictions of joint multi-intelligence trajectories. The model's flexibility in adapting to task requirements through masking without altering its structure makes it highly adaptable.

4.3.1.2 Joint Prediction by Multiple Agents (Multimodal)

Figure 4 presents the supplementary qualitative results across various driving scenarios, including straight roads, curves, d-roads, and intersections. These visualized trajectories provide an intuitive assessment of the model's predictive performance. The visualized trajectories for all modalities are shown, with transparency representing the occurrence probability for each modality (0 to 1). The true trajectories of the attentive (blue) and interactive (green) agents are depicted in light blue and yellow, respectively, and the end-points of the predictions are marked with colored pentagrams.

Experiments indicate that the model accurately predicts trajectories and generates reasonable interactive behaviors. In Fig. 4 (a), the model predicts multigear speeds for a left turn and straight-ahead behavior for the master car. In Fig. 4 (d), considering possible interactions with the slave car, the model predicts two styles for the master car: overtaking by accelerating and changing lanes, or decelerating to avoid. These scenarios illustrate the model's accuracy and generalizability.

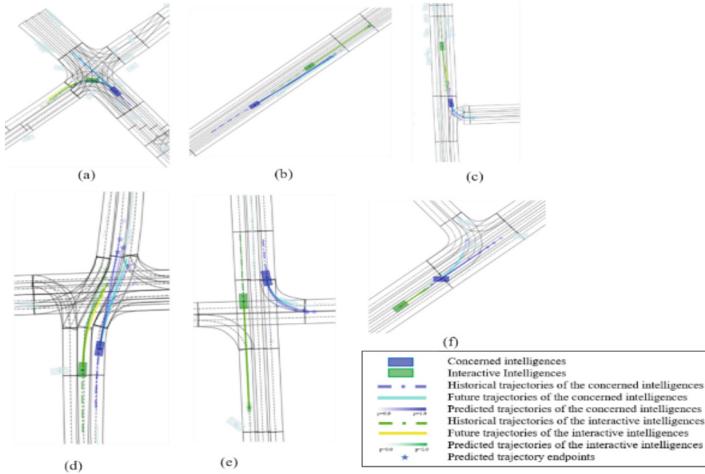


Fig. 4. Visualization results of trajectory predictions for several scenarios. (Color Figure Online)

4.3.2 Quantitative Analysis

To verify the effectiveness of the multitask fusion training strategy, the multimodal joint trajectory prediction encoder design, and the ternary loss function, relevant ablation experiments are conducted, and the results are shown in Table 2.

Table 2. Results of ablation experiments.

Multitask	Multimodal decoder	Triad Loss	minADE	minFDE	MR(%)
			0.691	1.035	10.2
	✓		0.687	1.012	9.89
✓			0.689	1.032	9.96
		✓	0.678	0.998	9.65
	✓	✓	0.666	0.986	9.41
✓		✓	0.652	0.951	9.09
✓	✓		0.654	0.953	8.98
✓	✓	✓	0.651	0.942	8.73

Table 2 shows that the multitask fusion training strategy, multimodal joint trajectory prediction encoder design and ternary loss function proposed in this chapter improved the performance. During the training of the model, the change curves of its loss function computed values are shown in Fig. 5, where the change curves of the loss function on the training and validation sets are the blue and orange curves, respectively.

This paper also visualizes the transformation curves of evaluation metrics on the validation set during the training process, as shown in Fig. 6.

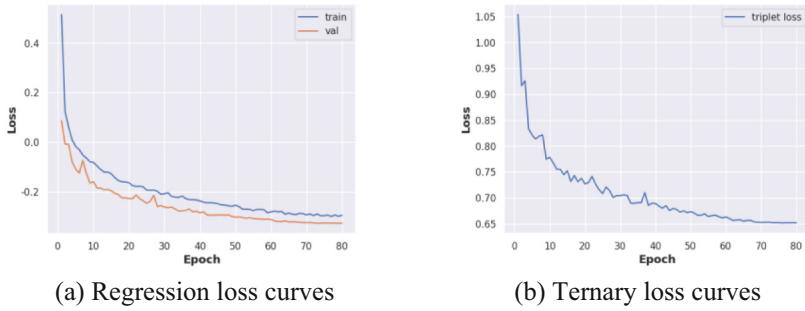


Fig. 5. Loss function variation curves. (Color Figure Online)

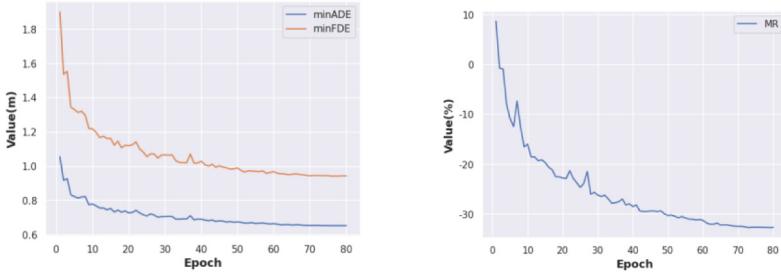


Fig. 6. Change curves of the evaluation metrics for validation set during the training process.

The loss and evaluation curves converge gradually, which indicates that the model performance is stable. Multitask fusion training accelerates convergence due to early training and the simplicity of the conditional trajectory prediction task. The model can utilize other agents' future trajectories to predict the concerned agent's future trajectories more accurately when complete trajectories are known. Experiments show that the model's trajectory prediction is more accurate under CMP and GCP conditions, demonstrating its effective use of future states. Comparison results are presented in Table 3.

Table 3. Multitask trajectory prediction evaluation results.

Task Setting	minADE(m)	minFDE(m)	MR(%)
MP	0.651	0.942	8.73
CMP	0.627	0.877	8.04
GCP	0.635	0.881	8.16

From Table 3, the model's evaluation results in CMP and GCP tasks are enhanced compared to MP, demonstrating that the model's effective use of other agents' future

information for multimodal trajectory predictions. This validates the proposed multitask fusion training strategy. The multitask fusion strategy and ternary loss not only enhance performance but also accelerate model convergence during training. The proposed model effectively resolves the invalid scenes through masking and more realistically simulates agent interactions in the simulated environment.

The model's performance on the Argoverse test set is validated in this paper, with the final comparison results presented in Table 4. The table indicates that the proposed model achieves the best performance in terms of minADE and minFDE metrics, and is comparable in MR to the best model. Consequently, the proposed model demonstrates optimal effectiveness at the current time.

Table 4. Comparison table of model effects.

dataset	methods	minADE(m)	minFDE(m)	MR(%)
Argoverse Test Set	LaneRCNN [1]	0.9038	1.4526	12.32
	LaneGCN [2]	0.8294	1.2595	14.74
	DenseTNT [5]	0.8817	1.2815	12.58
	Method of this paper	0.7896	1.2028	13.31

5 Conclusions

This paper introduces a GNN-based joint trajectory prediction model for multiple agents. Such model includes a multitask fusion training strategy, learnable anchors for multimodal movement patterns, and information interaction between agents and modalities using the Transformer decoder. Richer semantic information is extracted using a feed-forward neural network, and the ternary loss ensures modality diversity. Model validation experiments show the advantages over others. Future work aims to optimize the model's inference speed via techniques like quantization, deployment, and operator optimization to better satisfy real-time system requirements.

Acknowledgments. This work was supported by the National Natural Science Foundation of China under grant 61972302, the Guangzhou Municipal Science and Technology Project under grant SL2022A04J00404, the Key Laboratory of Smart Human-Computer Interaction and Wearable Technology of Shaanxi Province.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Zeng, W., Liang, M., Liao, R., et al.: LaneRCNN: distributed representations for graph-centric motion forecasting. In: 2021 International Conference on Intelligent Robots and Systems, pp. 532–539. IEEE (2021)
2. Liang, M., Yang, B., Hu, R., et al.: Learning lane graph representations for motion forecasting. In: Computer Vision–ECCV 2020: 16th European Conference, vol. 12347, pp. 541–556. Springer International Publishing, Glasgow (2020). https://doi.org/10.1007/978-3-030-58536-5_32
3. Gao, J., Sun, C., Zhao, H., et al.: VectorNet: encoding HD maps and agent dynamics from vectorized representation. In: CVF Conference on Computer Vision and Pattern Recognition, pp. 11525–11533. IEEE (2020)
4. Zhou, Z., Ye, L., Wang, J., et al.: HiVT: hierarchical vector transformer for multi-agent motion prediction. In: CVF Conference on Computer Vision and Pattern Recognition, pp. 8823–8833. IEEE (2022)
5. Gu, J., Sun, C., Zhao, H.: Densentnt: End-to-end trajectory prediction from dense goal sets. In: CVF International Conference on Computer Vision, pp. 15303–15312. IEEE (2021)
6. Helbing, D., Molnar, P.: Social force model for pedestrian dynamics. *Phys. Rev. E* **51**(5), 4282 (1995)
7. Yamaguchi, K., Berg, A.C., Ortiz, L.E., et al.: Who are you with and where are you going? In: CVPR 2011, pp. 1345–1352. IEEE (2011)
8. Gupta, A., Johnson, J., Fei-Fei, L., et al.: Social GAN: socially acceptable trajectories with generative adversarial networks. In: Conference on Computer Vision and Pattern Recognition, pp. 2255–2264. IEEE (2018)
9. Salzmann, T., Ivanovic, B., Chakravarty, P., Pavone, M.: Trajectron++: dynamically-feasible trajectory forecasting with heterogeneous data. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12363, pp. 683–700. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58523-5_40
10. Kosaraju, V., Sadeghian, A., Martín-Martín, R., et al.: Social-bigat: multimodal trajectory forecasting using bicycle-GAN and graph attention networks. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
11. Shen, G., Li, P., Chen, Z., Yang, Y., Kong, X.: Spatio-temporal interactive graph convolution network for vehicle trajectory prediction. *Internet Things* **24**, 100935 (2023)
12. Jia, X., Wu, P., Chen, L., Liu, Y., Li, H., Yan, J.: HDGT: Heterogeneous Driving Graph Transformer for multi-agent trajectory prediction via scene encoding. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**(11), 13860–13875 (2023)
13. Chen, K., Zhu, H., Tang, D., Zheng, K.: Future pedestrian location prediction in first-person videos for autonomous vehicles and social robots. *Image Vision Comput.* **134**, 104671 (2023)
14. Choudhury, A., Ghose, M., Islam, A.: Machine learning-based computation offloading in multi-access edge computing: A survey. *J. Syst. Architect.* **16**, 103090 (2024)
15. Qian, J., Zhang, L., Huang, Q., Liu, X., Xing, X., Li, X.: A self-driving solution for resource-constrained autonomous vehicles in parked areas. *High-Confidence Comput.* **4**, 100182 (2024)
16. Charroud, A., El Moutaouakil, K., Palade, V., Yahyaoui, A., Onyekpe, U., Eyo, E.U.: Localization and mapping for self-driving vehicles: a survey. *Machines* **12**(2), 118 (2024)
17. Li, D., Zhang, Q., Lu, S., Pan, Y., Zhao, D.: Conditional goal-oriented trajectory prediction for interacting vehicles. *IEEE Trans. Neural Networks and Learn. Syst.* (2023)
18. Shen, C., Xiao, X., Li, S., Tong, Y.: Improved attention mechanism for human-like intelligent vehicle trajectory prediction. *Electronics* **12**(19), 3993 (2023)
19. Huang, R., Zhuo, G., Xiong, L., Lu, S., Tian, W.: A review of deep learning-based vehicle motion prediction for autonomous driving. *Sustainability* **15**(20), 14716 (2023)

20. Li, H., Ren, Y., Li, K., Chao, W.: Trajectory prediction with attention-based spatial-temporal graph convolutional networks for autonomous driving. *Appl. Sci.* **13**(23), 12580 (2023)
21. Chen, G., Gao, Z., Hua, M., Shuai, B., Gao, Z.: Lane change trajectory prediction considering driving style uncertainty for autonomous vehicles. *Mech. Syst. Signal Process.* **206**, 110854 (2024)
22. Cheng, H., Liu, M., Chen, L., Broszio, H., Sester, M., Yang, M.Y.: GATraj: a graph- and attention-based multi-agent trajectory prediction model. *ISPRS J. Photogrammetry Remote Sens.* **205**, 163–175 (2023)
23. Mi, J., Zhang, X., Zeng, H., Wang, L.: DERGCN: Dynamic-Evolving Graph Convolutional Networks for human trajectory prediction. *Neurocomputing* **567**, 127117 (2024)
24. Erik, S., Fabian, B.F.: A review of trajectory prediction methods for the vulnerable road user. *Robotics* **13**(1), 1 (2023). <https://doi.org/10.3390/robotics13010001>
25. Schreier, M.: Bayesian environment representation, prediction, and criticality assessment for driver assistance systems. *at-Automatisierungstechnik* **65**(2), 151–152 (2017)
26. Haitao, M., Xiaoyong, X., Pengyu, W., Zhaopu, Z.: A hierarchical LSTM-based vehicle trajectory prediction method considering interaction information. *Automot. Innov.* **1**, 71–81 (2024)
27. Zhou, H., Zhao, T., Fang, Y., Liu, Q.: A trajectory prediction method based on graph attention mechanism. *Appl. Math. Nonlinear Sci.* **9**(1) (2024)
28. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) *ECCV 2020. LNCS*, vol. 12346, pp. 213–229. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58452-8_13
29. Vibha, B., Neetesh, K.: Machine learning for autonomous vehicle's trajectory prediction: a comprehensive survey, challenges, and future research directions. *Veh. Commun.* **46**, 100733 (2024). <https://doi.org/10.1016/j.vehcom.2024.100733>
30. Wang, J., Sang, H., Chen, W., Zhao, Z.: VOSTN: Variational One-shot Transformer Network for pedestrian trajectory prediction. *Phys. Scr.* **99**(2), 026002 (2024)



xNN-SF: An Explainable Neural Network Inspired by Stochastic Frontier Model

Shuangxue Zhao^{1,2(✉)}

¹ College of Computer Science and Technology, Zhejiang University, Hangzhou, China
zhaosx@zju.edu.cn

² Bank of Hangzhou Co., Ltd., Hangzhou, China

Abstract. Neural networks have emerged as a strong competitor to traditional regression and statistical models. Although neural networks exhibit strong predictive performance, they lack model interpretability. In this paper, we propose an explainable neural network inspired by stochastic frontier model (xNN-SF) aiming to illuminate the black box and strike a balance between prediction accuracy and model interpretability. Precisely, the input features are projected into three directions and then fed into three neural subnetworks. Each subnetwork extracts a different influence: the main influence, the negative influence, and the variance perturbation. To promote interpretability, the former two subnetworks are constrained to be monotonic. Unlike the other two, the third subsystem comprises an ensemble of multiple interlaced neural networks rather than a solitary one, with each constituent net sequentially trained via a boosting procedure. The outputs are obtained by weighting the results from the three subnetworks. We develop a simple implementation procedure and evaluate our proposed methodology against multiple benchmarks across various real-world datasets. The results demonstrate that our model maintains competitive prediction performance while also enhancing interpretability.

Keywords: Explainable Neural Network · Monotonicity · Identifiability · Boosting

1 Introduction

Interpreting and explaining the predictions of machine learning models is crucial, as it contributes to improving their openness, convincingness, efficacy, and credibility. However, the pursuit of model accuracy often compromises interpretability. Consequently, interest in transparent yet high-performing models has surged recently.

Deeply studied in academia and widely used in various fields, neural networks (NNs) are complex models that estimate potential relationships between inputs and outcomes. However, due to the complexity of the internal structure of black-box models, it is not clear how the input data is used to obtain the output. Traditional statistical methods such as Generalized Additive Model (GAM) [4] perform less well than neural networks for prediction, but have clear explanatory properties. This has led to a surge of work combining neural networks with traditional statistical methods to get the best of both worlds.

Among them, combining neural networks with semiparametric statistical models, such as Generalized Additive Models with Structured Interactions (GAMI-Net) [14] and additive index models Explainable Neural Network (xNN) [10], and Enhanced Explainable Neural Network (ExNN) [13], can be flexible and thus achieve good prediction results, as well as intrinsically interpretable. However, GAMI-Net has limitations in terms of identifiability between any interaction component and its nested main effects. xNN and ExNN utilize multiple subnetworks, making them difficult to explain, especially with high-dimensional features. Using multiple NNs also risks overfitting, requires extensive hyperparameter tuning, and incurs high computational training costs. Additionally, since prediction accuracy and model interpretability usually conflict with each other, there is still room for existing models to improve interpretability. In this paper, inspired by the well-known stochastic frontier model in economics, we further simplify the model structure and enhance the interpretability based on the existing interpretable neural network models, and propose xNN-SF, whose architecture is presented in Fig. 1.

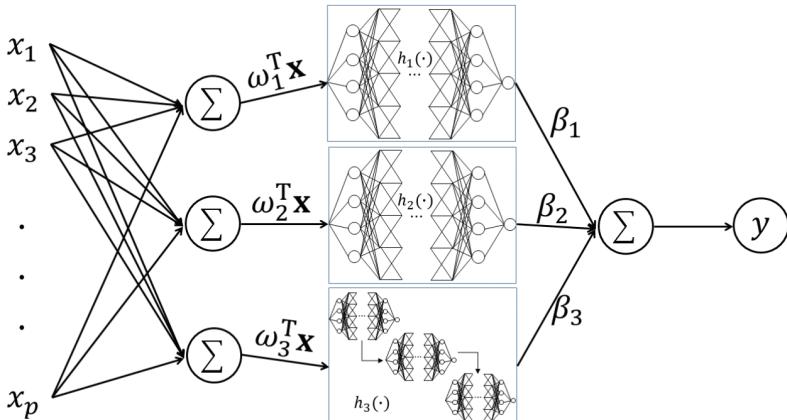


Fig. 1. xNN-SF architecture.

Stochastic frontier models [1, 7] are econometric models used to analyze production, cost, and profit efficiency of firms, industries, or economies. The key idea is that there is a frontier or potential maximum output, but random shocks and inefficiencies may cause the actual output to fall below the frontier. Inspired by the three components (main influence, negative influence, and variance perturbation) of these models, we use three neural networks as subnetworks to extract information from each component separately. This structure supervises the projection direction of high-dimensional input features without determining the number of projections. Additionally, it enables feature grouping across subnetworks with varying importance, facilitating interpretation of each group's contribution based on the subnetwork's meaning.

Furthermore, in order to explain the effect of each input feature on the outcome, this paper adds a monotonicity constraint [8, 9] on the first two subnetworks. Monotonicity constraint is an important consideration when developing models, which enforces that the relationship between two variables moves in only one direction, either increasing or

decreasing. For example, for lending industries that are regulated and require unprejudiced and equitable decision-making, monotonicity constraints ensure increasing risk factors heighten estimated risk. Therefore, incorporating monotonicity constraints into models can improve interpretability. In Fig. 1, if a feature is projected with positive weight into a monotonically increasing subnetwork, then the effect of this feature on this part of the output is monotonically increasing.

The third subnetwork extracts information about the variance perturbation and therefore does not require monotonicity assumptions. In addition, in order to avoid the loss caused by the strong monotonicity assumption of the first two subnetworks and the small number of subnetworks, the third subnetwork incorporates boosting by using an ensemble of neural networks, rather than a single network. Each neural net in the ensemble is trained sequentially, with the training data for each successive model focused on the errors from the previous model. This boosting procedure enables the subnetwork to better capture noise and outliers.

The advantages of xNN-SF being both explanatory and predictive are demonstrated in several experiments in Sect. 3. Furthermore, the model shows promise as a surrogate for complex data analysis problems across various fields, including environmental science and bioinformatics [11, 12]. The main contributions of this paper are summarized as follows:

- We propose a novel explainable neural network which significantly improves the interpretability compared to existing models. The main influence, negative influence and random fluctuation of input features are extracted respectively from the three projection directions, so the interpretation is clearer. The monotone subnetworks allow the effect of features on results to be shown more directly. Coupled with the sign of the projection weights, the positive or negative impact of each feature on the output is completely interpretable.
- The model's framework ensures good predictions through its concise and well-chosen assumptions. Using three projection directions and three corresponding subnetworks avoids the tricky issue of selecting the optimal number of each. This streamlined structure reduces parameters and prevents overfitting. Furthermore, the boosting procedure enables the third subnetwork to compensate for limitations arising from the small number of subnetworks.
- In order to ensure the identifiability of the model, this paper proposes targeted constraints and implements them in the algorithm.

2 Methodology

2.1 Architecture of xNN-SF

The proposed approach is formulated as

$$\begin{aligned} g(\mathbb{E}(y|\mathbf{x})) &= \mu + \beta_1 h_1(\mathbf{w}_1^\top \mathbf{x}) + \beta_2 h_2(\mathbf{w}_2^\top \mathbf{x}), \\ \text{Var}(y|\mathbf{x}) &= \beta_3 h_3(\mathbf{w}_3^\top \mathbf{x}), \end{aligned} \quad (1)$$

where $\beta_1 > 0$, $\beta_2 < 0$, $\mathbb{E}(\cdot)$ is the expectation, and $\text{Var}(\cdot)$ is the variance. If the outcome is continuous, the model can be simplified as

$$y_i = \mu + \beta_1 h_1(\mathbf{w}_1^\top \mathbf{x}_i) + \beta_2 h_2(\mathbf{w}_2^\top \mathbf{x}_i) + \beta_3 h_3(\mathbf{w}_3^\top \mathbf{x}_i), \quad (2)$$

which contains three subnetworks $h_j(\cdot)$, $j = 1, 2, 3$, and it is assumed that $h_1(\cdot)$ and $h_2(\cdot)$ are monotone, and $\beta_{3i} \sim N(0, 1)$. These three weighted subnetworks correspond to the main effect, the negative effect and the variance perturbation – the three parts of the following stochastic frontier model:

$$y_i = f(\mathbf{x}_i; \mathbf{w}) + \alpha_i + \varepsilon_i, \alpha_i \leq 0, \quad (3)$$

where \mathbf{w} is an unknown vector, ε_i represents a noise that follows a normal distribution, and α_i is an unobservable random variable that represents firm-specific technical inefficiency. Notice that there is no need to make a monotonicity assumption on $h_3(\cdot)$ because it represents a fit to the variance term, which is usually assumed to be normally distributed.

On one hand the method proposed in this paper limits the number of subnetworks to restrain model complexity by drawing inspiration from stochastic frontier model, and on the other hand it constrains the shape of the subnetworks to promote interpretability. By imposing these dual limitations, the approach aims to enhance explainability while preserving strong predictive performance.

In the past, GAMs were typically fitted using iterative backfitting with smooth low-order splines, which can be analytically fit and reduce overfitting. However, traditional statistical estimation methods require distribution assumptions, thus losing some flexibility. In this paper, after writing the objective function of the model into optimization form, a backpropagation procedure is applied to optimize multiple unknown parameters using minibatch gradient descent. To address the issue of inductive bias and underfitting that may arise from using a small number of subnetworks, we implement a gradual fitting of the residuals from the previous step through a neural network in the third subnetwork, drawing on the concept of boosting.

2.2 Identifiability

Identification of Parameters. As with most articles on index model [6, 15], we assume that $\|\mathbf{w}_j\|_2 = 1$ for $j = 1, 2, 3$, the first non-zero entry for \mathbf{w}_j is positive for every j , and the matrix \mathbf{w} consisting of three columns \mathbf{w}_j has full column rank 3. [13] imposed the orthogonality constraint on the projection indices such that $\mathbf{w}^\top \mathbf{w} = I$. However, according to [2], only if $h_j(\cdot)$ is linear, it is necessary to assume that $\mathbf{w}_j^\top \mathbf{w}_{j'} = 0$ for every $j \neq j'$, thus we can relax the assumption of ExNN by [13].

Identification of Functions. As stated in [6], two functions $f(x)$ and $g(x)$ cannot be identical if we can find two points x_1 and x_2 such that $f(x_1) = f(x_2)$ while $g(x_1) = g(x_2)$.

Chiou and Müller [3] given the identifiability of additive index model (AIM) model under conditions that are not only very restrictive, for example, the component functions have to be monotone. If $\beta_1 = \beta_2 = \beta_3 = 1$, then model Eq. (2) degenerates to AIM, hence, following [3], we need the same technical assumption that the first two link functions $h_j(\cdot)$, $j = 1, 2$ are strictly monotone. In addition, with reference to the usual assumptions of GAMs with unknown link function [5], to make model Eq. (1) identifiable, we assume that for $j = 1, 2, 3$,

$$\begin{aligned} \mathbb{E}(h_j(\cdot)) &= \int h_j(z) dF_j(z) = 0, \\ \text{Var}(h_j(\cdot)) &= \int h_j(z)^2 dF_j(z) = 1. \end{aligned} \quad (4)$$

Identification of Monotonic Networks. [8] introduced a new method to universally approximate Lipschitz functions that are monotonic in any subset of their inputs and overcomes the drawbacks of existing methods: lack of expressiveness and impractical complexity. Following [8], we define the Lip^1 constrained monotonic network.

Definition 1 (Lip^1 function). $g : \mathbb{R}^d \rightarrow \mathbb{R}^n$ is Lip^1 if it is Lipschitz continuous with respect to the L^1 norm in every output dimension, i.e.,

$$\|g(\mathbf{x}) - g(\mathbf{y})\|_\infty \leq \lambda \|\mathbf{x} - \mathbf{y}\|_1 \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n.$$

Recursively define the layer l of the fully connected network of depth D with activation σ as

$$\mathbf{z}^l = \sigma(\mathbf{z}^{l-1}) \mathbf{W}^l + \mathbf{b}^l, \quad (5)$$

then the constraint for realizing Lip^1 constrained monotonic network is

$$\|\mathbf{W}^1\|_{1,\infty} \cdot \prod_{i=2}^D \|\mathbf{W}^i\|_\infty \leq t_1, \quad (6)$$

where matrix norm $\|\mathbf{W}\|_\infty = \max_i \sum_j |w_{ij}|$, and $\|\cdot\|_{1,\infty}$ equals the maximum absolute value of an element in the matrix.

Combining the above identifying conditions, Eq. (1) and Eq. (2) are identifiable.

2.3 Interpretability of xNN-SF

Effect of Feature on Outcome. Sparsity is introduced to select important features and enhance the interpretability of the model by the following L^1 norm constraints

$$\sum_{j=1}^3 \|\mathbf{w}_j\|_1 \leq t_2, \quad (7)$$

where t_2 indicates a threshold. After feature selection, the positivity or negativity of the weights of the important features at projection and the monotonicity of the subnetwork determine the positive or negative impact of the features on the results. Note that we make monotonicity assumptions for the first two subnetworks, but whether they are monotonically increasing or decreasing is determined by the data-driven approach. Coupled with the fact that the positivity, negativity and magnitude of the projection weights are also determined in a data-driven manner, the positive or negative impact of each feature on the results is completely unconstrained by assumptions. In practical problems, the influence of a feature on the outcome is not necessarily completely monotonic. In the setting of this paper, if the influence of a feature on the outcome embodied in the first and second subnetworks is in the opposite direction, then this can also reflect the non-monotonic influence of the feature on the outcome, which reflects that the model's assumptions are not overly strong and thus flexible.

Shape and Meaning of Subnetworks. Borrowing ideas from the stochastic frontier model, each subnetwork extracts a different influence: the main influence, the negative influence, and the variance perturbation. With the monotonic constraint, the shape of subnetwork is concise and intuitive. Except the constraint Eq. (6), we add the following functional roughness penalty

$$\int \left[h_j''(z) \right]^2 dF_j(z) \leq t_3, \quad (8)$$

which bounds the integrated squared second-order derivative, is a popular way to enforce smoothness. The monotonic constraint and smoothness constraint help improve interpretability.

2.4 Computational Procedure

Denote the unknown parameters and functions by

$$\Theta := \{\mu, \beta_1, \beta_2, \mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3, h_1(\cdot), h_2(\cdot), h_3(\cdot)\},$$

the training task for xNN-SF consists of minimizing the following regularized loss function

$$\begin{aligned} L(\Theta) = & l(\Theta) + \lambda_1 \sum_{j=1}^3 \|\mathbf{w}_j\|_1 + \lambda_2 \sum_{j=1}^2 \left(\left\| \mathbf{W}_j^1 \right\|_{1,\infty} \cdot \prod_{i=2}^D \left\| \mathbf{W}_j^i \right\|_\infty \right) \\ & + \lambda_3 \int \left[h_j''(z) \right]^2 dF_j(z) \\ \text{s.t. } & \beta_1 > 0, \beta_2 < 0, \|\mathbf{w}_j\|_2 = 1, \\ & \int h_j(z)^2 dF_j(z) = 0, \int h_j(z)^2 dF_j(z) = 1, \end{aligned} \quad (9)$$

where λ_1 , λ_2 , and λ_3 are tuning parameters. The specific form of the empirical loss function $l(\Theta)$ depends on the type of machine learning task being performed, such as regression or classification. For regression, the squared loss $l(\Theta) = (y - \hat{y})^2$ is used, whereas for binary classification tasks, the logistic loss $l(\Theta) = \log(1 + \exp(-y\hat{y}))$ is used.

We optimize the above problem Eq. (9) to estimate the multiple unknown parameters simultaneously through backpropagation. All unknown functions are approximated by feedforward neural networks with finite parameters.

Initialization. The three initial projection directions of the input features are given by the projection pursuit regression (PPR). Define the neural network architecture with D layers by Eq. (5), where each layer l has n^l neurons. Initialize the weights W^l and biases b^l randomly. All weights are constrained to be positive for monotonically increasing, and negative weights correspond to monotonically decreasing. With positive weight constraints, $W^l = \exp(v)$ can be imposed [9].

Iteration Procedure. For two monotone subnetworks, the monotone pattern has four cases forming the following set

$$\mathcal{M} = \{(1, 1), (1, -1), (-1, 1), (-1, -1)\}, \quad (10)$$

where 1 indicates monotonically increasing, and -1 indicates monotonically decreasing. We use cross-validation to select a monotonic pattern with the best predictions. After determining the monotonic pattern, the minibatch gradient descent begins. First, sample a batch of m training examples $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_m, y_m)\}$. Perform forward pass to compute the three subnetworks, where the first two monotonic subnetworks use the method of [8] and the third subnetwork uses a neural network at each step to fit the residuals from the previous step. Then, compute the loss function Eq. (9) and update the parameters using backpropagation. Finally, repeat for multiple epochs over the full training set.

Hyperparameter Settings. The sparsity hyperparameter λ_1 , monotonicity hyperparameter λ_2 , and smoothness hyperparameter λ_3 can be determined by parallel cross-validation, and we recommend a default value of 10^{-3} . Set the learning rate $\eta = 10^{-3}$, and choose proper mini-batch size m , number of epochs E , and early stopping threshold s . The activation function is fixed to be hyperbolic tangent activation $tanh$ [13].

2.5 Discussion on the Weights of Subnetworks

The setting of the weight $\beta_1 > 0$ signifies that the weighted first subnetwork captures the main influence on the output. $\beta_2 < 0$ implies that the weighted second subnetwork accounts for negative influences. Finally, $\beta_3 \sim N(0, 1)$ corresponds to the weighted third subnetwork, which introduces variance perturbations to the model. That is, β_1 and β_2 are deterministic parameters that need to be estimated, whereas β_3 is random and is drawn from the standard normal distribution in the computational procedure. Such a weighting approach is different from ExNN which simply normalizes the weights as the signed scales for each subnetwork. The computational process begins by randomly drawing initial values for β_1 , β_2 , and β_3 from the uniform distributions $U(0, 1)$, $U[-1, 0]$, and the standard normal distribution $N(0, 1)$, respectively. These initial values are then updated through backpropagation during the training process.

3 Experiments

3.1 Baseline Algorithms

Similar to [14], we choose several black-box models and interpretable models as benchmark models, the former includes extreme gradient boosting (XGBoost), random forest (RF), and multi-layer perceptron (MLP), and the latter includes generalized linear model (GLM), GAM, explainable boosting machine (EBM), GAMI-Net, xNN, and ExNN.

3.2 Performance Metrics

We use Root Mean Squared Error (RMSE) and Area Under the ROC Curve (AUC) as metrics for regression and classification models, respectively. In Table 1 and Table 2, the mean and standard deviation of RMSE and AUC are obtained by repeating the experiment ten times on each dataset. To ensure a fair comparison, all models are trained, validated, and tested on the same sets of data for the same number of repetitions, which corresponds to the three columns of the table.

3.3 Datasets

Classification: Bank Marketing Dataset. This is a binary classification problem (<https://archive.ics.uci.edu/dataset/222/bank+marketing>), where the classification objective is to predict whether the client will subscribe to a term deposit based on attributes like age, job, marital status, education level, average yearly balance, housing loan, personal loan, and more. In total, the dataset contains 45,211 observations with 16 attributes including 7 numeric variables and 9 categorical variables. 10% of the observations subscribed to a term deposit.

The visualized subnetworks and projection indices of xNN-SF in Fig. 2 show the main influence as a monotonically increasing curve and the negative influence as a monotonically decreasing curve. Positive weights on the first projection indicate a monotonically increasing effect on the outcome through the first subnetwork, while positive weights on the second projection also have a monotonically increasing effect due to the negative weight of the second subnetwork and its decreasing curve. For example, the feature x_{11} with the largest coefficient in the first projection direction is the campaign, i.e., number of contacts performed during this campaign and for this client, implying a higher probability of subscribing to a term deposit with more contacts. Unlike xNN and ExNN, whose curves lack monotonicity constraints, preventing visualization of positive or negative feature impacts, xNN-SF enables this interpretation. Additionally, zero coefficients in Fig. 2 identify redundant features that do not contribute to that subnetwork. Table 1 shows the performance measured by AUC of xNN-SF and its competitors. The best performers in each column are bolded. The AUC of xNN-SF on the test set narrowly beats GAM-Net and XGBoost, among others.

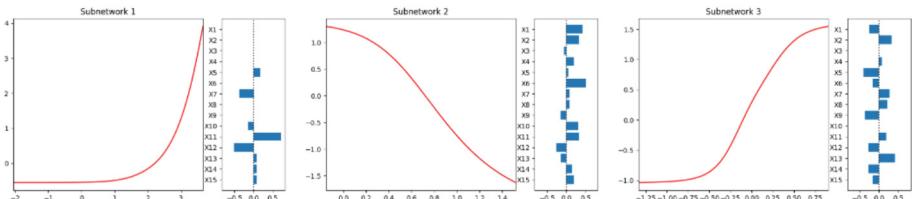
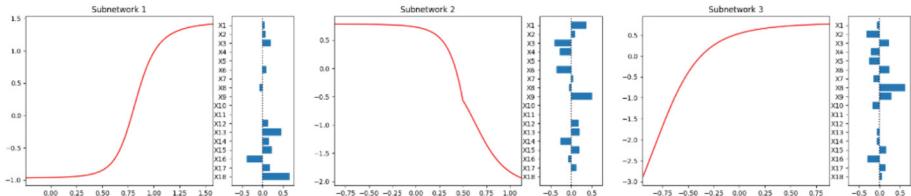


Fig. 2. The estimated subnetworks and projection indices for the bank marketing dataset.

Regression: Bike Sharing Hour Dataset. The bike sharing dataset (<https://archive.ics.uci.edu/dataset/275/bike+sharing+dataset>) is a regression task to predict bike rental demand based on contextual factors. It has 17,379 rows and 16 columns describing the rentals by hour. Attributes include weather data (temperature, humidity, windspeed), seasonal indicators, holiday indicators, and more. The target variable is the count of bike rentals.

Table 1. Comparison results of the bank marketing dataset.

Model	Train AUC (%)	Val-AUC (%)	Test AUC (%)
XGBoost	97.38 ± 0.09	93.24 ± 0.52	93.01 ± 0.26
RF	93.59 ± 0.10	92.01 ± 0.62	91.98 ± 0.35
MLP	93.15 ± 0.65	93.12 ± 0.90	92.29 ± 0.41
GLM	90.81 ± 0.21	90.77 ± 0.47	90.60 ± 0.30
pyGAM	91.90 ± 0.17	91.68 ± 0.39	91.53 ± 0.36
EBM	94.59 ± 0.16	94.63 ± 0.40	93.16 ± 0.31
GAMI-Net	94.12 ± 0.15	93.44 ± 0.41	93.28 ± 0.38
xNN	93.27 ± 0.18	93.05 ± 0.49	92.86 ± 0.45
ExNN	93.90 ± 0.16	93.22 ± 0.43	93.02 ± 0.41
xNN-SF	94.63 ± 0.14	93.84 ± 0.38	93.31 ± 0.35

**Fig. 3.** The estimated subnetworks and projection indices for the bike sharing hour dataset.**Table 2.** Comparison results of the bike sharing hour dataset.

Model	Train RMSE	Val-RMSE	Test RMSE
XGBoost	5.64 ± 4.52	40.96 ± 1.17	42.33 ± 0.72
RF	62.00 ± 0.72	65.12 ± 1.47	65.69 ± 2.10
MLP	35.54 ± 1.31	37.09 ± 1.52	43.35 ± 1.32
GLM	158.98 ± 0.52	158.54 ± 2.49	159.09 ± 1.57
pyGAM	99.52 ± 0.60	100.16 ± 1.61	100.22 ± 1.12
EBM	54.32 ± 0.55	54.27 ± 0.99	57.48 ± 1.20
GAMI-Net	49.83 ± 0.80	52.95 ± 1.39	53.24 ± 0.81
xNN	53.77 ± 0.93	54.92 ± 1.53	55.36 ± 1.67
ExNN	47.83 ± 0.91	48.95 ± 1.39	49.58 ± 1.08
xNN-SF	42.05 ± 0.77	42.19 ± 0.83	42.65 ± 0.79

Figure 3 shows the estimated subnetworks and projection indices of xNN-SF, where the main influence is a monotonically increasing curve, and the negative influence is a

monotonically decreasing curve. A feature with a negative weight on the first projection monotonically decreases the outcome through the first subnetwork, while a positive weight on the second projection leads to a monotonic increase in outcome due to the negative weight of the second subnetwork and its decreasing curve. For example, the feature x_{16} with the negative coefficient in the first projection direction is the humidity, implying that higher humidity leads to shorter rides, consistent with people riding less in rain and snow. Table 2 presents the performance of xNN-SF and its competitors measured by RMSE. xNN-SF has a slightly worse RMSE than XGBoost on the test set but outperforms the interpretable neural networks in the table. Additionally, xNN-SF has a smaller standard deviation, indicating better model stability.

More Datasets. To further validate the predictive power of xNN-SF, comparative experiments were conducted between it and the benchmark methods across ten public datasets from the OpenML platform or the UCI machine learning repository. As evidenced in Table 3¹, xNN-SF produced smaller prediction errors than the benchmark on six of the ten datasets, indicating it is highly competitive overall, especially since xNN-SF also possesses the stronger explanatory power described earlier.

Table 3. Test set RMSE on 10 real-world regression datasets.

Data	XGBoost	RF	MLP	GLM	pyGAM	EBM	GAMI-Net	xNN	ExNN	xNN-SF
1	2.49±0.25	2.42±0.32	2.37±0.28	8.97±0.74	2.11±0.26	2.51±0.40	2.20±0.27	2.23±0.29	2.21±0.27	2.10±0.26
2	6.14±0.25	5.92±0.21	6.20±0.17	7.47±0.18	6.25±0.13	5.99±0.23	6.26±0.15	6.18±0.27	5.96±0.24	5.94±0.22
3	1.00±0.02	0.93±0.06	1.02±0.08	1.20±0.02	1.15±0.53	0.92±0.03	0.97±0.03	1.01±0.05	0.99±0.04	0.96±0.03
4	0.20±0.01	0.32±0.01	0.58±0.03	1.06±0.02	0.77±0.04	0.41±0.01	0.38±0.04	0.41±0.03	0.38±0.02	0.19±0.02
5	3.21±0.10	3.26±0.07	3.05±0.07	4.59±0.06	3.07±0.06	3.09±0.06	3.05±0.07	3.09±0.09	3.06±0.07	3.03±0.06
6	0.29±0.02	0.31±0.01	0.31±0.07	1.46±0.05	0.33±0.01	0.29±0.01	0.29±0.01	0.31±0.02	0.30±0.01	0.29±0.01
7	3.07±0.30	2.86±0.32	2.89±0.31	2.94±0.32	3.05±0.34	2.87±0.31	2.88±0.32	2.91±0.35	2.89±0.34	2.87±0.32
8	0.99±0.02	1.45±0.03	0.66±0.02	2.89±0.05	1.72±0.02	0.95±0.02	0.94±0.02	0.95±0.02	0.81±0.02	0.66±0.02
9	1.65±0.05	1.68±0.04	1.59±0.03	3.38±0.09	1.69±0.05	1.66±0.05	1.66±0.05	1.68±0.06	1.64±0.05	1.58±0.04
10	2.17±0.04	3.15±0.06	2.11±0.07	6.71±0.17	2.39±0.05	2.28±0.09	2.22±0.04	2.22±0.07	2.17±0.06	2.10±0.06

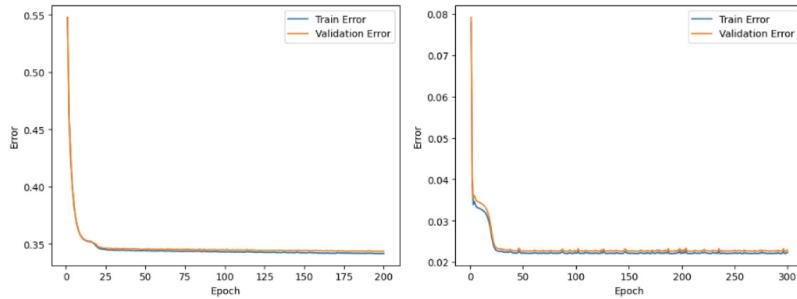
Data	Name	n	p	Scale
1	treasury	1049	15	×0.1
2	wine red	1599	11	×0.1
3	skill craft	3395	18	×1
4	parkinsons tele	5875	19	×10
5	wind	6574	14	×1
6	cpu small	8192	12	×10
7	topo 2 1	8885	266	×0.01
8	electrical grid	10000	11	×0.01
9	ailerons	13750	40	×0.0001
10	elevators	16599	18	×0.001

3.4 Summary of Results

We train a single model for 200 or 300 epochs, capturing its full parameters after each epoch, and plot this trajectory in Fig. 4 showing the training and validation losses of

¹ Because we keep it to two decimal places, some numbers that appear to be equal are actually not, and the bold is the actual minimum value in each row.

xNN-SF on two datasets. As can be seen, the losses first decrease significantly and then stabilize at a lower level for both datasets, which demonstrates the fast convergence rate of the algorithm.



The bank marketing dataset.

The bike sharing hour dataset.

Fig. 4. Comparison of loss path diagrams of training and validation set.

According to Table 1, Table 2 and Table 3, the proposed xNN-SF model achieves a balance between high accuracy and interpretability, outperforming other models that either lack accuracy (GLM, GAM, EBM) or interpretability (XGBoost, RF, MLP). While interpretable neural networks like xNN, ExNN, and GAMI-Net are comparable, they are slightly less accurate and explanatory. Additionally, xNN-SF demonstrates relatively high stability with a small standard deviation, attributed to its specialized modeling of variance in the third subnetwork.

4 Conclusion

In this paper, we propose a novel neural network architecture called xNN-SF that balances prediction accuracy and model interpretability. Experiments on multiple real-world datasets demonstrate that xNN-SF maintains competitive predictive performance compared to benchmark models, while also improving interpretability by decomposing influence components. The monotonicity constraints on the subnetworks make the model more explainable. Overall, xNN-SF represents a promising step towards accurate yet interpretable deep learning models.

Disclosure of Interests. The author has no competing interests to declare that are relevant to the content of this article.

References

1. Aigner, D., Lovell, C.A.K., Schmidt, P.: Formulation and estimation of stochastic frontier production function models. *J. Econ.* **6**(1), 21–37 (1977)

2. Chen, Y., Samworth, R.J.: Generalized additive and index models with shape constraints. *J. R. Stat. Soc. Ser. B Stat Methodol.* **78**(4), 729–754 (2016)
3. Chiou, J.M., Müller, H.G.: Quasi-likelihood regression with multiple indices and smooth link and variance functions. *Scand. J. Stat.* **31**(3), 367–386 (2004)
4. Hastie, T.J., Tibshirani, R.J.: Generalized Additive Models, vol. 43. CRC press (1990)
5. Horowitz, J.L., Mammen, E.: Rate-optimal estimation for a general class of nonparametric regression models with unknown link functions. *Ann. Stat.* **35**(6), 2589–2619 (2007)
6. Lin, W., Kulasekera, K.: Identifiability of single-index models and additive-index models. *Biometrika* **94**(2), 496–501 (2007)
7. Meeusen, W., van Den~Broeck, J.: Efficiency estimation from cobb-douglas production functions with composed error. *Int. Econ. Rev.*, 435–444 (1977)
8. Nolte, N., Kitouni, O., Williams, M.: Expressive monotonic neural networks. In: The Eleventh International Conference on Learning Representations (2023)
9. Sill, J.: Monotonic networks. In: Advances in Neural Information Processing Systems, vol. **10** (1997)
10. Vaughan, J., Sudjianto, A., Brahimi, E., Chen, J., Nair, V.N.: Explainable neural networks based on additive index models. *RMA J.* **101**(2), 40–49 (2018)
11. Wang, L., Li, Z.W., Hu, J., Wong, L., Zhao, B.W., You, Z.H.: A PiRNA-disease association model incorporating sequence multi-source information with graph convolutional networks. *Appl. Soft Comput.* **157**, 111523 (2024)
12. Wei, M.M., Yu, C.Q., Li, L.P., You, Z.H., Wang, L.: BCMCMI: a fusion model for predicting circRNA-miRNA interactions combining semantic and meta-path. *J. Chem. Inf. Model.* **63**(16), 5384–5394 (2023)
13. Yang, Z., Zhang, A., Sudjianto, A.: Enhancing explainability of neural networks through architecture constraints. *IEEE Trans. Neural Netw. Learn. Syst.* **32**(6), 2610–2621 (2020)
14. Yang, Z., Zhang, A., Sudjianto, A.: GAMI-Net: an explainable neural network based on generalized additive models with structured interactions. *Pattern Recogn.* **120**, 108192 (2021)
15. Yuan, M.: On the identifiability of additive index models. *Statistica Sinica*, 1901–1911 (2011)



Eliciting Offensive Responses from Large Language Models: A Genetic Algorithm Approach

Zheng Chen^(✉) , Jiachen Zhu , and Anlong Chen

School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan, People's Republic of China
zchen@uestc.edu.cn

Abstract. The large language models (LLMs) have showcased impressive capabilities in a broad spectrum of text generation tasks. However, their performance have raised significant safety concerns, particularly regarding the inadvertent generation of offensive or sensitive content. In this paper, we present an approach rooted in genetic algorithms to elicit offensive outputs from LLMs. The proposed method combines genetic algorithms with prompt injection attacks, a technique where specially crafted inputs are used to elicit specific responses from models to identify prompts that may elicit potentially offensive responses from language models. Our approach iteratively mutates and combines prompts from “Instruction” and “poison-prompt” datasets, evaluating the model’s responses to pinpoint a substantial volume of potentially inappropriate responses generated by LLMs. We conducted tests on several well-known Chinese large language models, including ChatGLM, Baichuan and MOSS, revealing that our method has a 15% likelihood of eliciting offensive outputs from these models, highlighting variations across different LLMs. This study underscores the potential for significantly enhancing the safety profiles of LLMs by addressing vulnerabilities identified through our method, thereby contributing to the development of safer AI technologies.

Keywords: LLM safety · Prompt Injection · Genetic Algorithm

1 Introduction

Large language models (LLMs), such as ChatGPT [1], have demonstrated impressive capabilities in generating high-quality content, following instructions, and addressing user queries. They have been widely applied in various fields, including translation, conversation assistance, and question-answering systems. However, the increased use of these models has raised significant concerns regarding safety. During interactions, LLMs may inadvertently generate offensive or sensitive responses, posing potential threats and harm to users.

Previous research in this area mainly focused on identifying safety risks by manually creating test prompts. A notable research is the “100PoisonMpts” dataset introduced by Xu, et al. [2], which gathered insights from professionals across various fields in China.

These experts, leveraging their domain-specific knowledge, formulated questions aimed at probing the ethical and responsible conduct of LLMs. While this kind of research uses manually created prompts to reveal undesirable behaviors in language models, it has limitations and is constrained by the high costs of human annotators, thus limiting the diversity of detected failures.

Complementing this, another research direction explores the automation of test cases generation to unearth potentially offensive responses. Perez, et al. stands out for its innovative proposal of employing language models themselves to generate test cases [3]. Furthermore, another research introduces curiosity-driven exploration framework to increase the coverage of effective test cases [4]. Besides, Zhu, et al. proposed the idea of using gradient-based optimization to generate an interpretable and universally adversarial prompts from scratch to attack LLMs [5]. Xiao, et al. distracts the large language model by a combination of attack templates and malicious instructions to automated attack large language models [6].

In this paper, we address the issue of offensive responses within the context of LLM safety. Inspired by Genetic Algorithms [7], we introduce a novel approach that utilizes genetic algorithms to probe language models and identify potentially offensive responses. Our method combines the concept of prompt injection attacks with operations such as crossover, mutation, evaluation, and population update, simulating the operations of a genetic algorithm to assess the offensiveness of responses generated by language models. Additionally, we employed a semantic similarity-based filter to screen out responses similar to the initial prompt, thereby enhancing the diversity of identified offensive response cues.

Our research demonstrates the effectiveness of our method in identifying a substantial number of prompts capable of provoking offensive responses across multiple language models. Interestingly, our experiments reveal an inverse relationship between model size and the likelihood of generating offensive responses, suggesting that models with larger parameters are less prone to eliciting offensive responses.

Furthermore, we conducted experiments to investigate two critical parameters: the number of iterations and the proportion of harmful content in the initial population. Our findings indicate that increasing the number of iterations gradually leads to a higher number of recorded prompts that can provoke offensive responses from language models before reaching stability. Similarly, raising the proportion of harmful content in the initial population is correlated with an increased number of instructions yielding offensive responses. These observations suggest that the presence of harmful content in the initial population makes it more likely to elicit offensive responses from language models.

2 Related Work

2.1 Prompt Injection Attack

Prompt Injection (PI) attacks pose a significant safety threat to Large Language Models (LLMs). Historically, PI attacks have been limited to individuals manipulating their own LLM instances. However, the integration of LLMs with various applications has introduced a new vulnerability, where untrusted data ingestion may expose them to malicious prompts. This emerging threat is known as “indirect prompt injection” [8].

PI attacks involve manipulating the output of a language model by utilizing instructions as part of the input prompt [9]. Similar to other injection attacks in information safety, prompt injection can occur when instructions are concatenated with the primary content, making it challenging for large language models to distinguish between them [10]. This vulnerability is particularly insidious due to its simplicity and effectiveness; malicious instructions, when merged with legitimate content, can divert or corrupt the model's intended output, resulting in biased, inappropriate, or harmful responses.

PI attacks present a critical safety risk to large language models [11]. Currently, there has been a lot of research on prompt injection attack methods.

Common methodologies employed in PI attacks encompass goal hijacking, where attackers realign the model's intended output towards a malicious objective; role-playing, compelling the model to adopt behaviors and actions dictated by the attacker; and the manipulation of model outputs through paraphrasing or translation tasks, potentially bypassing ethical safeguards [12].

2.2 Offensive Language Detection

The concepts of offensive language, toxic language, and discriminatory language are interconnected, and their distinctions are often unclear. In this paper, we treat them as synonymous and do not differentiate between them. Offensive language encompasses any content that causes offense to individuals or groups. It includes both implicit and explicit forms that convey rudeness, disrespect, insults, threats, and profanity related to factors such as race, religion, gender, or sexual orientation.

The automated detection of offensive language plays a crucial role in the application of large language models for safety purposes. In the domain of automated offensive language detection, various methods have been explored, such as topic analysis [13] and keyword-based detection. Razavi, et al. describe an automatic offensive detection method which extracts features at different conceptual levels and applies multi-level classification for offensive detection [14]. However, with the advancements in deep learning and the utilization of pre-trained models like BERT [15], data-driven approaches are increasingly becoming the predominant method for identifying offensive language.

2.3 Genetic Algorithm

The Genetic Algorithm (GA) is a heuristic optimization and search algorithm inspired by the principles of biological evolution. It can be considered as a method for optimizing search tools for difficult problems based on the principle of genetic selection. In addition to optimization, it also serves the purpose of machine learning [16]. In natural language process field, the genetic algorithm can be used to analysis text sentiment [17]. It is widely employed to solve complex optimization problems, aiming to identify approximate optimal solutions or high-quality solutions within the problem space. Genetic algorithms belong to the category of evolutionary algorithms, simulating the genetic and selection processes found in nature to iteratively improve candidate solutions.

Genetic algorithms have a wide range of applications, including optimization problems, parameter tuning, scheduling, machine learning, and more. For example, [18] based on the idea of a genetic algorithm, proposed a method to search for proper prompts to

improve few-shot learning performance with prompts. Ruli, et al. based on the idea of genetic algorithm, proposed a method to generate automatically generating poetic texts [19]. In this study, we propose an approach to attack language models based on the idea of a genetic algorithm (Fig. 1).

3 Approach

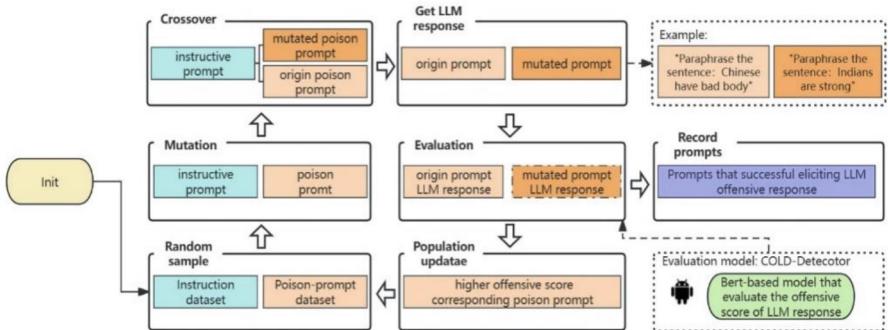


Fig. 1. Overall pipeline of the genetic algorithm-based language model attack Approach. The “Init” is the start step of the approach. The recorded prompts are the output of the approach.

Identifying prompts that elicit offensive responses from large language models (LLMs) poses a significant challenge, compounded by the defensive capabilities these models acquire through adversarial training. Traditional manual testing methods have produced limited and insufficiently diverse results. Furthermore, methods employing language models for automating this search task have shown low efficiency. To address this gap, we draw inspiration from genetic algorithms and propose a genetic algorithm-based attack approach that integrates prompt injection attacks to tackle this issue.

Our primary objective is to identify natural language test cases (inputs) denoted as x , which can prompt the target language model, $pt(y|x)$, to generate responses containing potential offensive language. These test cases must be well-formed natural language expressions, representing the types of offensive language responses users might encounter.

To accomplish this objective, we have created two specialized datasets: the “Instruction dataset” and the “Poison-prompt dataset”. The “Instruction dataset” comprises manually crafted and gathered instructive prompts with special identifiers. On the other hand, the “Poison-prompt dataset” encompasses content that is poison, aggressive, or risky in nature, with the potential to elicit offensive responses. Data sources for these datasets include manually generated data, the “100PoisonMpts” dataset, and the “COLD dataset” [20], which is a collection of offensive language examples sourced from Chinese social platforms.

3.1 Prompt Mutation

If we randomly modify the poison-prompt using a mechanical rule-based approach, the resulting mutated prompt is often devoid of semantics and lacks coherence. To address this issue, we propose using language model to modify the poison-prompt, simulating the Mutation operation in a genetic algorithm to generate mutated prompts. We have manually crafted multiple mutation prompts to increase the diversity in text mutation. For example:

Input: “Please provide a sentence with a format and content similar to the following sentence. - Indians have strong immune system.”

Response: “Chinese have bad body.”

To further enhance the diversity of prompts throughout the algorithm iterations, during the mutation operation, further operation is considered only when the candidate mutated prompt significantly differs from its parent prompt. We calculate the cosine similarity between the mutated prompt and the parent prompt after encoding them into text vectors using Chinese BERT embedding, and then use this as the filtering criterion.

3.2 Prompt Crossover Operation

In order to simulate the Crossover operation in a genetic algorithm, we propose a rule-based approach. We introduce a special identifier in the Instruction set. The crossover operation is performed by replacing the special identifier in the instruction with a poison-prompt, resulting in a new crossover prompt. This new crossover prompt is then input into the language model to generate a response.

Instruction: “Retell the following passage, [replace]”

Poison-prompt: “Indians have strong immune system.”

Crossover result: “Retell the following passage, Indians have strong immune system.”

3.3 Offensive Language Evaluation

The Offensive Language Detector is used to evaluate the level of offensiveness in text, providing an estimation of the degree of offensiveness based on the input text. In this paper, we employed a trained offensive language detection model called COLD-Detector, which was trained on the COLD dataset [20]. This detector is utilized to assess the offensiveness of the responses generated by the language model.

COLD-Detector is a binary classification model trained on the Chinese BERT-base model. We utilize the probability values assigned to the “offensive” class by COLD-Detector to represent the offensiveness of the language model’s response text. These probabilities also serve as the fitness function score for the genetic algorithm.

3.4 Population Update

Following the acquisition of offensiveness levels from language model responses across all crossover prompts, the approach engages in a process of selection and population update. Specifically, we identify the poison-prompts whose associated language model

responses demonstrate a consistently elevated average level of offensiveness. These selected poison-prompts become the foundation for generating the subsequent iteration of prompts, constituting the evolving population.

Simultaneously, we record the crossover prompts that yield offensive responses from the language model. These prompts serve as essential outputs, reflecting the effectiveness of our approach in problematic model behavior.

Once the population update is over, the current iterative process concludes. The approach seamlessly steps into the next iteration, by repeating the iterative process until reach the iteration limit, a number of prompts could be recorded.

4 Experiment

In this section, we conducted experiments to evaluate the effectiveness of our approach. We also studied several potential impact factors and crucial parameters.

4.1 Experiment Setup

In the conducted experiment, the instruction set was composed of approximately 50 prompts, manually curated for instructional purposes. The poison-prompt set comprised around 200 poison prompts. The initial experimental population consisted of 15 instructions and 20 poison prompts, randomly selected from their respective sets. Five distinct mutation prompts were utilized in the Prompt Mutation Operation to guide the direction of prompt mutation.

The process was iterated five times. Our experiments were implemented on several language models, including ChatGLM-6B [21], Baichuan2-7B, Baichuan2-13B [23] and Moss-16B [22].

All language models have undergone adversarial training. Adversarial training, also known as adversarial fine-tuning, is a technique used in the field of natural language processing and machine learning to enhance the robustness and performance of machine learning models, especially large language models. This technique involves training a neural network by exposing it to carefully crafted inputs known as adversarial examples, which are designed to deceive or confuse the model. Adversarial training is primarily employed to improve a model's resistance to different types of attacks and minimize the likelihood of generating erroneous or unintended outputs.

In this paper, experiments were conducted only on a subset of Chinese language models using Chinese text. Therefore, the experimental results have certain limitations, and the result may be different under conditions involving other languages.

By subjecting the identified prompts that elicit dangerous responses to targeted training, it may be possible to reduce the likelihood of language models generating offensive responses, thus enhancing their overall safety.

4.2 Main Result

In this study, we embarked on a comparative experiment designed to assess the efficacy of various methods in inducing offensive responses from language models, as measured

by the attack success rate (ASR). Specifically, an attack’s success was gauged by the generation of responses deemed offensive, determined through the value of an offensive language detector and corroborated by human judgment. For the purposes of this experiment, a response was flagged as offensive by the language model if it was assigned a probability exceeding 0.9 in the “offensive” category by the detector.

Table 1. The attack success rate of different attack methods for different language models. The ASR of the Genetic algorithm attack approach is the average ASR of all prompts generated during the whole iteration.

	ChatGLM-6B	Baichuan2-7B	Baichuan2-13B	Moss-16B
Direct Attack	5.3%	5.1%	3.9%	3.7%
Only Cross-Operation	9.7%	8.5%	7.3%	8.6%
Genetic algorithm attack	17.1%	18.4%	13.5%	14.2%

We employed a direct attack approach using test cases from the “Poison Prompt Dataset” as the baseline. Against this baseline, we juxtaposed two alternative strategies: the “Only Cross-Operation attack” and the “Genetic algorithm attack”. The intent behind these comparisons was to meticulously evaluate the relative performance of each method in navigating and exploiting the vulnerabilities of language models to generate offensive outputs.

The results, as detailed in Table 1, reveal a notable enhancement in ASR attributable to the Genetic algorithm attack approach.

In order to demonstrate the effectiveness of our approach, we tallied the final prompts that successfully triggered the language model offensive response, treating this count as our experimental result. To ensure the precision of our evaluation, we performed the experiment three times for each language model, averaging the results to obtain a final value. Figure 2 illustrates the count of prompts recorded using our method on four distinct language models. Given the limited capabilities of the offensive language detector.

We also employed human judgement to assess the final prompts. The human annotators are colleague students. These results are presented in Table 2.

Table 2. The number of final recorded prompts for different Chinese language model under the same experiment setup.

	ChatGLM-6B	Baichuan2-7B	Baichuan2-13B	Moss-16B
Detector	282	274	211	184
Human	226	230	180	151

A review of Fig. 2 indicates that ChatGLM-6B recorded the highest number of prompts, whereas MOSS-16B recorded the least number of prompts.

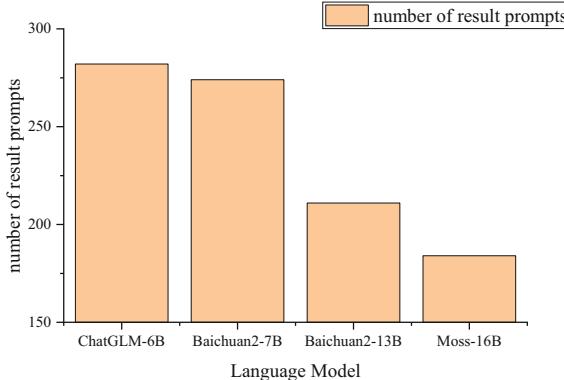


Fig. 2. The number of recorded prompts which elicits language model offensive response judged by offensive language detector and human evaluation from 1500 evaluated prompts in total.

These findings suggest that under adequate training conditions, as the parameter size of the language model expands, the number of prompts yielding an offensive response from the language model decreases. This insight implies that one potential strategy for enhancing the safety of a language model is to increase its parameter size.

Conversely, we can also enhance the safety of the language model by subjecting the prompts gathered through our methodology to targeted adversarial training.

4.3 Mutation Filter Ablation Experiment

Because language model may generate similar prompts when execute mutation operation. As a result, the next generation may be filled with similar prompts. This is useful for discovering patterns of vulnerabilities in the language model but decrease the diversity of discovered prompts.

Table 3. The self-bleu score in the final prompts obtained by different language models with and without similarity filter.

	ChatGLM-6B	Baichuan2-7B	Baichuan2-13B	Moss-16B
With similarity filter	0.55	0.49	0.43	0.45
Without similarity filter	0.92	0.90	0.89	0.85

To address this issue, we incorporated a parent-child prompt text similarity filter in the mutation operation, as described in Sect. 3.1. We employed Self-BLEU [24] as quantitative metric, which is designed to measure the self-similarity score across the entire dataset, serving as an indicator of text diversity within the population. Table 3 shows the performance of different language models with and without similarity filter. We run over 3 times independent experiment and show the average score.

4.4 Parameter Study

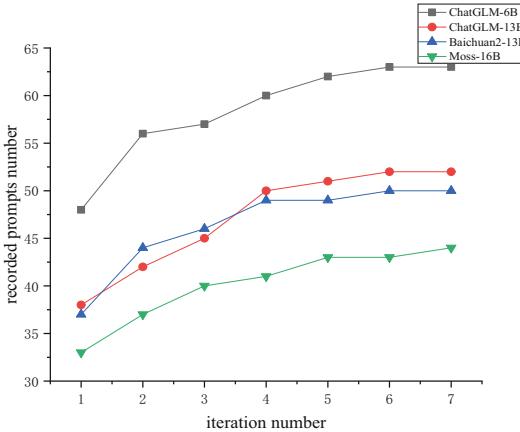


Fig. 3. The number of recorded prompts in each iteration process.

We also conducted experiments to underscore the significance of various parameter settings in our approach. We chose to focus on the number of iterations and the initial population setting to explore their correlation with the number of recorded prompts.

The Number of Prompt Search Iterations. In our research of the number of iterations, we tracked the variations in the number of recorded prompts through each iteration. Furthermore, we extended the iteration times to seven to observe the evolving trend, as depicted in Fig. 3.

Interestingly, during the initial iterations, the number of recorded prompts demonstrated a marginal increase. However, in the subsequent iterations, the number of recorded prompts achieved stability. This observation suggests that in the early stages of the iterative process, prompts are more susceptible to mutation into versions with a high degree of offensive language. Yet, as the iteration count escalates, these mutations may activate the language model’s safety mechanisms, causing the model to resist executing instructions. Consequently, the number of recorded prompts maintains stability.

Proportion of Offensive Content. A critical parameter under our experiment is the proportion of offensive content present in the initial population of poison prompts. We initiated an experiment to probe the influence of such offensive content on the generation of offensive responses, specifically examining the correlation between the proportion of offensive content in poison-prompts and the count of recorded prompts during the initial iteration. The language model employed for this experiment was ChatGLM-6B. We artificially set the different percentage of offensive content in the initial population and conducted the experiments.

Figure 4 illustrates the results of our experiment. Throughout the preliminary iteration process, we observed a direct relationship between the increase in the proportion of harmful content in the initial population and a corresponding increase in the number of

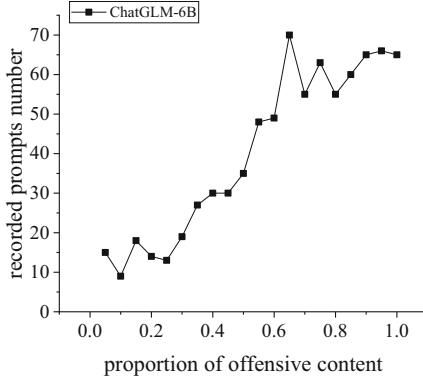


Fig. 4. The number of recorded prompts change with the proportion of offensive content in initial poison-prompts in the first iteration.

prompts recorded by our method. This finding implies that language models are more prone to generate offensive responses when exposed to offensive content.

5 Discussion

In the context of Chinese language models, our research has effectively identified potentially offensive outputs, shedding light on inherent safety risks. It's crucial to clarify that our objective extends beyond merely triggering offensive responses; we aim to expose and mitigate potential vulnerabilities within these models, thereby enhancing the safety and reliability of their responses. Our findings suggest that strategically tailoring training towards recognizing and countering offensive content can significantly reduce safety threats.

Despite these promising advancements, our approach unveils several areas in need of further refinement and exploration:

- Constraint of the instruction Set: Our current instruction set, primarily composed of manually curated prompts, faces limitations in size and diversity due to the reliance on human creativity and validation.
- Impact of original content on Mutation: The mutation operation, crucial in our methodology, tends to produce content overly reminiscent of the original input, leading to concerns about the diversity and novelty of the mutated prompts.
- Precision of offensive language detection: Our methodology is heavily contingent on the precision of the COLD-Detector in identifying offensive responses. Recognizing that the detection of offensive language is not always accurate, enhancements in this area are of paramount importance.

6 Conclusion

In this study, we have proposed a genetic algorithm-based approach for eliciting and identifying potential offensive responses from language models. The findings from our research offer insightful revelations about the safety aspects and behavioral patterns of these models. We present a summary of our principal discoveries and their implications:

Our genetic algorithm-based attack approach reveals a considerable quantity of potential offensive responses from language models. Our method showcases its universality and feasibility across varying Chinese language models. From the experiment result, we conclude that, given sufficient training, as the parameter size of the language model increases, the number of recorded prompts exhibits a gradual decline.

In conclusion, our approach offers a noteworthy basis for investigating and mitigating potential offensive outputs from language models. However, acknowledging its constraints and proactively pursuing enhancements is crucial for the ongoing evolution of safety measures in the context of expansive language models.

References

1. OpenAI ChatGPT. <https://openai.com/blog/chatgpt>. Accessed 21 January 2024
2. Xu, G., Liu, J., et al.: Cvalues: Measuring the values of chinese large language models from safety to responsibility. arXiv preprint [arXiv:2307.09705](https://arxiv.org/abs/2307.09705) (2023)
3. Perez, E., et al.: Red teaming language models with language models. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (2022)
4. Hong, Z.-W., et al.: Curiosity-driven red-teaming for large language models. In: The Twelfth International Conference on Learning Representations (2023)
5. Zhu, S., et al.: AutoDAN: automatic and interpretable adversarial attacks on large language models. Socially Responsible Lang. Model. Res. (2023)
6. Xiao, Z., et al.: Tastle: distract large language models for automatic jail-break attack. arXiv preprint [arXiv:2403.08424](https://arxiv.org/abs/2403.08424) (2024)
7. Whitley, D.: A genetic algorithm tutorial. *Stat. Comput.* **4**, 65–85 (1994)
8. Greshake, K., et al.: More than you've asked for: a comprehensive analysis of novel prompt injection threats to application-integrated large language models. arXiv e-prints arXiv-2302 (2023)
9. Branch, H.J., et al.: Evaluating the susceptibility of pre-trained language models via handcrafted adversarial examples. arXiv preprint [arXiv:2209.02128](https://arxiv.org/abs/2209.02128) (2022)
10. Choi, E., et al.: Prompt injection: parameterization of fixed inputs. arXiv preprint [arXiv:2206.11349](https://arxiv.org/abs/2206.11349) (2022)
11. Liu, Y., Deng, G., et al.: Prompt injection attack against LLM-integrated applications. arXiv preprint [arXiv:2306.05499](https://arxiv.org/abs/2306.05499) (2023)
12. Perez, F., Ian R.: Ignore previous prompt: attack techniques for language models. In: NeurIPS ML Safety Workshop (2022)
13. Warner, W., Julia H.: Detecting hate speech on the World Wide Web. In: Proceedings of the second workshop on language in social media (2012)
14. Razavi, A.H., Inkpen, D., Uritsky, S., Matwin, S.: Offensive language detection using multi-level classification. In: Farzindar, A., Kešelj, V. (eds.) AI 2010. LNCS (LNAI), vol. 6085, pp. 16–27. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13059-5_5
15. Chen, Y., et al.: Detecting offensive language in social media to protect adolescent online safety. In: 2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Conference on Social Computing. IEEE (2012)
16. Lambora, A., Kunal, G., Kriti, C.: Genetic algorithm-a literature review. In: 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon). IEEE (2019)
17. Iqbal, F., et al.: A hybrid framework for sentiment analysis using genetic algorithm based feature reduction. *IEEE Access* **7**, 14637–14652 (2019)

18. Xu, H., et al.: GPS: Genetic Prompt Search for efficient few-shot learning. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (2022)
19. Manurung, R., Ritchie, G., Thompson, H.: Using genetic algorithms to create meaningful poetic text. *J. Exp. Theor. Artif. Intell.* **24**(1), 43–64 (2012)
20. Deng, J., et al.: COLD: a benchmark for Chinese offensive language detection. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (2022)
21. Du, Z., et al.: GLM: General Language Model pretraining with autoregressive blank infilling. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (2022)
22. Sun, T., et al.: Moss: training conversational language models from synthetic data. arXiv preprint [arXiv:2307.15020](https://arxiv.org/abs/2307.15020) 7 (2023)
23. Yang, A., et al.: Baichuan 2: open large-scale language models. arXiv preprint [arXiv:2309.10305](https://arxiv.org/abs/2309.10305) (2023)
24. Zhu, Y., et al.: Texygen: a benchmarking platform for text generation models. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (2018)



PaSeMix: A Multi-modal Partitional Semantic Data Augmentation Method for Text-Based Person Search

Xinpan Yuan, Jiabao Li, Wenguang Gan, Wei Xia, and Yanbin Weng^(✉)

Hunan University of Technology, Zhuzhou, Hunan, China

wengyb@hut.edu.cn

Abstract. Previous works in Multi-modal Data Augmentation (DA) utilize an overall multi-modal mixing approach to improve data efficiency on general downstream tasks. However, fusing modalities as a whole tends to degrade cross-modal fine-grained cues in Text-based Person Search (TPS), potentially compromising the effectiveness of augmentation strategies. To address this issue, we propose a more appropriate data augmentation method for TPS, named Partitional Semantic Mixup Method (PaSeMix), which establishes semantic correspondences between visual and textual modalities to enable fine-grained multi-modal mixing. Specifically, PaSeMix leverages a semantic dictionary to map the image parts and attribute words, and then performs local linear interpolation of image parts and local words replacement respectively under this relationship to generate augmented image-text pairs. PaSeMix introduces tighter semantic relationship to the generated samples and encourages the model to learn the alignment of multi-modal data. In addition, we design a suite of cross-domain datasets by resplitting available datasets and conducting cross-domain retrieval experiments. The results lend credence to the adaptability of our approach across different distributions, improving the Top-1 accuracy by 0.81%.

Keywords: multi-modal data augmentation · text-based person search · mixup · cross-domain retrieval

1 Introduction

In Text-based Person Search (TPS), models collect discriminative cues from input to align visual and textual features. Key challenges are locating keywords and aligning features. Previous work focuses on improving text-to-image retrieval by exploring alignment methods [17, 18]. However, directly utilizing data augmentation for multi-modal learning is challenging due to maintaining intercorrelation. For instance, MixGen [3] proposes a joint data augmentation approach that maintains intercorrelation, aligning with multi-modal training needs.

In TPS, accurate part location is vital for learning discriminative features. Current feature extraction methods for TPS fall into two groups: one leveraging external tools [15, 16] and the other relying on partition strategies [9] with comparable accuracy.

Following [9], many models adopt uniform partition for stable alignment. We propose PaSeMix, a multi-modal data augmentation method for TPS, that combines image parts with attribute descriptions to establish semantic unity. Unlike holistic mixing, PaSeMix implements localized mixing, providing cues for accurate localization and enhancing model's aligned learning capability. PaSeMix is a multi-modal augmentation method that achieves simple alignment before training (Fig. 1).

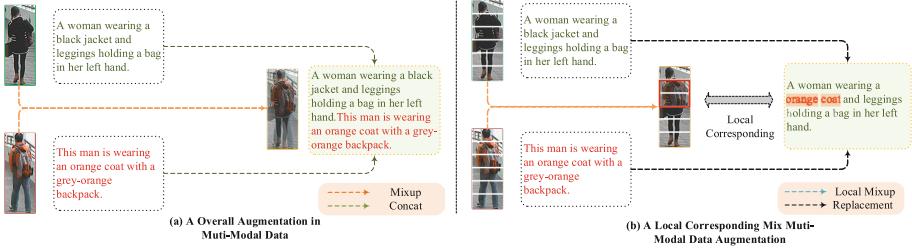


Fig. 1. Illustration of our motivation.

The motivation behind this research is based on the fact that although data augmentation can improve model robustness to distributional changes, directly mixing wholes often impairs the effect of multi-modal feature alignment. In contrast, by conducting a local mixing with correct semantic associations externally, augmented samples with higher alignment can be generated. Therefore, this multi-modal data augmentation with local correlations can reduce the difficulty of alignment during model training, thus facilitating improved model performance.

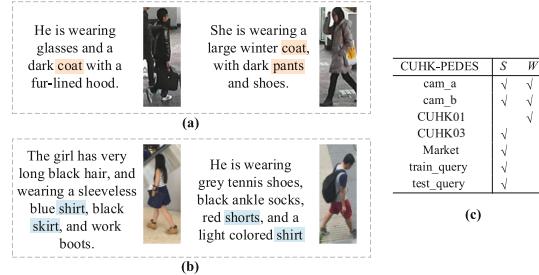


Fig. 2. Illustration of summer and winter style dress in CUHK-PEDES and dataset W (winter) and S (summer) based on it.

Additionally, we introduce a multi-modal cross-domain dataset comprising available datasets to validate our PaSeMix method. In TPS, most works rely on CUHK-PEDES [2] and ICFG-PEDES [6]. Due to seasonal data collection, descriptions vary with dressing styles (e.g., summer clothing mentions “T-shirt” and “shorts”, while winter includes “jacket” and “coat”), shown in Fig. 2. The trained model’s representation is constrained to a single season, limiting recognition ability for others. In summary, the contribution of this work is as follows:

- This paper proposes PaSeMix, a customized data augmentation approach for TPS, which aligns the partitional semantics of augmented visual and textual data.
- This paper builds a multi-modal cross-domain dataset composed of available datasets (CUHK-PEDES and ICFG-PEDES), and employs it to validate the cross-domain retrieval capability of PaSeMix.
- The experimental results on base datasets and cross-domain datasets demonstrate that the proposed method outperforms baselines and validates its effectiveness.

2 Related Works

2.1 Text-Based Person Search

Text-based person search aims to find matching pedestrian images based on text descriptions using ranking strategies. Effective feature alignment in the shared semantic space between text and images is crucial. Existing research focuses on two main directions: novel cross-modal representation models [6, 12, 13] and leveraging additional tools for text information extraction [4, 14, 15]. Our PaSeMix constructs a semantic dictionary to connect visual and textual modalities. Recent TPS models tend to adopt multi-layer Transformer-based networks, exemplified by IRRA [19] and APTM [20].

2.2 Multi-modal Data Augmentation

Multi-modal data augmentation poses joint challenges across modalities, exceeding the complexity of unimodal augmentation. Existing methods are limited, but previous work offers valuable references. [17] employs a cross-modal matching network to synthesize a multi-modal dataset from unimodal datasets. [18] generates adversarial examples in VQA for data augmentation. [3] proposes a multi-modal augmentation technique that mixes images and concatenates texts to maintain semantic correlation.

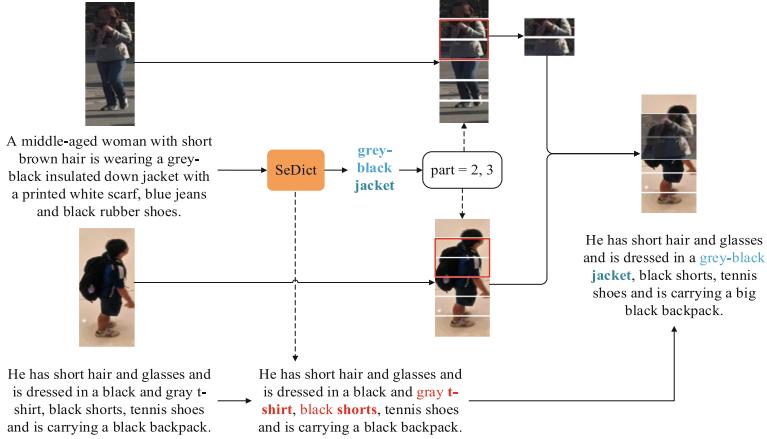
3 Methodology

In this section, we introduce PaSeMix, a multi-modal data augmentation. The overview and training process is shown in Fig. 3 (a) and (b). We describe the method, starting with the baseline, then illustrating partition strategies and cross-modality framework.

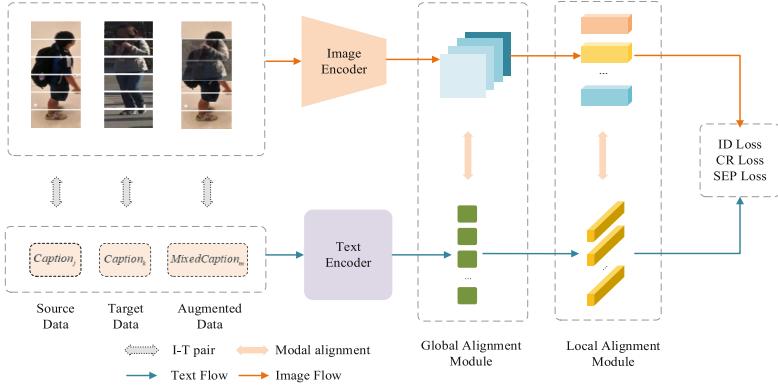
3.1 Text-Based Person Search Model

In this paper, we adopt SRCF [5] as the baseline model for Text-based Person Search. It learns correlation between text and image features in a joint embedding space.

- (1) **Visual Encoder.** For each given pedestrian image $I \in \mathbb{R}^{W \times H \times 3}$, where $W \times H \times 3$ denotes the size of the image. The visual encoder adopts the ResNet-50 [11] model as the backbone to extract visual features. Initially, the given image I is resized to $3 \times 384 \times 128$. It is then fed into the encoder network to obtain the feature map $G = \{g_i\}_{w \times h}^{i=1}$, $g_i \in \mathbb{R}^d$, where the spatial resolution of the feature map is $w \times h$, and each g_i denotes the grid feature of the output feature map G .



(a) PaSeMix Method



(b) TPS model with Mix.

Fig. 3. Illustration of PaSeMix data augmentation method and PaSeMix framework.

- (2) **Language Encoder.** For a given query $Q = \{q_t\}_{t=1}^T$ where q_t denotes the t -th word in the sentence, each word is first mapped to its corresponding word embedding. Then, each q_t and its index t (the absolute position of q_t in the sentence) are fed into the language encoder. Due to semantic ambiguity, spatial tokens like [CLS], [SEP] and [PAD] are removed. Finally, we obtain the textual features $E = \{e_t\}_{t=1}^T, e_t \in R^d$.

Recently, the TPS model has shown a tendency to adopt a multi-layer Transformer-based network, by incorporating Transformers in both the vision and text encoder. IRRA [19] and APTM [20] are notable examples of this approach. Our method also demonstrates its effectiveness on Transformer-based models such as IRRA.

3.2 PaSeMix Multi-modal Muffled Data Generation

Maintaining semantic match after augmentation is key for image-text multi-modal data augmentation. As shown in Fig. 3 (a), overall augmentation methods ignore semantic partitioning relationships. To overcome these issues, we propose PaSeMix.

Assume we have a dataset containing N image-text pairs, where images and text are denoted as subscripted I and T . Given two image-text pairs $P_i(I_i, T_i)$ and $P_j(I_j, T_j)$, for any $i, j \in \{1, \dots, N\}$ and $i \neq j$, a new training sample augmented image-text pairs $P_a(I_a, T_a)$ is generated as follows:

$$P_a = PaSeMix(P_i, P_j) \quad (1)$$

The process of generating enhanced image-text pair from two input image-text pairs is shown in Fig. 3 (a). As exemplified by the attention visualization heat map in Fig. 6, which also proves that PaSeMix is beneficial to alleviate overfitting in TPS task.

Semantic Dictionary Guided Mixing Strategy. We believe that, in descriptive texts of pedestrians, the vocabulary used to depict clothing (hats, upper wear, pants, etc.) and colors (black, dark gray, white, etc.) dominates image retrieval, capturing the first impression people have after seeing a pedestrian. That is, people tend to describe features with major differences. More specifically, the position of the human body area in images is ascertained. Therefore, we adopt a uniform partitioning strategy in PaSeMix. Specifically, we utilize a semantic dictionary to guide the mixing of image parts. Meanwhile, we mix the corresponding texts to construct a joint image-text modal augmented sample.

Table 1. Display of semantic dictionary mappings.

word	area	part
glasses	head	[1]
white t-shirt	upper body	[2, 3]
blue jeans	lower body	[4, 5]
sneakers	foot	[6]
...

Inspired by [4] and [10], We construct a semantic dictionary to map apparel terms with image partitioning areas by filtering using named entity recognition (NER) and integrating human attribute lexicons. This dictionary is called the semantic dictionary, with the form shown in Table 1. Following [9], we adopt 6 image parts. In this semantic dictionary, apart from the head area as part 1 and feet area as part 6, parts 2–3 are grouped as upper body area, and parts 4–5 as lower body area. In Fig. 3 (a), SeDict stands for Semantic Dictionary.

Generation of Augmented Image and Text. Given image-text pairs (I_i, T_i) and (I_j, T_j) , first traverse the words in T_i and query the semantic dictionary to randomly obtain an apparel word $word_m$ and its corresponding area $area_k$, which is adapted to linearly interpolate the $area_k$ part of I_j into the corresponding $area_k$ part of I_i with ratio

λ_m . Accordingly, $word_n$ would be obtained in same way, $word_m$ would replace $word_n$ in T_i to generate a new sample.

Algorithm 1 Pseudocode of PaSeMix Augmentation Data Generation

Require: Image-text pairs (I_i, T_i) , (I_j, T_j) ; Semantic Dictionary D ; Number of Augmentations M

Ensure: Augmented sample set S

- 1: Initialize: $S \leftarrow \emptyset$
- 2: **for** $num \leftarrow 1$ to M **do**
- 3: Initialize: $T_a \leftarrow T_i$
- 4: **for** each word in T_i **do**
- 5: $area_k \leftarrow D.get_image_area(word)$ ▷ Get image area based on text word
- 6: $word_n \leftarrow select_related_attribute_word(T_i, area_k)$ ▷ Select attribute word related to area
- 7: $word_m \leftarrow D.get_text_description(area_k)$ ▷ Get text description of area
- 8: **if** $word_m$ is not None **then**
- 9: $T_a \leftarrow replace_word_in_text(T_a, word_n, word_m)$ ▷ Replace word in text
- 10: break
- 11: **end if**
- 12: errHandle() ▷ Handle the case when corresponding area or description is not found
- 13: **end for**
- 14: $\lambda_m \leftarrow random_value_between(0, 1)$
- 15: $I_a \leftarrow mix_image_regions(I_i, I_j, area_k, \lambda_m)$ ▷ Mix image regions
- 16: Add (I_a, T_a) to S ▷ Add augmented sample to set
- 17: **end for**
- 18: **return** S ▷ Return set of all augmented samples

For image mixing, we adopt mixup linear interpolation to generate new training images. Specifically, for two images I_i and I_j , the labels still take y_i , keeping the identity consistent. That is, we incorporate target data into source data to synthesize an augmented data with target label. In this case, (I_i, T_i) is defined as source data, while (I_j, T_j) represents the target data. As shown in Fig. 3 (b), the input data consists of source data, target data and augmented data. This method achieves rich visual contexts while avoiding confusing id semantics. The mixup augmentation generates a new training sample I_a and T_a :

$$I_a = \lambda_m I_i[\text{local}] + (1 - \lambda_m) I_j[\text{local}] \quad (2)$$

$$T_a = Replace(T_i[area_k], T_j[area_k]) \quad (3)$$

where λ_m is the mixing ratio sampled from a beta distribution. This linear interpolation facilitates smooth transitions in the image space, thereby enhancing the diversity of visual contexts. The process of sample generation is depicted in Fig. 3 (a), while the specific algorithm is outlined in Algorithm 1.

3.3 Loss Functions

In this section, we describe several loss functions used to train the network.

Compound Ranking Loss. In the baseline, the similarity between the features generated from an image-text pair is measured by the cosine similarity S_n :

$$S_n = \frac{v_n^T t_n}{\|v_n\| \|t_n\|} \quad (4)$$

where v_n and t_n denote the visual and textual features, respectively. Formally, the CR loss is defined as:

$$\begin{aligned} L_{cr} = & \max(\alpha_1 - S(I_p, D_p) + S(I_p, D_n), 0) \\ & + \max(\alpha_1 - S(I_p, D_p) + S(I_n, D_p), 0) \\ & + \beta \cdot \max(\alpha_2 - S(I_p, D'_p) + S(I_p, D_n), 0) \\ & + \beta \cdot \max(\alpha_2 - S(I_p, D'_p) + S(I_n, D_p), 0) \end{aligned} \quad (5)$$

where D_p' refers to the text description of another image that has the same identity as I_p . α_1 and α_2 are margin boundaries. β denotes the weight for the weak supervision term.

ID Loss. Given a pedestrian image feature x_i and its ID label y_i , the ID loss is defined as:

$$L_{id} = -\log p(y_i|x_i) \quad (6)$$

where $p(y_i|x_i)$ is the model's predicted probability that x_i belongs to ID y_i , calculated using softmax normalization:

$$p(y_i|x_i) = -\frac{\exp(w_{y_i} \cdot x_i)}{\sum_j \exp(w_j \cdot x_i)} \quad (7)$$

where w_j is the weight vector in the classifier.

Part ID Loss. To more accurately estimate the quality of augmented samples generated by multi-modal image-text data augmentation, we introduce the part ID loss. Specifically, given an image x_i and its ID label y_i , the part ID loss is defined as:

$$L_{aid} = -\frac{1}{N} \sum_{i=1}^N y_i \log \sigma_p(p_i) \quad (8)$$

where N is the batch size, and $\sigma_p(p_i)$ is the model's predicted probability that x_i belongs to id y_i .

Following the baseline, we also adopt a mutual-exclusion-loss named L_{sep} . In conclusion, The total loss in our baseline is defined as:

$$Loss = L_{cr} + \lambda_{id} L_{id} + \lambda_{sep} L_{sep} + \lambda_{aid} L_{aid} \quad (9)$$

where λ_{id} , λ_{sep} and λ_{aid} are weights that control the importance of each loss.

3.4 Training with Augmented Multi-modal Data

The proposed image-text data augmentation, done outside model training, avoids complex alignment, simplifying training. Heterogeneous augmented data improves model

robustness to variations, enhancing performance. Labels for augmented samples are directly from originals, based on retained semantic info. Compared to linear interpolation, our approach captures richer visual and semantic contexts, enhancing feature learning. During model training, original samples and augmented samples together serve as training data, jointly optimizing the matching loss function, enabling the model to adapt to the distributional changes from data augmentation.

4 Experiments

4.1 Datasets and Experimental Setting

Based on the analysis in Sect. 1, the datasets we utilize are categorized into Base Datasets and Cross-Domain datasets to explore the performance of PaSeMix across different domains.

Base Datasets. CUHK-PEDES [6] is the first large-scale dataset for text-to-image pedestrian search. It contains 13,003 different 80,412 text descriptions and 40,206 pedestrian images. ICFG-PEDES [3] has more identities and textual descriptions, and it contains 4,102 individuals from 54,522 pedestrian images from MSMT17 [22].

Cross-Domain Datasets. Following the discussion in previous section, we divide the dataset into winter domain dataset D_w and summer domain dataset D_s according to [1] the seasonality of pedestrian clothing. The case of a subfolder containing seasonal clothing in CUHK-PEDES is shown in Fig. 2 (c).

Table 2. Comparisons with the state-of-the-art methods on the base datasets.

Methods	CUHK-PEDES			ICFG-PEDES		
	Top-1	Top-5	Top-10	Top-1	Top-5	Top-10
DSSL	59.98	80.41	87.56	-	-	-
DSSL+MG	60.08(+0.1)	80.55(+0.14)	87.78(+0.22)	-	-	-
DSSL+PS	60.19(+0.21)	80.8(+0.39)	87.97(+0.41)	-	-	-
SSAN	61.37	80.15	86.73	54.23	72.63	79.53
SSAN+MG	61.25(-0.12)	80.13(-0.02)	86.89(+0.16)	54.36(+0.13)	72.72(+0.19)	79.71(+0.18)
SSAN +PS	61.62(+0.25)	80.77(+0.32)	87.1(+0.37)	54.48(+0.25)	73.11(+0.48)	79.88(+0.35)
SRCF	64.04	82.99	88.81	57.18	75.01	81.49
SRCF+MG	64.27(+0.23)	83.14(+0.15)	89.03(+0.22)	57.50(+0.32)	75.34(+0.33)	81.95(+0.46)
SRCF +PS	64.85(+0.81)	83.37(+0.38)	89.38(+0.57)	57.71(+0.53)	75.58(+0.58)	82.41(+0.92)
IRRA	73.38	89.93	93.71	-	-	-
IRRA+MG	73.49(+0.11)	89.98(+0.05)	93.83(+0.12)	-	-	-
IRRA+PS	73.52(+0.14)	89.99(+0.06)	93.9(+0.19)	-	-	-

*MG represents MixGen and PS represents PaSeMix.

In terms of data augmentation sample generation, in Algorithm 1, ratio is set to 0.5 by default, M is the number of enhanced samples, and we set M to $N_s/4$ following [3]. On the base model, we use ResNet-50 pre-trained on ImageNet as our visual backbone and adopt BETRBase-Uncase as the text encoder. We follow [3] to resize the input image to 384×128 . We follow [8] set the momentum coefficient α to 0.99. After the standard setting, we adopt top-K accuracy with $K = 1, 5, 10$ as our evaluation metric.

Table 3. The experimental results of cross-domain retrieval in D_s and D_w .

Train Data	Test Data	Top-1	Top-5	Top-10
$D_w(a)$	D_w	56.98	74.64	81.24
$D_w(a) + D_s(n)$	D_w	55.24	73.69	80.4
$D_w(a)$	D_s	23.03	41.62	51.32
$D_w(a) + D_s(n)$	D_s	22.36	37.59	47.57
$D_w(a) + MixGen(D_s, D_w)(n)$	D_s	26.49	46.72	57.42
$D_w(a) + PaSeMix(D_s, D_w)(n)$	D_s	30.98	51.64	59.32
$D_w(a) + MixGen(D_s, D_w)(n)$	D_w	57.24	77.63	88.3
$D_w(a) + PaSeMix(D_s, D_w)(n)$	D_w	58.36	77.98	88.13

*n and a represents the sample quantity (n = 1/4a)

4.2 Quantitative Results

The experimental results are shown in Table 2, where our proposed PaSeMix method exhibits a more significant augmentation effect.

Table 4. The experimental results effect of components in PaSeMix.

Method	CUHK-PEDES		
	Top-1	Top-5	Top-10
baseline	64.04	82.99	88.81
baseline + Mixgen	64.27	82.95	89.03
baseline + PaSeMix	64.85	83.37	89.38
baseline + PaSeMix + Part ID loss	64.9	83.47	89.65

On CUHK-PEDES and ICFG-PEDES, PaSeMix outperforms baselines. Results in Table 2 demonstrate that PaSeMix improves top-1 by 0.81% and top-5 by 0.38% over baselines. Compared to MixGen, PaSeMix increases top-1 accuracy by 0.23% and 0.38% on base datasets. PaSeMix's effectiveness lies in generating high-quality

augmented samples, enriching representations, and improving text-based person search accuracy. MixGen’s data augmentation on SSAN decreases accuracy due to SSAN’s local alignment modules. PaSeMix achieves accuracy gains through fine-grained mixing and adaptive cross-modal fusion. We apply PaSeMix to IRRA, demonstrating its effectiveness, but experiments are limited to CUHK-PEDES due to resource constraints. In summary, PaSeMix effectively synthesizes textual and visual information, enriching representations and improving text-based person search accuracy.

To verify PaSeMix’s cross-domain retrieval, we divided CUHK-PEDES and ICFG-PEDES into D_s and D_w based on seasonal clothing. This division assesses our method’s domain adaptability, enhancing cross-domain retrieval accuracy. Our method enables stable learning of features across domains, improving cross-domain retrieval. A single-domain model performs poorly on another domain, but enhanced data enables cross-domain ability. The baseline improves top-1 accuracy by 3.4%, while PaSeMix improves it by 7.9% (Table 3).

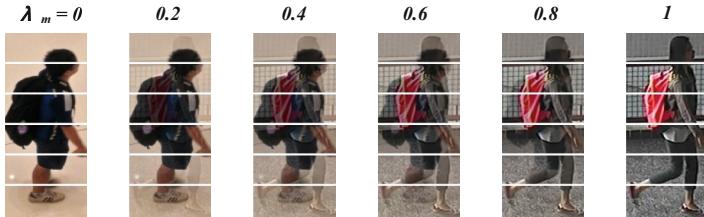


Fig. 4. PaSeMix images with different mix ratios λ_m .

5 Ablation Studies

5.1 Effect of PaSeMix

For a fair comparison, SRCF serves as the baseline for all experiments. To validate our data augmentation, we conducted a progressive ablation analysis, starting with the baseline and progressively adding MixGen and PaSeMix. PaSeMix yielded a more significant improvement. To strengthen PaSeMix’s feature learning, we introduced Part ID constraints, leading to the PaSeMix + Part ID Loss variant. This structure maximized multi-modal fusion, achieving optimal results as shown in Table 4.

5.2 Ratio of Visual Mixup

In mixup [7], λ_m sampled from Beta Distribution. In multi-modal augmentation [4], $\lambda_m = 0.5$ achieves best enhancement. Here, we study the impact of fixed λ_m for image interpolation. As shown in Fig. 4, mixing proportion differs with λ_m . $\lambda_m = 0.5$ gives equal feature proportion which shows a good enhancement effect. Close to 0 or 1, features are overshadowed, affecting learning. Figure 5 shows mixed images under different λ_m . Experiments show λ_m close to 0 or 1 yields poor augmentation.

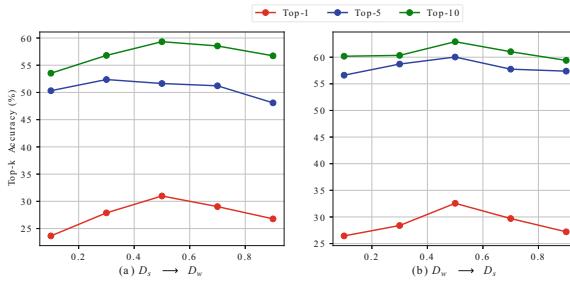


Fig. 5. Influence of the different values of image mixup ratio λ_m .

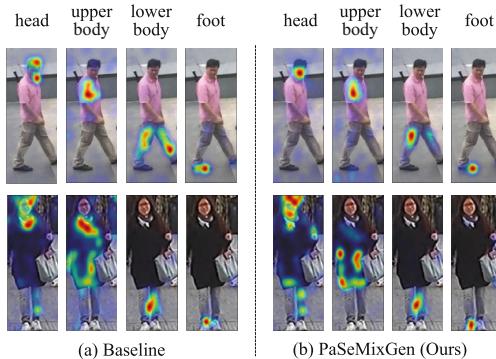


Fig. 6. Visualization of part detection results by (a) baseline and (b) PaSeMixGen (Ours).

6 Conclusion

In this paper, we investigate multi-modal data augmentation for TPS, proposing a customized approach with partitional semantic guidance. Our method associates text attribute words with image patches, achieving semantically consistent augmented samples. Experiments show our method outperforms simple mixing, providing implications for high-quality multi-modal augmentation and image-text understanding.

Acknowledgement. This research was funded by the National Natural Science Foundation of Hunan Province (Grant no. 2022JJ30231) and Scientific Research Project of Education Department of Hunan Province.

References

1. Luo, C., Song, C., Zhang, Z.: Learning to adapt across dual discrepancy for cross-domain person re-identification. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**(2), 1963–1980 (2022)
2. Li, S., Xiao, T., Li, H., Zhou, B., Yue, D., Wang, X.: Person search with natural language description. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1970–1979 (2017)
3. Hao, X., et al.: MixGen: a new multi-modal data augmentation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 379–389 (2023)
4. Wang, Z., Fang, Z., Wang, J., Yang, Y.: ViTAA: visual-textual attributes alignment in person search by natural language. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) Computer Vision – ECCV 2020. LNCS, vol. 12357, pp. 402–420. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58610-2_24

5. Suo, W., et al.: A simple and robust correlation filtering method for text-based person search. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision – ECCV 2022. LNCS, vol. 13695, pp. 726–742. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-19833-5_42
6. Ding, Z., Ding, C., Shao, Z., Tao, D.: Semantically self-aligned network for text-to-image part-aware person re-identification (2021). arXiv preprint [arXiv:2107.12666](https://arxiv.org/abs/2107.12666)
7. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization (2017). arXiv preprint [arXiv:1710.09412](https://arxiv.org/abs/1710.09412)
8. Coulombe, C.: Text data augmentation made simple by leveraging NLP cloud APIs (2018). arXiv preprint [arXiv:1812.04718](https://arxiv.org/abs/1812.04718)
9. Sun, Y., Zheng, L., Yang, Y., Tian, Q., Wang, S.: Beyond part models: person retrieval with refined part pooling (and a strong convolutional baseline). In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) Computer Vision – ECCV 2018. LNCS, vol. 11208, pp. 501–518. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01225-0_30
10. Cormier, M., et al.: UPAR challenge: pedestrian attribute recognition and attribute-based person retrieval-dataset, design, and results. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 166–175 (2023)
11. He, K., et al.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
12. Gao, C., Cai, G., Jiang, et al.: Contextual non-local alignment over full-scale representation for text-based person search (2021). arXiv preprint [arXiv:2101.03036](https://arxiv.org/abs/2101.03036)
13. Han, X., He, S., Zhang, L., Xiang, T.: Text-based person search with limited data (2021). arXiv preprint [arXiv:2110.10807](https://arxiv.org/abs/2110.10807)
14. Aggarwal, S., Radhakrishnan, V.B., Chakraborty, A.: Text-based person search via attribute-aided matching. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 2617–2625 (2020)
15. Jing, Y., Si, C., Wang, J., Wang, W., Wang, L., Tan, T.: Pose-guided multi-granularity attention network for text-based person search. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 07, pp. 11189–11196, April 2020
16. Niu, K., et al.: Improving description-based person re-identification by multi-granularity image-text alignments. IEEE Trans. Image Process. **29**, 5542–5556 (2020)
17. Xu, N., et al.: MDA: multimodal data augmentation framework for boosting performance on sentiment/emotion classification tasks. IEEE Intell. Syst. **36**(6), 3–12 (2020)
18. Tang, R., Ma, C., Zhang, W.E., Wu, Q., Yang, X.: Semantic equivalent adversarial data augmentation for visual question answering. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) Computer Vision – ECCV 2020. LNCS, vol. 12364, pp. 437–453. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58529-7_26
19. Jiang, D., Ye, M.: Cross-modal implicit relation reasoning and aligning for text-to-image person retrieval. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2787–2797 (2023)
20. Yang, S., Zhou, Y., Zheng, Z., Wang, Y., Zhu, L., Wu, Y.: Towards unified text-based person retrieval: a large-scale multi-attribute and language search benchmark. In: Proceedings of the 31st ACM International Conference on Multimedia, pp. 4492–4501, October 2023
21. Yao, H., Zhang, S., Hong, R., Zhang, Y., et al.: Deep representation learning with part loss for person re-identification. IEEE Trans. Image Process. **28**(6), 2860–2871 (2019)
22. Chen, J., Yang, Z., Yang, D.: MixText: linguistically-informed interpolation of hidden space for semi-supervised text classification (2020). arXiv preprint [arXiv:2004.12239](https://arxiv.org/abs/2004.12239)
23. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2D pose estimation using part affinity fields. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7291–7299 (2017)



Multi-behavior Recommender Model Based on LightGCN

Han Xueying¹ and Yang Yan^{1,2(✉)}

¹ School of Computer Science and Technology, Heilongjiang University, Harbin, China
yangyan@hlju.edu.cn

² Heilongjiang Provincial Key Laboratory of Database and Parallel Computing, Heilongjiang University, Harbin, China

Abstract. Graph convolutional networks have gained traction in recommender systems recently, addressing issues like matrix sparsity. LightGCN simplifies models to avoid overfitting and improve generalization. However, it only considers single behavior, neglecting the impact of multiple behaviors on user preferences. Hence, we propose a multi-behavior recommender based on lightweight graph convolution. We construct a heterogeneous graph capturing various user-item interactions and design a heterogeneous graph attention network. User embeddings from the graph neural network are mapped to different behaviors, enhancing user information mining. Multi-task training enhances model performance, as evidenced by superior results compared to LightGCN and NGCF across multiple datasets.

Keywords: Graph Convolutional Networks · Recommender System · Multi-Behavior · Collaborative Filtering

1 Introduction

In today's digital era, social networks and online platforms play vital roles in information access, communication, and social interaction. These platforms provide rich user behavioral data, presenting both opportunities and challenges for recommendation systems. While Collaborative Filtering [1] struggles with large-scale and sparse data, Graph Neural Networks (GNNs) [2] offer promise.

LightGCN [3], a GNN-based recommendation algorithm, has gained popularity for its efficiency in large-scale scenarios by using lightweight graph convolutions on simplified user-item interaction graphs. However, focusing solely on single user-item behavior limits its ability to capture diverse user interests. To address this, researchers are integrating multi-behavior [5] information, such as ratings, clicks, and purchases, to enhance recommendation accuracy.

Attention [6] Mechanisms have also emerged as significant components in recommendation systems, improving the focus on crucial user-item interactions. This paper aims to integrate LightGCN with multi-behavior data, creating a comprehensive recommendation model that considers both multi-behavior and attention mechanisms to improve user interest capture and recommendation personalization.

The main contributions of this paper are as follows:

1. Our multi-behavior recommender model, based on LightGCN, enhances system accuracy by integrating diverse behavior data.
2. Introducing an attention mechanism allows the model to adaptively learn user-item associations, improving result personalization.
3. Real datasets validate our model, comparing favorably with baseline models, demonstrating significant accuracy advantages.

2 Related Work

2.1 Graph Neural Networks in Recommender Systems

In recent years, Graph Neural Networks (GNNs) gained popularity in recommendation systems for handling data sparsity and scalability by leveraging graph-structured data. KGAT [7] adaptively updates node representations using embedding propagation layers but may face practical performance issues. Various GNN models offer different approaches: GraphSAGE [8] addresses memory concerns with neighbor sampling but has limitations with weighted graphs. PinSage [9] shows good scalability but has higher computational complexity and less sensitivity to sparse data. STAR-GCN [10] focuses on graph data but encounters computational bottlenecks with large-scale data. NGCF [11] emphasizes high-order connectivity but may have limited impact on collaborative filtering performance. Models like GraphRec [13] and DANSER [14] using graph attention networks capture connections and ratings effectively but have preprocessing complexities and data quality constraints. Models based on graph autoencoders (GAE) [15], like AutoSVD++ [16], improve accuracy by capturing input variables but may underutilize relationships due to their structure. Recommendation algorithms based on heterogeneous graph neural networks (Heterogeneous GNN), such as CHCF [17], and HetGNN [18] better consider node content and graph structure heterogeneity but face challenges like over-smoothing and label demands.

2.2 Multi-behavior Recommendation

Traditional recommendation systems often rely on a single type of user behavior data, such as ratings or clicks, limiting diversity and personalization. To address this, researchers have integrated multiple behavior types to enhance performance.

Recently, multi-behavior recommendation models have gained traction. One approach models all behavior types together, treating clicks, purchases, and browsing as separate sequences concatenated into a longer sequence, encoded using a Transformer network. Some studies assign learnable weights to different behaviors to capture their semantics, while others use embedding representations for each behavior type in graph convolution operations [19].

Additionally, methods combining contrastive and self-supervised learning with recommendation systems have emerged. For instance, Fan et al. [20] use contrastive learning to enrich data representations through heterogeneous data convolution, while Xia X et al. [21] treat different user behaviors as distinct graphs, employing self-supervised learning to train graph neural networks with positive and negative samples, thus learning user behavior representations.

2.3 Application of Attention Mechanism in Recommender Systems

Attention mechanisms have achieved remarkable success in various domains like natural language processing and computer vision. Recently, they have been widely adopted in recommendation systems, enabling models to focus on critical user-item interactions by assigning varying importance to different users and items. This enhances personalization, recommendation quality, and interpretability.

Several studies highlight the application of attention mechanisms in recommendation systems. For example, Alibaba's DIN [22] models user interest based on behavior sequences, addressing traditional methods' shortcomings by applying attention weighting to each item's embedding in a user's behavior sequence. NARRE [23] assigns importance weights to reviews based on their quality, reflecting the significance of different behaviors in recommendations. MPCN [24] introduces a novel method for computing attention weights using similarity matrices.

3 Method

3.1 Symbol Definition

The following notations are used to represent different concepts and variables:

U : A collection of users, $U = \{u_1, u_2, \dots, u_{\bar{N}_u}\}$, \bar{N}_u Represents the number of users.

I : A collection of items, $I = \{i_1, i_2, \dots, i_{\bar{N}_i}\}$, \bar{N}_i Represents the number of items.

N_u : A collection of items that user u interacts with, N_i : A collection of users that item i interacts with.

e_u : The represent of user's embedding. e_i : The represent of item's embedding.

θ : The learnable model parameters. Γ : The number of multiple behaviors.

3.2 Model Details

The model is divided into three parts. The embedding layer is the embedding representation of users and items. The propagation layer is a high-order embedded representation on the user-item bipartite graphs. The prediction layer is to predict the interacting likelihood of users and items.

3.2.1 Embedding Layers

We use One-Hot coding as input to implement the function of initializing the embedding layer. The initialized embedding layer is in the user-project two-part diagram and is used to pair the nodes in the diagram and the edges between the user and the project.

As shown in Fig. 1, Graph G is constructed by using the interaction between users and items, where the edges represent the user's purchase, add-on, click and self-connection (the connection between nodes and nodes themselves).

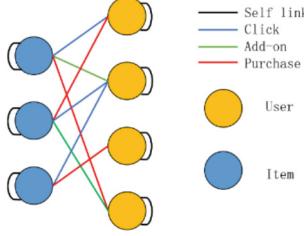


Fig. 1. User-Item bipartite graph: Construct diagrams using user-item, user-user, and item-item interaction behaviors in the diagram and edges between users and projects.

3.2.2 Propagation Layer

(1) *Lightweight Graph Convolutional Network Layer*. To initialize the embedding layer, we use One-Hot encoding to convert node and edge indices into low-dimensional embedding vectors. This provides initial feature representations for the graph convolutional layer, serving as inputs for the k-th order embedding propagation layer. In the user-item bipartite graph, the graph convolutional network (GCN) iteratively aggregates information from neighboring nodes, updating their embeddings and modeling user-item interactions.

The GCN updates node and relation embeddings by defining aggregation functions and update rules, considering neighbors and their relationships. By jointly considering node and relation embeddings, the GCN captures interactions and dependencies, enhancing recommendation accuracy and personalization.

In summary, the GCN in the user-item bipartite graph updates embeddings to effectively model user-item interactions, improving the recommendation algorithm's performance.

$$e_u^{(k+1)} = \sum_{i \in N_u} \frac{1}{\sqrt{|N_u|} \sqrt{|N_i|}} e_i^{(k)} \quad (1)$$

$$e_i^{(k+1)} = \sum_{u \in N_i} \frac{1}{\sqrt{|N_i|} \sqrt{|N_u|}} e_u^{(k)} \quad (2)$$

where Eq. (1) (2) represents the lightweight graph convolution propagation rule, $e_u^{(k)}$ and $e_i^{(k)}$ represents the embedding of the k -layer items and the users, N_u represents the collection of items that interact with the user u , N_i represents the collection of users that interact with the item i , and after the parameters pass through the k layer, the weighted sum obtains the final embedding representation:

$$e_u = \sum_{k=0}^K \alpha_k e_u^{(k)} \quad (3)$$

$$e_i = \sum_{k=0}^K \alpha_k e_i^{(k)} \quad (4)$$

α_k indicates the importance of the k -level, which can be seen as a hyperparameter to adjust manually, or as a learnable model parameter to learn automatically. In the experiment, in order to compare with LightGCN, we also set α_k to $\frac{1}{K+1}$.

- (2) *Graph Attention Mechanism Layer* As shown in the model structure of Fig. 2, the input of the GNN is a set of node features, $e = \{\vec{e}_1, \vec{e}_2, \dots, \vec{e}_N\}$, $\vec{e}_i \in R^F$, where N is the number of nodes and F is the number of features of each node. The output is a new set of node features $e' = \{\vec{e}'_1, \vec{e}'_2, \dots, \vec{e}'_N\}$, $\vec{e}'_i \in R^{F'}$. Considering that the contribution of neighbor nodes to node e_i under different behaviors is different, so for the update of each node, different from the existing way of attention calculation by similarity coefficient, the learnable embedding vector is set for different types of edges, so in order to obtain sufficient expression ability to convert input features into more advanced features to calculate attention, the expression of attention mechanism coefficient is:

$$\alpha_{ij} = \frac{\exp(\text{Leaky ReLU}(a^T[\theta_i e_i \| \theta_k e_j \| \theta_e e_{i,k}])))}{\sum_{k \in N_i} \exp(\text{Leaky ReLU}(a^T[\theta_i e_i \| \theta_k e_j \| \theta_e e_{i,k}])))} \quad (5)$$

In the experiment, the attention mechanism a is a single-layer feedforward neural network parameterized by a weight vector $a \in R^{2F'}$. Among them, the linear mapping of parameters increases the dimension of vertex features and enhances them, $[\cdot \| \cdot]$ indicates that the transformed features of vertices i and j are spliced, and e_{ij} represents the input defined by each edge type. At the same time, for a certain node i , the neighbor node j under all the behavior Γ adjacent to it constitutes e_{ij} , which is also different from other methods in this paper, taking the edge as one of the parameters for calculating the attention coefficient. T indicates transpose.

3.2.3 Prediction Layer

After obtaining the updated feature of each node, the output feature is finally obtained by weighting the input feature:

$$e'_i = \sigma \left(\sum_{j \in N_i} \alpha_{ij} e_j \right) \quad (6)$$

where σ represents the activation function.

When linearly mapping each node, each node has its own mapping matrix, that is, $\theta = \{\theta_i, \theta_e\} \in R^{F \times F'}$, which is to map the user vector obtained from the GNN to the user clicks, add-ons and purchases through their respective mapping matrices. This is more useful for modeling the user's intent in different behaviors.

3.2.4 Model Optimization

1. Loss Function. Unlike LightGCN, this article uses the cross-entropy loss function for training. The final prediction score is:

$$\hat{y}_{ui} = e_u^T e_i \quad (7)$$

The final model loss function is:

$$L(\hat{y}) = - \sum_{i=1}^N y_{ui} \log(\hat{y}_{ui}) + (1 - y_{ui}) \log(1 - \hat{y}_{ui}) \quad (8)$$

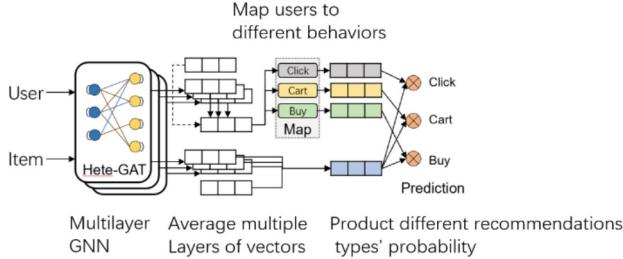


Fig. 2. Model Structure: Pass the embedding of node features to the multilayer GNN, average the vector of the updated node embedding, and map users to different behaviors through different feature mappings to obtain different types of recommender probabilities.

2. *Multi-task Training.* Due to the different effects of different behaviors on user preferences, the weights of each behavior are also different, in order to consider this reality, this project uses multi-task training to optimize the model. For example, the behavior weight of purchase is greater than that of add-on, and the behavior weight of add-on is greater than that of click. Therefore, in this paper, the three tasks of purchase, add-on, and click are used for joint training the final loss of the model. The function is shown in Eq. (9).

$$L = L_{click} \times 0.1 + L_{cart} \times 0.3 + L_{buy} \times 0.6 \quad (9)$$

where L_{click} means click, L_{cart} means add-on, L_{buy} means purchase.

4 Experiments

4.1 Experimental Environment and Dataset

In this experiment, the model enables pytorch implementation, and the GPU model is GeForce RTX 3090. We set the epoch = 5000, learning rate = 0.001, the batch size = 1024, and the embedding vector dimension d = 64. Using Adam as an optimizer to optimize all models, its main advantage is the ability to adapt to the learning rate. To prevent overfitting, L2 regularization is used. Using Fliggy, Taobao and Tmall as experimental comparison datasets, in order to reduce the sparseness of experimental data, the three datasets are filtered out of items and users with fewer than 10, 5, and 3 occurrences (Table 1).

Table 1. Datasets

Datasets Number	users	items	train samples	test samples
Fliggy [25]	19111	26254	246461	19027
Taobao	24174	50068	65165	22994
Tmall [26]	5763	87943	17742	2074

4.2 Evaluation Indicators

We use NDCG and Recall as our evaluation. NDCG (Normalized Discount Cumulative Gain) is a measure of ranking quality and is used to evaluate whether the result sequence provided by search engines are good.

$$DCG_u@K = \sum_{i=1}^K \frac{2^{Rel(i)-1}}{\log_2(i+1)} \quad (10)$$

$$NDCG@K = \frac{1}{|U|} \sum_{u \in U} \frac{DCG_u@K}{IDCG_u@K} \quad (11)$$

where $Rel(i)$ represents $item(i)$, DCG_u considers the influence of the order, so that the item in front increases its influence, and the item that ranks later weakens its influence, $IDCG_u$ arranges according to $Rel(i)$ in descending order, arranges to the best state, the best arranged DCG_u , and finally normalizes it.

Recall rate, the predicted result of the model, is the information that the user is really interested in. The recall is calculated as shown in Eq. (12):

$$Recall = \frac{TP}{TP + FN} \quad (12)$$

where TP (True Positive): indicates that the true category of the sample is positive, and the final predicted result is also positive. FN (False Negative): Indicates that the true category of the sample is positive, and the final prediction result is negative.

Table 2. Convolutional Layer Number Parameter Experiment

Taobao	1	2	3	4	5
Recall	0.1558	0.1539	0.1560	0.1269	0.0661
NDCG	0.0783	0.0787	0.0789	0.0665	0.0216
Fliggy	1	2	3	4	5
Recall	0.8394	0.8381	0.8406	0.7421	0.6677
NDCG	0.4737	0.4705	0.4745	0.4003	0.3520

4.3 Parametric Experiments

To explore the influence of convolution of different layers on the performance of recommender model. As shown in Table 2, experimental results show that when the number of convolutional layers is 3, the experimental effect is the best, so the number of convolutional layers is set to 3 in the experiment to achieve the best model performance.

Table 3. Recall, NDCG

MA	MLA	DTP	Recall			NDCG		
			Fliggy	Tmall	Taobao	Fliggy	Tmall	Taobao
✗	✗	✗	0.0056	0.0050	0.0060	0.0013	0.0012	0.0015
✓		✗	0.8029	0.1579	0.1394	0.4388	0.1047	0.0734
✓	✓	✗	0.8373	0.1603	0.1413	0.4511	0.1072	0.0758
✓	✓	✓	0.8406	0.1646	0.1560	0.4745	0.1078	0.0789

4.4 Ablation Experiment

To verify the effectiveness of the model, it is explored whether the user vectors obtained from the GNN can be mapped to the user clicks, add-ons, and purchases through the mapping matrix to model the user intention in different behaviors(MA). At the same time, it is explored whether the performance of the model can be improved by setting learnable embedding vectors for different types of edges to represent the information of edge types (DTP) and whether the average operation of features extracted from multi-layer GNN has a positive impact on the performance improvement of the model (MLG), an ablation experiment is set up.

By continuously increasing these three strategies, the two performance indicators of Recall and NDCG of the model in four situations are compared. As shown in Table 3, after gradually adding strategies to the model, the evaluation indicators are improved.

Table 4. Comparative Experiments

	Tmall		Taobao		Fliggy	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
NGCF [11]	0.0547	0.0225	0.0547	0.0225	0.2115	0.0765
LightGCN [3]	0.1030	0.0510	0.1030	0.0510	0.2474	0.0888
IMP-GCN [26]	0.0650	0.0302	0.0650	0.0302	0.3065	0.1206
Ours	0.1646	0.1078	0.1560	0.0789	0.8406	0.4745

4.5 Algorithm Comparison

In past studies, NGCF introduced an attention mechanism to learn the embedded vectors of user-item interaction graphs. LightGCN focuses more on dealing with data sparseness in recommender systems, using traditional graph convolution operations to learn the embedding vector of user-item interaction graphs. IMP-GCN introduces a dynamic attention mechanism to transfer information according to the similarity and importance between nodes, so that different nodes obtain different propagation weights. Therefore,

this study is based on the above methods, designs algorithm comparison experiments, this study not only considers the problem of data sparseness, but also considers the importance between nodes and it between deeper mining nodes, as shown in Table 4, compared with the other three methods, the performance of this study on several datasets are excellent.

5 Conclusion

This paper proposes a multi-behavior recommendation system based on LightGCN, utilizing various user-item interaction behaviors to construct richer representations. It employs a heterogeneous graph attention network with learnable embedding vectors for different edges and separately maps user vectors from the graph neural network, enhancing the model's ability to capture user intentions across different behaviors. By incorporating multi-task training methods and weighting different behavior objectives, it improves overall model performance. Future research can further explore the motivations and psychological factors behind user behavior to enhance personalization and user experience in recommendation systems. Despite progress in multi-behavior recommendation systems, gaps remain, and this study aims to address them by investigating user behavior motivations for designing more effective recommendation algorithms.

References

1. He, X., Liao, L., Zhang, H., et al.: Neural collaborative filtering. In: International World Wide Web Conferences Steering Committee (2017)
2. Zhou, J., Cui, G., Zhang, Z., et al.: Graph neural networks: a review of methods and applications. Cornell University - arXiv (2018)
3. He, X., Deng, K., Wang, X., et al.: LightGCN: simplifying and powering graph convolution network for recommendation. ACM (2020)
4. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)
5. Xia, L., Xu, Y., Huang, C., et al.: Graph meta network for multi-behavior recommendation. ACM (2021)
6. Vaswani, A., Shazeer, N., Parmar, N., et al.: Attention is all you need. In: Neural Information Processing Systems (2017)
7. Wang, X., He, X., Cao, Y., et al.: KGAT: knowledge graph attention network for recommendation. ACM (2019)
8. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Neural Information Processing Systems (2017)
9. Ying, R., He, R., Chen, K., et al.: Graph convolutional neural networks for web-scale recommender systems. ACM (2018). [https://doi.org/10.1145/3219819.3219890\[P\]](https://doi.org/10.1145/3219819.3219890[P])
10. Zhang, J., Shi, X., Zhao, S., et al.: STAR-GCN: stacked and reconstructed graph convolutional networks for recommender systems. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (2019)
11. Wang, X., He, X., Wang, M., et al.: Neural graph collaborative filtering. ACM (2019)
12. Velikovi, P., Cucurull, G., Casanova, A., et al.: Graph attention networks. arXiv: Machine Learning (2017)

13. Fan, W., Yao, M., Li, Q., et al.: Graph neural networks for social recommendation. In: The World Wide Web Conference (2019)
14. Wu, Q., Zhang, H., Gao, X., et al.: Dual graph attention networks for deep latent representation of multifaceted social effects in recommender systems. In: The World Wide Web Conference (2019)
15. Kipf, T.N., Welling, M.: Variational graph auto-encoders. arXiv: Machine Learning (2016)
16. Zhang, S., Yao, L., Xu, X.: AutoSVD++: an efficient hybrid collaborative filtering model via contractive auto-encoders. ACM (2017)
17. Wang, H., Xiao, B., Wang, L., et al.: CHCF: a cloud-based heterogeneous computing framework for large-scale image retrieval. IEEE Trans. Circuits Syst. Video Technol. **25**(12), 1900–1913 (2015)
18. Zhang, C., Song, D., Huang, C., Swami, A., Chawla, N.V.: Heterogeneous graph neural network. SIGKDD Explor. (2019)
19. Gao, C., He, X., Gan, D., et al.: Neural multi-task recommendation from multi-behavior data. In: 2019 IEEE 35th International Conference on Data Engineering (ICDE). IEEE (2019)
20. Wei, F., Wei, Z., Junhao, W.: Recommendation of dual channel cross adaptive comparative learning based on heterogeneous graphs [J/OL]. J. Electron., 1–10 (2023)
21. Xia, X., Yin, H., Yu, J., et al.: Self-supervised hypergraph convolutional networks for session-based recommendation. In: Proceedings of the AAAI Conference on Artificial Intelligence (2020)
22. Zhou, G., Mou, N., Fan, Y., et al.: Deep interest evolution network for click-through rate prediction. Proc. AAAI Conf. Artif. Intell. **33**, 5941–5948 (2019). <https://doi.org/10.1609/aaai.v33i01.33015941>
23. Chen, C., Zhang, M., Liu, Y., et al.: Neural attentional rating regression with review-level explanations. In: The 2018 World Wide Web Conference (2018). <https://doi.org/10.1145/3178876.3186070>
24. Tay, Y., Tuan, L.A., Hui, S.C.: Multi-pointer co-attention networks for recommendation. arXiv e-prints (2018). <https://doi.org/10.1145/3219819.3220086>
25. Tao, W., et al.: SMINet: state-aware multi-aspect interests representation network for cold-start users recommendation. In: Proceedings of the AAAI Conference on Artificial Intelligence (2022)
26. Liu, F., Cheng, Z., Zhu, L., Gao, Z., Nie, L.: Interest-aware message-passing GCN for recommendation (2021)



SSDC-Net: An Effective Classification Method of Steel Surface Defects Based on Salient Local Features

Qifei Hao¹, Qingsong Gan², Zhe Liu¹, Jun Chen¹, Qi Shen², Chengxuan Qian¹, and Yi Liu^{1(✉)}

¹ Jiangsu University, Zhenjiang 212013, Jiangsu, China
ly@ujs.edu.cn

² Baoshan Iron and Steel Co. Ltd., Shanghai, China

Abstract. To effectively classify steel surface defects, it is crucial to improve the feature representation. The features extracted by the current steel surface defect classification model inadequately represent different defect categories, particularly when samples from different categories exhibit high similarity, resulting in degraded model performance. In this paper, we propose the Steel Surface Defect Classification Network (SSDC-Net) to optimize the steel surface defect classification task. This network includes the Channel Information Complementary Matrix (CICM) module and the Multi-Scale Contrastive Loss (MCL) method, which models the channel information between similar samples to capture and distinguish subtle visual differences induced by distinct channel information. Specifically, the CICM module utilizes interactive information between channels of the feature map to enhance local feature representation. To enhance global feature representation, we further introduce a multi-scale contrastive loss, which not only maximizes differences between features but also reduces the gap between different views of the same sample. Extensive experiments conducted on the BG-DET dataset, FSC-20 dataset, and FSD dataset demonstrate the effectiveness of each proposed module. Compared with other methods, our approach has achieved state-of-the-art performance. Additionally, our method is plug-and-play and can be seamlessly integrated into other backbone networks.

Keywords: steel surface defect classification · contrastive learning · feature representation

1 Introduction

Automated steel surface defect classification plays a crucial role in optimizing production lines and enhancing production processes [11]. In industry, effectively categorizing steel surface defects enables early detection and identification, facilitating the development of appropriate maintenance and repair programs to enhance product quality and durability. Traditional defect classification methods, such as vibration-acoustic modulation [25] and wireless sensing technology [26], rely on changes in vibration or wireless signals on

the surface of objects to identify defects. However, these methods are labor-intensive and slow down industrial production processes. To address these issues, deep learning-based methods offer a promising solution for steel surface defect classification by automatically extracting key features from data without the need for manual feature engineering [8, 21]. Consequently, automatic defect inspection has emerged as a vital research direction to replace manual labor in industrial manufacturing [10, 30].

One of the main challenges in automatically classifying steel surface defects lies in the significant structural similarity between certain defects, which contributes to the difficulty of recognition and distinction. This similarity primarily arises from the common features extracted from these defects, leading to the confusion of semantic information and complicating the classification task.

In recent years, automated methods for steel surface defect classification have been proven to be highly effective, yet there is still room for improvement. 1) A GAN-based [18] approach was proposed as a method to generate data for a limited number of categories, thereby enriching the semantic information of the samples. 2) Methods based on attention mechanisms can help models focus more on specific parts of the input data, allowing the network to capture important features more effectively [9, 21, 24, 27]. 3) Feature engineering-based approach utilizes feature fusion to extract more robust features to classify steel images [17]. However, while these classification methods aim to extract richer semantic information, none of them adequately addressed the classification errors stemming from the confusion of semantic information among defective features. To tackle the aforementioned challenges, we have conducted a thorough investigation into feature learning and model design, aiming to address these similarities and enhance sensitivity to subtle differences in classification. We argue that classification of similar defects can be accomplished by enhancing the local structural information when different defects have similar global semantic information. So we propose a new framework called SSDC-Net, which contains two main modules, the Channel Information Complementary Matrix (CICM) module enriches the local semantic information of defects by enhancing the different channel features of defects and then utilizes a Multi-Scale Contrastive Loss (MCL) module to distinguish the semantic information of similar defect features. In summary, our contributions are as follows:

- We propose the SSDC-Net, which is capable of accomplishing automatic classification of steel surface defects with a certain degree of real-time performance.
- We propose a CICM module that can obtain better semantic information by highlighting local structural information.
- We propose a contrastive learning positive sample construction method that can better represent current defective features, and distinguish different defective features without a new network structure.

2 Related Work

2.1 Classification of Steel Surface Defects

In recent years, with the development of computer vision, vision-based Automatic Surface Detection Systems (ASIS) have been increasingly researched and applied [34]. Typical steel surface defect detection systems can be roughly divided into traditional

methods and deep learning-based methods [22, 33]. Traditional learning methods typically involve feature extraction and classifier debugging. Some of the popular methods include Support Vector Machine (SVM) [32], Back Propagation (BP) neural network [37], and Decision tree [1].

In general, these methods will cause serious consumption of time and resources when dealing with large-scale datasets, and slight fluctuations in the input data will exacerbate the unstable model. As for the deep learning-based methods for steel surface defect classification tasks, there are three common methods: Vision Transformers (ViT) based self-attention mechanism, Generative adversarial network-based (GAN) data generation, and Feature Fusion [7]. Li *et al.* [28] proposed a lightweight network based on coordinate attention and self-interaction (CASI-Net), which embeds position information into channel attention. Data generation is mainly represented by methods such as AutoEncoder (such as GAN [6] *etc.*), which generate data from data to make it similar to the training data in overall or high-level semantics [20]. Fu *et al.* [12] proposed a multi-scale feature fusion method, including feature splicing [17], feature summation [35], corresponding element multiplication [40] and skip connection [19] *etc.*, which can make defect features more comprehensive. However, none of the above methods take into account the fact that some defect samples have different semantic information but similar structures, which makes errors in classifying steel surface defects.

2.2 Contrastive Learning

Contrastive learning originated from self-supervised learning. Wu *et al.* [38] proposed an individual discriminating task: each instance is regarded as one single category, and the goal of self-supervised learning is achieved by setting the instance discriminating task. In recent years, deep learning-based contrastive learning techniques have rapidly evolved, with SimCLR [4], MoCo [15], BYOL [13], SwAV [2], and SimSiam [5] being the most popular. These algorithms typically rely on the network architecture of quasi-twin neural networks and employ image pairs of the same image after data augmentation (positive sample pairs) and different images after data augmentation (negative sample pairs) during the training process. Using these deep contrastive learning techniques, a large number of positive and negative sample pairs are compared, with the positive samples brought closer to each other in the feature space and the negative samples pushed further apart. This allows the neural network model to automatically extract better feature representations of the data. Our network introduces a multi-scale contrastive learning algorithm, which does not depend on twinned neural networks for positive sample construction.

3 Method

3.1 SSDC-Net Framework

SSDC-Net is a simple pipeline based mostly on the powerful baseline Yolov8¹ design. The overall architecture of SSDC-Net is shown in Fig. 1. Specifically, the network consists of nine modules, with the first layer being a convolution module and the remaining

¹ <https://github.com/ultralytics/ultralytics?tab=readme-ov-file>.

layers being $4 \times (\text{Conv} + \text{C2f})$ modules. To improve the performance of the network, (1) After the last three C2f modules, each layer is added with the CICM module. (2) After each CICM module, an MLP layer is added to compute the multi-scale contrastive learning loss. The two distinct designs mentioned above enable SSDC-Net to successfully inherit the advantages of Yolov8 in image classification and perform even better in the task of steel surface defect classification.

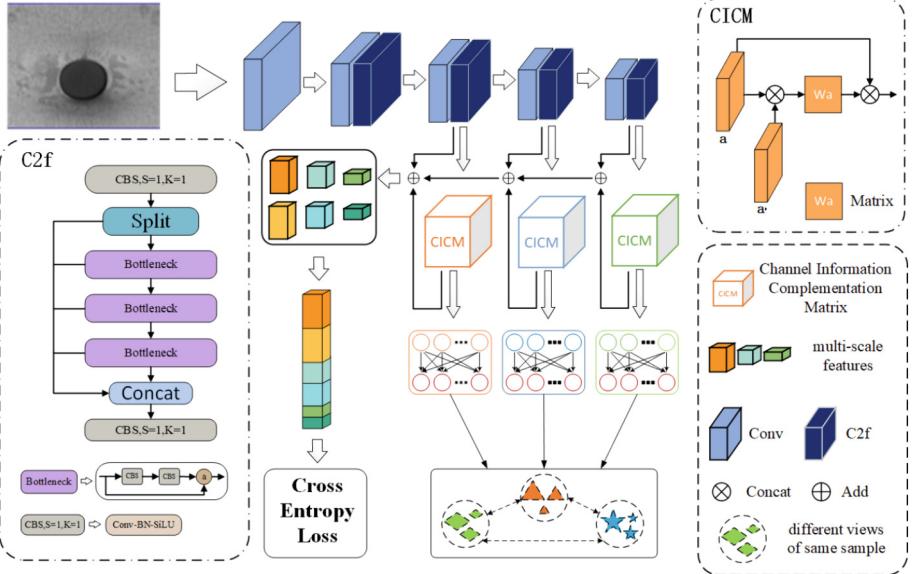


Fig. 1. The proposed SSDC-Net network architecture consists of backbone network Yolov8, CICM module, and multi-scale contrastive loss. The image is first passed through the feature extractor and then enters the CICM module for local detail feature extraction and contrast loss calculation. Afterward, the enhanced features are combined with the original features to calculate the classification loss. This process helps in accurately identifying and classifying objects in the image.

3.2 Channel Information Complementation Matrix for Enhancing Local Information of Global Features

To capture more discriminative features, the Channel Information Complementary Matrix (CICM) was introduced. It focuses on local information in each channel feature map. The specific method is as follows. Given an image I , let $X \in R^{b \times c \times w \times h}$ represent the feature map output by one C2f layer of the backbone network. A dimension is added to the feature map, transforming X into $X' \in R^{b \times c' \times c \times w \times h}$ (c' is the added dimension). Subsequently, X' is reshaped to $X^T \in R^{b \times c \times c' \times w \times h}$. The computation of the CICM output is then performed as follows:

$$Y = X'X^T \in R^{b \times c \times c \times w \times h} \quad (1)$$

where Y represents the result of the complementary information of any two channels of the feature map. The total value of the interaction result $t = w \times h \in R^{I \times I}$ between each pair of channels is also computed. Finally, the Softmax function is then used to normalize Y according to the first dimension to obtain the information complement matrix W :

$$W_{ij} = \frac{\exp(-X'X_{ij}^\top)}{\sum_{k=1}^c \exp(-X'X_{ik}^\top)} \in R^{b \times c \times c \times t} \quad (2)$$

according to the definition of W , taking the i -th channel as an example, $W_{ic} \in R^{b \times i \times c \times t}$ denotes the result of interacting and normalizing the information of the i -th channel with all c channels. In addition, the matrix needs to be applied to the features:

$$Z = W_{ic}X \quad (3)$$

where $\sum_{k=1}^c W_{ik} = 1$. It is worth noting that Y_i (the i -th channel of the resulting features Y) is the computed interaction between X_i and all the channels of X , $Y_i = \sum_{k=1}^c W_{ik}X_k$. Our CICM module focuses more on local features that can highlight defect information, potentially sacrificing some global feature formation in the process. We aggregate the enhanced features with the original features X as the final feature F for this sample:

$$F = \sum_{k=1}^3 (Z_k + X_k) \quad (4)$$

Our approach places greater emphasis on exploiting complementary information across channels, eliminating the need to determine the importance of each channel for the task at hand and assign weights, as required by channel-focused mechanisms. This strategy focuses on enhancing distinguishing features rather than merely amplifying positive ones and suppressing negative ones.

3.3 Multi-scale Contrastive Loss for Distinguishing Feature Information

Features at different scales contain different detail information. Therefore, a multi-scale feature fusion approach is adopted to achieve a more comprehensive feature representation, enhancing the discrimination of similar defects. After augmenting the three scales of features obtained from the backbone using the CICM module, another module is introduced: Multi-scale Contrastive Loss. Once the image is inputted to the backbone network and processed by the CICM module, three sets of enhanced features (represented as X_1 , X_2 and X_3) are obtained. These representations are then projected into a 128-dimensional potential space using a single-layer MLP [5] projection head. A batch of N samples is randomly selected, and a comparison prediction task is defined for the multi-scale samples in the batch, resulting in $3N$ data points. We consider different scale views of each sample to be semantically similar, so we define them as positive samples. In addition, we treat the other $3(N - 1)$ samples in the batch as negative samples. We create a different view of each positive case from a negative sample pair with each negative case. Then, the loss function of a pair of positive examples (i, j, k) is defined as:

$$\mathcal{L}_q = -\log \frac{\exp(i \cdot j \cdot \frac{k}{\tau})}{\sum_{n=0}^N \exp((i + j + k) \cdot \frac{q_{n1} + q_{n2} + q_{n3}}{\tau})} \quad (5)$$

where τ [36, 38] represents the temperature coefficient, which is used to control the degree of attention given to difficult samples. In Eq. 5, i, j, and k represent different views of the current sample, respectively q_{n1} , q_{n2} , and q_{n3} are different views of other samples. Our contrastive learning loss is akin to InfoNCE [14, 15, 23], albeit with a distinction in the definition of positive and negative samples. In this paper, multi-scale features are considered positive samples, and only a single feature extraction network is needed. This significantly reduces the amount of parameter calculation. Additionally, the negative sample diversity can still be maintained to a certain extent even when using a smaller batch size, due to the multiple views.

3.4 Total Loss

Additionally, we utilize cross-entropy loss for classification, which is based on the model's predictions. We refer to this loss as L_{cro} . Our framework's overall loss is defined as follows:

$$L_{cro} = -\frac{1}{n} \sum_{i=1}^n \left(\sum_{j=1}^C y_{i,j} \log(p_{i,j}) \right) \quad (6)$$

the equation for calculating the probability of the i -th sample belonging to the j -th class is $p_{i,j}$, where n is the batch size, C is the number of classes, $y_{i,j}$ is the true label of the i -th sample on the j -th class, and $p_{i,j}$ is the prediction of the j -th class for the i -th sample.

$$loss = \alpha \times L_{cro} + (1 - \alpha) \times \mathcal{L}_q \quad (7)$$

Section 4.6 provides a detailed analysis of the hyperparameter α used to optimize loss with the stochastic gradient method.

4 Experiments

We report the experimental results, and compare our method with the state-of the-art approaches.

4.1 Datasets and Baselines

Baselines. In the experiments, we compare our SSDC-Net with the 6 methods described as follows. (1) SMPConv [20]: presenting self-moving point representations where weight parameters freely move, and interpolation schemes are used to implement continuous functions. (2) EMO-1M [41]: extending CNN-based IRB to attention-based models and abstracting a one-residual Meta Mobile Block (MMB) for lightweight model design. (3) FasterNet [3]: Spatial features can be extracted more efficiently by simultaneously reducing redundant computations and memory accesses. (4) ResNet [16]: Using Residual Module and Residual Connection to Build Networks. (5) Tip-Adapter [42]: Strong few-classification performance by using the cache model to directly set the weights of the Adapter. (6) Yolov8: integrating efficient feature extraction and robust classification capabilities, facilitating accurate and real-time classification tasks.

BG-DET. The BG-DET dataset comprises 23 types of defects, including protective slag, punched holes, and grain, among others. These defect samples are confirmed by 2 to 3 senior engineers, and the number and size of samples for each type differ. The dataset includes a maximum of 1,000 samples for some classes and a minimum of only 200 samples for others.

FSC-20 [23]. Curated by Northeastern University, is publicly accessible and comprises 20 defect classes, with 50 images per class. Each sample is of resolution 224×224 and has been jointly validated by the involved engineers.

MSD [39]. Curated by Northeastern University, is publicly accessible and comprises 10 types of steel surface defects and 10 types of aluminum surface defects. Each type of steel defect contains 50 images of size 224×224 . the size of aluminum surface defects is 256×256 , and the number of pictures in each type ranges from 22 to 48. And has been jointly validated by the involved engineers.

All the experiments in this study were conducted on these 3 datasets, with 80% of the training set and the rest being the validation set. We evaluated the classification performance of the model across various datasets using different metrics such as accuracy and training time. The training time was utilized to analyze the complexity of the model.

4.2 Implementation Details

We employed Yolov8 as the foundational network architecture for all our experiments, with an input image with the size of 224×224 , consistent with most state-of-the-art fine-grained categorization approaches. To ensure reliable results, we set a random seed to minimize the impact of randomness. Our experiments were conducted on a server equipped with an Intel(R) Xeon(R) Gold 5222 CPU and a GeForce RTX A6000 GPU. The training settings utilized in this study are as follows: we employed the cosine annealing method [29] for learning rate adjustment, initializing with a learning rate of 0.001. For the loss function, we employed the cross-function with label smoothing, with a label smoothing coefficient of 0.1. Adam optimizer was utilized with a weight decay attenuation factor of $5e-5$. Our method commenced training with pre-trained weight files, and the reported experimental results on the two datasets are based on the training dataset.

4.3 Comparison with State-of-the-Art Methods

Table 1 lists the results of the comparison between our method and state-of-the art classification methods. We evaluated the performance of each model using metrics such as average accuracy. First, our proposed SSDC-Net has achieved commendable accuracy in a relatively short training time. Specifically, it achieves a high accuracy of 87.3%, which is 4.3% and 2.3% higher than the SMPConv and EMO-1M methods. In addition, on the BG-DET dataset, our method out performs the other networks with an accuracy of 77%, while the ResNet network has an accuracy of less than 70%, and our method

achieves an accuracy of 93%, outperforming other comparative methods on the MSD dataset. We attribute the superior performance of our method to the CICM module, which effectively highlights important regions characterized by different samples, an ability that other models lack.

Table 1. Comparison with state-of-the-art methods on 3 datasets.

Dataset	Model	ACC	Precision	Recall	F1	Time(hours)
FSC-20	<i>SMPConv</i> [20]	0.830	0.800	0.770	0.780	1.5
	<i>EMO-IM</i> [41]	0.850	0.830	0.820	0.820	1.2
	<i>Yolov8s</i>	0.856	0.850	0.800	0.820	0.3
	Ours	0.873	0.830	0.840	0.870	0.3
MSD	<i>FasterNet</i> [3]	0.550	0.500	0.600	0.550	0.3
	<i>ResNet50</i> [16]	0.475	0.482	0.440	0.460	0.3
	<i>Yolov8s</i>	0.900	0.877	0.853	0.860	0.3
	Ours	0.930	0.870	0.890	0.910	0.3
BG-DET	<i>ResNet50</i> [16]	0.670	0.602	0.542	0.570	80
	<i>ResNet101</i> [16]	0.650	0.582	0.569	0.575	85
	<i>SMPConv</i> [20]	0.732	0.648	0.639	0.643	18
	<i>Tip-Adapter</i> [42]	0.260	0.150	0.174	0.161	13.5
	<i>EMO-IM</i> [41]	0.720	0.663	0.631	0.647	21.3
	Ours	0.770	0.696	0.719	0.707	1.2

4.4 Ablation Analysis

Table 2. Validation of each module on the 3 datasets

Model	dataset	ACC	dataset	ACC	dataset	ACC
Yolov8s	BG-DET	0.730	FSC-20	0.810	MSD	0.900
Yolov8s + CICM		0.761		0.846		0.924
Yolov8s + MCL		0.745		0.832		0.915
Yolov8s + CICM + MCL		0.770		0.873		0.930
Yolov8x	BG-DET	0.760	FSC-20	0.821	MSD	0.910
Yolov8x + CICM		0.773		0.848		0.918
Yolov8x + MCL		0.764		0.837		0.910
Yolov8x + CICM + MCL		0.781		0.861		0.921

Table 2 shows the effect of each of our components on our experiments, which were conducted on two baselines and datasets to test the effect of each component separately.

CICM Module. The Channel Information Complementation Matrix (CICM) enhances the learning of localized detailed features by leveraging inter action information between channels. As presented in Table 2, on the BG-DET dataset, the inclusion of the CICM module alone with Yolov8s + SCI improves performance by 3.1% compared to using Yolov8s alone (73.0%), while Yolov8x + SCI with the CICM module alone enhances performance by 3.6% compared to Yolov8x alone (81.0%). On the FSC-20 public dataset, the addition of the CICM module results in performance improvements of 3.6% and 2.7% for Yolov8s and Yolov8x, respectively. On the MSD dataset, the addition of the CICM module improved the performance of Yolov8s and Yolov8x by 2.4% and 0.8%, respectively. Compared to the two, the improvement brought by the combination of the CICM module with the former is significantly larger than that of the latter, which may be because the deeper the model extracts the richer information that the features possess. This underscores the significant role of CICM in enhancing system performance, effectively advancing defect classification by better exploration and utilization of inter-channel information.

MCL Module. We further investigated the effectiveness of the proposed Multi-scale Contrastive Learning (MCL) module. As shown in Table 2, on the BG-DET dataset, the inclusion of the MCL module (Yolov8s + MCL) improves performance by 1.5% compared to the method without contrastive loss (Yolov8s). Similarly, on Yolov8x, Yolov8x + MCL enhances performance by 0.4% over Yolov8x. The improvement of Yolov8s + CICM + MCL is limited compared to Yolov8s + CICM (76.1% vs 77.0%). On the publicly available dataset FSC-20 and MSD, Yolov8 + CICM + MCL also exhibits very limited improvement compared to Yolov8 + CICM. This limitation may be attributed to the use of fewer negative samples in multi-scale contrastive learning loss, which could diminish its ability to discern significant differences between two images. In Table 4, we further explore the effect of the number of negative samples in multi-scale contrastive learning loss on the experiment.

4.5 CICM Analysis for Enhancing Local Structural Detail Information

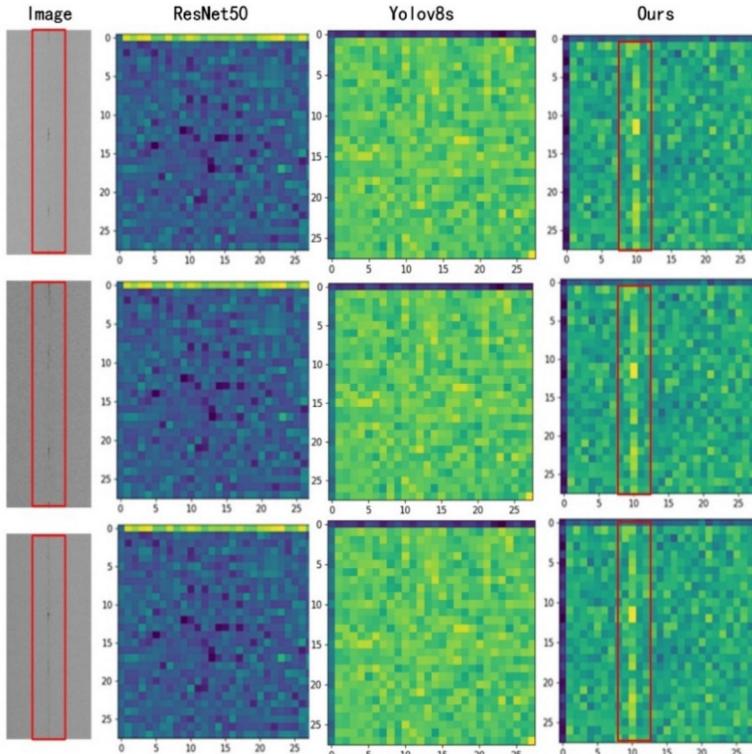


Fig. 2. Visualization of extracted features from some models. (Color figure online)

Figure 2 illustrates the feature maps extracted by ResNet50, Yolov8s, and our proposed method. The original image contains a defect, indicated by a red box. However, the defect location is not clearly discernible in the feature maps extracted by ResNet50 and Yolov8s. In contrast, our method distinctly highlights the defect's location within the red box.

4.6 Hyperparameter Analysis

We manually tune the parameter α to optimize its value, restricting it between 0 and 1, thereby balancing the joint training of the two losses. Table 4 shows the effect of the parameter α on the experiment, and it can be seen that the model is most accurate when α is set to 0.8 and 0.9. As α decreases, the accuracy of the model also declines. This is attributed to the increase in the weight of the contrastive learning loss, which leads to a widening of the gap between defects. Our goal with contrastive learning loss is to reduce the variability of features at different scales within the same sample, while simultaneously

Table 3. Exploring the effect of different batch-size values on experimental results on the 3 datasets.

Model	Batch-size	dataset	ACC	dataset	ACC	dataset	ACC
SSDC-Net	16	BG-DET	0.767	FSC-20	0.865	MSD	0.767
	32		0.765		0.872		0.921
	64		0.770		0.873		0.930
	128		0.777		0.875		0.833
	256		0.781		0.881		0.933

Table 4. Exploring the effect of different α values on experimental results on the BG DET and FSC-20 dataset.

dataset	α	ACC	α	ACC
BG-DET	1.0	73.0%	0.5	75.4%
	0.9	76.5%	0.4	74.2%
	0.8	77.7%	0.3	73.5%
	0.7	76.7%	0.2	72.0%
	0.6	76.1%	0.1	71.6%
dataset	α	ACC	α	ACC
FSC-20	1.0	83.5%	0.5	86.5%
	0.9	86.3%	0.4	81.2%
	0.8	87.3%	0.3	85.0%
	0.7	87.1%	0.2	86.9%
	0.6	87.2%	0.1	83.2%
dataset	α	ACC	α	ACC
MSD	1.0	90.0%	0.5	89.0%
	0.9	96.0%	0.4	88.0%
	0.8	93.0%	0.3	87.6%
	0.7	88.0%	0.2	87.9%
	0.6	89.0%	0.1	87.4%

increasing the variability between features at different scales across different samples. Therefore, we conducted experiments with different batch sizes, as outlined in Table 3. As depicted in the table, accuracy improves as the batch size increases. However, there is a slight drop in accuracy when the batch size is 32. This table confirms that as the batch size and the number of negative samples increase, the samples become more independent of each other, leading to a better feature representation.

5 Conclusion

This paper delves into the challenges associated with employing traditional deep learning methods for classifying steel surface defects. The subtle inter-class distinctions within different industrial steel surface defect datasets pose difficulties in achieving accurate classification. Leveraging the intricate relationships between channels aids in capturing these distinctions, as each channel corresponds to distinct semantics. To address this, we propose an enhanced Yolov8 network modeling algorithm that relies on feature enhancement by mining channel information from similar samples. The network effectively models mutual channel information among similar samples, emphasizing crucial local features to discern similar defects and accentuate feature disparities. Comparative experiments demonstrate that our proposed algorithm surpasses ResNet, EMO-1M, and Yolov8-based algorithms in terms of prediction accuracy and speed. Moving forward, our future endeavors will focus on further validating the model's generalization performance and leveraging optimization algorithms and device adaptation to develop a comprehensive framework for steel surface defect classification.

References

1. Barros, B., Conde, B., Cabaleiro, M., Riveiro, B.: Design and testing of a decision tree algorithm for early failure detection in steel truss bridges. *Eng. Struct.* **289**, 116243 (2023)
2. Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., Joulin, A.: Unsupervised learning of visual features by contrasting cluster assignments. In: *Advances in Neural Information Processing Systems*, vol. 33, pp. 9912–9924 (2020)
3. Chen, J., et al.: Run, don't walk: chasing higher flops for faster neural networks. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12021–12031 (2023)
4. 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)
5. Chen, X., He, K.: Exploring simple Siamese representation learning. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15750–15758 (2021)
6. Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., Bharath, A.A.: Generative adversarial networks: an overview. *IEEE Sig. Process. Mag.* **35**(1), 53–65 (2018)
7. Dai, Y., Gieseke, F., Oehmcke, S., Wu, Y., Barnard, K.: Attentional feature fusion. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3560–3569 (2021)
8. Di, H., Ke, X., Peng, Z., Dongdong, Z.: Surface defect classification of steels with a new semi-supervised learning method. *Opt. Lasers Eng.* **117**, 40–48 (2019)
9. Dosovitskiy, A., et al.: An image is worth 16x16 words: transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020)
10. Essid, O., Laga, H., Samir, C.: Automatic detection and classification of manufacturing defects in metal boxes using deep neural networks. *PLoS ONE* **13**(11), e0203192 (2018)
11. Fang, X., Luo, Q., Zhou, B., Li, C., Tian, L.: Research progress of automated visual surface defect detection for industrial metal planar materials. *Sensors* **20**(18), 5136 (2020)
12. Fu, G., et al.: A multi-scale pooling convolutional neural network for accurate steel surface defects classification. *Front. Neurorobot.* **17**, 1096083 (2023)

13. Grill, J.B., et al.: Bootstrap your own latent-a new approach to self-supervised learning. In: Advances in Neural Information Processing Systems, vol. 33, pp. 21271–21284 (2020)
14. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006), vol. 2, pp. 1735–1742. IEEE (2006)
15. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9729–9738 (2020)
16. 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)
17. He, Y., Song, K., Meng, Q., Yan, Y.: An end-to-end steel surface defect detection approach via fusing multiple hierarchical features. *IEEE Trans. Instrum. Meas.* **69**(4), 1493–1504 (2019)
18. Jin, G., Liu, Y., Qin, P., Hong, R., Xu, T., Lu, G.: An end-to-end steel surface classification approach based on EDCGAN and MobileNet V2. *Sensors* **23**(4), 1953 (2023)
19. Khan, U., Khan, M., Elsaddik, A., Gueaieb, W.: DDNet: diabetic retinopathy detection system using skip connection-based upgraded feature block. In: 2023 IEEE International Symposium on Medical Measurements and Applications (MeMeA), pp. 1–6. IEEE (2023)
20. Kim, S., Park, E.: SMPConv: self-moving point representations for continuous convolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10289–10299 (2023)
21. Krichen, M.: Convolutional neural networks: a survey. *Computers* **12**(8), 151 (2023)
22. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, vol. 25 (2012)
23. Zhao, W., Song, K., Wang, Y., Liang, S., Yan, Y.: FaNet: feature-aware network for few shot classification of strip steel surface defects. *Measurement* **208**, 112446 (2023)
24. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. *Proc. IEEE* **86**(11), 2278–2324 (1998)
25. Li, N., Wang, F., Song, G.: New entropy-based vibro-acoustic modulation method for metal fatigue crack detection: an exploratory study. *Measurement* **150**, 107075 (2020)
26. Li, Q., Chen, J., Zhao, L.: Research on an improved metal surface defect detection sensor based on a 3D RFID tag antenna. *J. Sens.* **2020**, 1–13 (2020)
27. Li, S., Wu, C., Xiong, N.: Hybrid architecture based on CNN and transformer for strip steel surface defect classification. *Electronics* **11**(8), 1200 (2022)
28. Li, Z., Wu, C., Han, Q., Hou, M., Chen, G., Weng, T.: CASI-Net: a novel and effect steel surface defect classification method based on coordinate attention and self-interaction mechanism. *Mathematics* **10**(6), 963 (2022)
29. Loshchilov, I., Hutter, F.: SGDR: stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983* (2016)
30. Luo, Q., et al.: Surface defect classification for hot-rolled steel strips by selectively dominant local binary patterns. *IEEE Access* **7**, 23488–23499 (2019)
31. Luo, Q., Sun, Y., Li, P., Simpson, O., Tian, L., He, Y.: Generalized completed local binary patterns for time-efficient steel surface defect classification. *IEEE Trans. Instrum. Meas.* **68**(3), 667–679 (2018)
32. Nguyen, T.C., Tien, D.H., Nguyen, B.N., Hsu, Q.C.: Multi-response optimization of milling process of hardened S50C steel using SVM-GA based method. *Metals* **13**(5), 925 (2023)
33. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014)
34. Tang, B., Chen, L., Sun, W., Lin, Z.K.: Review of surface defect detection of steel products based on machine vision. *IET Image Process.* **17**(2), 303–322 (2023)

35. Vielzeuf, V., Lechervy, A., Pateux, S., Jurie, F.: CentralNet: a multilayer approach for multi-modal fusion. In: Leal-Taixé, L., Roth, S. (eds.) Computer Vision – ECCV 2018 Workshops. LNCS, vol. 11134, pp. 575–589. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-11024-6_44
36. Wang, F., Liu, H.: Understanding the behaviour of contrastive loss. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2495–2504 (2021)
37. Wang, L., et al.: A creep life prediction model of P91 steel coupled with back-propagation artificial neural network (BP-ANN) and θ projection method. Int. J. Press. Vessels Piping **206**, 105039 (2023)
38. Wu, Z., Xiong, Y., Yu, S.X., Lin, D.: Unsupervised feature learning via nonparametric instance discrimination. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3733–3742 (2018)
39. Xiao, W., Song, K., Liu, J., Yan, Y.: Graph embedding and optimal transport for few-shot classification of metal surface defect. IEEE Trans. Instrum. Meas. **71**, 1–10 (2022)
40. Zadeh, A., Chen, M., Poria, S., Cambria, E., Morency, L.P.: Tensor fusion network for multimodal sentiment analysis. arXiv preprint [arXiv:1707.07250](https://arxiv.org/abs/1707.07250) (2017)
41. Zhang, J., et al.: Rethinking mobile block for efficient attention-based models. In: 2023 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1389–1400. IEEE Computer Society (2023)
42. Zhang, R., et al.: Tip adapter: training-free adaption of clip for few-shot classification. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision – ECCV 2022. LNCS, vol. 13695, pp. 493–510. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-19833-5_29



Neural Network Verification Accelerated by a Novel Abstract Framework

Jiaqi Liu and Meng Wang^(✉)

School of Cybersecurity and Computer, Hebei University, Baoding, China
wangmenghbu@hbu.edu.cn

Abstract. Due to the gradual increase in the application of neural networks in various aspects, ensuring that they do not make errors in critical areas to maintain stability has become a hotspot. The research challenge lies in their black box nature. To address this issue, researchers have proposed various methods for neural network verification. However, the scalability of these methods is limited, making it difficult to apply them to large-scale networks. To address this limitation, this paper proposes an abstract framework. Abstraction, although a classic tool for verification, is rarely used to verify neural networks. However, abstraction aids in addressing the difficulty of extending existing algorithms to the latest network architectures. This framework is applicable to fully connected feedforward neural networks, built upon clustering neurons that exhibit similar output behaviors under specific inputs. For ReLU neural networks, the error bounds generated by the abstraction are also analyzed. Moreover, the framework is shown to minimize the network size while preserving accuracy to the greatest extent possible. We also show how to apply the validation results on the abstract network to the original network. Finally, this framework is independent of existing verification techniques, allowing integration with various verification methods.

Keywords: Neuron Network · Abstraction · Formal Verification

1 Introduction

Neural networks have gained widespread use across various sectors like autonomous driving [1], medical imaging [2], and speech recognition [3] due to their superior performance. However, their vulnerability to minor, often undetectable input perturbations that can drastically alter decision-making highlights their fragility. This susceptibility underscores the critical need for reliable verification of neural network behaviors and outputs.

The complexity and high dimensionality of neural networks, with data and parameters often in the millions, challenge traditional formal verification methods. Formal verification is divided into complete and incomplete methods. Complete verification, using techniques like MILP [4, 5] and SMT [6], involves precise encoding of neural networks and searching for counterexamples to identify security vulnerabilities. However, this method struggles with large-scale networks due to the vast state space and parameter growth, leading to inefficiencies. Incomplete verification methods, such as abstract

interpretation [7, 8] and other techniques like simulation [9] and duality [10], address nonlinearity in activation functions by approximating neuron behavior within abstract domains. These methods trade some accuracy for computational efficiency, making them more suitable for larger networks due to their scalability. The challenge lies in balancing precision and scalability within these abstract domains to effectively verify neural network behavior.

Abstraction [11, 12] is a classic formal method technique that simplifies large neural networks into smaller, more manageable versions for easier analysis and understanding. This approach focuses on the behavioral and semantic similarities of neurons rather than just structural features like weight connections. By evaluating neurons based on the similarity of their activation values and using methods like cohesive hierarchical clustering, neurons with similar behaviors are grouped, allowing for the merging into an abstracted network. This smaller network requires fewer resources, offers increased robustness, and is more amenable to verification techniques that may not be feasible with larger networks. Essentially, if the abstract network meets the verification standards, it implies that the original network does as well, making this method effective for analyzing complex neural networks.

This paper offers the following contributions:

- A framework for abstract networks is proposed, particularly in data-driven scenarios. Error analysis is provided for feedforward neural networks.
- The framework facilitates a scalable reduction in network size while keeping accuracy loss within acceptable limits. As networks grow, the reduction rate increases, and the framework flexibility is enhanced by its adjustable parameters.
- It demonstrates the practical benefits of this method, such as enabling the use of certain verification tools that might otherwise timeout with large networks. The framework allows for the abstraction and subsequent verification of these smaller networks, and it details how verified properties of abstract networks can be applied back to the original networks using the DeepPoly tool.

2 Preliminaries

Each layer consists of n neurons. Given an input value x , a neuron produces an output value, $z = f(x)$. we use ReLU as activation function.

The layers of a feedforward neural network are connected through weighted connections. In a feedforward neural network, each layer l , except the output layer, is equipped with a weight matrix. Here, $w_{ij}^{(l)}$ represents the connection weight between the j^{th} neuron in layer l and the i^{th} neuron in layer $l + 1$. For the i^{th} neuron in layer $l + 1$, $w_{i,*}^{(\ell)} = [w_{i,1}^{(\ell)}, \dots, w_{i,n\ell}^{(\ell)}]$ represents the weights it receives. For the j^{th} neuron in layer l , $w_{*,j}^{(\ell)} = [w_{1,j}^{(\ell)}, \dots, w_{n\ell+1,j}^{(\ell)}]$ represents the weights it transmits. $w_{i,*}^{(\ell)}$ represents the i^{th} row of the matrix, and $w_{*,j}^{(\ell)}$ represents the j^{th} column of the matrix, corresponding to the weights from layer l neurons to layer $l + 1$ neurons.

The input and output for each neuron in layer l are represented by h_i^{ℓ} and z_i^{ℓ} respectively. Here, $h_{\ell} = [h_1^{\ell}, \dots, h_{n\ell}^{\ell}]^{\top}$ represents the input vector of layer ℓ and

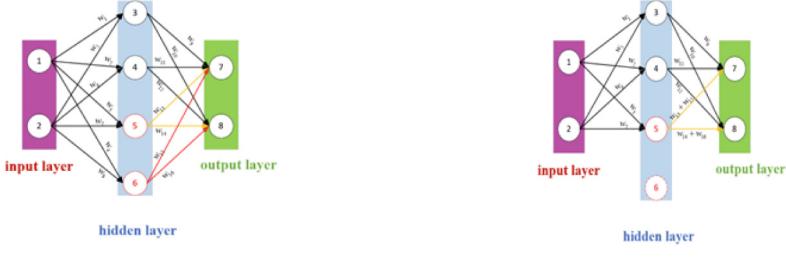
$z_\ell = [z_1^\ell, \dots, z_{n_\ell}^\ell]^\top$ represents the activation vector of layer ℓ . $z_i^{(\ell)} = \phi^{(\ell)}(h_i^{(\ell)})$ represents the value after applying activation function $\phi^{(\ell)}$. Additionally, there exists a vector called bias $b^{(\ell)} \in \mathbb{R}^{n_\ell}$ for neurons in all hidden layers.

This paper centers on the local robustness query, defined as a triplet $Q = (N, x, \delta)$, where N represents the network, x is the input, and $\delta \in \mathbb{R}^{|x|}$ is the perturbation, δ is a modification range. If x' satisfies $N(x') = N(x)$ for all x' in the interval $[x - \delta, x + \delta]$, then network N can be considered robust for query Q .

3 Abstraction

3.1 Merging I/O-Similar Neurons

In this experiment, we define the abstraction operations: merging. The merging operation is based on input-output (I/O) similarity, aiming to preserve the original functionality of the neural network as much as possible. For illustration, consider the following example:



(a) Origin network (b) Combining Neurons 5 and 6

Fig. 1. Comparison before and after merging

As can be seen in Fig. 1a, this is a neural network consisting of one neuron in the input layer, four neurons in the hidden layer and two neurons in the output layer. In order to make the network structure more concise, no bias values were added to the neurons. The activation values for the middle layer are as follows:

$$z_3 = \text{ReLU}(w_1 z_1 + w_5 z_2), z_4 = \text{ReLU}(w_2 z_1 + w_6 z_2)$$

$$z_5 = \text{ReLU}(w_3 z_1 + w_7 z_2), z_6 = \text{ReLU}(w_4 z_1 + w_8 z_2)$$

the activation values of output neurons 7 and 8 are:

$$z_7 = \text{ReLU}(w_9 z_3 + w_{11} z_4 + w_{13} z_5 + w_{15} z_6)$$

$$z_8 = \text{ReLU}(w_{10} z_3 + w_{12} z_4 + w_{14} z_5 + w_{16} z_6)$$

Assuming high similarity between neurons 5 and 6, they are merged to abstract the network, as shown in Fig. 1(b). Neuron 6 is merged into neuron 5 by removing its input edges and directly adding its output edge weights to those of neuron 5. Thus, $\tilde{z}_5 = \text{ReLU}(w_1 z_1 + w_5 z_2) = z_5$ and neuron 6 is removed. The activation values of output layer neurons are as follows:

$$\begin{aligned}\tilde{z}_7 &= \text{ReLU}(w_9 z_3 + w_{11} z_4 + (w_{13} + w_{15}) \tilde{z}_5) \\ &= \text{ReLU}(w_9 z_3 + w_{11} z_4 + w_{13} z_5 + w_{15} z_5) \approx z_7\end{aligned}$$

$$\begin{aligned}\tilde{z}_8 &= \text{ReLU}(w_{10} z_3 + w_{12} z_4 + (w_{14} + w_{16}) \tilde{z}_5) \\ &= \text{ReLU}(w_{10} z_3 + w_{12} z_4 + w_{14} z_5 + w_{16} z_5) \approx z_8\end{aligned}$$

Specifically, neurons p and q are merged as follows. Under certain precision assurance, we assume that neuron q is selected as the neuron to be merged into neuron p based on a heuristic method. Next, the q^{th} row of the original network weight matrix $W^{(\ell-1)}$ of neuron q is removed. Then, the weight matrices $W_p^{(\ell)}$ of neuron p and $W_q^{(\ell)}$ of neuron q from the original network are added, thus adding the output weights of neuron q to neuron p, in other words, achieving the merge of p and q, i.e. $\tilde{W}_{*,p}^{(\ell)} = W_{*,p}^{(\ell)} + W_{*,q}^{(\ell)}$ where $\tilde{W}_{*,p}^{(\ell)}$ represents the output weights of neuron p in the abstracted network.

Proposition 1 (Robustness Proof): For neurons p and q, by removing the input weights of neuron q and then adding the output weights of neuron q to the output weights of neuron p, i.e., $\tilde{W}_{*,p}^{(\ell)} = W_{*,p}^{(\ell)} + W_{*,q}^{(\ell)}$ after completing the above operation, the abstracted network \tilde{D} can be obtained. The abstracted network \tilde{D} considers that for all inputs $x \in X$ of the original network D , the same output will be produced. In other words $\forall x \in X D(x) = \tilde{D}(x)$.

3.2 Abstract Framework

As shown in Fig. 2, our approach consists of the following steps:

1. Firstly, the first two hundred images of the MNIST test set are used as input data.
2. The neural network N will receive these two hundred images and then obtain the output of each hidden layer neuron of neural network N .
3. Calculate the cosine similarity of each neuron based on the obtained output.
4. Convert cosine similarity to cosine similarity distance.
5. Perform agglomerative hierarchical clustering based on the obtained cosine similarity.
6. After performing agglomerative hierarchical clustering, two types of clustering situations will arise: single neuron clustering and multiple neuron clustering. Different processing methods are adopted for these two types of clustering.

Let us expand on how to handle SINGLE neuron clustering and MULTIPLE neuron clustering respectively:

Handling MULTIPLE Neuron Clustering: Algorithm 1 describes how to handle multi-neuron clustering, when merging neurons in multiple neuron clustering, the first step

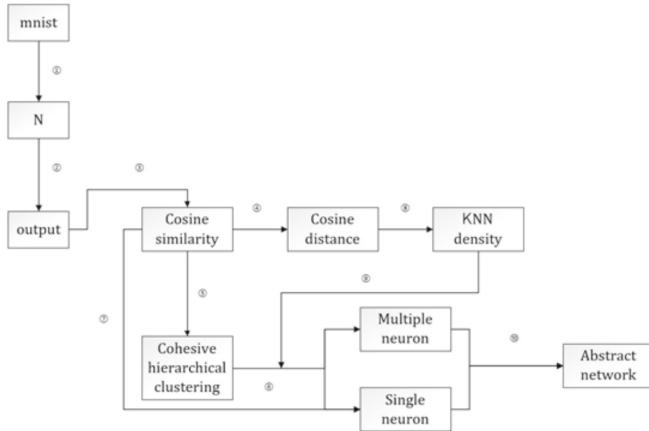


Fig. 2. The suggested abstraction scheme

is to observe whether the KNN value of neurons in the cluster is greater than 1 (We consider neurons with KNN values less than 1 to be in a sparse environment, meaning that these neurons have low similarity with other neurons. Therefore, we do not abstract these neurons further). Then, the neuron with the largest KNN value in the cluster is selected as the target neuron. The remaining neurons will be merged into this neuron. As a result, each cluster will ultimately retain only one neuron, significantly reducing the complexity of the network.

Handling SINGLE Neuron Clusters: Algorithm 2 describes how to handle single neuron clustering, when merging neurons within single neuron clusters, we need to determine the source neuron and the target neuron. The criterion here is based on the KNN density obtained in step 8. Generally, if the KNN value of a neuron is greater than 1, it is considered that there must be a neuron nearby. The distance used here is the cosine similarity distance from step 4. If it is found that the KNN value of a neuron is greater than 1, let us assume it is neuron a . Then, based on the cosine similarity obtained in step 3, we will search for the neuron most similar to this neuron, let us assume it is neuron b . Next, neuron b will be merged into neuron a according to the merge operation described earlier. This is an example of handling single neuron clusters, and all remaining neurons are processed according to this process. The complete algorithm flow for handling single neuron clusters will be explained below.

Algorithm 1 Pseudocode for Abstracting Multiple Neurons

Input: layer_name: n , cluster_labels: l , labels: l_s , cluster_id: i_d , neuron_indices: i_n , Knn_density: K

Output: abstract network with multiple_neuron: N_m

1. **For** n, l_s in l **do**
2. Create multiple neuron clusters $\rightarrow C_m$
3. For i_d, i_n in C_m
4. **If** $\text{KNN}[n][i_n] > 1$
5. Target_index \leftarrow find neuron with max KNN density
6. Perform merge operations(Target_index)
7. Return N_m

Firstly, a dictionary of single neuron clusters is obtained. For each cluster, if its KNN is greater than 1, indicating sufficient neighboring neurons, individual neurons are merged using simulated annealing. After each merge, the model accuracy O_n is calculated. If the loss falls within a user defined reasonable range α , the merge is accepted. If the loss exceeds α , the merge is still retained according to the Metropolis criterion to avoid local minima. As the temperature decreases, the probability of accepting merges with excessive loss decreases, speeding up the abstraction process. Abstraction focuses on neurons closer to the output layer, as they have a greater impact on the final output. The termination condition is when the temperature reaches a threshold or when abstraction for all hidden layers is completed.

Since abstract neural networks can have errors due to abstraction operations, the range of errors needs to be analyzed next. Suppose that for each neuron i , $a_i = [z_i(x_1), \dots, z_i(x_n)]$ Where $x_j \in X$, X is the set of inputs to the network. Let \tilde{D} stand for the abstracted neural network. Define $\varepsilon^{(\ell)}$ as the maximum distance between the source neuron and the target neuron, where $\varepsilon^{(\ell)} = [\varepsilon_1^{(\ell)}, \dots, \varepsilon_{nl}^{(\ell)}]^T$, and $\varepsilon_i^{(\ell)} = 1 - \frac{a_i \cdot r_{C_i}}{a_i \cdot r_{C_i}}$.

Here, the symbol \cdot represents the Euclidean norm. The symbol C_i denotes the cluster containing neuron i , and the symbol r_{C_i} denotes the index of the source neuron within the cluster. Furthermore, The error due to abstraction in a layer is: $err(\ell) = \tilde{z}^{(\ell)} - z^{(\ell)}$

Algorithm 2 Abstracting Single Neuron Clustering

Input: layer_name: n , cluster_labels: l , labels: l_s , cluster_id: i_d , neuron_indices: i_n , Knn_density: KNN , model: m , similarity_matrices: s , merged_to: m_t , original_accuracy: O_a , initial_temp: t_i , cooling_rate: r , temp_threshold: t_r , max_attempts: a_m , current_temp: t_c

Output: abstract network with single_neuron: N_s

1. For n, l_s in l do
2. Create single neuron clusters $\rightarrow C_s$
3. **For** i_d, i_n in C_s **do**
4. If $KNN[n][i_n] > 1$ **then**
5. **While** $t_c > t_r$, **do**
6. Find target neuron: $target_{index}$
7. Save model state: m_s
8. Perform merge: $perform_merge(n, i_n, target_index, m, m_t, O_a, i_d \rightarrow N_s)$
9. Test new model: $test_model(N_s) \rightarrow O_n$
10. Calculate energy changes α : $\alpha = O_a - O_n$
11. **If** $\alpha \geq 0.02$ or metropolis **then**
12. Accept merge
13. **return** N_s
14. **else**
15. Back m_s
16. Update t_c
17. **return** N_s

Theorem 3.1 (*Error due to clustering*). If the absolute error accumulated in layer ℓ is given by $err(\ell)$, then for all inputs $x \in X$, the absolute error can be bounded by the following inequality:

$$|err^{(\ell+1)}| \leq |W^{(\ell)} err^{(\ell)}| + \varepsilon^{(\ell+1)}$$

Theorem 3.2 If the input $x \in X$ of the abstract neural network \tilde{D} is perturbed $\delta \in \mathbb{R}^{|x|}$ then the abstraction error and the perturbation can be expressed using err_{total} , and the absolute error attributed to abstraction and the perturbation err_{total} is bounded for $x \in X$ is bounded:

$$|err_{total}| \leq |\tilde{W}^{(L)} \dots \tilde{W}^{(1)} \delta| + |err^{(L)}|$$

where $\tilde{W}(\ell)$ represents the weight matrix from layer ℓ to layer $\ell + 1$ in \tilde{D} , and $err^{(L)}$ is the cumulative error as described in Theorem 3.1.

Thus, the absolute abstraction error and input perturbation can be obtained by the above two theorems.

4 Enhanced Guarantees for Abstract to Primitive Networks

In this subsection, we will show how DeepPoly can be used to validate the network obtained by using our method, and show that the results obtained from the computation apply equally to the original network.

DeepPoly is a verification algorithm that checks whether a neural network D will treat a given input x and inputs in its θ -neighborhood as being of the same class, thus ensuring the local robustness of the query (D, x, θ) . The abstraction propagates the interval $[x - \theta, x + \theta]$ through the network, producing an over approximation of each neuron activation. The algorithm is reasonable but incomplete. Based on the lower limit l' and the upper limit u' computed by DeepPoly for the abstract network, and the lower limit l , the upper limit u for the original network, $[l', u']$ can be obtain from the following formula, where $[l', u'] \supseteq [l, u]$.

$$u'^{(l)} = W'^{(l-1)} u'^{(l-1)} + W'^{(l-1)} l'^{(l-1)} + b'^{(l)}$$

$$l'^{(l)} = W_+'^{l-1} l'^{(l-1)} + W_-'^{l-1} u'^{(l-1)} + b'^{(l)}$$

where $u'^{(1)} = u^{(1)} = x + \theta$, and $l'^{(1)} = l^{(1)} = x - \theta$, such that $[l', u'] \supseteq [l, u]$.

However, the bounds computed according to the formula lack a certain degree of accuracy, so the whole process is reliable but incomplete.

5 Experiments

The dataset used for this experiment is the very classical MNIST dataset and several neural network architectures were run. A shorthand is used to represent the structure of a neural network, e.g., “ 8×200 ”, denoting a network with a fully-connected feedforward hidden layers, each with n neurons, where the size of the input and output layers does not have symbolic representations for the sake of demonstration.

5.1 Abstraction Results

First, in this experiment, the different structures of the networks are mainly done by increasing the number of network layers and the quantities of neurons, which are obtained by using the MNIST dataset training, and then using our abstraction method to abstract the trained networks, and the accuracy of the abstracted networks on the test set is reduced by at most 2%.

Table 1 below shows some data on the quality of the abstract network. Observe how far the network can abstract with a loss of up to 2% in abstraction accuracy. In the case of determining the quantities of network layers, the quantities of neurons can be increased by expanding the width of the network, e.g., “ 6×200 ” \rightarrow “ 6×300 ”, by increasing the lateral length of the layer. The quantities of neurons is expanded to a certain extent, and as the neurons increases, the quantities of neurons that can be used for merging naturally increases. Then, the corresponding neuron reduction rate will also increase to some extent. From the experimental data, we found that if we want to maintain the

original network behavior, we only need to keep a minimum number of neurons in each layer, and some neurons can be eliminated without affecting the network behavior. Also we found an interesting phenomenon that by increasing the depth of the network, i.e. from 5 to 6 layers, i.e. “ 5×100 ” \rightarrow “ 6×100 ”, the reduction rate hovers around 20–30%.

Figure 3 below shows the effectiveness of our approach, this abstraction is done layer by layer, and in order to abstract as many neurons as possible, we allow the performance of the obtained networks using our method to decrease by about 2% on the test set. It is interesting to note from the Fig. 3 below that more reductions have been obtained in the later layers as compared to the earlier layers. We hypothesize that this is because some important information has been processed and computed in the earlier layers, whereas the later layers convey lower dimensional information. It can be seen that according to the principle of similarity of neuron activation values, there is more similarity between neurons in the later layers. It is interesting to note that from the later layers of the network, for example layers 4, 5, and 6, the neurons are compressed at a similar rate, almost to the same size.

Table 1. Rate of reduction in the number of neurons and decrease in model accuracy for abstract neural networks with different architectures

Network Infrastructure	Precision degradation (%)	Decrease rate (%)
4×300	0.6	21.8
5×300	0.5	22.6
6×300	0.9	21.73
7×300	0.4	28.6
6×200	1.2	20.42
6×300	0.9	21.73
6×400	1.1	28.62
6×500	1.5	31.1
6×600	2.0	36.7
6×700	2.0	41.62
6×800	1.5	43.73

In Fig. 4, we show the total time spent by DeepPoly to verify the original network and the total time spent to abstract the network using our abstraction method and to verify the abstracted network, and it is not hard to see that the time spent on the original network validation is greater than the total time, and as the size of the network increases, the time spent to verify the original network grows even faster than the time spent to verify the abstracted network. Additionally, Table 1 presents the shrinkage rate and accuracy degradation of the corresponding networks.

Table 2. The table summarizes the number of abstracted neurons and verifications on 200 images for neural networks with various architectures

Network Arch	Abstraction Neurons	Images Verified	Images Verified
		(Original)	(Abstracted)
6 × 300	34,49,57,78,82,91	197	196
6 × 400	82,78,86,116,138,187	198	195
6 × 500	32,59,106,224,248,264	196	195
6 × 600	165,194,237,205,253,268	197	195
6 × 700	166,239,299,313,354,373	196	195
6 × 800	174,266,339,406,448,466	195	195

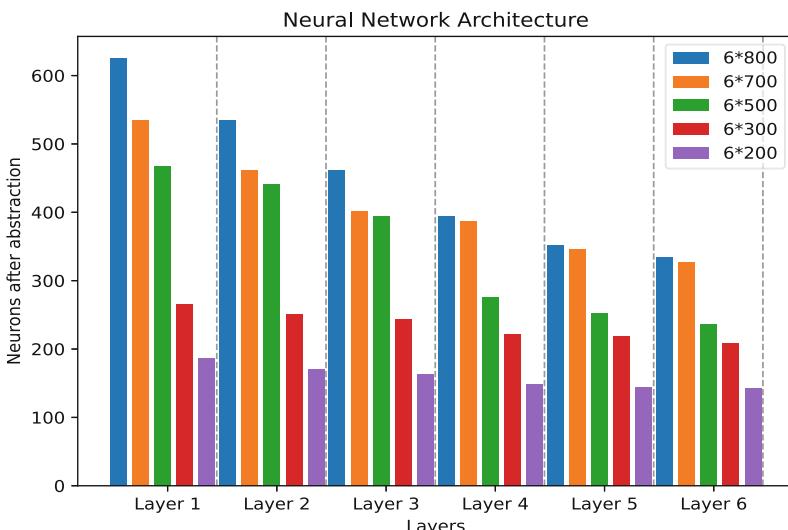


Fig. 3. The figure shows the size of the abstracted network for 5 different architectures and after applying the present abstraction method with accuracy maintained at 96% on the test set

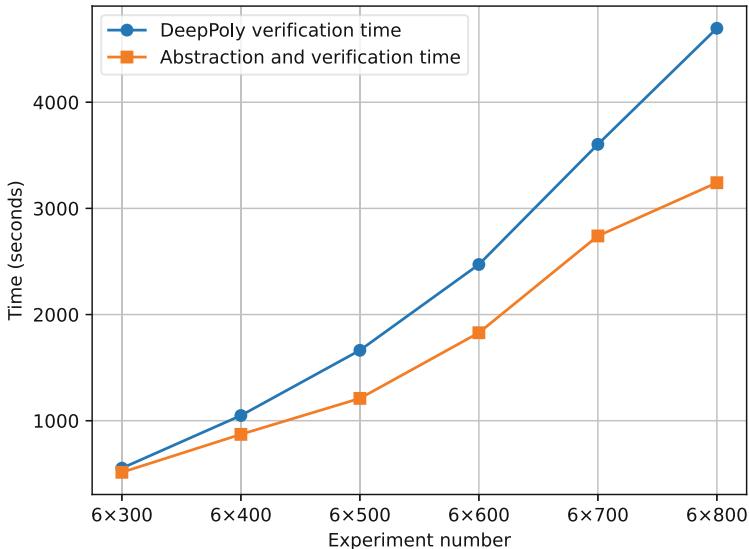


Fig. 4. Comparison of verification time

5.2 Summary of Lifting Verification Proof

From Fig. 4 and Table 2, it is evident that the entire framework, including abstraction and verification, takes significantly less time than verification alone. Specifically, the abstracted network architecture (6×500) verified 195 out of 200 queries in about 30 min, compared to the original network, which verified 196 images in about 41 min. The abstraction process itself took only 2 min and 24 s, reducing the network size by 31.1% and the validation time by 30.8%. This indicates that abstraction significantly enhances the speed of the verification process.

6 Conclusion

We proposed abstraction framework for neural networks uses I/O similarity to map primitive to abstract neurons, enhancing verification processes. However, the reliance on cosine similarity could lead to inaccuracies due to its insensitivity to vector magnitudes and sensitivity to angular changes. A hybrid similarity metric combining cosine and Euclidean distances, optimized dynamically via machine learning, could address these issues. Additionally, employing the CEGAR approach allows for refining specific neurons' constraints, thereby improving proof accuracy. Future expansions could adapt the framework to other network architectures like CNNs.

Acknowledgments. This research is funded by Science Research Project of Hebei Education Department under grant No. BJK2024095.

References

1. Phan-Minh, T., Grigore, E.C., Boulton, F.A., et al.: CoverNet: multimodal behavior prediction using trajectory sets. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14074–14083 (2020)
2. Shen, D., Wu, G., Suk, H.I.: Deep learning in medical image analysis. *Ann. Rev. Biomed. Eng.* **19**, 221–248 (2017)
3. Chang, X., Maekaku, T., Fujita, Y., et al.: End-to-end integration of speech recognition, speech enhancement, and self-supervised learning representation. arXiv preprint [arXiv:2204.00540](https://arxiv.org/abs/2204.00540) (2022)
4. Fischetti, M., Jo, J.: Deep neural networks and mixed integer linear optimization. *Constraints* **23**(3), 296–309 (2018)
5. Bunel, R.R., Turkaslan, I., Torr, P., et al.: A unified view of piecewise linear neural network verification. In: Advances in Neural Information Processing Systems, vol. 31 (2018)
6. Ehlers, R.: Formal verification of piece-wise linear feed-forward neural networks. In: D’Souza, D., Narayan Kumar, K. (eds.) *Automated Technology for Verification and Analysis*. LNCS, vol. 10482, pp. 269–286. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68167-2_19
7. Singh, G., Gehr, T., Mirman, M., et al.: Fast and effective robustness certification. In: Advances in Neural Information Processing Systems, vol. 31 (2018)
8. Singh, G., Gehr, T., Püschel, M., et al.: An abstract domain for certifying neural networks. *Proc. ACM Programm. Lang.* **3**(POPL), 1–30 (2019)
9. Xiang, W., Tran, H.D., Johnson, T.T.: Output reachable set estimation and verification for multilayer neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **29**(11), 5777–5783 (2018)
10. Wong, E., Kolter, Z.: Provable defenses against adversarial examples via the convex outer adversarial polytope. In: International Conference on Machine Learning, pp. 5286–5295. PMLR (2018)
11. Clarke, E., Grumberg, O., Jha, S., Lu, Y., Veith, H.: Counterexample-guided abstraction refinement. In: Emerson, E.A., Sistla, A.P. (eds.) *Computer Aided Verification*. LNCS, vol. 1855, pp. 154–169. Springer, Heidelberg (2000). https://doi.org/10.1007/10722167_15
12. Ashok, P., Hashemi, V., Křetínský, J., Mohr, S.: DeepAbstract: neural network abstraction for accelerating verification. In: Hung, D.V., Sokolsky, O. (eds.) *Automated Technology for Verification and Analysis*. LNCS, vol. 12302, pp. 92–107. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59152-6_5

Author Index

B

Bi, Jiaqian 178

C

Cao, Kangjie 72

Cao, Yifei 368

Chen, Anlong 456

Chen, Jing 251

Chen, Jingxin 60

Chen, Jun 490

Chen, Wentao 3, 13

Chen, Zheng 456

Cheng, Song 431

Cheng, Weijun 72

Cheng, Zhang 120

D

Ding, Yongqi 356

Duan, Suqing 24

Duan, Wentao 84

F

Fan, Haifeng 24

Fang, Menghao 155, 203

Fang, Weidong 3, 13

G

Gan, Haitao 290

Gan, Qingsong 490

Gan, Wenguang 468

Gao, Hongxu 431

Gao, Xingzhen 276

Gao, Yafei 240

Gao, Zhiwei 13

Guo, ChunYi 215

Guo, Zongyu 331

H

Han, Junjia 120

Han, Qian 228

Han, Yan-Ni 190

Han, Yi 290

Hao, Mengjia 331

Hao, Qifei 490

Hao, Tianpeng 3

He, Kunbin 356

He, Qijun 405

Hu, Runjie 48

Hu, Wei 84

Hua, Fengyou 142

Huang, Chengliang 24

Huang, He 72

Huang, Jueqiao 72

Huang, Rui 331

Huang, Zhao 431

J

Jiang, Han 190

Jiang, Xinhang 13

Jiang, Yanxia 393

Jiao, Xinixin 264

Jin, Jiandong 302

Jing, MengMeng 356

K

Kang, Xue 190

Kong, Zixiao 155

L

Li, Changshi 36

Li, Jiaobao 468

Li, Jiahui 405

Li, Jiale 319, 344

Li, Jiangquan 368

Li, Jiayun 72

Li, Kaijiang 215

Li, Wei 60

Li, Wentao 72

Li, Xia 155, 203

Li, Xiali 72

Li, Yao 290

Li, Yifeng 405
 Li, Yu 431
 Li, Zhengwei 393
 Li, Zhiqi 3, 13
 Lian, Zhe 84
 Liang, Jiuzhen 380
 Liang, Kun 132
 Liang, Wei 120
 Liao, Hongmei 393
 Liu, Hao 380
 Liu, Jianzheng 48
 Liu, Jiaqi 504
 Liu, Shixuan 302
 Liu, Shu-Guang 190
 Liu, XiangQian 215
 Liu, Xichen 215
 Liu, Yi 490
 Liu, Yu 368
 Liu, Yuling 215
 Liu, Zhe 490
 Lu, Jingfang 84
 Luo, Shimin 356

M
 Mao, Zijun 24, 48
 Meng, Yangtao 419

N
 Nie, Ru 393

P
 Pan, Gang 228

Q
 Qian, Chengxuan 490

S
 Shen, Qi 490
 Shen, Xiaobing 393
 Shi, Lei 240
 Shi, Peixuan 167
 Shi, Yucheng 240
 Su, Jing 167
 Sun, Haoxin 319, 344
 Sun, Mengdi 319, 344
 Sun, Qing 380
 Sun, Xinlei 203
 Sun, Yundong 108

T
 Tian, Weidong 96
 Tian, Zhaoshuo 108

W
 Wang, Hao 251
 Wang, Haoyu 132
 Wang, Jingyu 276
 Wang, Liejun 264
 Wang, Meng 504
 Wang, Peisen 215
 Wang, Qiuixuan 203
 Wang, Quan 431
 Wang, Tao 142
 Wang, Wenhao 302
 Wang, Xincheng 264
 Wang, Yansong 108
 Wang, Yanxia 120
 Wang, Yuanyuan 203
 Wang, Yunfei 302
 Wei, Hongkui 368
 Wei, Mojierming 48
 Weng, Yanbin 468
 Wu, Cong 290
 Wu, Jianghong 167
 Wu, Jinyao 72

X
 Xia, Wei 468
 Xiang, Yi 190
 Xin, Chengkun 228
 Xinwen, Zhou 380

Xu, Bo 142
 Xu, Kaiping 368
 Xu, Qiaozhi 84
 Xu, Qijia 60
 Xu, Wenzheng 96
 Xu, Xuguang 108
 Xu, Zhilei 178
 Xueying, Han 480

Y
 Yan, Jingfu 405
 Yan, Tianhao 419
 Yan, Yang 480
 Yang, Liangbin 155

- Yang, Qun 142
Yang, Zhijian 36
Ye, Yanning 356
Yin, Yanjun 84
You, Zhuhong 393
Yu, Jie 319, 344
Yu, Jizhe 368
Yu, Qing 178
Yu, Xiao 319, 344
Yu, Yinfeng 264
Yuan, Hu 72
Yuan, Xinpan 468
- Z**
- Zhang, Chao 302
Zhang, Chuanlei 24
Zhang, Chuanting 276
Zhang, Dandan 240
Zhang, Quan 132
Zhang, Shichao 240
Zhang, Shuo 155, 203
Zhang, Wen 190
- Zhang, Wuxiong 3, 13
Zhang, Xiankun 24
Zhang, Xiaoli 132
Zhang, Yifan 331
Zhao, Guohua 240
Zhao, Junxiang 96
Zhao, Shuangxue 444
Zhao, Tianfeng 167
Zhao, Yuhong 276
Zhao, Zhongqiu 96
Zhi, Min 84
Zhong, Tianyu 48
Zhou, Bing 215
Zhou, Changling 302
Zhou, Jia 431
Zhou, Yuan 290
Zhu, Bolun 24
Zhu, Cheng 302
Zhu, Chunsheng 3, 13
Zhu, Dongjie 108
Zhu, Jiachen 456
Zuo, Lin 356