

LNAI 14646

De-Nian Yang · Xing Xie ·
Vincent S. Tseng · Jian Pei ·
Jen-Wei Huang · Jerry Chun-Wei Lin (Eds.)

Advances in Knowledge Discovery and Data Mining

28th Pacific-Asia Conference
on Knowledge Discovery and Data Mining, PAKDD 2024
Taipei, Taiwan, May 7–10, 2024
Proceedings, Part II



 Springer

MOREMEDIA



Lecture Notes in Computer Science

Lecture Notes in Artificial Intelligence

14646

Founding Editor

Jörg Siekmann

Series Editors

Randy Goebel, *University of Alberta, Edmonton, Canada*

Wolfgang Wahlster, *DFKI, Berlin, Germany*

Zhi-Hua Zhou, *Nanjing University, Nanjing, China*

The series Lecture Notes in Artificial Intelligence (LNAI) was established in 1988 as a topical subseries of LNCS devoted to artificial intelligence.

The series publishes state-of-the-art research results at a high level. As with the LNCS mother series, the mission of the series is to serve the international R & D community by providing an invaluable service, mainly focused on the publication of conference and workshop proceedings and postproceedings.

De-Nian Yang · Xing Xie · Vincent S. Tseng ·
Jian Pei · Jen-Wei Huang · Jerry Chun-Wei Lin
Editors

Advances in Knowledge Discovery and Data Mining

28th Pacific-Asia Conference
on Knowledge Discovery and Data Mining, PAKDD 2024
Taipei, Taiwan, May 7–10, 2024
Proceedings, Part II



Springer

Editors

De-Nian Yang 
Academia Sinica
Taipei, Taiwan

Vincent S. Tseng 
National Yang Ming Chiao Tung University
Hsinchu, Taiwan

Jen-Wei Huang 
National Cheng Kung University
Tainan, Taiwan

Xing Xie 
Microsoft Research Asia
Beijing, China

Jian Pei 
Duke University
Durham, NC, USA

Jerry Chun-Wei Lin 
Silesian University of Technology
Gliwice, Poland

ISSN 0302-9743

ISSN 1611-3349 (electronic)

Lecture Notes in Artificial Intelligence

ISBN 978-981-97-2252-5

ISBN 978-981-97-2253-2 (eBook)

<https://doi.org/10.1007/978-981-97-2253-2>

LNCS Sublibrary: SL7 – Artificial Intelligence

© 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

Paper in this product is recyclable.

General Chairs' Preface

On behalf of the Organizing Committee, we were delighted to welcome attendees to the 28th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2024). Since its inception in 1997, PAKDD has long established itself as one of the leading international conferences on data mining and knowledge discovery. PAKDD provides an international forum for researchers and industry practitioners to share their new ideas, original research results, and practical development experiences across all areas of Knowledge Discovery and Data Mining (KDD). This year, after its two previous editions in Taipei (2002) and Tainan (2014), PAKDD was held in Taiwan for the third time in the fascinating city of Taipei, during May 7–10, 2024. Moreover, PAKDD 2024 was held as a fully physical conference since the COVID-19 pandemic was contained.

We extend our sincere gratitude to the researchers who submitted their work to the PAKDD 2024 main conference, high-quality tutorials, and workshops on cutting-edge topics. The conference program was further enriched with seven high-quality tutorials and five workshops on cutting-edge topics. We would like to deliver our sincere thanks for their efforts in research, as well as in preparing high-quality presentations. We also express our appreciation to all the collaborators and sponsors for their trust and cooperation. We were honored to have three distinguished keynote speakers joining the conference: Ed H. Chi (Google DeepMind), Vipin Kumar (University of Minnesota), and Huan Liu (Arizona State University), each with high reputations in their respective areas. We enjoyed their participation and talks, which made the conference one of the best academic platforms for knowledge discovery and data mining. We would like to express our sincere gratitude for the contributions of the Steering Committee members, Organizing Committee members, Program Committee members, and anonymous reviewers, led by Program Committee Chairs De-Nian Yang and Xing Xie. It is through their untiring efforts that the conference had an excellent technical program. We are also thankful to the other Organizing Committee members: Workshop Chairs, Chuan-Kang Ting and Xiaoli Li; Tutorial Chairs, Jiun-Long Huang and Philippe Fournier-Viger; Publicity Chairs, Mi-Yen Yeh and Rage Uday Kiran; Industrial Chairs, Kun-Ta Chuang, Wei-Chao Chen and Richie Tsai; Proceedings Chairs, Jen-Wei Huang and Jerry Chun-Wei Lin; Registration Chairs, Chih-Ya Shen and Hong-Han Shuai; Web and Content Chairs, Cheng-Te Li and Shan-Hung Wu; Local Arrangement Chairs, Yi-Ling Chen, Kuan-Ting Lai, Yi-Ting Chen, and Ya-Wen Teng. We feel indebted to the PAKDD Steering Committee for their constant guidance and sponsorship of manuscripts. We are also grateful to the hosting organizations, National Yang Ming Chiao Tung University and Academia Sinica, and all our sponsors for continuously providing institutional and financial support to PAKDD 2024.

May 2024

Vincent S. Tseng
Jian Pei

PC Chairs' Preface

It is our great pleasure to present the 28th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2024) as Program Committee Chairs. PAKDD is one of the longest-established and leading international conferences in the areas of data mining and knowledge discovery. It provides an international forum for researchers and industry practitioners to share their new ideas, original research results, and practical development experiences in all KDD-related areas, including data mining, data warehousing, machine learning, artificial intelligence, databases, statistics, knowledge engineering, big data technologies, and foundations.

This year, PAKDD received a record number of 720 submissions, among which 86 submissions were rejected at a preliminary stage due to policy violations. There were 595 Program Committee members and 101 Senior Program Committee members involved in the double-blind reviewing process. For submissions entering the double-blind review process, each one received at least three quality reviews from PC members. Furthermore, each valid submission received one meta-review from the assigned SPC member, who also led the discussion with the PC members. The PC Co-chairs then considered the recommendations and meta-reviews from SPC members and looked into each submission as well as its reviews and PC discussions to make the final decision.

As a result of the highly competitive selection process, 175 submissions were accepted and recommended to be published, with 133 oral-presentation papers and 42 poster-presentation papers. We would like to thank all SPC and PC members whose diligence produced a high-quality program for PAKDD 2024. The conference program also featured three keynote speeches from distinguished data mining researchers, eight invited industrial talks, five cutting-edge workshops, and seven comprehensive tutorials.

We wish to sincerely thank all SPC members, PC members, and external reviewers for their invaluable efforts in ensuring a timely, fair, and highly effective paper review and selection procedure. We hope that readers of the proceedings will find the PAKDD 2024 technical program both interesting and rewarding.

May 2024

De-Nian Yang
Xing Xie

Organization

Organizing Committee

Honorary Chairs

Philip S. Yu
Ming-Syan Chen

University of Illinois at Chicago, USA
National Taiwan University, Taiwan

General Chairs

Vincent S. Tseng
Jian Pei

National Yang Ming Chiao Tung University,
Taiwan
Duke University, USA

Program Committee Chairs

De-Nian Yang
Xing Xie

Academia Sinica, Taiwan
Microsoft Research Asia, China

Workshop Chairs

Chuan-Kang Ting
Xiaoli Li

National Tsing Hua University, Taiwan
A*STAR, Singapore

Tutorial Chairs

Jiun-Long Huang
Philippe Fournier-Viger

National Yang Ming Chiao Tung University,
Taiwan
Shenzhen University, China

Publicity Chairs

Mi-Yen Yeh
Rage Uday Kiran

Academia Sinica, Taiwan
University of Aizu, Japan

Industrial Chairs

Kun-Ta Chuang	National Cheng Kung University, Taiwan
Wei-Chao Chen	Inventec Corp./Skywatch Innovation, Taiwan
Richie Tsai	Taiwan AI Academy, Taiwan

Proceedings Chairs

Jen-Wei Huang	National Cheng Kung University, Taiwan
Jerry Chun-Wei Lin	Silesian University of Technology, Poland

Registration Chairs

Chih-Ya Shen	National Tsing Hua University, Taiwan
Hong-Han Shuai	National Yang Ming Chiao Tung University, Taiwan

Web and Content Chairs

Shan-Hung Wu	National Tsing Hua University, Taiwan
Cheng-Te Li	National Cheng Kung University, Taiwan

Local Arrangement Chairs

Yi-Ling Chen	National Taiwan University of Science and Technology, Taiwan
Kuan-Ting Lai	National Taipei University of Technology, Taiwan
Yi-Ting Chen	National Yang Ming Chiao Tung University, Taiwan
Ya-Wen Teng	Academia Sinica, Taiwan

Steering Committee

Chair

Longbing Cao	Macquarie University, Australia
--------------	---------------------------------

Vice Chair

Gill Dobbie	University of Auckland, New Zealand
-------------	-------------------------------------

Treasurer

Longbing Cao

Macquarie University, Australia

Members

Ramesh Agrawal

Jawaharlal Nehru University, India

Gill Dobbie

University of Auckland, New Zealand

João Gama

University of Porto, Portugal

Zhiguo Gong

University of Macau, Macau SAR

Hisashi Kashima

Kyoto University, Japan

Hady W. Lauw

Singapore Management University, Singapore

Jae-Gil Lee

KAIST, Korea

Dinh Phung

Monash University, Australia

Kyuseok Shim

Seoul National University, Korea

Geoff Webb

Monash University, Australia

Raymond Chi-Wing Wong

Hong Kong University of Science and

Technology, Hong Kong SAR

Min-Ling Zhang

Southeast University, China

Life Members

Longbing Cao

Macquarie University, Australia

Ming-Syan Chen

National Taiwan University, Taiwan

David Cheung

University of Hong Kong, China

Joshua Z. Huang

Chinese Academy of Sciences, China

Masaru Kitsuregawa

Tokyo University, Japan

Rao Kotagiri

University of Melbourne, Australia

Ee-Peng Lim

Singapore Management University, Singapore

Huan Liu

Arizona State University, USA

Hiroshi Motoda

AFOSR/AOARD and Osaka University, Japan

Jian Pei

Duke University, USA

P. Krishna Reddy

IIIT Hyderabad, India

Jaideep Srivastava

University of Minnesota, USA

Thanaruk Theeramunkong

Thammasat University, Thailand

Tu-Bao Ho

JAIST, Japan

Vincent S. Tseng

National Yang Ming Chiao Tung University,
Taiwan

Takashi Washio

Osaka University, Japan

Kyu-Young Whang

KAIST, Korea

Graham Williams

Australian National University, Australia

Chengqi Zhang

University of Technology Sydney, Australia

Ning Zhong
Zhi-Hua Zhou

Maebashi Institute of Technology, Japan
Nanjing University, China

Past Members

Arbee L. P. Chen
Hongjun Lu

Asia University, Taiwan
Hong Kong University of Science and
Technology, Hong Kong SAR
Tokyo Institute of Technology, Japan

Takao Terano

Senior Program Committee

Aijun An	York University, Canada
Aris Anagnostopoulos	Sapienza Università di Roma, Italy
Ting Bai	Beijing University of Posts and Telecommunications, China
Elisa Bertino	Purdue University, USA
Arnab Bhattacharya	IIT Kanpur, India
Albert Bifet	Université Paris-Saclay, France
Ludovico Boratto	Università degli Studi di Cagliari, Italy
Ricardo Campello	University of Southern Denmark, Denmark
Longbing Cao	University of Technology Sydney, Australia
Tru Cao	UTHealth, USA
Tanmoy Chakraborty	IIT Delhi, India
Jeffrey Chan	RMIT University, Australia
Pin-Yu Chen	IBM T. J. Watson Research Center, USA
Bin Cui	Peking University, China
Anirban Dasgupta	IIT Gandhinagar, India
Wei Ding	University of Massachusetts Boston, USA
Eibe Frank	University of Waikato, New Zealand
Chen Gong	Nanjing University of Science and Technology, China
Jingrui He	UIUC, USA
Tzung-Pei Hong	National University of Kaohsiung, Taiwan
Qinghua Hu	Tianjin University, China
Hong Huang	Huazhong University of Science and Technology, China
Jen-Wei Huang	National Cheng Kung University, Taiwan
Tsuyoshi Ide	IBM T. J. Watson Research Center, USA
Xiaowei Jia	University of Pittsburgh, USA
Zhe Jiang	University of Florida, USA

Toshihiro Kamishima	National Institute of Advanced Industrial Science and Technology, Japan
Murat Kantarcioglu	University of Texas at Dallas, USA
Hung-Yu Kao	National Cheng Kung University, Taiwan
Kamalakar Karlapalem	IIIT Hyderabad, India
Anuj Karpatne	Virginia Tech, USA
Hisashi Kashima	Kyoto University, Japan
Sang-Wook Kim	Hanyang University, Korea
Yun Sing Koh	University of Auckland, New Zealand
Hady Lauw	Singapore Management University, Singapore
Byung Suk Lee	University of Vermont, USA
Jae-Gil Lee	KAIST, Korea
Wang-Chien Lee	Pennsylvania State University, USA
Chaozhuo Li	Microsoft Research Asia, China
Gang Li	Deakin University, Australia
Jiuyong Li	University of South Australia, Australia
Jundong Li	University of Virginia, USA
Ming Li	Nanjing University, China
Sheng Li	University of Virginia, USA
Ying Li	AwanTunai, Singapore
Yu-Feng Li	Nanjing University, China
Hao Liao	Shenzhen University, China
Ee-peng Lim	Singapore Management University, Singapore
Jerry Chun-Wei Lin	Silesian University of Technology, Poland
Shou-De Lin	National Taiwan University, Taiwan
Hongyan Liu	Tsinghua University, China
Wei Liu	University of Technology Sydney, Australia
Chang-Tien Lu	Virginia Tech, USA
Yuan Luo	Northwestern University, USA
Wagner Meira Jr.	UFMG, Brazil
Alexandros Ntoulas	University of Athens, Greece
Satoshi Oyama	Nagoya City University, Japan
Guansong Pang	Singapore Management University, Singapore
Panagiotis Papapetrou	Stockholm University, Sweden
Wen-Chih Peng	National Yang Ming Chiao Tung University, Taiwan
Dzung Phan	IBM T. J. Watson Research Center, USA
Uday Rage	University of Aizu, Japan
Rajeev Raman	University of Leicester, UK
P. Krishna Reddy	IIIT Hyderabad, India
Thomas Seidl	LMU München, Germany
Neil Shah	Snap Inc., USA

Yingxia Shao	Beijing University of Posts and Telecommunications, China
Victor S. Sheng	Texas Tech University, USA
Kyuseok Shim	Seoul National University, Korea
Arlei Silva	Rice University, USA
Jaideep Srivastava	University of Minnesota, USA
Masashi Sugiyama	RIKEN/University of Tokyo, Japan
Ju Sun	University of Minnesota, USA
Jiliang Tang	Michigan State University, USA
Hanghang Tong	UIUC, USA
Ranga Raju Vatsavai	North Carolina State University, USA
Hao Wang	Nanyang Technological University, Singapore
Hao Wang	Xidian University, China
Jianyong Wang	Tsinghua University, China
Tim Weninger	University of Notre Dame, USA
Raymond Chi-Wing Wong	Hong Kong University of Science and Technology, Hong Kong SAR
Jia Wu	Macquarie University, Australia
Xindong Wu	Hefei University of Technology, China
Xintao Wu	University of Arkansas, USA
Yiqun Xie	University of Maryland, USA
Yue Xu	Queensland University of Technology, Australia
Lina Yao	University of New South Wales, Australia
Han-Jia Ye	Nanjing University, China
Mi-Yen Yeh	Academia Sinica, Taiwan
Hongzhi Yin	University of Queensland, Australia
Min-Ling Zhang	Southeast University, China
Ping Zhang	Ohio State University, USA
Zhao Zhang	Hefei University of Technology, China
Zhongfei Zhang	Binghamton University, USA
Xiangyu Zhao	City University of Hong Kong, Hong Kong SAR
Yanchang Zhao	CSIRO, Australia
Jiayu Zhou	Michigan State University, USA
Xiao Zhou	Renmin University of China, China
Xiaofang Zhou	Hong Kong University of Science and Technology, Hong Kong SAR
Feida Zhu	Singapore Management University, Singapore
Fuzhen Zhuang	Beihang University, China

Program Committee

Zubin Abraham	Robert Bosch, USA
Pedro Henriques Abreu	CISUC, Portugal
Muhammad Abulaish	South Asian University, India
Bijaya Adhikari	University of Iowa, USA
Karan Aggarwal	Amazon, USA
Chowdhury Farhan Ahmed	University of Dhaka, Bangladesh
Ulrich Aïvodji	ÉTS Montréal, Canada
Esra Akbas	Georgia State University, USA
Shafiq Alam	Massey University Auckland, New Zealand
Giuseppe Albi	Università degli Studi di Pavia, Italy
David Anastasiu	Santa Clara University, USA
Xiang Ao	Chinese Academy of Sciences, China
Elena-Simona Apostol	Uppsala University, Sweden
Sunil Aryal	Deakin University, Australia
Jees Augustine	Microsoft, USA
Konstantin Avrachenkov	Inria, France
Goonmeet Bajaj	Ohio State University, USA
Jean Paul Barddal	PUCPR, Brazil
Srikanta Bedathur	IIT Delhi, India
Sadok Ben Yahia	University of Southern Denmark, Denmark
Alessandro Berti	Università di Pisa, Italy
Siddhartha Bhattacharyya	University of Illinois at Chicago, USA
Ranran Bian	University of Sydney, Australia
Song Bian	Chinese University of Hong Kong, Hong Kong SAR
Giovanni Maria Biancofiore	Politecnico di Bari, Italy
Fernando Bobillo	University of Zaragoza, Spain
Adrian M. P. Brasoveanu	Modul Technology GmbH, Austria
Krisztian Buza	Budapest University of Technology and Economics, Hungary
Luca Cagliero	Politecnico di Torino, Italy
Jean-Paul Calbimonte	University of Applied Sciences and Arts Western Switzerland, Switzerland
K. Selçuk Candan	Arizona State University, USA
Fuyuan Cao	Shanxi University, China
Huiping Cao	New Mexico State University, USA
Jian Cao	Shanghai Jiao Tong University, China
Yan Cao	University of Texas at Dallas, USA
Yang Cao	Hokkaido University, Japan
Yuanjiang Cao	Macquarie University, Australia

Sharma Chakravarthy	University of Texas at Arlington, USA
Harry Kai-Ho Chan	University of Sheffield, UK
Zhangming Chan	Alibaba Group, China
Snigdhansu Chatterjee	University of Minnesota, USA
Mandar Chaudhary	eBay, USA
Chen Chen	University of Virginia, USA
Chun-Hao Chen	National Kaohsiung University of Science and Technology, Taiwan
Enhong Chen	University of Science and Technology of China, China
Fanglan Chen	Virginia Tech, USA
Feng Chen	University of Texas at Dallas, USA
Hongyang Chen	Zhejiang Lab, China
Jia Chen	University of California Riverside, USA
Jinjun Chen	Swinburne University of Technology, Australia
Lingwei Chen	Wright State University, USA
Ping Chen	University of Massachusetts Boston, USA
Shang-Tse Chen	National Taiwan University, Taiwan
Shengyu Chen	University of Pittsburgh, USA
Songcan Chen	Nanjing University of Aeronautics and Astronautics, China
Tao Chen	China University of Geosciences, China
Tianwen Chen	Hong Kong University of Science and Technology, Hong Kong SAR
Tong Chen	University of Queensland, Australia
Weitong Chen	University of Adelaide, Australia
Yi-Hui Chen	Chang Gung University, Taiwan
Yile Chen	Nanyang Technological University, Singapore
Yi-Ling Chen	National Taiwan University of Science and Technology, Taiwan
Yi-Shin Chen	National Tsing Hua University, Taiwan
Yi-Ting Chen	National Yang Ming Chiao Tung University, Taiwan
Zheng Chen	Osaka University, Japan
Zhengzhang Chen	NEC Laboratories America, USA
Zhiyuan Chen	UMBC, USA
Zhong Chen	Southern Illinois University, USA
Peng Cheng	East China Normal University, China
Abdelghani Chibani	Université Paris-Est Créteil, France
Jingyuan Chou	University of Virginia, USA
Lingyang Chu	McMaster University, Canada
Kun-Ta Chuang	National Cheng Kung University, Taiwan

Robert Churchill	Georgetown University, USA
Chaoran Cui	Shandong University of Finance and Economics, China
Alfredo Cuzzocrea	Università della Calabria, Italy
Bi-Ru Dai	National Taiwan University of Science and Technology, Taiwan
Honghua Dai	Zhengzhou University, China
Claudia d'Amato	University of Bari, Italy
Chuangyin Dang	City University of Hong Kong, China
Mrinal Das	IIT Palakkad, India
Debanjan Datta	Virginia Tech, USA
Cyril de Runz	Université de Tours, France
Jeremiah Deng	University of Otago, New Zealand
Ke Deng	RMIT University, Australia
Zhaohong Deng	Jiangnan University, China
Anne Denton	North Dakota State University, USA
Shridhar Devamane	KLE Institute of Technology, India
Djellel Difallah	New York University, USA
Ling Ding	Tianjin University, China
Shifei Ding	China University of Mining and Technology, China
Yao-Xiang Ding	Zhejiang University, China
Yifan Ding	University of Notre Dame, USA
Ying Ding	University of Texas at Austin, USA
Lamine Diop	EPITA, France
Nemanja Djuric	Aurora Innovation, USA
Gillian Dobbie	University of Auckland, New Zealand
Josep Domingo-Ferrer	Universitat Rovira i Virgili, Spain
Bo Dong	Amazon, USA
Yushun Dong	University of Virginia, USA
Bo Du	Wuhan University, China
Silin Du	Tsinghua University, China
Jiuding Duan	Allianz Global Investors, Japan
Lei Duan	Sichuan University, China
Walid Durani	LMU München, Germany
Sourav Dutta	Huawei Research Centre, Ireland
Mohamad El-Hajj	MacEwan University, Canada
Ya Ju Fan	Lawrence Livermore National Laboratory, USA
Zipei Fan	Jilin University, China
Majid Farhadloo	University of Minnesota, USA
Fabio Fassetti	Università della Calabria, Italy
Zhiquan Feng	National Cheng Kung University, Taiwan

Len Feremans	Universiteit Antwerpen, Belgium
Edouard Fouché	Karlsruher Institut für Technologie, Germany
Dongqi Fu	UIUC, USA
Yanjie Fu	University of Central Florida, USA
Ken-ichi Fukui	Osaka University, Japan
Matjaž Gams	Jožef Stefan Institute, Slovenia
Amir Gandomi	University of Technology Sydney, Australia
Aryya Gangopadhyay	UMBC, USA
Dashan Gao	Hong Kong University of Science and Technology, China
Wei Gao	Nanjing University, China
Yifeng Gao	University of Texas Rio Grande Valley, USA
Yunjun Gao	Zhejiang University, China
Paolo Garza	Politecnico di Torino, Italy
Chang Ge	University of Minnesota, USA
Xin Geng	Southeast University, China
Flavio Giobergia	Politecnico di Torino, Italy
Rosalba Giugno	Università degli Studi di Verona, Italy
Aris Gkoulalas-Divanis	Merative, USA
Djordje Gligorijevic	Temple University, USA
Daniela Godoy	UNICEN, Argentina
Heitor Gomes	Victoria University of Wellington, New Zealand
Maciej Grzenda	Warsaw University of Technology, Poland
Lei Gu	Nanjing University of Posts and Telecommunications, China
Yong Guan	Iowa State University, USA
Riccardo Guidotti	Università di Pisa, Italy
Ekta Gujral	University of California Riverside, USA
Guimu Guo	Rowan University, USA
Ting Guo	University of Technology Sydney, Australia
Xingzhi Guo	Stony Brook University, USA
Ch. Md. Rakin Haider	Purdue University, USA
Benjamin Halstead	University of Auckland, New Zealand
Jinkun Han	Georgia State University, USA
Lu Han	Nanjing University, China
Yufei Han	Inria, France
Daisuke Hatano	RIKEN, Japan
Kohei Hatano	Kyushu University/RIKEN AIP, Japan
Shogo Hayashi	BizReach, Japan
Erhu He	University of Pittsburgh, USA
Guoliang He	Wuhan University, China
Pengfei He	Michigan State University, USA

Yi He	Old Dominion University, USA
Shen-Shyang Ho	Rowan University, USA
William Hsu	Kansas State University, USA
Haoji Hu	University of Minnesota, USA
Hongsheng Hu	CSIRO, Australia
Liang Hu	Tongji University, China
Shizhe Hu	Zhengzhou University, China
Wei Hu	Nanjing University, China
Mengdi Huai	Iowa State University, USA
Chao Huang	University of Hong Kong, Hong Kong SAR
Congrui Huang	Microsoft, China
Guangyan Huang	Deakin University, Australia
Jimmy Huang	York University, Canada
Jinbin Huang	Hong Kong Baptist University, Hong Kong SAR
Kai Huang	Hong Kong University of Science and Technology, China
Ling Huang	South China Agricultural University, China
Ting-Ji Huang	Nanjing University, China
Xin Huang	Hong Kong Baptist University, Hong Kong SAR
Zhenya Huang	University of Science and Technology of China, China
Chih-Chieh Hung	National Chung Hsing University, Taiwan
Hui-Ju Hung	Pennsylvania State University, USA
Nam Huynh	JAIST, Japan
Akihiro Inokuchi	Kwansei Gakuin University, Japan
Atsushi Inoue	Eastern Washington University, USA
Nevo Itzhak	Ben-Gurion University, Israel
Tomoya Iwakura	Fujitsu Laboratories Ltd., Japan
Divyesh Jadav	IBM T. J. Watson Research Center, USA
Shubham Jain	Visa Research, USA
Bijay Prasad Jayswal	National Cheng Kung University, Taiwan
Kishlay Jha	University of Iowa, USA
Taoran Ji	Texas A&M University - Corpus Christi, USA
Songlei Jian	NUDT, China
Gaoxia Jiang	Shanxi University, China
Hansi Jiang	SAS Institute Inc., USA
Jiaxin Jiang	National University of Singapore, Singapore
Min Jiang	Xiamen University, China
Renhe Jiang	University of Tokyo, Japan
Yuli Jiang	Chinese University of Hong Kong, Hong Kong SAR
Bo Jin	Dalian University of Technology, China

Ming Jin	Monash University, Australia
Ruoming Jin	Kent State University, USA
Wei Jin	University of North Texas, USA
Mingxuan Ju	University of Notre Dame, USA
Wei Ju	Peking University, China
Vana Kalogeraki	Athens University of Economics and Business, Greece
Bo Kang	Ghent University, Belgium
Jian Kang	University of Rochester, USA
Ashwin Viswanathan Kannan	Amazon, USA
Tomi Kauppinen	Aalto University School of Science, Finland
Jungeun Kim	Kongju National University, Korea
Kyoung-Sook Kim	National Institute of Advanced Industrial Science and Technology, Japan
Primož Kocbek	University of Maribor, Slovenia
Aritra Konar	Katholieke Universiteit Leuven, Belgium
Youyong Kong	Southeast University, China
Olivera Kotevska	Oak Ridge National Laboratory, USA
P. Radha Krishna	NIT Warangal, India
Adit Krishnan	UIUC, USA
Gokul Krishnan	IIT Madras, India
Peer Kröger	CAU, Germany
Marzena Kryszkiewicz	Warsaw University of Technology, Poland
Chuan-Wei Kuo	National Yang Ming Chiao Tung University, Taiwan
Kuan-Ting Lai	National Taipei University of Technology, Taiwan
Long Lan	NUDT, China
Duc-Trong Le	Vietnam National University, Vietnam
Tuan Le	New Mexico State University, USA
Chul-Ho Lee	Texas State University, USA
Ickjai Lee	James Cook University, Australia
Ki Yong Lee	Sookmyung Women's University, Korea
Ki-Hoon Lee	Kwangwoon University, Korea
Roy Ka-Wei Lee	Singapore University of Technology and Design, Singapore
Yue-Shi Lee	Ming Chuan University, Taiwan
Dino Lenco	INRAE, France
Carson Leung	University of Manitoba, Canada
Boyu Li	University of Technology Sydney, Australia
Chaojie Li	University of New South Wales, Australia
Cheng-Te Li	National Cheng Kung University, Taiwan
Chongshou Li	Southwest Jiaotong University, China

Fengxin Li	Renmin University of China, China
Guozhong Li	King Abdullah University of Science and Technology, Saudi Arabia
Huaxiong Li	Nanjing University, China
Jianxin Li	Beihang University, China
Lei Li	Hong Kong University of Science and Technology (Guangzhou), China
Peipei Li	Hefei University of Technology, China
Qian Li	Curtin University, Australia
Rong-Hua Li	Beijing Institute of Technology, China
Shao-Yuan Li	Nanjing University of Aeronautics and Astronautics, China
Shuai Li	Cambridge University, UK
Shuang Li	Beijing Institute of Technology, China
Tianrui Li	Southwest Jiaotong University, China
Wengen Li	Tongji University, China
Wentao Li	Hong Kong University of Science and Technology (Guangzhou), China
Xin-Ye Li	Bytedance, China
Xiucheng Li	Harbin Institute of Technology, China
Xuelong Li	Northwestern Polytechnical University, China
Yidong Li	Beijing Jiaotong University, China
Yinxiao Li	Meta Platforms, USA
Yuefeng Li	Queensland University of Technology, Australia
Yun Li	Nanjing University of Posts and Telecommunications, China
Panagiotis Liakos	University of Athens, Greece
Xiang Lian	Kent State University, USA
Shen Liang	Université Paris Cité, France
Qing Liao	Harbin Institute of Technology (Shenzhen), China
Sungsu Lim	Chungnam National University, Korea
Dandan Lin	Shenzhen Institute of Computing Sciences, China
Yijun Lin	University of Minnesota, USA
Ying-Jia Lin	National Cheng Kung University, Taiwan
Baodi Liu	China University of Petroleum (East China), China
Chien-Liang Liu	National Yang Ming Chiao Tung University, Taiwan
Guiquan Liu	University of Science and Technology of China, China
Jin Liu	Shanghai Maritime University, China
Jinfei Liu	Emory University, USA
Kunpeng Liu	Portland State University, USA

Ning Liu	Shandong University, China
Qi Liu	University of Science and Technology of China, China
Qing Liu	Zhejiang University, China
Qun Liu	Louisiana State University, USA
Shenghua Liu	Chinese Academy of Sciences, China
Weifeng Liu	China University of Petroleum (East China), China
Yang Liu	Wilfrid Laurier University, Canada
Yao Liu	University of New South Wales, Australia
Yixin Liu	Monash University, Australia
Zheng Liu	Nanjing University of Posts and Telecommunications, China
Cheng Long	Nanyang Technological University, Singapore
Haibing Lu	Santa Clara University, USA
Wenpeng Lu	Qilu University of Technology, China
Simone Ludwig	North Dakota State University, USA
Dongsheng Luo	Florida International University, USA
Ping Luo	Chinese Academy of Sciences, China
Wei Luo	Deakin University, Australia
Xiao Luo	UCLA, USA
Xin Luo	Shandong University, China
Yong Luo	Wuhan University, China
Fenglong Ma	Pennsylvania State University, USA
Huifang Ma	Northwest Normal University, China
Jing Ma	Hong Kong Baptist University, Hong Kong SAR
Qianli Ma	South China University of Technology, China
Yi-Fan Ma	Nanjing University, China
Rich Maclin	University of Minnesota, USA
Son Mai	Queen's University Belfast, UK
Arun Maiya	Institute for Defense Analyses, USA
Bradley Malin	Vanderbilt University Medical Center, USA
Giuseppe Manco	Consiglio Nazionale delle Ricerche, Italy
Naresh Manwani	IIIT Hyderabad, India
Francesco Marcelloni	Università di Pisa, Italy
Leandro Marinho	UFCG, Brazil
Koji Maruhashi	Fujitsu Laboratories Ltd., Japan
Florent Masseglia	Inria, France
Mohammad Masud	United Arab Emirates University, United Arab Emirates
Sarah Masud	IIIT Delhi, India
Costas Mavromatis	University of Minnesota, USA

Maxwell McNeil	University at Albany SUNY, USA
Massimo Melucci	Università degli Studi di Padova, Italy
Alex Memory	Johns Hopkins University, USA
Ernestina Menasalvas	Universidad Politécnica de Madrid, Spain
Xupeng Miao	Carnegie Mellon University, USA
Matej Miheli	University of Zagreb, Croatia
Fan Min	Southwest Petroleum University, China
Jun-Ki Min	Korea University of Technology and Education, Korea
Tsunenori Mine	Kyushu University, Japan
Nguyen Le Minh	JAIST, Japan
Shuichi Miyazawa	Graduate University for Advanced Studies, Japan
Songsong Mo	Nanyang Technological University, Singapore
Jacob Montiel	Amazon, USA
Yang-Sae Moon	Kangwon National University, Korea
Sebastian Moreno	Universidad Adolfo Ibáñez, Chile
Daisuke Moriwaki	CyberAgent, Inc., Japan
Tsuyoshi Murata	Tokyo Institute of Technology, Japan
Charini Nanayakkara	Australian National University, Australia
Mirco Nanni	Consiglio Nazionale delle Ricerche, Italy
Wilfred Ng	Hong Kong University of Science and Technology, Hong Kong SAR
Cam-Tu Nguyen	Nanjing University, China
Canh Hao Nguyen	Kyoto University, Japan
Hoang Long Nguyen	Meharry Medical College, USA
Shiwen Ni	Chinese Academy of Sciences, China
Jian-Yun Nie	Université de Montréal, Canada
Tadashi Nomoto	National Institute of Japanese Literature, Japan
Tim Oates	UMBC, USA
Eduardo Ogasawara	CEFET-RJ, Brazil
Kouzou Ohara	Aoyama Gakuin University, Japan
Kok-Leong Ong	RMIT University, Australia
Riccardo Ortale	Consiglio Nazionale delle Ricerche, Italy
Arindam Pal	CSIRO, Australia
Eliana Pastor	Politecnico di Torino, Italy
Dhaval Patel	IBM T. J. Watson Research Center, USA
Martin Pavlovski	Yahoo Inc., USA
Le Peng	University of Minnesota, USA
Nhan Pham	IBM T. J. Watson Research Center, USA
Thai-Hoang Pham	Ohio State University, USA
Chengzhi Piao	Hong Kong Baptist University, Hong Kong SAR
Marc Planeyvit	EPITA, France

Mario Prado-Romero	Gran Sasso Science Institute, Italy
Bardh Prenkaj	Sapienza Università di Roma, Italy
Jianzhong Qi	University of Melbourne, Australia
Buyue Qian	Xi'an Jiaotong University, China
Huajie Qian	Columbia University, USA
Hezhe Qiao	Singapore Management University, Singapore
Biao Qin	Renmin University of China, China
Zengchang Qin	Beihang University, China
Tho Quan	Ho Chi Minh City University of Technology, Vietnam
Miloš Radovanović	University of Novi Sad, Serbia
Thilina Ranbaduge	Australian National University, Australia
Chotirat Ratanamahatana	Chulalongkorn University, Thailand
Chandra Reddy	IBM T. J. Watson Research Center, USA
Ryan Rossi	Adobe Research, USA
Morteza Saberi	University of Technology Sydney, Australia
Akira Sakai	Fujitsu Laboratories Ltd., Japan
David Sánchez	Universitat Rovira i Virgili, Spain
Maria Luisa Sapino	Università degli Studi di Torino, Italy
Hernan Sarmiento	UChile & IMFD, Chile
Badrul Sarwar	CloudAEye, USA
Nader Shakibay Senobari	University of California Riverside, USA
Nasrin Shabani	Macquarie University, Australia
Ankit Sharma	University of Minnesota, USA
Chandra N. Shekar	RGUKT RK Valley, India
Chih-Ya Shen	National Tsing Hua University, Taiwan
Wei Shen	Nankai University, China
Yu Shen	Peking University, China
Zhi-Yu Shen	Nanjing University, China
Chuan Shi	Beijing University of Posts and Telecommunications, China
Yue Shi	Meta Platforms, USA
Zhenwei Shi	Beihang University, China
Motoki Shiga	Tohoku University, Japan
Kijung Shin	KAIST, Korea
Kai Shu	Illinois Institute of Technology, USA
Hong-Han Shuai	National Yang Ming Chiao Tung University, Taiwan
Zeren Shui	University of Minnesota, USA
Satyaki Sikdar	Indiana University, USA
Dan Simovici	University of Massachusetts Boston, USA
Apoorva Singh	IIT Patna, India

Bikash Chandra Singh	Islamic University, Bangladesh
Stavros Sintos	University of Illinois at Chicago, USA
Krishnamoorthy Sivakumar	Washington State University, USA
Andrzej Skowron	University of Warsaw, Poland
Andy Song	RMIT University, Australia
Dongjin Song	University of Connecticut, USA
Arnaud Soulet	Université de Tours, France
Ja-Hwung Su	National University of Kaohsiung, Taiwan
Victor Suciú	University of Wisconsin, USA
Liang Sun	Alibaba Group, USA
Xin Sun	Technische Universität München, Germany
Yuqing Sun	Shandong University, China
Hirofumi Suzuki	Fujitsu Laboratories Ltd., Japan
Anika Tabassum	Oak Ridge National Laboratory, USA
Yasuo Tabei	RIKEN, Japan
Chih-Hua Tai	National Taipei University, Taiwan
Hiroshi Takahashi	NTT, Japan
Atsuhiro Takasu	National Institute of Informatics, Japan
Yanchao Tan	Fuzhou University, China
Chang Tang	China University of Geosciences, China
Lu-An Tang	NEC Laboratories America, USA
Qiang Tang	Luxembourg Institute of Science and Technology, Luxembourg
Yiming Tang	Hefei University of Technology, China
Ying-Peng Tang	Nanjing University of Aeronautics and Astronautics, China
Xiaohui (Daniel) Tao	University of Southern Queensland, Australia
Vahid Taslimitehrani	PhysioSigns Inc., USA
Maguelonne Teisseire	INRAE, France
Ya-Wen Teng	Academia Sinica, Taiwan
Masahiro Terabe	Chugai Pharmaceutical Co. Ltd., Japan
Kia Teymourian	University of Texas at Austin, USA
Qing Tian	Nanjing University of Information Science and Technology, China
Yijun Tian	University of Notre Dame, USA
Maksim Tkachenko	Singapore Management University, Singapore
Yongxin Tong	Beihang University, China
Vicenç Torra	University of Umeå, Sweden
Nhu-Thuat Tran	Singapore Management University, Singapore
Yash Travadi	University of Minnesota, USA
Quoc-Tuan Truong	Amazon, USA

Yi-Ju Tseng	National Yang Ming Chiao Tung University, Taiwan
Turki Turki	King Abdulaziz University, Saudi Arabia
Ruo-Chun Tzeng	KTH Royal Institute of Technology, Sweden
Leong Hou U	University of Macau, Macau SAR
Jeffrey Ullman	Stanford University, USA
Rohini Uppuluri	Glassdoor, USA
Satya Valluri	Databricks, USA
Dinusha Vatsalan	Macquarie University, Australia
Bruno Veloso	FEP - University of Porto and INESC TEC, Portugal
Anushka Vidanage	Australian National University, Australia
Herna Viktor	University of Ottawa, Canada
Michalis Vlachos	University of Lausanne, Switzerland
Sheng Wan	Nanjing University of Science and Technology, China
Beilun Wang	Southeast University, China
Changdong Wang	Sun Yat-sen University, China
Chih-Hang Wang	Academia Sinica, Taiwan
Chuan-Ju Wang	Academia Sinica, Taiwan
Guoyin Wang	Chongqing University of Posts and Telecommunications, China
Hongjun Wang	Southwest Jiaotong University, China
Hongtao Wang	North China Electric Power University, China
Jianwu Wang	UMBC, USA
Jie Wang	Southwest Jiaotong University, China
Jin Wang	Megagon Labs, USA
Jingyuan Wang	Beihang University, China
Jun Wang	Shandong University, China
Lizhen Wang	Yunnan University, China
Peng Wang	Southeast University, China
Pengyang Wang	University of Macau, Macau SAR
Sen Wang	University of Queensland, Australia
Senzhang Wang	Central South University, China
Shoujin Wang	Macquarie University, Australia
Sibo Wang	Chinese University of Hong Kong, Hong Kong SAR
Suhang Wang	Pennsylvania State University, USA
Wei Wang	Fudan University, China
Wei Wang	Hong Kong University of Science and Technology (Guangzhou), China
Weicheng Wang	Hong Kong University of Science and Technology, Hong Kong SAR

Wei-Yao Wang	National Yang Ming Chiao Tung University, Taiwan
Wendy Hui Wang	Stevens Institute of Technology, USA
Xiao Wang	Beihang University, China
Xiaoyang Wang	University of New South Wales, Australia
Xin Wang	University of Calgary, Canada
Xinyuan Wang	George Mason University, USA
Yanhao Wang	East China Normal University, China
Yuanlong Wang	Ohio State University, USA
Yuping Wang	Xidian University, China
Yuxiang Wang	Hangzhou Dianzi University, China
Hua Wei	Arizona State University, USA
Zhewei Wei	Renmin University of China, China
Yimin Wen	Guilin University of Electronic Technology, China
Brendon Woodford	University of Otago, New Zealand
Cheng-Wei Wu	National Ilan University, Taiwan
Fan Wu	Central South University, China
Fangzhao Wu	Microsoft Research Asia, China
Jiansheng Wu	Nanjing University of Posts and Telecommunications, China
Jin-Hui Wu	Nanjing University, China
Jun Wu	UIUC, USA
Ou Wu	Tianjin University, China
Shan-Hung Wu	National Tsing Hua University, Taiwan
Shu Wu	Chinese Academy of Sciences, China
Wensheng Wu	University of Southern California, USA
Yun-Ang Wu	National Taiwan University, Taiwan
Wenjie Xi	George Mason University, USA
Lingyun Xiang	Changsha University of Science and Technology, China
Ruliang Xiao	Fujian Normal University, China
Yanghua Xiao	Fudan University, China
Sihong Xie	Lehigh University, USA
Zheng Xie	Nanjing University, China
Bo Xiong	Universität Stuttgart, Germany
Haoyi Xiong	Baidu, Inc., China
Bo Xu	Donghua University, China
Bo Xu	Dalian University of Technology, China
Guandong Xu	University of Technology Sydney, Australia
Hongzuo Xu	NUDT, China
Ji Xu	Guizhou University, China

Tong Xu	University of Science and Technology of China, China
Yuanbo Xu	Jilin University, China
Hui Xue	Southeast University, China
Qiao Xue	Nanjing University of Aeronautics and Astronautics, China
Akihiro Yamaguchi	Toshiba Corporation, Japan
Bo Yang	Jilin University, China
Liangwei Yang	University of Illinois at Chicago, USA
Liu Yang	Tianjin University, China
Shaofu Yang	Southeast University, China
Shiyu Yang	Guangzhou University, China
Wanqi Yang	Nanjing Normal University, China
Xiaoling Yang	Southwest Jiaotong University, China
Xiaowei Yang	South China University of Technology, China
Yan Yang	Southwest Jiaotong University, China
Yiyang Yang	Guangdong University of Technology, China
Yu Yang	City University of Hong Kong, Hong Kong SAR
Yu-Bin Yang	Nanjing University, China
Junjie Yao	East China Normal University, China
Wei Ye	Tongji University, China
Yanfang Ye	University of Notre Dame, USA
Kalidas Yeturu	IIT Tirupati, India
Ilkay Yildiz Potter	BioSensics LLC, USA
Minghao Yin	Northeast Normal University, China
Ziqi Yin	Nanyang Technological University, Singapore
Jia-Ching Ying	National Chung Hsing University, Taiwan
Tetsuya Yoshida	Nara Women's University, Japan
Hang Yu	Shanghai University, China
Jifan Yu	Tsinghua University, China
Yanwei Yu	Ocean University of China, China
Yongsheng Yu	Macquarie University, Australia
Long Yuan	Nanjing University of Science and Technology, China
Lin Yue	University of Newcastle, Australia
Xiaodong Yue	Shanghai University, China
Nayyar Zaidi	Monash University, Australia
Chengxi Zang	Cornell University, USA
Alexey Zaytsev	Skoltech, Russia
Yifeng Zeng	Northumbria University, UK
Petros Zerfos	IBM T. J. Watson Research Center, USA
De-Chuan Zhan	Nanjing University, China

Huixin Zhan	Texas Tech University, USA
Daokun Zhang	Monash University, Australia
Dongxiang Zhang	Zhejiang University, China
Guoxi Zhang	Beijing Institute of General Artificial Intelligence, China
Hao Zhang	Chinese University of Hong Kong, Hong Kong SAR
Huaxiang Zhang	Shandong Normal University, China
Ji Zhang	University of Southern Queensland, Australia
Jianfei Zhang	Université de Sherbrooke, Canada
Lei Zhang	Anhui University, China
Li Zhang	University of Texas Rio Grande Valley, USA
Lin Zhang	IDEA Education, China
Mengjie Zhang	Victoria University of Wellington, New Zealand
Nan Zhang	Wenzhou University, China
Quangui Zhang	Liaoning Technical University, China
Shichao Zhang	Central South University, China
Tianlin Zhang	University of Manchester, UK
Wei Emma Zhang	University of Adelaide, Australia
Wenbin Zhang	Florida International University, USA
Wentao Zhang	Mila, Canada
Xiaobo Zhang	Southwest Jiaotong University, China
Xuyun Zhang	Macquarie University, Australia
Yaqian Zhang	University of Waikato, New Zealand
Yikai Zhang	Guangzhou University, China
Yiqun Zhang	Guangdong University of Technology, China
Yudong Zhang	Nanjing Normal University, China
Zhiwei Zhang	Beijing Institute of Technology, China
Zike Zhang	Hangzhou Normal University, China
Zili Zhang	Southwest University, China
Chen Zhao	Baylor University, USA
Jiaqi Zhao	China University of Mining and Technology, China
Kaiqi Zhao	University of Auckland, New Zealand
Pengfei Zhao	BNU-HKBU United International College, China
Pengpeng Zhao	Soochow University, China
Ying Zhao	Tsinghua University, China
Zhongying Zhao	Shandong University of Science and Technology, China
Guanjie Zheng	Shanghai Jiao Tong University, China
Lecheng Zheng	UIUC, USA
Weiguo Zheng	Fudan University, China

Aoying Zhou	East China Normal University, China
Bing Zhou	Sam Houston State University, USA
Nianjun Zhou	IBM T. J. Watson Research Center, USA
Qinghai Zhou	UIUC, USA
Xiangmin Zhou	RMIT University, Australia
Xiaoping Zhou	Beijing University of Civil Engineering and Architecture, China
Xun Zhou	University of Iowa, USA
Jonathan Zhu	Wheaton College, USA
Ronghang Zhu	University of Georgia, China
Xingquan Zhu	Florida Atlantic University, USA
Ye Zhu	Deakin University, Australia
Yihang Zhu	University of Leicester, UK
Yuanyuan Zhu	Wuhan University, China
Ziwei Zhu	George Mason University, USA

External Reviewers

Zihan Li	University of Massachusetts Boston, USA
Ting Yu	Zhejiang Lab, China

Sponsoring Organizations



Accton



ACSI



Appier



中華電信
Chunghwa Telecom

Chunghwa Telecom Co., Ltd



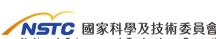
DOIT, Taipei



ISCOM



Metaage



NSTC



Pegatron



Quanta Computer



TWS



Wavenet Co., Ltd

Contents – Part II

Deep Learning

AdaPQ: Adaptive Exploration Product Quantization with Adversary-Aware Block Size Selection Toward Compression Efficiency	3
<i>Yan-Ting Ye, Ting-An Chen, and Ming-Syan Chen</i>	
Ranking Enhanced Supervised Contrastive Learning for Regression	15
<i>Ziheng Zhou, Ying Zhao, Haojia Zuo, and Wenguang Chen</i>	
Treatment Effect Estimation Under Unknown Interference	28
<i>Xiaofeng Lin, Guoxi Zhang, Xiaotian Lu, and Hisashi Kashima</i>	
A New Loss for Image Retrieval: Class Anchor Margin	43
<i>Alexandru Ghiță and Radu Tudor Ionescu</i>	
Personalized EDM Subject Generation via Co-factored User-Subject Embedding	55
<i>Yu-Hsiu Chen, Zhi Rui Tam, and Hong-Han Shuai</i>	
Spatial-Temporal Bipartite Graph Attention Network for Traffic Forecasting	68
<i>Dimuthu Lakmal, Kushani Perera, Renata Borovica-Gajic, and Shanika Karunasekera</i>	
CMed-GPT: Prompt Tuning for Entity-Aware Chinese Medical Dialogue Generation	81
<i>Zhijie Qu, Juan Li, Zerui Ma, and Jianqiang Li</i>	
MvRNA: A New Multi-view Deep Neural Network for Predicting Parkinson’s Disease	93
<i>Lin Chen, Yuxin Zhou, Xiaobo Zhang, Zhehao Zhang, and Hailong Zheng</i>	
Path-Aware Cross-Attention Network for Question Answering	105
<i>Ziye Luo, Ying Xiong, and Buzhou Tang</i>	
StyleAutoEncoder for Manipulating Image Attributes Using Pre-trained StyleGAN	118
<i>Andrzej Bedychaj, Jacek Tabor, and Marek Śmieja</i>	

SEE: Spherical Embedding Expansion for Improving Deep Metric Learning ... <i>Binh Minh Le and Simon S. Woo</i>	131
Multi-modal Recurrent Graph Neural Networks for Spatiotemporal Forecasting <i>Nicholas Majeske and Ariful Azad</i>	144
Layer-Wise Sparse Training of Transformer via Convolutional Flood Filling ... <i>Bokyeong Yoon, Yoonsang Han, and Gordon Euhyun Moon</i>	158
Towards Cost-Efficient Federated Multi-agent RL with Learnable Aggregation <i>Yi Zhang, Sen Wang, Zhi Chen, Xuwei Xu, Stano Funiak, and Jiajun Liu</i>	171
LongStory: Coherent, Complete and Length Controlled Long Story Generation <i>Kyeongman Park, Nakyeong Yang, and Kyomin Jung</i>	184
Relation-Aware Label Smoothing for Self-KD <i>Jeongho Kim and Simon S. Woo</i>	197
Bi-CryptoNets: Leveraging Different-Level Privacy for Encrypted Inference <i>Man-Jie Yuan, Zheng Zou, and Wei Gao</i>	210
Enhancing YOLOv7 for Plant Organs Detection Using Attention-Gate Mechanism <i>Hanane Ariouat, Youcef Sklab, Marc Pignal, Florian Jabbour, Régine Vignes Lebbe, Edi Prifti, Jean-Daniel Zucker, and Eric Chenin</i>	223
On Dark Knowledge for Distilling Generators <i>Chi Hong, Robert Birke, Pin-Yu Chen, and Lydia Y. Chen</i>	235
RPH-PGD: Randomly Projected Hessian for Perturbed Gradient Descent <i>Chi-Chang Li, Jay Huang, Wing-Kai Hon, and Che-Rung Lee</i>	248
Transformer based Multitask Learning for Image Captioning and Object Detection <i>Debolena Basak, P. K. Srijith, and Maunendra Sankar Desarkar</i>	260
Communicative and Cooperative Learning for Multi-agent Indoor Navigation <i>Fengda Zhu, Vincent CS Lee, and Rui Liu</i>	273

Enhancing Continuous Domain Adaptation with Multi-path Transfer Curriculum <i>Hanbing Liu, Jingge Wang, Xuan Zhang, Ye Guo, and Yang Li</i>	286
Graphs and Networks	
Enhancing Network Role Modeling: Introducing Attributed Multiplex Structural Role Embedding for Complex Networks <i>Lili Wang, Chenghan Huang, Ruiye Yao, Chongyang Gao, Weicheng Ma, and Soroush Vosoughi</i>	301
Query-Decision Regression Between Shortest Path and Minimum Steiner Tree <i>Guangmo Tong, Peng Zhao, and Mina Samizadeh</i>	314
Enhancing Policy Gradient for Traveling Salesman Problem with Data Augmented Behavior Cloning <i>Yunchao Zhang, Kewen Liao, Zhibin Liao, and Longkun Guo</i>	327
Leveraging Transfer Learning for Enhancing Graph Optimization Problem Solving <i>Hui-Ju Hung, Wang-Chien Lee, Chih-Ya Shen, Fang He, and Zhen Lei</i>	339
SD-Attack: Targeted Spectral Attacks on Graphs <i>Xianren Zhang, Jing Ma, Yushun Dong, Chen Chen, Min Gao, and Jundong Li</i>	352
Improving Structural and Semantic Global Knowledge in Graph Contrastive Learning with Distillation <i>Mi Wen, Hongwei Wang, Yunsheng Xue, Yi Wu, and Hong Wen</i>	364
DEGNN: Dual Experts Graph Neural Network Handling both Edge and Node Feature Noise <i>Tai Hasegawa, Sukwon Yun, Xin Liu, Yin Jun Phua, and Tsuyoshi Murata</i>	376
Alleviating Over-Smoothing via Aggregation over Compact Manifolds <i>Dongzhuoran Zhou, Hui Yang, Bo Xiong, Yue Ma, and Evgeny Kharlamov</i>	390
Are Graph Embeddings the Panacea?: An Empirical Survey from the Data Fitness Perspective <i>Qiang Sun, Du Q. Huynh, Mark Reynolds, and Wei Liu</i>	405
Revisiting Link Prediction with the Dowker Complex <i>Jae Won Choi, Yuzhou Chen, José Frías, Joel Castillo, and Yulia Gel</i>	418

GraphNILM: A Graph Neural Network for Energy Disaggregation	431
<i>Rui Shang, Siji Chen, Zhiqian Chen, and Chang-Tien Lu</i>	
A Contraction Tree SAT Encoding for Computing Twin-Width	444
<i>Yinon Horev, Shiraz Shay, Sarel Cohen, Tobias Friedrich, Davis Issac, Lior Kamma, Aikaterini Niklanovits, and Kirill Simonov</i>	
Author Index	457

Deep Learning



AdaPQ: Adaptive Exploration Product Quantization with Adversary-Aware Block Size Selection Toward Compression Efficiency

Yan-Ting Ye^(✉), Ting-An Chen, and Ming-Syan Chen

National Taiwan University, Taipei, Taiwan
`{ytye,tachen}@arbor.ee.ntu.edu.tw, mschen@ntu.edu.tw`

Abstract. *Product Quantization (PQ)* has received an increasing research attention due to the effectiveness on bit-width compression for memory efficiency. PQ is developed to divide weight values into blocks and adopt clustering algorithms dividing them into groups assigned with quantized values accordingly. Existing research mainly focused on the clustering strategy design with a minimal error between the original weights and the quantized values for the performance maintenance. However, the block division, i.e., the selection of block size, determines the choices of number of clusters and compression rate which has not been fully studied. Therefore, this paper proposes a novel scheme *AdaPQ* with the process, *Adaptive Exploration Product Quantization*, to first flexibly construct varying block sizes by padding the filter weights, which enlarges the search space of quantization result of PQ and avoids being suffered from a sub-optimal solution. Afterward, we further design a strategy, *Adversary-aware Block Size Selection*, to select an appropriate block size for each layer by evaluating the sensitivity on performance under perturbation for obtaining a minor performance loss under a high compression rate. Experimental results show that AdaPQ achieves a higher accuracy under a similar compression rate compared with the state-of-the-art.

Keywords: Model compression · Product quantization

1 Introduction

In recent research, neural network models, such as ResNet [1] and MobileNet [37], have been demonstrated with remarkable performances across various applications [2,3]. However, due to a large number of model parameters and numerous multiply-accumulate (MAC) operations, it poses significant challenges on deployment of these models onto resource-limited edge devices [4]. Accordingly, several techniques have been developed to address the issues of high memory and computational costs, including quantization [5,11,12], knowledge distillation [23–25], and pruning [6–8]. In particular, quantization has obtained an increasing attention since its efficacy on memory reduction and inference acceleration by decreasing the bit-widths of model weights [5].

Scalar quantization (SQ) is a process which divides the range of numerical values into a set of discrete levels and maps the original values to the nearest level individually [5, 10]. However, the *one-to-one* mapping causes inefficiency due to a large amount of network parameters [30]. Therefore, another branch of research *product quantization (PQ)* has been developed to address this issue. Instead of the projection of single values, product quantization process divides similar values into groups and then represents each group with a value. [16] is the very beginning PQ work proposed to split weights into block units and then operate the k-means algorithm [31] to cluster the blocks. The centroids of clusters are adopted as the representative values. Therefore, it only needs the storage for the footprints of the cluster indices and centers rather than individual quantized values, which derives a higher compression rate and more memory reduction than the prior works in scalar quantization [17, 19].

According to the demonstrated breakthrough on the model compression rate in [16], more research focuses on studying PQ subsequently [17, 19, 20]. Q-CNN [20] focuses on storing the partial results of clustering to save computation costs which are derived from the linear combination of output feature maps. iPQ [17] iteratively quantize and finetune the model weights by Expectation-Maximization (EM) algorithm [18] to find the cluster centroids to minimize the output feature map error during quantization. Recently, QuantNoise [19] addresses the biased gradient issue during quantization by randomly selecting weights to maintain the precision of the gradients during backpropagation.

In general, the existing PQ research is developed based on the process of weight block division and clustering. However, there are two main issues in PQ not fully studied. The first is *the constraint of the block size*. The block size is selected so that the size of filter weights is divisible by the block size, which limits the choices of number of clusters, i.e., the choices of the compression rate and may lead to compression inefficiency. Moreover, another issue studied in this paper is *the fixed block size*. The block size is fixed for each network layer, which results in uniform compression rate for different layers. Nevertheless, the sensitivities to the compression for separate layers are varying. That is, the performance degradation is different in each layer. Accordingly, current quantization algorithms based on the fixed block size may suffer from a large accuracy loss resulting from the sensitive layers, which may derive a sub-optimal result.

In this paper, to mitigate the problems of compression inefficiency and a notable accuracy loss from the choices of the block size, we propose a novel PQ algorithm, ***Adaptive Exploration Product Quantization with Adversary-aware Block Size Selection Toward Compression Efficiency (AdaPQ)***. To address the first issue, the constraint of the block size, we propose ***Adaptive Exploration Quantization***. We adaptively expanding the filter sizes by padding to flexibly explore the clustering (quantization) results under varying block sizes for each layer based on the new filters. Without the constraint of the block size, there are more choices of number of clusters, which enables higher compression efficiency. Moreover, for the issue of fixed block size, we design a strategy of block size selection, ***Adversary-aware Block Size Selection***, to

construct a larger search space with variant choices of block sizes for each network layer to explore the quantization configuration closer to the optimum to achieve a minimal accuracy loss at a high compression rate. We compute the perturbation during PQ process to evaluate the sensitivity of each layer and further determine the block size. The layers with smaller perturbation are more robust and can be represented with fewer values, i.e., with a larger block size.

Our contributions are summarized as follows.

1. We make the first attempt to address the issue of fixed network block sizes to break the limitation of compression rate of existing product quantization (PQ) research.
2. We propose mixed block size PQ and a simple PQ block size selection scheme to achieve a better tradeoff between compression rate and performance, which is orthogonal to existing PQ approaches.
3. Experiments demonstrate that AdaPQ can achieve an outstanding compression rate without an accuracy loss. For example, a 32-bit ResNet18 on tiny-ImageNet is compressed with 50.90x rate and 57.94% accuracy, 1.56% accuracy increment compared with the state-of-the-art.

2 Related Works

Scalar Quantization. There are numerous of studies on Scalar Quantization (SQ) [5, 10, 27]. Jacob *et al.* [5] propose a complete 8-bit quantization pipeline, including quantization-aware training and efficient integer-only inference. DoReFaNet [10] explores lower-bit quantization and uses quantized gradient during training to further reduce the computation cost. HAWQ [11] and its following work [28] achieve a better trade-off between compression rate and accuracy by using a mixed precision quantization scheme. HAQ [12] employs reinforcement learning (RL) to combine information of both hardware and software implementation, enabling a hardware-friendly computation process.

Product Quantization. Product Quantization (PQ) is developed to improve the compression efficiency of SQ. Gong *et al.* [16] is regarded as the first PQ utilized for model compression. They propose to adopt the K-means algorithm [38] to cluster model weights as the quantized values. The clustering enables a more efficient compression process than SQ which quantizes each individual weight value. iPQ [17] quantizes and fine-tunes the model weights iteratively with the Expectation-Maximization (EM) algorithm [18] to find the cluster centroids and indices with a minimal error of output feature maps during quantization. QuantNoise [19] points out that the biased gradients from using STE during quantization-aware training cause performance loss, especially under a high compression rate. Hence, they randomly select weights to keep full-precision to mitigate the bias. However, existing PQ research employs a consistent block size setting for each layer, which leads to a limited compression rate. Moreover, the choice of block size depends on the size of filter weights, which induces a small search space of compression rates.

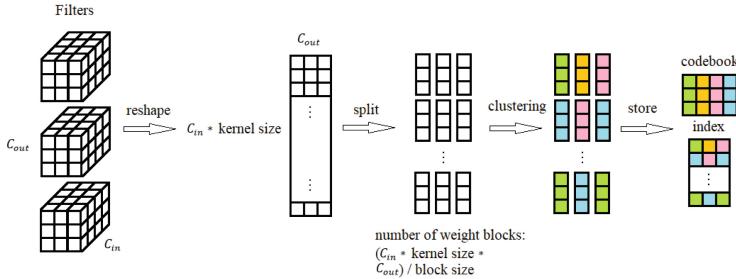


Fig. 1. The basic PQ procedure.

3 Preliminary

We first introduce the PQ procedure and notations. The primary concept behind PQ in model compression is to represent weights with few indices. This is achieved by partitioning weights and clustering them. Since PQ only needs to store the indices and the codebook, the compression rate can achieve remarkably high (over 100x), making PQ receive an increasing research attention.

Clustering. The first step of PQ is splitting the weight matrix into blocks. Consider applying PQ on a single layer in a neural network. Suppose the weight matrix W has a size (C_{in}, C_{out}) and the block size b is given, W is first split into C_{out} vectors along the output channel dimension and then each vector is further split into blocks with size b . Now we have $S = C_{out} * \frac{C_{in}}{b}$ blocks, denote as w_j , $j = 1 \dots S$.

Then a k-means algorithm [31] is applied to these blocks to determine cluster centers and the cluster index of each blocks. Given the bit-width of the cluster indices n , i.e. there are 2^n clusters, the quantized weight now contains cluster centers $q_1 \dots q_{2^n}$ and cluster indices $i_1 \dots i_S$. The objective is to minimize the weight distortion:

$$\min \sum_{j=1}^S \|w_j - q_{i_j}\|_2^2 \quad (1)$$

Inference. After clustering, we can recover the weight matrix \hat{W} from the stored codebook and index. We retrieve weight blocks $\hat{w}_j = q_{i_j}$, $j = i_1 \dots i_S$ then concatenate and reshape them to size (C_{in}, C_{out}) . Then we can inference with the recovered weight matrix \hat{W} .

Block Size Constraint. Each weight block should have the same size in order to perform k-means clustering. Given a weight matrix's size, it must be divisible by the block size. The available block sizes for a given weight matrix is $\{b | C_{in} \% b = 0\}$. This is called the block size constraint.

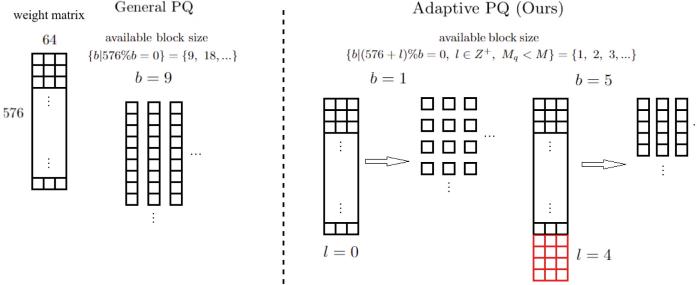


Fig. 2. The difference of the block size search spaces between previous PQ methods and our AdaPQ.

4 Methodology

Section 4.1 will introduce that our proposed scheme AdaPQ with an adaptive padding method that allows PQ with varying block sizes to select from. In Sect. 4.2, we will illustrate the evaluation of the sensitivity of each layer under separate block sizes and the optimization of the size configurations for more accuracy recovery within a desired compression rate.

4.1 Adaptive Exploration Quantization

To relax the block size constraint and enable PQ with more choices of block sizes to explore the optimal solution of compression result, we adaptively add padding values to filter weights for the PQ process illustrated in Fig. 2, where l represents the padding length. The block size is varying with different numbers of padding values. The available block sizes for any given layer are now $\{b | C_{in}\%(b + l) = 0, l \in Z^+\} = Z^+$. Compared to the ones mentioned in Sect. 3, more block sizes are able to be searched now. Accordingly, more compression rates can be explored. In other words, it is more likely to search a compression result closer to the optimum with the minimal performance degradation. Note that the padded values are employed merely in obtaining the cluster centroids, i.e., the quantized weight values, during the learning process of PQ. During inference, the padding values are removed so that no additional memory storage is required.

A potential problem behind the PQ algorithm with padding is that the padding values can affect the searching result of cluster centroids (demonstrated and discussed in ablation study, the last subsection in Sect. 5). To avoid the impact of padding value, we modify the clustering procedure in our Adaptive Exploration Quantization. We ignore padding values when computing the distance and updating centroids. For weight blocks without padding, the distance computation remains unchanged (see Lines 3-4 in Algorithm 1). Otherwise, we ignore the padded values and their corresponding elements in the calculation of cluster centroids (see Lines 5-6 in Algorithm 1). During centroid updates, we average the elements of filters without padding values to determine the new centroids (see Lines 6-7 in Algorithm 2).

Algorithm 1. Adaptive padding distance computing

Input: weight blocks $\tilde{w}_1 \dots \tilde{w}_S$
 1: current cluster centers $q_1 \dots q_{2^n}$
 2: padding length l

Output:

3: **if** \tilde{w}_i does not contain padding **then**
 4: $d_{ij} = \|\tilde{w}_i - q_j\|_2^2$, $j = 1 \dots 2^n$
 5: **else**
 6: $d_{ij} = \|\tilde{w}_i[:l] - q_j[:l]\|_2^2$, $j = 1 \dots 2^n$
 7: **end if**

Algorithm 2. Adaptive padding centroid update

Input: weight blocks $\tilde{w}_1 \dots \tilde{w}_S$
 1: current cluster indices $i_1 \dots i_S$
 2: padding length l

Output:

3: **for** cluster index $c = 1 \dots 2^n$ **do**
 4: Suppose in cluster c , there are a weight blocks without padding (denoted \tilde{w}_{c_i} , $i = 1 \dots a$) and b weight blocks with padding (denoted \tilde{w}_{c_i} , $i = (a+1) \dots (a+b)$).
 5: Compute new cluster center q_c .
 6: $q_c[:l] = \sum_{i=1}^{a+b} \tilde{w}_{c_i}[:l] / (a+b)$
 7: $q_c[l:] = \sum_{i=a+1}^{a+b} \tilde{w}_{c_i}[l:] / (b)$
 8: **end for**

4.2 Adversary-Aware Block Size Selection

In previous PQ works [16, 17, 19], the block size is a predefined hyperparameter, and each layer uses the same block size. Uniform block sizes, however, may not be the best quantization configuration. In neural network models, it is discovered that different layers can have varying influences on the performance [11, 12] in previous scalar quantization research. The sensitivities toward quantization error are also different across layers. Assigning each layer a suitable compression rate, i.e., the block size, is an effective approach achieving a higher compression rate while a minimal performance loss. Therefore, we propose Adversary-aware Block Size Selection in PQ process for block size selection within each network layer. We evaluate the weight distortion of the quantized model as the sensitivity toward quantization layerwisely given a memory budget.

The procedure of our Adversary-aware Block Size Selection is outlined in Algorithm 3. We first apply PQ with a uniform block size to compute the weight distortion. We set initial minimum and maximum block sizes. The layer with the highest distortion, i.e. highest sensitivity, will have the minimum block size; And the layer with the lowest distortion will have the maximum block size. Block sizes of remaining layers are computed by linear interpolation with their distortion and the current maximum and minimum block sizes. With a given bit-width of cluster index n and block size b , the size of a quantized layer is $(\lceil \frac{C_{in}}{b} \rceil * C_{out} * n + 2^n * b * 32)$ bits. We can compute the size of quantized models accordingly. We

Algorithm 3. Adversary-aware Block Size Selection

Input: weight matrix W_l , $l = 1 \dots L$
 1: model size constraint M
Output:
 2: Compute weight perturbation $e_l = ||W_l - \hat{W}_l||_2^2$, $l = 1 \dots L$ under a uniform block size scheme, and let $e_{min} = \min_l e_l$, $e_{max} = \max_l e_l$
 3: Set the minimum block size $b_{min} = 1$ and the maximum block size $b_{max} = 1$, quantized model size $m = inf$
 4: **while** $m > M$ **do**
 5: $b_{max} = b_{max} + 1$
 6: Determine block size $b_l = b_{min} + \frac{e_{max}-e_l}{e_{max}-e_{min}} * (b_{max} - b_{min})$, $l = 1 \dots L$
 7: Compute the quantized model size m_{new} with $b_1 \dots b_L$
 8: **if** $m_{new} > m$ **then**
 9: $b_{min} = b_{min} + 1$
 10: $b_{max} = b_{min} + 1$
 11: $m = inf$
 12: **else**
 13: $m = m_{new}$
 14: **end if**
 15: **end while**

keep increasing the maximum block size until the memory budget requirement is met. If the maximum block size exceeds the valid block size range, which means the model size become dominated by the cluster center and start increasing, we increase the minimum block size instead.

5 Experiments

Benchmark Datasets and Architectures. We conduct several experiments on three benchmark datasets, CIFAR-10 [33], CIFAR-100 [33], and tiny-ImageNet dataset [35], to validate the effectiveness of AdaPQ. CIFAR-10 dataset contains 32×32 colored images in 10 classes. There are 50,000 training images and 10,000 for testing. CIFAR-100 dataset also contains the same amount of 32×32 colored images as CIFAR-10 but in 100 classes. Tiny-ImageNet is a subset of ImageNet [36] with 10,000 64×64 images in 200 classes in both training and testing sets. For model architectures, we evaluate AdaPQ and baselines on the benchmarks ResNet18 [1] and MobileNet-v2 [37].

Experimental Settings. Before quantization, models are trained from scratch on three benchmark datasets separately with SGD optimization in 100 epochs. The learning rate is 0.01, and the batch size is 128. A multi-step learning rate decay is applied during training for better convergence. All experiments are conducted on an NVIDIA RTX3060 GPU.

Table 1. Comparisons with baseline PQ methods.

Quantization scheme	(a) Resnet18										
	CIFAR-10			CIFAR-100			tiny-ImageNet				
	Compression	Top-1(%)	Top-1 drop(%)		Compression	Top-1(%)	Top-1 drop(%)		Compression	Top-1(%)	Top-1 drop(%)
Full precision model	1.00x	94.62	—	1.00x	75.44	—	1.00x	62.92	—		
iPQ <small>(small block)</small> [17]	31.97x	93.49	1.13	31.97x	72.30	3.14	31.81x	58.67	4.25		
QN <small>(small block)</small> [19]	32.10x	93.58	1.04	32.10x	73.91	1.53	31.95x	58.36	4.56		
Ours	33.08x	94.05	0.57	34.61x	73.35	2.09	30.21x	58.70	4.22		
Ours+QN	33.08x	94.16	0.46	34.61x	73.99	1.45	30.21x	60.22	2.70		
iPQ <small>(large block)</small> [17]	51.74x	91.78	2.84	51.74x	71.03	4.41	51.19x	54.28	8.64		
QN <small>(large block)</small> [19]	53.32x	93.44	1.18	53.32x	72.16	3.28	53.13x	56.38	6.54		
Ours	53.31x	93.65	0.97	51.87x	72.18	3.26	50.90x	55.78	7.14		
Ours+QN	53.31x	93.98	0.64	51.87x	72.76	2.68	50.90x	57.94	4.98		

Quantization scheme	(b) Mobilenet-v2								
	CIFAR-10			CIFAR-100					
	Compression	Top-1(%)	Top-1 drop(%)		Compression	Top-1(%)	Top-1 drop(%)		
Full precision model	1.00x	90.43	—	1.00x	67.49	—			
iPQ <small>(small block)</small> [17]	20.08x	*	*	20.08x	*	*			
QN <small>(small block)</small> [19]	20.08x	*	*	20.08x	*	*			
Ours	22.67x	87.81	2.62	20.43x	61.60	5.89			

Comparisons with the State-of-the-Art. Table 1 illustrates the quantization results of our proposed AdaPQ compared with the previous works, Quant-Noise [17] and iPQ [19], where the compression rates are obtained from dividing the original model sizes by the quantized model sizes. For comparisons, we present the accuracy improvements under similar compression rates. Table 1 (a) shows that our proposed AdaPQ based on the basic PQ procedure obtains 0.5%–2% accuracy increment compared with previous works on benchmark datasets under Resnet18 architecture. It is worth mentioning that when the previous methods, iPQ [17] and QN [17], are applied to the lightweight architecture Mobilenet-v2, their performances deteriorate during the quantization procedure mainly due to the small weight block size. Therefore, these prior works fail to inference data as shown with the notation * in Table 1 (b). By contrast, it reveals that our AdaPQ with the block size exploration and optimization processes can search a quantization configuration with higher compression rate and no significant accuracy loss accordingly.

Effectiveness of Search Space Expansion Under Adaptive Exploration Quantization.

Adaptive Exploration Quantization is proposed to explore more compression results by padding values to filter weights to find a solution close to the optimum (introduced in Sect. 4.1). Figure 3 presents the quantization results of PQ with and without considering the padding. The horizontal axis shows the compression rate, and the vertical axis presents accuracy. In particular, the black dots in the figure manifest the quantization results that are explored without padding. The compression results depend on the block sizes which are limited to the factors of the filter size. However, the PQ process under padding on the other hand, can have much more choices of block sizes and search more results with higher compression efficiency. The additional search results are marked in

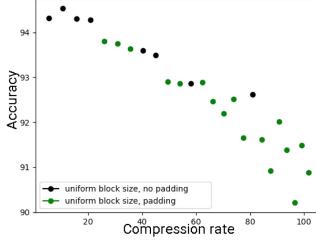


Fig. 3. Quantization results of Resnet18 on CIFAR-10 with generalized PQ. Black dots represent the performances without padding, and green dots manifest the results using adaptive padding. (Color figure online)

Table 2. Accuracy improvement of mixed block size PQ. We train Resnet18 (a) and Mobilenet-v2 (b) on CIFAR-10, CIFAR-100, and tinyImageNet. Models quantized with uniform block size and mixed block size have a close compression rate. So the accuracy difference can show an improvement.

Quantization scheme	(a) Uniform block size and mixed block size PQ for Resnet18.								
	CIFAR-10			CIFAR-100			tiny-ImageNet		
	Compression	Top-1(%)	Top-1 drop(%)	Compression	Top-1(%)	Top-1 drop(%)	Compression	Top-1(%)	Top-1 drop(%)
full precision model	1.00x	94.62	—	1.00x	75.44	—	1.00x	62.92	—
Ours, uniform block size	53.87x	92.87	1.75	41.96x	72.82	2.62	41.81x	56.32	6.60
Ours, mixed block size	53.31x	93.65	0.97	40.56x	73.03	2.41	42.92x	57.27	5.65
Ours, uniform block size	101.76x	90.88	3.74	80.75x	67.88	7.56	72.83x	51.66	11.26
Ours, mixed block size	100.90x	92.37	2.25	81.85x	70.58	4.86	72.50x	53.45	9.47

Quantization scheme	(b) Uniform block size and mixed block size PQ for Mobilenet-v2.								
	CIFAR-10			CIFAR-100			tiny-ImageNet		
	Compression	Top-1(%)	Top-1 drop(%)	Compression	Top-1(%)	Top-1 drop(%)	Compression	Top-1(%)	Top-1 drop(%)
full precision model	1.00x	90.43	—	1.00x	67.49	—	1.00x	49.48	—
Ours, uniform block size	11.77x	88.27	2.16	10.30x	64.37	3.12	10.42x	42.09	7.39
Ours, mixed block size	11.77x	89.05	1.38	10.06x	64.56	2.93	10.58x	43.37	6.11
Ours, uniform block size	22.44x	85.39	5.04	18.99x	60.86	6.63	18.90x	38.96	10.52
Ours, mixed block size	22.57x	88.33	2.10	19.05x	63.26	4.23	19.26x	40.74	8.74

green in Fig. 3. From the results, we can observe that Adaptive Exploration Quantization can effectively explore more compression rates without the block size constraint, especially the higher compression rates from 60x to 100x.

Effectiveness of Performance Improvements by Adversary-Aware Block Size Selection. Table 2 shows the accuracy improvements by using mixed block size configurations which are found under Adversary-aware Block Size Selection (introduced in Sect. 4.2). At the similar compression rates, models quantized with mixed block sizes increase accuracy by 1–3% compared with the performances under the setting of uniform block size for each network layers. In other words, the enhancement demonstrates the effectiveness of the mixed block size by leveraging the sensitivity of each layer.

Table 3. Quantiation result of Resnet18 on CIFAR-10 with different padding methods.

Padding method	Compression	Top-1(%)	Top-1 drop(%)
Non-quantized	1.00x	94.62	–
OOD padding (uniform)	101.76x	89.73	4.89
OOD padding (random)	101.76x	85.43	9.19
Zero padding	101.76x	90.11	4.51
Mean padding	101.76x	90.34	4.28
Adaptive padding (Ours)	101.76x	90.88	3.74

Ablation Study on Padding Values. We employ different padding values in AdaPQ to evaluate the effects. The experiment is conducted on CIFAR-10 using Resnet18. The block size is uniformly 25. We use zero padding, mean padding, uniformly out-of-distribution padding, and randomly out-of-distribution padding. From Table 3, we can observe that if the padding value is too far away from the weight distribution, e.g., OOD padding, the performance will deteriorate significantly. Zero padding and mean padding can achieve better performances than OOD padding. However, there still exists performance degradation due to the bias of the clustering result. In contrast, our proposed adaptive padding (detailed in Algorithm 1 in Sect. 4.1), can achieve the highest accuracy at the same compression rate, since we skip the padding values when calculating and updating the clustering distances and centroids. In other words, the converged clustering result can be guaranteed as same as the PQ process without padding to mitigate the problem of bias clustering result derived from padding.

6 Conclusion

In this paper, we propose an algorithm, AdaPQ, to improve the compression efficiency of existing PQ research. To mitigate the problem of the block size constraint, we employ padding technique to adaptively expanding the filter weights to explore a larger search space of block sizes. Furthermore, instead of adoption of fixed block size for the whole network, we dynamically select the appropriate size for each layer by evaluating the sensitiveness, i.e., the distortion of quantized weights under perturbation. Experimental results demonstrate that the removal of the block size constraint and the adversary-aware block size selection can effectively enhance the performance under the similar compression rate.

Acknowledgement. This work was supported in part by the National Science and Technology Council, Taiwan, under grant NSTC-112-2223-E-002-015, and by the Ministry of Education, Taiwan, under grant MOE 112L9009.

References

1. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, pp. 770–778 (2016). <https://doi.org/10.1109/CVPR.2016.90>
2. Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., Terzopoulos, D.: Image segmentation using deep learning: a survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **44**(7), 3523–3542 (2022)
3. Anwar, S., Barnes, N.: Real image denoising with feature attention. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 3155–3164 (2019)
4. Li, E., Zeng, L., Zhou, Z., Chen, X.: Edge AI: on-demand accelerating deep neural network inference via edge computing. *IEEE Trans. Wireless Commun.* **19**(1), 447–457 (2019)
5. Jacob, B., et al.: Quantization and training of neural networks for efficient integer-arithmetic-only inference. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2704–2713 (2018)
6. Anwar, S., Hwang, K., Sung, W.: Structured pruning of deep convolutional neural networks (2015). <https://doi.org/10.48550/arXiv.1512.08571>
7. Luo, J., Wu, J., Lin, W.: ThiNet: a filter level pruning method for deep neural network compression. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2017)
8. Blalock, D., Gonzalez, O., Jose, J., Frankle, J., Guttag, J.: What is the state of neural network pruning? *Proc. Mach. Learn. Syst.* **2**, 129–146 (2020)
9. Uhlich, S., et al.: Mixed Precision DNNs: all you need is a good parametrization (2020). <https://doi.org/10.48550/arXiv.1905.11452>
10. Zhou, S., Wu, Y., Ni, Z., Zhou, X., Wen, H., Zou, Y.: DoReFa-Net: training low bitwidth convolutional neural networks with low bitwidth gradients (2018). <https://doi.org/10.48550/arXiv.1606.06160>
11. Dong, Z., Yao, Z., Gholami, A., Mahoney, M., Keutzer, K.: HAWQ: Hessian Aware Quantization of neural networks with mixed-precision. In: Proceedings of The IEEE/CVF International Conference on Computer Vision (ICCV) 2019, pp. 293–302 (2019)
12. Wang, K., Liu, Z., Lin, Y., Lin, J., Han, S.: HAQ: Hardware-aware automated quantization with mixed precision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 2019, pp. 8604–8612 (2019)
13. Lin, X., Zhao, C., Pan, W.: Towards accurate binary convolutional neural network. In: Advances in Neural Information Processing Systems 2017, vol. 30, pp. 345–353 (2017)
14. Lin, M., et al.: Rotated binary neural network. In: Advances in Neural Information Processing Systems 2020, vol. 33, pp. 7474–7485 (2020)
15. Jégou, H., Douze, M., Schmid, C.: Product quantization for nearest neighbor search. *IEEE Trans. Pattern Anal. Mach. Intell.* **33**, 117–128 (2011)
16. Gong, Y., Liu, L., Yang, M., Bourdev, L.: Compressing Deep Convolutional Networks using Vector Quantization (2014). <https://doi.org/10.48550/arXiv.1412.6115>
17. Stock, P., Joulin, A., Gribonval, R., Graham, B., Jégou, H.: And the bit goes down: revisiting the quantization of neural networks. In: International Conference on Learning Representations. (2020)

18. Moon, T.K.: The expectation-maximization algorithm. *IEEE Sig. Process. Mag.* **13**, 47–60 (1996)
19. Stock, P., et al.: Training with quantization noise for extreme model compression. In: International Conference on Learning Representations 2021 (2021)
20. Wu, J., Leng, C., Wang, Y., Hu, Q., Cheng, J.: Quantized convolutional neural networks for mobile devices. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2016, pp. 4820–4828 (2016)
21. Yoshua, B., Léonard, N., Courville, A.: Estimating or propagating gradients through stochastic neurons for conditional computation (2013). <https://doi.org/10.48550/arXiv.1308.3432>
22. Wu, Y., Lee, H., Lin, Y., Chien, S.: Accelerator design for vector quantized convolutional neural network. In: 2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS), pp. 46–50 (2019)
23. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. In: NIPS 2014 Deep Learning Workshop (2014)
24. Lopes, R., Fenu, S., Starner, T.: Data-free knowledge distillation for deep neural networks (2017). <https://doi.org/10.48550/arXiv.1710.07535>
25. Gou, J., et al.: Knowledge distillation: a survey. *Int. J. Comput. Vision* **129**(6), pp. 1789–1819 (2021). <https://doi.org/10.1007/s11263-021-01453-z>
26. Gholami, A., Kim, S., Dong, Z., Yao, Z., Mahoney, M., Keutzer, K.: A survey of quantization methods for efficient neural network inference (2021). <https://doi.org/10.48550/arXiv.2103.13630>
27. Sung, W., Shin, S., Hwang, K.: Resiliency of deep neural networks under quantization (2016). <https://doi.org/10.48550/arXiv.1511.06488>
28. Yao, Z., et al.: HAWQ-V3: dyadic neural network quantization. In: Proceedings of the 38th International Conference on Machine Learning, vol. 139, pp. 11875–11886 (2021)
29. Johnson, J., Douze, M., Jégou, H.: Billion-scale similarity search with GPUs (2017) <https://doi.org/10.48550/arXiv.1702.08734>
30. Han, S., Mao, H., Dally, W.: Deep compression: compressing deep neural network with pruning, trained quantization and huffman coding. In: 4th International Conference on Learning Representations (ICLR) (2016)
31. MacQueen, J., et al.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297 (1967)
32. Ren, P., et al.: A comprehensive survey of neural architecture search: challenges and solutions. *ACM Comput. Surv.* **54**(4) (2021). <https://doi.org/10.1145/3447582>
33. Krizhevsky, A., Nair, V., Hinton, G.: CIFAR-10 (Canadian Institute for Advanced Research). <http://www.cs.toronto.edu/~kriz/cifar.html>
34. Krizhevsky, A., Nair, V., Hinton, G.: CIFAR-100 (Canadian Institute for Advanced Research). <http://www.cs.toronto.edu/~kriz/cifar.html>
35. Le, Y., Yang, X.: Tiny ImageNet Visual Recognition Challenge (2015)
36. Deng, J., Dong, W., Socher, R., Li, L., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255 (2009)
37. Howard, A., et al.: MobileNets: efficient convolutional neural networks for mobile vision applications (2017). <https://doi.org/10.48550/arXiv.1704.04861>
38. Pelleg, D., Moore, A.: Accelerating exact k-means algorithms with geometric reasoning. In: Proceedings of the fifth ACM Sigkdd International Conference on Knowledge Discovery And Data Mining, pp. 277–281 (1999)



Ranking Enhanced Supervised Contrastive Learning for Regression

Ziheng Zhou, Ying Zhao^(✉), Haojia Zuo, and Wenguang Chen

Tsinghua University, Beijing, China

{zhouzihe18,zuohj19}@mails.tsinghua.edu.cn, {yingz,cwg}@tsinghua.edu.cn

Abstract. Supervised contrastive learning has shown promising results in image classification tasks where the representations are pulled together if they share same labels or otherwise pushed apart. Such dispersion process in the representation space benefits the downstream classification tasks. However, when applied to regression tasks directly, such dispersion lacks guidance of the relationship among target labels (i.e. the label distances), which leads to the disalignment between representation distances and label distances. Achieving such alignment without compromising the dispersion of learned representations is challenging. In this paper, we propose a Ranking Enhanced Supervised Contrastive Loss (RESupCon) to empower the representation dispersion process with ranking alignment between representation distances and label distances in a controlled fashion. We demonstrate the effectiveness of our method in image regression tasks on four real-world datasets with various interests, including meteorological, medical and human facial data. Experimental results of our method show that representations with better ranking are learned and improvements are made over other baselines in terms of RMSE on all four datasets.

Keywords: contrastive learning · representation learning · regression · representation order

1 Introduction

Recent years have seen rapid progress in the field of representation learning, where self-supervised learning methods such as SimCLR [4] and the MoCo family [5, 10] have gained popularity and shown remarkable abilities to extract features automatically from unlabeled data. Generally speaking, representation learning maps data samples to the feature space and learns their positions based on a certain policy, e.g. samples augmented from the same image should be neighbors in the feature space. Specifically, contrastive learning serves to compare similar and dissimilar samples to which different policies are applied [12]. In

This work was supported by the National Key Research and Development Program of China (2022YFC3004102) and Qinghai Kunlun Talents Program.

the context of supervised learning, modifications can be made upon those self-supervised methods in order to utilize the extra knowledge provided by some kind of supervision. For example, Supervised Contrastive Learning (SupCon) [11] extends SimCLR and takes not only augmentation but also the class labels into account. Contrastive Language-Image Pre-training (CLIP) [17] encourages alignment between two modalities by conducting contrastive training on image-text-paired datasets. Both methods report their promising abilities on the task of image classification. In theory, Graf et al. [8] prove that SupCon forces the representations to collapse according to classes and to be pushed away from those of different classes. Such dispersion in high dimensional space could benefit downstream classifiers as features of different classes become more divisible.

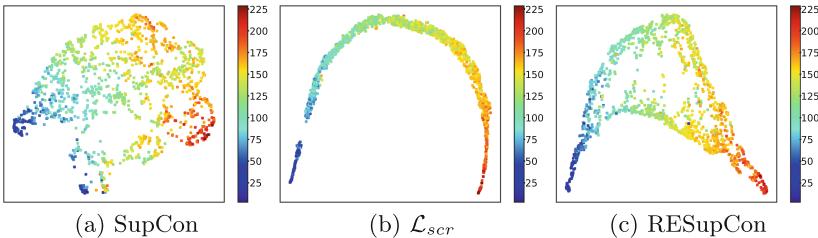


Fig. 1. UMAP [13] visualization of the representations learned by SupCon, \mathcal{L}_{scr} and RESupCon on BoneAge validation dataset. Each dot represents the representation of an input image and its color represents the corresponding target label.

However, the advantages that Supervised Contrastive Learning can offer are mainly confined to classification tasks. Its effectiveness in regression tasks could be doubted, given the continuous nature of real-valued labels. A naive way to apply SupCon to regression tasks is to simply redefine the strategy of selecting positive and negative pairs, e.g. binning on the label values to construct artificial classes. In this setting, the class representations learned by SupCon lack a discernible order, whereas the relative relationships between real-valued labels are intrinsically meaningful in regression tasks. As shown in Fig. 1(a), representations trained with SupCon are dispersed, however ones with similar target labels are often separated, suggesting that the relationships between label values are not well preserved in the feature space. On the other hand, enforcing the ranking of representations in the feature space, either by using a ranking loss alone as shown in Fig. 1 (b) or by contrasting samples against each other based on their rankings in the target space as in [20], compromises the feature space dispersion that is initially achieved by contrastive learning.

In this paper, we propose the Ranking Enhanced Supervised Contrastive Loss (RESupCon), a simple yet effective method to improve the performance of regression by regulating the feature space learned by supervised contrastive learning. As shown in Fig. 1 (c), representations learned by our RESupCon method are well organized in accordance with the labels and still retain the dispersion in contrast with using the ranking loss (\mathcal{L}_{scr}) alone.

The contributions of our work are:

- We define the Spearman correlation ranking loss for measuring and optimizing the alignment between the ranking of representation distances and that of label distances, which helps to generate an orderly representation space.
- We propose RESupCon, a supervised contrastive learning method for regression under the guidance of the Spearman correlation ranking loss, which is capable of learning representations with ranking while retaining dispersion in the representation space.
- We demonstrate the superiority of our method over state-of-the-art baseline methods on four real world datasets through extensive experiments. Further experiments also show that the effectiveness of our method comes from the controlled combination of the dispersion process and the ranking alignment, whereas using either of them alone does not lead to optimal representations for downstream regression tasks.

The rest of paper is organized as follows. We first give a brief overview of related work in Sect. 2. Then, we introduce the preliminary knowledge on both unsupervised and supervised contrastive learning in Sect. 3. Next, we present our proposed method in Sect. 4. Experimental results of our proposed method on four benchmark datasets are shown in Sect. 5. Finally, we make concluding remarks in Sect. 6.

2 Related Work

Supervised Contrastive Learning for Classification. Self-supervised contrastive learning methods, e.g. SimCLR [4] and the MoCo family [5, 10], extract informative representations by contrasting similar samples (“positive pair”), which in common practice are produced by augmenting the same input, with dissimilar ones (“negative pairs”). SupCon [11], further makes use of labels while keeping the contrastive basis. Samples with the same label are defined as positive pairs, and vice versa. SupCon outperforms end-to-end classification with raw Cross-Entropy loss on datasets. [8] show that SupCon encourages features to collapse by classes and then be dispersed in the feature space, increasing the feature diversity of distinct labels. [3] explore the coarse-to-fine transfer learning ability of SupCon on the basis of the aforementioned geometric properties.

Supervised Contrastive Learning for Regression. Despite the success of SupCon on classification, it is non-trivial to transplant supervised contrastive learning onto the field of regression. Given the continuous nature of real-valued labels, related work focus on strengthening the relationship between feature similarities and label distances, which is absent in the concept of SupCon. AdaCon [7] modifies the SupCon loss by adding a term to force the feature similarities of negative samples to reduce according to their label distances. AdaCon reports a minor improvement over end-to-end regression with raw L1 and L2 losses.

[20] argue that the learned representations should be ordered in accordance with label values to benefit regression, and propose RNC as a solution. However, RNC selects any pair of samples as positive pairs, which somewhat compromises the dispersion.

Ranking as a Loss. Learning to Rank (LTR), which targets at building ranking models from training data, has long been a research concern in the field of information retrieval [16]. Ranking (and sorting) operations are typically non-differentiable which cannot be used as a loss to directly optimize a model via back-propagation. Various differentiable approaches have been proposed as proxies to ranking, e.g. [6, 19], often at the cost of computational complexity. [1] propose a fast differentiable approximate algorithm for ranking and sorting in $O(n \log n)$ time, enabling the use of ranking as a practical loss component in deep learning.

3 Preliminaries

We give a brief introduction to the self-supervised method SimCLR [4], as well as the supervised method SupCon [11]. This line of work forms the basis of our proposed method.

Contrastive learning as hinted by its name, contrasts “positive” samples against “negative” ones. In absence of labeled data, SimCLR constructs positive and negative samples by means of data augmentation. Given input $\mathbf{x} \in X$, the same data augmentation operator is applied on \mathbf{x} twice, producing two “views” of the input, $\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j$, which are then passed through a network to obtain projected vectors¹ $\mathbf{z}_i, \mathbf{z}_j$. Following the procedure, a batch of N inputs is processed into a two-viewed batch $\{\tilde{\mathbf{x}}_k\}_{k=1\dots 2N}$ and then $\{\mathbf{z}_k\}_{k=1\dots 2N}$. SimCLR designates $\tilde{\mathbf{x}}_i$ as the “anchor”, relative to which positive and negative samples are defined: $\tilde{\mathbf{x}}_j$ is defined positive while every other sample in the batch is defined negative. The SimCLR loss is defined as the following,

$$\mathcal{L}_{SimCLR} = - \sum_{i \in I} \log \frac{\exp(s(\mathbf{z}_i, \mathbf{z}_{j(i)})/\tau)}{\sum_{k \in A(i)} \exp(s(\mathbf{z}_i, \mathbf{z}_k)/\tau)}, \quad (1)$$

where $I \equiv \{1\dots 2N\}$, $A(i) \equiv I \setminus \{i\}$, $s(\cdot, \cdot)$ is the similarity function (e.g. the cosine similarity), $j(i)$ is the index of the positive counterpart of sample i , and τ is the temperature hyper-parameter.

On the basis of SimCLR, altering the definition of positive and negative pairs yields variants of contrastive losses, among which the SupCon loss is the first to utilize labels, formulated as the following,

¹ Note that they are outputs of the projection head which is omitted after the contrastive training. Network before the projection head is called the encoder, whose outputs we denote by “representations”. We generally refer to both as “feature”.

$$\mathcal{L}_{SupCon} = \sum_{i \in I} \frac{-1}{|P(i)|} \sum_{j \in P(i)} \log \frac{\exp(s(\mathbf{z}_i, \mathbf{z}_j)/\tau)}{\sum_{k \in A(i)} \exp(s(\mathbf{z}_i, \mathbf{z}_k)/\tau)}, \quad (2)$$

where $P(i)$ is the index set of batch samples that share the same label as sample i . It differs from \mathcal{L}_{SimCLR} in that $|P(i)|$ could be greater than 1, and thereby the cross-entropy loss is revised accordingly with a non-one-hot ground truth.

4 Methodology

4.1 Motivation

Graf et al. [8] inspect into the geometry of the SupCon loss and prove that minimizing the SupCon loss is equivalent to forming a regular simplex with the projected vectors $\{\mathbf{z}_i\}$. In other words, SupCon forces the features to collapse according to their corresponding labels, and meanwhile be dispersed on the unit sphere. In general, contrastive learning leads to *the dispersion of learned features*, along with uniformity in SimCLR or class collapse in SupCon. Such dispersion enhances the divisibility of features in the high dimensional space, reducing the need for a complicated downstream classifier.

As aforementioned, applying SupCon on regression tasks directly using pre-defined bins on target labels would produce a representation space that are not well organized by their label distances. Zha et al. [20] prove that the alignment between the ranking of label distances and that of representation distances matters for regression tasks. However in [20], with any pair of samples can be selected as positive pairs, the dispersion is somewhat compromised.

Inspired by the pros and cons of these works, we believe if the dispersion procedure of SupCon could be guided by label distances, the undesirable distributions of representations as shown in Fig. 1 (a) could be mitigated. Towards this end, we propose to enhance SupCon by adding a ranking correlation term to its loss function, which is formally defined as Ranking Enhanced Supervised Contrastive Loss (RESupCon), to ensure that the distances between representations learned by SupCon are aligned with their label distances and still maintain the efficacy in learning distinguishable and scattered representations.

4.2 Ranking Enhanced Supervised Contrastive Learning (RESupCon)

Overall Structure. For a regression task, as shown in Fig. 2, we aim to train a neural network composed of two components: a representation encoder network $f : X \rightarrow \mathcal{R}^{de}$ and a regression prediction network $g : \mathcal{R}^{de} \rightarrow \mathcal{R}^{dt}$ to predict the target $y \in \mathcal{R}^{dt}$ with input $\mathbf{x} \in X$. The encoder network can use any backbone structure to extract representations corresponding to \mathbf{x} . The regression prediction network can simply be a linear regressor. Same as in other works, we deploy the two stage training strategy to firstly train the upstream encoder network and then fix the encoder to train the downstream regressor.

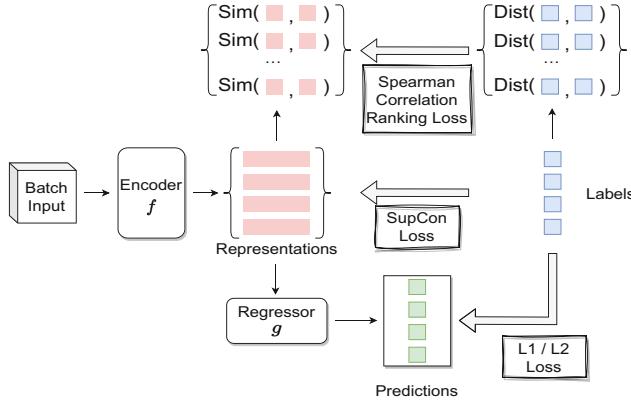


Fig. 2. Overall structure

For the upstream encoder training, RESupCon loss is imposed, which is formally defined as the following,

$$\mathcal{L}_{RESupCon} = \mathcal{L}_{SupConM} + \lambda \mathcal{L}_{scr}, \quad (3)$$

where \mathcal{L}_{scr} is the Spearman correlation ranking loss, λ is the weight for \mathcal{L}_{scr} and $\mathcal{L}_{SupConM}$ is the SupCon loss with a modified $P(i)$. Here, $P(i)$ is the index set of batch samples satisfying $|y_i - y_j| < \theta$, where y_i and y_j are the labels corresponding to sample i and sample j , θ is the predefined bin size. The Spearman correlation ranking loss \mathcal{L}_{scr} is defined in the next subsection.

The representations learned with the Spearman correlation ranking loss shall exhibit a good correlation between their representation distances and label distances while still maintaining the efficacy of supervised contrastive learning that pulls together representations within the same bin and pushes apart other representations. These representations with better ranking correlation and dispersion shall then be used directly to train the downstream regressor. Hence, instead of imposing the loss on the projected vectors generated by a projection head, we impose both $\mathcal{L}_{SupConM}$ and \mathcal{L}_{scr} directly on the features generated by the encoder and these features are used as the representations to train the downstream regressor subsequently.

Spearman Correlation Ranking Loss. How well the ranking of the representation distances are in accordance with the ranking of the label distances can be represented using the Spearman's rank correlation coefficient. The Spearman's rank correlation coefficient is defined as the following,

$$r_s(A, B) = \frac{Cov(R(A), R(B))}{\sigma_{R(A)} \sigma_{R(B)}}, \quad (4)$$

where A , B are two sequence variables and $R(A)$ and $R(B)$ are the rank values of A and B , respectively. Note that Spearman's rank correlation coefficient

is actually the Pearson correlation coefficient between the ranks of two variables. Intuitively, a high Spearman’s rank correlation coefficient (maximum at 1) between two variables indicates that they have similar or identical ranks. Conversely, a low correlation coefficient suggests dissimilar or completely opposite ranks of the two variables, with a correlation coefficient of -1 indicating perfect opposition. Formally, the Spearman correlation ranking loss is defined as the following,

$$\mathcal{L}_{scr} = -\frac{1}{2N} \sum_{i \in I} r_s(d_z(i), d_y(i)), \quad (5)$$

where $i \in I \equiv \{1...2N\}$ is the index set of augmented samples in the two-viewed batch. Taking sample i as an anchor, $d_z(i)$ is the distance function that returns the sequence of distances between anchor representation \mathbf{z}_i and all other representations \mathbf{z}_j ($j \in I$). Distances between representations are computed using negative cosine similarity. $d_y(i)$ is the distance function that returns the sequence of distances between anchor label y_i and all other labels y_j ($j \in I$). Distance between labels is computed using absolute difference. $d_z(i)$ and $d_y(i)$ are defined as the following,

$$\begin{cases} d_z(i) = [-s(\mathbf{z}_i, \mathbf{z}_1), -s(\mathbf{z}_i, \mathbf{z}_2), \dots, -s(\mathbf{z}_i, \mathbf{z}_{2N})] \\ d_y(i) = [|y_i - y_1|, |y_i - y_2|, \dots, |y_i - y_{2N}|] \end{cases}. \quad (6)$$

Since the Spearman’s rank correlation coefficient between two variables is mathematically non-differentiable, we use the implementation of the differentiable ranking algorithm proposed by [1] to approximate it.

Time Complexity. We analyze the time complexity of RESupCon in the serialized scenario. As formulated in Eq. 2, SupCon involves $P(i)$ operations for positive samples and $O(2N - 1)$ operations for negative samples per anchor choice i , where $2N$ is the two-viewed batch size. A summation is done over the $2N$ anchor choices, thus giving us a total of $O(\sum_{i=1}^{2N} (P(i) + 2N - 1)) = O(N^2 + NL)$ operations for the SupCon loss term, where L denotes the expected number of samples in a batch that is annotated with the same label. As stated by [1], the Spearman correlation ranking loss has a time complexity of $O(n \log n)$, where $n = 2N - 1$ in our case of in-batch ranking, per anchor. Therefore the Spearman correlation ranking loss term yields a time complexity of $O(N^2 \log N)$ for a batch. To summarize, the time complexity of the RESupCon loss is $O(N^2 + NL + N^2 \log N) = O(N^2 \log N)$ for a batch of size $2N$.

5 Experiments

5.1 Datasets

We evaluate our method on four regression datasets in various area of interests, e.g. meteorological data, medical data and human facial data.

AgeDB-32. AgeDB [15] dataset contains a total of 16,488 in-the-wild human facial images and their corresponding ages as labels ranging from 1 to 101. We split it into a 12,208-image training set, a 2,140-image validation set and a 2140-image test set. We filtered out images with channels less than 3 and resize all the images to 32×32 before feeding into the network. We name this dataset AgeDB-32.

TCIR. TCIR [2] is a publicly available dataset for tropical cyclone (TC) Intensity Regression (TCIR). We use 36,566 frames of TCs from 2003-2013 as training data, 3,245 frames from 2014 as validation data, and 7,570 frames from 2015-2016 as testing data. Infrared and passive microwave rain-rate channels are used as input and the corresponding intensity from best track intensities (in kts) are used as labels. Each image contains 201×201 pixels and the central area of 65×65 pixels is cropped as the input.

WIKI. IMDB-WIKI [18] is another dataset that contains 523,051 human facial images and corresponding ages as labels. In our experiment, we only use its WIKI dataset, which contains 62,328 images and is split into a 49,692-image training set, a 6,335-image validation set and a 6,301-image test set.

BoneAge. The 2017 RSNA Pediatric Bone Age Challenge Dataset [9] contains X-ray images of hands and their skeletal age annotations. The dataset is split into a 12,611-image training set, a 1,425-image validation set and a 200-image test set. Skeletal ages range from 0 to 216 months.

Data Preprocessing and Augmentation: For TCIR, normalization is applied on each channel of a single sample to have zero mean and unit standard deviation. Random rotation augmentation is used during training. For all other datasets, we normalize the dataset with means and standard deviations calculated on the training set. Random resized cropping, random horizontal flipping, random color jittering and random gray-scaling are applied during training. The input size is resized to 32×32 for TCIR, AgeDB-32 and WIKI, and 64×64 for BoneAge.

5.2 Baselines and Settings

We compare our method with various baselines.

We use ResNet-18 as backbone and use the cosine similarity function for all contrastive losses including ours. Baselines are configured as follows.

E2E-L1 and E2E-L2. Features generated by the encoder are L2-normalized and connected directly to a downstream linear regressor and are trained in an end-to-end fashion using L1 and L2 losses.

SupCon. As recommended in [11], we append a two-layer projection head (Linear, ReLU, Linear) to the encoder and apply the SupCon loss on the L2-normalized output. Every distinct value is considered as a class when calculating the SupCon loss. After training the upstream network, a linear layer is appended to the encoder and is trained to produce a single value regression output.

AdaCon. We implement the model and loss as described in [7]. AdaCon adopts the same projection head mechanism as SupCon. We also set every distinct integer label to be a class, as AdaCon relies on a certain binning policy to convert regression to classification.

RNC. We implement the model and loss as described in [20]. According to the authors, RNC does not include a projection head in the model, same as our method. The L2-normalized representations trained with the RNC loss are used for downstream regressors directly. We train the model following the two-stage scheme of upstream contrastive training and downstream linear probing.

We implement all methods in PyTorch and conduct experiments on RTX 2080Ti GPUs. For E2E-L1 and E2E-L2, models are trained for 400 epochs. For the contrastive methods, upstream training runs for 400 epochs followed by a downstream training (linear probing) stage for 100 epochs. Best model is selected according to the loss on the validation set. All models are optimized with SGD. Hyper-parameters required for the optimizer, namely learning rate, momentum and weight decay, are chosen by conducting grid searches using NNI [14] on all methods to find their best settings respectively (on validation set). The temperature τ is set to 0.1. All experiments are repeated five times and the average results on test sets are reported.

5.3 Overall Performance

As outlined above, we train the models using one of two approaches: an end-to-end manner employing L1 (E2E-L1) and L2 (E2E-L2) losses, or a two-stage scheme that involves upstream contrastive training followed by downstream linear probing (with either L1 or L2 losses). The overall regression performance as measured by RMSE and R^2 is reported in Table 1. Our results show that SupCon underperforms when compared to end-to-end regression with either L1 or L2 losses. This suggests that failing to capture the relationship between representation similarities and label distances is suboptimal for regression tasks. AdaCon performs better than SupCon, but it does not constantly surpass the performance of E2E-L1 or E2E-L2. RNC outperforms the end-to-end methods on three datasets, while slightly worse on the WIKI dataset. It is worth mentioning that training with the Spearman correlation ranking loss (\mathcal{L}_{scr}) alone does not outperform end-to-end methods. In contrast, our proposed method RESupCon exhibits superior performance when compared to all baselines in terms of both RMSE and R^2 , regardless of the loss used in the downstream phase. Specifically, RESupCon (+L2) improves over SupCon (+L2) in terms of RMSE by 3.9%,

Table 1. RMSE and R² results on various test datasets

Method	AgeDB-32		TCIR		WIKI		BoneAge	
	RMSE↓	R ² ↑	RMSE↓	R ² ↑	RMSE↓	R ² ↑	RMSE↓	R ² ↑
E2E-L1	9.93	0.628	11.57	0.852	10.76	0.590	14.70	0.883
E2E-L2	10.04	0.620	11.24	0.861	10.73	0.593	15.07	0.877
SupCon(+L1)	10.04	0.620	11.64	0.850	11.24	0.552	17.60	0.832
SupCon(+L2)	10.06	0.618	11.46	0.855	11.05	0.568	17.32	0.837
AdaCon(+L1)	9.91	0.630	11.41	0.856	11.02	0.570	15.53	0.870
AdaCon(+L2)	9.97	0.626	11.57	0.852	10.91	0.579	16.34	0.856
RNC(+L1)	9.84	0.634	11.21	0.861	10.88	0.580	14.14	0.891
RNC(+L2)	9.78	0.640	11.23	0.861	10.87	0.582	14.21	0.891
\mathcal{L}_{scr} (+L1)	10.19	0.609	12.15	0.837	11.08	0.566	15.25	0.874
\mathcal{L}_{scr} (+L2)	10.25	0.604	11.38	0.857	10.95	0.576	14.98	0.878
RESupCon(+L1)	9.64	0.650	11.07	0.865	10.66	0.597	13.86	0.896
RESupCon(+L2)	9.67	0.647	10.98	0.867	10.60	0.603	13.40	0.903

4.2%, 4.1% and 22.6% on four datasets respectively, and improves over E2E-L2 on RMSE by 3.7%, 2.3%, 1.2% and 11% respectively.

5.4 Comparison on Spearman’s Rank Correlation Coefficients

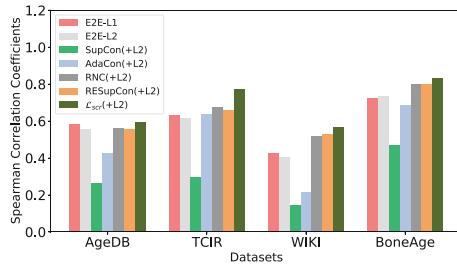


Fig. 3. Spearman’s rank correlation coefficients between representation and label distances on validation sets.

Here we further analyze the correlation between representation and label distances. We feed the validation set $\{\mathbf{x}_i, y_i\}$ into the trained encoder network (or up to the penultimate layer for end-to-end models), and collect the output representations $\{\mathbf{z}_i\}$. The Spearman’s rank correlation coefficient between representation and label distances, $r_s(d_z(i), d_y(i))$, is calculated upon each sample i and averaged over the whole validation set. Figure 3 presents the results on all datasets and methods. As shown in the figure, E2E-L1 and E2E-L2 losses

already result in good ranking at the penultimate layer. It is worth noting that SupCon method harms ranking greatly, confirming the statement that SupCon only disperses representations with little concern about the continuous nature of labels. AdaCon improves the ranking over SupCon but still fails to catch up with E2E-L1 or E2E-L2. RNC and RESupCon both concentrate on the ranking and succeed to retain the coefficient, even higher than that of E2E-L1 and E2E-L2 in most cases. It is worth noting that, training with \mathcal{L}_{scr} alone achieves the highest Spearman’s rank correlation coefficient but does not achieve the best downstream performance. This indicates that a good ranking itself does not lead to optimal representations, without the help of dispersion gained from contrastive learning.

5.5 Parameter Study and Loss Curve

A parameter study is conducted on all four datasets on the weight of \mathcal{L}_{scr} . For illustration, we show the prediction performance in RMSE on AgeDB validation set with respect to different weights in Fig. 4 (a). The RMSE begins to drop initially as λ increases until a certain value, in this case $\lambda = 4$, and becomes worse afterwards. It complies with our assumption that emphasis on ranking alone or using SupCon alone does not achieve the optimal representation space.

The loss curves of both $\mathcal{L}_{SupConM}$ and \mathcal{L}_{scr} of the RESupCon method on AgeDB training set and validation set and the loss curve of the SupCon method on AgeDB validation set are shown in Fig. 4 (b). It shows that $\mathcal{L}_{SupConM}$ trained alone and $\mathcal{L}_{SupConM}$ within RESupCon converge to a similar value on the validation set, while \mathcal{L}_{scr} also decreases along training in a similar pace. The above observation implies that $\mathcal{L}_{SupConM}$ and \mathcal{L}_{scr} can indeed be optimized simultaneously.

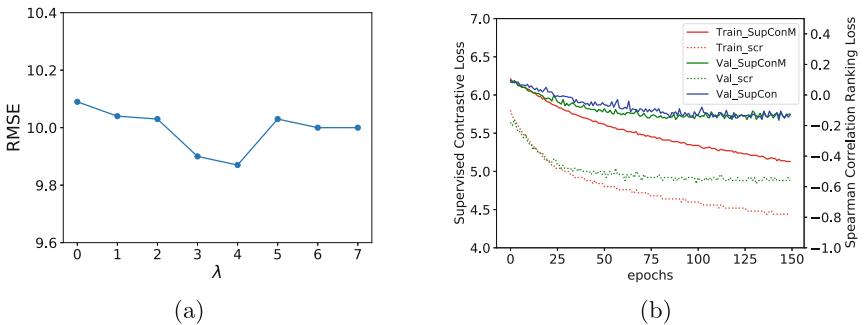


Fig. 4. (a) Parameter Study on λ using AgeDB-32 validation dataset. (b) Training and validation loss curves on AgeDB-32.

6 Conclusion

We propose a new method RESupCon to improve the alignment between representations and labels and to leverage contrastive learning in a controlled fashion, designed to not only learn a representation space with better ranking, but also preserve the dispersion of the learned representations. We verify that the properties of *dispersion* and *ranking* of the representation space do complement each other when conducting contrastive learning on datasets with ordinal labels. RESupCon improves performance on regression tasks by combining and balancing both properties rather than focusing on either one in isolation.

References

1. Blondel, M., Teboul, O., Berthet, Q., Djolonga, J.: Fast differentiable sorting and ranking. In: International Conference on Machine Learning, pp. 950–959. PMLR (2020)
2. Chen, B., Chen, B.F., Lin, H.T.: Rotation-blended CNNs on a new open dataset for tropical cyclone image-to-intensity regression. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 90–99 (2018)
3. Chen, M., Fu, D.Y., Narayan, A., Zhang, M., Song, Z., Fatahalian, K., Ré, C.: Perfectly balanced: improving transfer and robustness of supervised contrastive learning. In: International Conference on Machine Learning, pp. 3090–3122. PMLR (2022)
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., Xie, S., He, K.: An empirical study of training self-supervised vision transformers. In: 2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021, pp. 9620–9629. IEEE (2021). <https://doi.org/10.1109/ICCV48922.2021.00950>
6. Cuturi, M., Teboul, O., Vert, J.P.: Differentiable ranking and sorting using optimal transport. In: Advances in Neural Information Processing Systems. vol. 32 (2019)
7. Dai, W., Li, X., Chiu, W.H.K., Kuo, M.D., Cheng, K.T.: Adaptive contrast for image regression in computer-aided disease assessment. IEEE Trans. Med. Imaging **41**(5), 1255–1268 (2021)
8. Graf, F., Hofer, C., Niethammer, M., Kwitt, R.: Dissecting supervised contrastive learning. In: International Conference on Machine Learning, pp. 3821–3830. PMLR (2021)
9. Halabi, S.S., et al.: The RSNA pediatric bone age machine learning challenge. Radiology **290**(2), 498–503 (2019)
10. 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)
11. Khosla, P., et al.: Supervised contrastive learning. Adv. Neural. Inf. Process. Syst. **33**, 18661–18673 (2020)
12. Le-Khac, P.H., Healy, G., Smeaton, A.F.: Contrastive representation learning: a framework and review. IEEE Access **8**, 193907–193934 (2020)

13. McInnes, L., Healy, J., Melville, J.: UMAP: uniform manifold approximation and projection for dimension reduction. [arXiv:1802.03426](https://arxiv.org/abs/1802.03426) (2020)
14. Microsoft: Neural Network Intelligence (2021). <https://github.com/microsoft/nni>. Accessed 02 Aug 2023
15. Moschoglou, S., Papaioannou, A., Sagonas, C., Deng, J., Kotsia, I., Zafeiriou, S.: AgeDB: the first manually collected, in-the-wild age database. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop, vol. 2, p. 5 (2017)
16. Qin, T., Liu, T.Y., Xu, J., Li, H.: LETOR: a benchmark collection for research on learning to rank for information retrieval. Inf. Retrieval **13**, 346–374 (2010)
17. Radford, A., et al.: Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning, pp. 8748–8763. PMLR (2021)
18. Rothe, R., Timofte, R., Van Gool, L.: DEX: deep expectation of apparent age from a single image. In: Proceedings of the IEEE international conference on computer vision workshops, pp. 10–15 (2015)
19. Taylor, M., Guiver, J., Robertson, S., Minka, T.: SoftRank: optimizing non-smooth rank metrics. In: Proceedings of the 2008 International Conference on Web Search and Data Mining, pp. 77–86 (2008)
20. Zha, K., Cao, P., Son, J., Yang, Y., Katabi, D.: Rank-n-contrast: learning continuous representations for regression. In: Thirty-seventh Conference on Neural Information Processing Systems (2023)



Treatment Effect Estimation Under Unknown Interference

Xiaofeng Lin^(✉), Guoxi Zhang, Xiaotian Lu, and Hisashi Kashima

Graduate School of Informatics, Kyoto University, Kyoto, Japan
`{lxf,guoxi,lu}@ml.ist.i.kyoto-u.ac.jp, kashima@i.kyoto-u.ac.jp`

Abstract. Causal inference is a powerful tool for effective decision-making in various areas, such as medicine and commerce. For example, it allows businesses to determine whether an advertisement has a role in influencing a customer to buy the advertised product. The influence of an advertisement on a particular customer is considered the advertisement's individual treatment effect (ITE). This study estimates ITE from data in which units are potentially connected. In this case, the outcome for a unit can be influenced by treatments to other units, resulting in inaccurate ITE estimation, a phenomenon known as interference. Existing methods for ITE estimation that address interference rely on knowledge of connections between units. However, these methods are not applicable when this connection information is missing due to privacy concerns, a scenario known as unknown interference. To overcome this limitation, this study proposes a method that designs a graph structure learner, which infers the structure of interference by imposing an L_0 -norm regularization on the number of potential connections. The inferred structure is then fed into a graph convolution network to model interference received by units. We carry out extensive experiments on several datasets to verify the effectiveness of the proposed method in addressing unknown interference.

Keywords: Causal inference Individual · Treatment Effect Estimation · Interference

1 Introduction

Estimating treatment effects is crucial in various fields, including medicine [22], education [20], and e-commerce [18]. In commerce, it helps assess whether advertisements influence customer purchases. Treatment effects can be estimated at different levels: average treatment effect (ATE) for the average overall population effect of a treatment and individual treatment effect (ITE) for specific individual effects.

This study focuses on estimating treatment effects using observational data, which includes records of treatment assignments, outcomes, and covariates of units. In particular, we consider scenarios where units are potentially connected. In this case, the outcome of a unit can be influenced by the treatments assigned

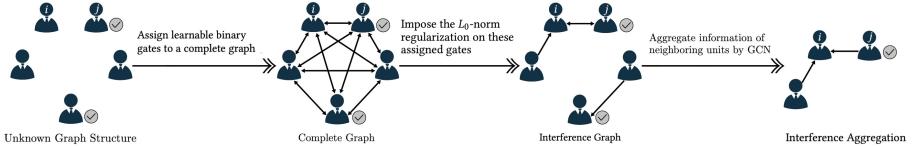


Fig. 1. Overview of the proposed method that models unknown interference. A unit with a checkmark (\checkmark) represents a treated unit, e.g., a unit that receives an advertisement, and a unit without a checkmark represents a control unit.

to other units. Consider an advertising example, users who do not see an advertisement can receive influence from their friends who did, so their responses are indirectly affected by the advertisement, a concept known as *interference* [25]. If such interference is not properly modeled, the estimation of the treatment effects will be inaccurate. However, the potential connections among units are usually hidden, owing to issues such as privacy preservation, a scenario known as *unknown interference* [25]. Existing studies [2, 25] estimated ATE under unknown interference but lacked a solution for ITE, which is crucial in practical applications.

This study proposes an end-to-end method to model unknown interference, aiming to automatically identify significant connections that cause interference (Fig. 1). We utilize a graph structure learner (GSL) to construct an *interference graph* by initially creating a complete graph, as any pair of units could potentially be interconnected. The GSL assigns learnable binary gates to the edges in this complete graph to discern which edges cause interference. During training, an L_0 -norm regularization is applied to these binary gates and minimized using an approximate solution [15]. Additionally, the GSL assigns varying weights to the remaining edges by employing a method similar to graph attention mechanisms [27, 29]. To accurately model interference, our approach utilizes the inferred interference graph, edge weights, and a graph convolution network (GCN) [28] to aggregate interference received by units. The model architecture is shown in Fig. 2, with further details provided in Sect. 4. Upon the training of the model is complete, the proposed model is used for ITE and ATE estimation, as elaborated in Sect. 4.4.

The contributions of this study can be summarized as follows:

- This study overcomes a novel challenge: ITE estimation under unknown interference.
- This study proposes a new method to address unknown interference.
- Extensive experiments verify the efficacy of the proposed methods for ITE and ATE estimation in the presence of unknown interference.

2 Related Work

Many existing methods estimate ITE and ATE without accounting for interference [6, 7, 10, 12, 23]. They estimate treatment effects under the assumption that

there is no interference among units. This is often unrealistic in real-world data where units are interconnected and propagate information.

Previous studies on modeling interference can be categorized into three types based on their assumptions of interference: *group-level interference* [9, 14, 26], *pairwise interference* [1, 4], and *networked interference* [13, 16, 17]. Group-level interference assumes that interference exists within subgroups of units but not across them, which is a partial interference and does not align well with real-world scenarios. Pairwise interference presumes that units can be influenced by their immediate neighbors, limiting their ability to capture the broader propagation of interference. Networked interference is the most general assumption, allowing interference to propagate through a graph of units. Ma et al. [17] applied an aggregation mechanism of GCNs [28] to model the propagation of interference on graphs, enabling their model to capture networked interference. Ma et al. [16] proposed a hypergraph-GCN-based method for modeling high-order interference on hypergraphs. Lin et al. [13] modeled interference across different views of multi-view graphs using a heterogeneous GCN-based method. However, these GCN-based methods need knowledge about graph structure, which is not always available due to privacy concerns.

Several existing studies have considered the issue of unknown interference, but they still have certain limitations. Rakesh et al. [19] constructed a graph based on the similarity of unit covariates to fill in the missing graph information. Their approach cannot discover the interference among dissimilar units and only considers pairwise interference. Bhattacharya et al. [2] assumed partial interference rather than networked interference and proposed a greedy network structure search method to find a partial interference structure. Furthermore, this study only estimated ATE. In a different approach, Sävje et al. [25] did not construct interference structures but proposed an expected average treatment effect (EATE) estimation method under unknown interference. However, this method still cannot estimate ITE, which is often crucial for many applications, such as advertising pushing. To overcome their limitations, this study proposes an end-to-end method that models the networked interference and estimates ITE without relying on information about graph structures. In our experiments, we compare the proposed method with applying the existing GCN-based methods [17] on the similarity graph [19] of units.

3 Preliminaries

Let $\mathbf{x}_i \in \mathbb{R}^d$ be the covariates (e.g., age) of a unit i , $t_i \in \{0, 1\}$ be the treatment assigned to the unit i (e.g., an advertisement), \mathbf{X} be the covariates of all units, and \mathbf{T} be the treatments of all units. We use non-bold, italicized, and capitalized letters to denote random variables, such as X_i , and use subscript $-i$ to denote all the other units except i , such as \mathbf{T}_{-i} . This study considers that units are potentially connected, and we use \mathbf{s}_i to represent the summary of neighbor information (i.e., information of units connected to the unit i) for the unit i . Potential outcomes can be indexed only by the individual treatment

and the neighbor information [4], denoted as $y_i^{T_i}(S_i)$. Let $y_i^t(\mathbf{s}_i)$ be the simplified expression for $y_i^{T_i=t}(S_i = \mathbf{s}_i)$. Let $y_i^{\text{obs}} \in \mathbb{R}$ be the observed outcome of the unit i under the actual value of t_i and \mathbf{s}_i . The covariates, treatments, and observed outcomes of units constitute observational data $\{(\mathbf{x}_i, t_i, y_i^{\text{obs}})\}_{i=1}^N$.

Conventionally, the ITE is defined in a scenario where there is no interference among units [21], under the so-called stable unit treatment value assumption (SUTVA) [21]. Such a definition of the ITE characterizes the intrinsic treatment effect of every unit. We can formulate such an ITE by masking out neighbor information S in the potential outcomes using $\tau(X_i = \mathbf{x}_i, S_i = \mathbf{0}) = \mathbb{E}[Y_i^1(S_i = \mathbf{0})|X_i = \mathbf{x}_i] - \mathbb{E}[Y_i^0(S_i = \mathbf{0})|X_i = \mathbf{x}_i]$.

We now introduce graphs. A graph contains information about both units and their relationships, represented by connections or edges among units. Let $\mathbf{A} \in \{0, 1\}^{N \times N}$ be the adjacency matrix of the graph. If there is a direct edge from a unit j to a unit i , i.e., the unit j is a neighbor of the unit i , $A_{ij} = 1$; otherwise, $A_{ij} = 0$. In this study, we assume that the units are potentially connected through an interference graph \mathbf{A}^{itf} . However, due to observational limitations and privacy protection issues, \mathbf{A}^{itf} is unknown.

In the situation where units are associated with unknown graphs, the SUTVA cannot hold, as connected units often propagate information. Therefore, this study assumes that there exists unknown interference, where the outcome of a unit is not only influenced by its own treatment and covariates but also influenced by those of its neighbors on the unknown interference graph \mathbf{A}^{itf} , i.e., $S \neq \mathbf{0}$. In this case, the ITE under $S_i = \mathbf{s}_i$ can be formulated using the following equation, similar to the definition of ITE under interference in the existing study [17]: $\tau(X_i = \mathbf{x}_i, S_i = \mathbf{s}_i) = \mathbb{E}[Y_i^1(S_i = \mathbf{s}_i)|X_i = \mathbf{x}_i] - \mathbb{E}[Y_i^0(S_i = \mathbf{s}_i)|X_i = \mathbf{x}_i]$.

Apart from interference, the issue of confounders is a well-known challenge in observational data [17]. Confounders are parts of covariates and affect both the treatment assignment and outcome. Confounders result in an imbalance in the population distributions with different treatment assignments [10]. Without properly addressing confounders, ITE estimation will be biased.

4 Proposed Method: Treatment Effect Estimation Under Unknown Interference

This study aims to estimate both the ITE without interference $\tau(X_i = \mathbf{x}_i, S_i = \mathbf{0})$ and ITE with unknown interference $\tau(X_i = \mathbf{x}_i, S_i = \mathbf{s}_i)$ from observational data $\{(\mathbf{x}_i, t_i, y_i^{\text{obs}})\}_{i=1}^N$. To this end, we propose a method called Treatment Effect Estimator under Unknown Interference (UNITE). We prove that ITE is identifiable in Appendix A. A diagram for the proposed UNITE is provided in Fig. 2. We released codes of UNITE at <https://github.com/LINXF208/UNITE>. UNITE contains four components: a covariate representation learner ϕ , a GSL, an aggregation function ψ , and two outcome predictors. The first component ϕ learns representations of covariates to address confounders, and it also makes the representations interference-free in order to estimate $\tau(X_i = \mathbf{x}_i, S_i = \mathbf{0})$. The second component GSL infers an interference graph from observed data.

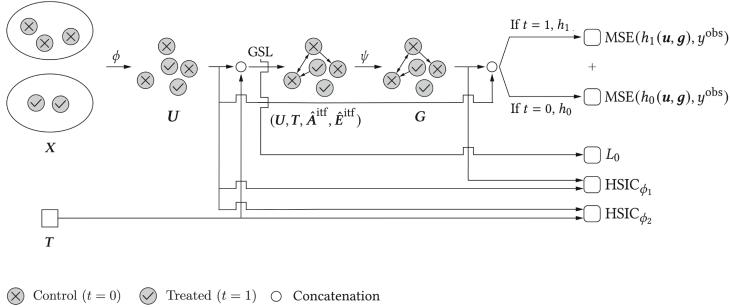


Fig. 2. Model architecture of UNITE.

The third component ψ utilizes a multi-layered GCN to aggregate interference received by units and transform aggregated results into unit representations, referred to as interference representations. Finally, the last component predicts potential outcomes using the covariate and interference representations.

4.1 Covariate Representation Learner

The covariate representation learner ϕ serves two primary purposes. Firstly, it aims to mitigate the imbalance in the distributions of different treatment groups caused by confounders. To this end, ϕ maps covariates to a representation space where treatment assignments and covariate representations become approximately independent [17]. Secondly, ϕ also aim to generate interference-free covariate representations for estimating $\tau(X_i = \mathbf{x}_i, S_i = \mathbf{0})$. This is similar to finding a representation space where interference representations and covariate representations are approximately independent.

To be specific, let ϕ be a covariate representation learner, $\mathbf{u}_i = \phi(\mathbf{x}_i; \mathbf{W}_\phi)$ be the covariate representation of a unit i , \mathbf{W}_ϕ be the paramaters of the ϕ , and \mathbf{U} be the covariate representations of all units.

We now introduce a Hibelt-Schmidt independence criterion (HSIC) regularization [5], a criterion for testing the independence of two random variables, with values ranging from 0 to 1. If two random variables are independent, the HSIC equals 0. Thus, minimizing the HSIC between treatment assignments and covariate representations, and minimizing the HSIC between interference representations and covariate representations can achieve the goals of ϕ . We use an efficient approach for calculating HSIC [5], as described in Appendix B. Let HSIC_{ϕ_1} denote the HSIC between covariate representations and treatment assignments, HSIC_{ϕ_2} denote the HSIC between covariate and interference representations, and $\mathcal{L}_\phi = \beta_1 \text{HSIC}_{\phi_1} + \beta_2 \text{HSIC}_{\phi_2}$ denote the entire HSIC regularization, where β_1 and β_2 are hyperparameters.

4.2 Graph Structure Learner

The challenge of GSL is that, although a unit might be potentially connected to all other units in the data, the outcome of a unit usually receives influence from only parts of other units. Furthermore, the level of influence can be different for different influencers. For example, suppose we want to estimate the effect of a promotion for a specific computer on all the students in a university. A student can receive more influence from peers who take the same classes than students he or she does not meet on campus. The level of influence received will depend on the proximity between the influencer and the influenced units. Thus, the goal of our GSL is to infer the structure of the interference graph \mathbf{A}^{itf} and the level of interference \mathbf{E}^{itf} , which are scalar weights for edges of \mathbf{A}^{itf} .

Inspired by Ye et al. [29], we infer the interference graph from the complete graph of units by associating binary gates to edges. Let $\mathbf{A}^{\text{com}} \in \{1\}^{N \times N}$ be the adjacency matrix of the complete graph, $z_{ij} \in \{0, 1\}$ be a binary gate between unit i and j , and \mathbf{Z} be all the binary gates. $z_{ij} = 1$ if the unit i receives interference from the unit j , and $z_{ij} = 0$ otherwise.¹ Then, the \mathbf{A}^{itf} is given by $\mathbf{A}^{\text{itf}} = \mathbf{A}^{\text{com}} \odot \mathbf{Z}$, where the \odot represents the Hadamard product. To infer the edges that actually cause interference, the UNITE imposes the L_0 -norm regularization on the gates, which is defined as $\mathcal{L}_{\text{GSL}} = \frac{1}{N^2} \sum_{(i,j)} 1_{[z_{ij} \neq 0]}$, where $1_{[\cdot]}$ is the indicator function. The gates are computed using the information of units available to the GSL.

However, direct optimization of the \mathcal{L}_{GSL} is intractable. As the interference graph \mathbf{A}^{itf} is determined by the \mathbf{Z} and used by other modules of the UNITE, direct optimization of the \mathcal{L}_{GSL} requires enumerating all possible binary matrices in $\{0, 1\}^{N \times N}$. Thus, we apply a hard concrete estimator technique proposed by Louizos et al. [15] for approximating the L_0 -norm regularization. This technique is based on an observation that, for any distribution over the binary gates $\Pr(\mathbf{Z})$, we have $\min_{\mathbf{Z}} \mathcal{L}_{\text{GSL}} \leq \mathbb{E}_{\mathbf{Z} \sim \Pr(\mathbf{Z})} [\mathcal{L}_{\text{GSL}}]$, which means that the minimum of the \mathcal{L}_{GSL} is upper bounded by its expectation over a sampling distribution $\Pr(\mathbf{Z})$. We approximate z_{ij} by sampling \hat{z}_{ij} from a hard concrete distribution and use \hat{z}_{ij} to compute a sample for the interference graph $\hat{\mathbf{A}}^{\text{itf}}$ during learning.

Specifically, the hard concrete distribution $\Pr(\mathbf{Z}; \log \boldsymbol{\alpha})$ can be considered as a continuous approximation to the Bernoulli distribution, which admits using a reparameterization trick for efficient optimization. Suppose v is a random variable that follows the uniform distribution over $[0, 1]$, i.e. $v \sim \text{Unif}(0, 1)$. Then, we can express a sample of \hat{z}_{ij} using a sample of v as follows:

$$\hat{z}_{ij} = \min \{1, \max \{0, \text{sigmoid}((\log v - \log(1-v) + \log \alpha_{ij})/\eta_1)(\eta_3 - \eta_2) + \eta_2\}\}, \quad (1)$$

where $\eta_1 = 2/3$, $\eta_2 = -0.1$, $\eta_3 = 1.1$ are typical parameters for the hard concrete distribution [15]. Here, $\log \alpha_{ij}$ can be approximated by a function of the information for unit i and unit j available to the GSL. Let $\mathbf{p}_i, \mathbf{p}_j \in \mathbb{R}^{d'}$ be the information of the unit i and j , which is the concatenation of corresponding unit covariate

¹ Note that we assume the edges of the interference graph to be directed.

representation \mathbf{u} and treatment t . Here, $\log \alpha_{ij} = [\mathbf{p}_i^T \mathbf{W}_{\text{GSL}} \| \mathbf{p}_j^T \mathbf{W}_{\text{GSL}}] \mathbf{o}_{\text{GSL}}$, where $\mathbf{o}_{\text{GSL}} \in \mathbb{R}^{2d''}$ and $\mathbf{W}_{\text{GSL}} \in \mathbb{R}^{d' \times d''}$ are parameters to be learned. $\|$ means the concatenation operation for two vectors. In this case, during training \mathcal{L}_{GSL} can be approximated by $\hat{\mathcal{L}}_{\text{GSL}} = \frac{\lambda}{N^2} \sum_{(i,j)} \text{sigmoid} \left(\log \alpha_{ij} - \eta_1 \log \frac{-\eta_2}{\eta_3} \right)$, where λ is a hyperparameter. Ideally, most of the elements in the $\hat{\mathbf{Z}}$ will be zeros under the L_0 -norm regularization, i.e., L_{GSL} ; thus, the model will preserve only a few crucial edges. Units j with remaining edges, i.e., $\hat{A}_{ij}^{\text{itf}} > 0$ are considered to cause interference to a unit i , where $\hat{A}_{ij}^{\text{itf}} = 1_{[\hat{z}_{ij} > 0]}$.

A sample \hat{z}_{ij} is a continuous approximation of the binary gate to the edge from the unit j to i . The larger \hat{z}_{ij} is, the more confident the proposed UNITE is about the unit j causing interference to unit i . In other words, a higher value of \hat{z}_{ij} implies a higher importance for the interference from the j to i . However, there usually exist discrepancies in the number of remaining edges and neighbors for different units. In this case, if we use \hat{z}_{ij} to aggregate interference and compute interference representations, there will be significant numerical discrepancies in the representations of different units. This can impact the performance and stability of the proposed method. To overcome this issue, we approximate the level of interference \hat{e}_{ij} between unit i and j by normalizing \hat{z}_{ij} by $\hat{e}_{ij} = \frac{\hat{z}_{ij}}{\sum_{\hat{z}_{ik} > 0} \hat{z}_{ik}}$.² After normalizing the \hat{z}_{ij} , the value of \hat{e}_{ij} is the level of interference relative to other remaining neighbors of i . This is similar to graph attention mechanisms [27, 29].

Once the training of the GSL under the L_{GSL} is complete, most of the elements in the $\hat{\mathbf{Z}}$ are zeros; therefore, only a few units j with remaining edges (i.e., $\hat{A}_{ij}^{\text{itf}} > 0$) are considered to cause interference to a unit i . In the test phase, we stop sampling v from $\text{Unif}(0, 1)$. Then, \hat{z}_{ij} is generated by $\hat{z}_{ij} = \min \{1, \max \{0, \text{sigmoid}((\log \alpha_{ij})/\eta_1)(\eta_3 - \eta_2) + \eta_2\}\}$.

4.3 Aggregation Function

Given a sampled interference graph $\hat{\mathbf{A}}^{\text{itf}}$, the level of interference $\hat{\mathbf{E}}^{\text{itf}}$, covariate representations \mathbf{U} , and treatments \mathbf{T} , the aggregation function ψ computes unit representations \mathbf{G} as the summary of the neighbor information: $\mathbf{G} = \psi(\mathbf{U}, \mathbf{T}, \hat{\mathbf{A}}^{\text{itf}} \odot \hat{\mathbf{E}}^{\text{itf}})$. Here, we use \mathbf{W}_ψ to denote all parameters of the ψ . For a unit i , its representation \mathbf{g}_i is supposed to capture the interference of its potential neighbors, called the interference representation of the unit i .

The ψ of the UNITE employs a multi-layered GCN [28] to aggregate and model interference. A GCN layer can aggregate information from unit neighbors into a representation for each unit via a message-propagation mechanism. Let $\mathbf{g}_i^{(l)}$ be the representation of unit i computed by the l -th layer, and let $\mathbf{G}^{(l)}$ be the representation of all units computed by this layer. We use $\mathbf{g}_i^{(0)}$ and $\mathbf{G}^{(0)}$ as

² As the information of unit i itself is always important for computing the level of interference, we set \hat{z}_{ii} to 1.

input to the first layer of GCN, which is the concatenation of the covariate representations \mathbf{U} and treatments \mathbf{T} . The message-passing result, i.e., the calculation of the l -th GCN layer for a unit i can be expressed as follows:

$$\mathbf{g}_i^{(l)} = \sigma \left((\hat{\mathbf{A}}^{\text{itf}} \odot \hat{\mathbf{E}}^{\text{itf}}) \mathbf{G}^{(l-1)} \mathbf{W}_{\psi}^{(l)} \right),$$

where $\mathbf{W}_{\psi}^{(l)}$ is a parameter matrix for the l -th layer, and $\sigma(\cdot)$ is a non-linear activation function. This equation shows that $\mathbf{g}_i^{(l)}$ is a weighted sum of the representations of neighbors of the unit i , weighted by their level of interference. A single GCN layer captures one-hop neighbor information, and to capture networked interference, we stack multiple GCN layers.

4.4 Outcome Predictors and ITE Estimators

Inspired by prior works [10, 17, 23], given \mathbf{U} and \mathbf{G} , we use feedforward neural networks to parameterize predictors h_0 and h_1 for predicting potential outcomes of $t = 0$ and $t = 1$. Denote by \mathbf{W}_h for their parameters. The outcome predictors are trained by minimizing the mean square error (MSE) between their outputs and the observed outcomes, denoted by $\mathcal{L}_h = \frac{1}{n} \sum_{i=1}^n (h_{t_i}(\mathbf{u}_i, \mathbf{g}_i) - y_i^{\text{obs}})^2$.

To learn the model parameters (\mathbf{W}_{ϕ} , \mathbf{W}_{GSL} , \mathbf{o}_{GSL} , \mathbf{W}_{ψ} , and \mathbf{W}_h) of UNITE, we now introduce the model optimization. As it is intractable to directly optimize the L_0 -norm regularization of the GSL directly, we sample $\hat{\mathbf{Z}}$ using Eq. (1) to construct samples of interference graphs during training. We optimizes the above parameters by minimizing the loss function $\hat{\mathcal{L}}$, defined as

$$\hat{\mathcal{L}}(\mathbf{W}_{\phi}, \mathbf{W}_{\text{GSL}}, \mathbf{o}_{\text{GSL}}, \mathbf{W}_{\psi}, \mathbf{W}_h) = \mathbb{E}_{\hat{\mathbf{Z}}} [\mathcal{L}_{\phi} + \mathcal{L}_{\psi} + \mathcal{L}_h] + \hat{\mathcal{L}}_{\text{GSL}}.$$

With the outcome predictors, we can estimate the ITE under interference using $\hat{\tau}(X = \mathbf{x}_i, S_i = \mathbf{s}_i) = h_1(\mathbf{u}_i, \mathbf{g}_i) - h_0(\mathbf{u}_i, \mathbf{g}_i)$, and estimate the interference-free ITE using $\hat{\tau}(X_i = \mathbf{x}_i, S_i = \mathbf{0}) = h_1(\mathbf{u}_i, \mathbf{0}) - h_0(\mathbf{u}_i, \mathbf{0})$. In addition, ATE estimation with or without interference is performed by averaging the outputs of $\hat{\tau}(X_i = \mathbf{x}_i, S_i = \mathbf{s}_i)$ or $\hat{\tau}(X_i = \mathbf{x}_i, S_i = \mathbf{0})$.

5 Experiments

We carried out experiments on three public datasets and compared the performance of the proposed methods with that of various existing methods to verify the effectiveness of the proposed method.

5.1 Experiment Settings

Datasets. We evaluated the proposed method using the following datasets.

Jobs Dataset [11]: The Jobs dataset is a widely used benchmark in the causal inference for observational studies [19, 23]. It comprises a random control trial (RCT) subset, known as the LaLonde experimental sample, which has 297

treated and 425 control units, and the PSID comparison group, which includes 2,490 control units [24]. The dataset includes information about workers (covariates), the status of job training participation (treatment), and employment status (outcomes), gathered from real-world observations. The objective is to estimate the effect of job training on unemployment. Following Rakesh [19] who considered interference in the Jobs dataset, we also consider interference for this dataset, although it lacks information about connections. We averaged results over ten different training/validation/testing splits with ratios of 64/16/20.

Amazon Datasets [8]: Rakesh [19] used the co-purchase graph from the Amazon datasets [8] to explore the effect of positive and negative reviews on product sales and interference issues. To study these effects separately, Rakesh [19] divided the dataset into Amazon^- (comprising only negative reviews) and Amazon^+ (comprising only positive reviews). Units in these datasets are connected through directed graphs. In Amazon^- , there are 14,538 units with 15,011 connections. For the Amazon^+ dataset, we extract the first 10,000 units with 3,000 connections. The treatment t (with values 0 or 1) depends on the number of negative (or positive) reviews a unit has: $t = 1$ if a unit has more than three negative (or positive) reviews, and $t = 0$ if it has fewer than three [19]. Each unit's covariate vector \mathbf{x} consists of 300 features generated from their reviews using the doc2vec method. The ITE under interference (i.e., $\tau(X_i = \mathbf{x}_i, S_i = s_i)$) for each unit is estimated using matching methods [19]. We follow Ma [17], conducting three repeated experiments and averaging the results for the Amazon^- and Amazon^+ datasets, with a train/validation/test ratio of 80/5/15.

Baselines. We compare UNITE with various baseline methods, and they can be divided into the following categories:

No Graph: We compare UNITE with traditional methods, which do not consider graphs and ignore interference. BNN [10] addresses confounders by minimizing distribution discrepancies of covariate representation between different treatment groups but does not account for interference. We consider two BNN structures: BNN-4-0 with four representation layers and a linear output layer, and BNN-2-2 with two representation layers, two hidden prediction layers, and a linear output layer. CFR [23] employs a strategy similar to BNN, using the Maximum Mean Distance (MMD) and Wasserstein distance to penalize distribution discrepancies in the representation space. We consider two CFR schemes: CFR-MMD, using MMD for distribution discrepancy, and CFR-Wass, using the Wasserstein distance. TARNet [23] has the same model architecture as the CFR but removes the distribution discrepancy penalty term.

GCN-Based Methods [17]: GCN-based methods use GCNs [28] to model interference. To verify their performance under unknown interference, we construct k -NN graphs (with $k=10$) based on cosine similarity between units, which is inspired by Chen et al. [3] and Rakesh et al. [19]. The same graph structures are used throughout the training, validation, and testing phases. Two regularization schemes are considered: $\text{GCN}\phi$, which uses HSIC to balance outputs of representation layers, and $\text{GCN}\psi$, which uses HSIC to balance outputs of GCN layers. We use $\text{GCN}\phi/\psi + k\text{-NN}$ to denote GCN-based methods with k -NN graphs.

Metrics. To verify the quality of $\hat{\tau}(X = \mathbf{x}, S_i = \mathbf{0})$ estimation, we calculate $\hat{R}^{\text{pol}}(\pi_f)$ in the balanced and interference-free RCT subset, following prior work [23]: $\hat{R}^{\text{pol}}(\pi_f) = 1 - (\mathbb{E}[y_1^f | \pi_f(\mathbf{x}) = 1, t = 1] \cdot p(\pi_f = 1) + \mathbb{E}[y_0^f | \pi_f(\mathbf{x}) = 0, t = 0] \cdot p(\pi_f = 0))$. Here, $\pi_f(\mathbf{x}) = 1$ if $\hat{\tau}(X = \mathbf{x}, S_i = \mathbf{0}) > 0$, and $\pi_f(\mathbf{x}) = 0$, otherwise. We also calculate ϵ^{ATT} , which represents the error in ATE estimation for the treated group (ATT) in the balanced and interference-free RCT subset: $\epsilon^{\text{ATT}} = \left| \frac{1}{|\mathbf{R}^t|} \sum_{i \in \mathbf{R}^t} \tau(X_i = \mathbf{x}_i, S_i = \mathbf{0}) - \frac{1}{|\mathbf{R}^t|} \sum_{i \in \mathbf{R}^t} \hat{\tau}(X_i = \mathbf{x}_i, S_i = \mathbf{0}) \right|$, where \mathbf{R}^t represents the set of treated units of the RCT subset. For Amazon⁻, and Amazon⁺ datasets, we calculate ϵ^{PEHE} , which represents the error in estimation of $\tau(X_i = \mathbf{x}_i, S_i = \mathbf{s}_i)$ and is defined as $\epsilon^{\text{PEHE}} = \frac{1}{n} \sum_{i=1}^n (\hat{\tau}(X_i = \mathbf{x}_i, S_i = \mathbf{s}_i) - \hat{\tau}(X_i = \mathbf{x}_i, S_i = \mathbf{s}_i))^2$. Further, we compute ϵ^{ATE} , which represents the error in ATE estimation under interference, and is defined as $\epsilon^{\text{ATE}} = \left| \frac{1}{n} \sum_{i=1}^n \tau(X_i = \mathbf{x}_i, S_i = \mathbf{s}_i) - \frac{1}{n} \sum_{i=1}^n \hat{\tau}(X_i = \mathbf{x}_i, S_i = \mathbf{s}_i) \right|$.

The implementation details are elaborated in Appendix C.

5.2 Results

To verify the claim that existing unknown interference and the proposed methods can address unknown interference, we conducted experiments on the Jobs, Amazon⁻, and Amazon⁺ datasets. The Amazon⁻ and Amazon⁺ datasets contain interference and provide graph structures [17, 19]. We hid the graph structure to simulate the unknown interference. Table 1 shows that the proposed UNITE yields better results than the baseline methods on the Jobs. Comparing the results of the UNITE on the Jobs dataset with those of the baseline methods (such as CFR) that ignore interference, we can observe that considering addressing interference can help improve the performance of the ITE and ATE estimation. It implies that there might be interference between units even if there is no observed information on the connection between units. The results also reveal that UNITE achieves lower errors in ITE estimation on the Amazon⁻ and Amazon⁺ datasets than the other baseline methods. Meanwhile, the error in

Table 1. Results on the test sets of the Jobs, Amazon⁻, and Amazon⁺ datasets under unknown interference. The proposed UNITE outperforms the baseline methods in the ITE estimation under unknown interference.

Method	Jobs		Amazon ⁻		Amazon ⁺	
	$\hat{R}^{\text{pol}}(\pi_f)$	ϵ^{ATT}	ϵ^{PEHE}	ϵ^{ATE}	ϵ^{PEHE}	ϵ^{ATE}
TARNet	0.22 ± 0.04	0.05 ± 0.02	1.79 ± 0.01	0.24 ± 0.01	1.80 ± 0.02	0.06 ± 0.00
BNN-2-2	0.26 ± 0.07	0.07 ± 0.01	2.01 ± 0.02	0.17 ± 0.01	1.93 ± 0.01	0.06 ± 0.01
BNN-4-0	0.23 ± 0.05	0.07 ± 0.00	1.96 ± 0.00	0.06 ± 0.04	1.92 ± 0.00	0.04 ± 0.01
CFR-MMD	0.21 ± 0.05	0.10 ± 0.05	1.79 ± 0.01	0.25 ± 0.02	1.84 ± 0.00	0.07 ± 0.01
CFR-Wass	0.21 ± 0.04	0.09 ± 0.05	1.79 ± 0.01	0.24 ± 0.01	1.85 ± 0.02	0.05 ± 0.00
GCN $_{\phi}$ + k-NN	0.22 ± 0.04	0.06 ± 0.01	1.87 ± 0.04	0.06 ± 0.01	1.85 ± 0.03	0.04 ± 0.03
GCN $_{\psi}$ + k-NN	0.22 ± 0.04	0.06 ± 0.01	1.90 ± 0.03	0.19 ± 0.03	1.85 ± 0.01	0.17 ± 0.02
UNITE (ours)	0.19 ± 0.03	0.04 ± 0.02	1.57 ± 0.06	0.07 ± 0.03	1.72 ± 0.01	0.07 ± 0.02

ATE estimation is comparable to the other baseline methods. These results verify that the proposed methods can address unknown interference. In addition, these results indicate that the GCN-based methods with k -NN graphs cannot correctly address the unknown interference, as they cannot achieve better performance in the ITE estimation than other baseline methods, such as CFR-MMD.

We also conducted ablation experiments, which are presented in Appendix D.

6 Conclusion

In this study, we proposed UNITE, which models unknown interference and estimates the ITE in the presence of unknown interference. We performed and evaluated the proposed method on three public datasets to verify the effectiveness of the proposed method. The results indicate that UNITE is powerful in addressing unknown interference.

Currently, the proposed method is limited to the binary treatment setting; extending the setting to multi-valued treatments is a possible future work.

Acknowledgements. This study was supported by JSPS KAKENHI Grant Number 20H04244.

A Identifiability of the Expectation of Potential Outcomes

Here, we show that the expectation of potential outcomes $Y_i^t(\mathbf{s}_i)$ can be identified from observed data. To this end, we need some assumptions.

Inspired by Chen and Ji [3] who learn graph structures using the features of units, we have the following assumption.

Assumption 1 (A1). *The unknown structure \mathbf{A}^{itf} can be discovered from the \mathbf{X} and \mathbf{T} using a graph structure learner $GSL(\cdot)$, i.e., $\mathbf{A}^{\text{itf}} = GSL(\mathbf{X}, \mathbf{T})$.*

Similar to the existing studies on addressing interference [16, 17], we assume the interference can be aggregated, as the following assumption.

Assumption 2 (A2). *There exists an aggregation function $\psi(\cdot)$, which can aggregate information of other units on the graph \mathbf{A}^{itf} while outputting the \mathbf{s} , i.e., $\mathbf{s}_i = \psi(\mathbf{T}_{-i}, \mathbf{X}_{-i}, \mathbf{A}^{\text{itf}})$.*

We extend the neighbor interference assumption [4] to the networked interference [17].

Assumption 3 (A3). *For $\forall i$, $\forall \mathbf{T}_{-i}, \mathbf{T}'_{-i}, \forall \mathbf{X}_{-i}, \mathbf{X}'_{-i}$, and $\forall \mathbf{A}^{\text{itf}}, \mathbf{A}^{\text{itf}'} : \text{when } \mathbf{s}_i = \mathbf{s}'_i$, i.e., $\psi(\mathbf{T}_{-i}, \mathbf{X}_{-i}, \mathbf{A}^{\text{itf}}) = \psi(\mathbf{T}'_{-i}, \mathbf{X}'_{-i}, \mathbf{A}^{\text{itf}'})$, $Y_i^t(S_i = \mathbf{s}_i) = Y_i^t(S_i = \mathbf{s}'_i)$ holds.*

We use the consistency assumption which is similar to the consistency assumption in the existing study on interference [4].

Assumption 4 (A4). $Y_i^{\text{obs}} = Y_i^{t_i}(S_i = \mathbf{s}_i)$ on the graph \mathbf{A} for a unit i under t_i and \mathbf{s}_i .

We take a similar unconfoundedness assumption of the existing study on addressing interference [4].

Assumption 5 (A5). There is no hidden confounder. For any unit i , given the covariates, the treatment assignment and output of the aggregation function are independent of potential outcomes, i.e., $T_i, S_i \perp\!\!\!\perp Y_i^1(\mathbf{s}_i), Y_i^0(\mathbf{s}_i) | X_i$.

We now prove the identification of the expectation of potential outcomes $Y_i^t(\mathbf{s}_i)$ ($t = 1$ or $t = 0$) as follows:

$$\mathbb{E}[Y_i^{\text{obs}} | X_i = \mathbf{x}_i, T_i = t, X_{-i} = \mathbf{X}_{-i}, T_{-i} = \mathbf{T}_{-i}, X = \mathbf{X}, T = \mathbf{T}]$$

$$= \mathbb{E}[Y_i^{\text{obs}} | X_i = \mathbf{x}_i, T_i = t, X_{-i} = \mathbf{X}_{-i}, T_{-i} = \mathbf{T}_{-i}, A^{\text{itf}} = \mathbf{A}^{\text{itf}}] \quad (A1)$$

$$= \mathbb{E}[Y_i^{\text{obs}} | X_i = \mathbf{x}_i, T_i = t, S_i = \mathbf{s}_i] \quad (A2)$$

$$= \mathbb{E}[Y_i^t(\mathbf{s}_i) | X_i = \mathbf{x}_i, T_i = t, S_i = \mathbf{s}_i] \quad (A3, A4)$$

$$= \mathbb{E}[Y_i^t(\mathbf{s}_i) | X_i = \mathbf{x}_i]. \quad (A5)$$

Inspired by the above proof, if unknown interference is properly modeled, the potential outcomes can be modeled.

B HSIC

The calculation of HSIC is as follows:

$$\text{HSIC}(\mathbf{B}, \mathbf{C}) = \frac{1}{(n-1)^2} \text{tr}(\mathbf{K} \mathbf{H} \mathbf{L} \mathbf{H}), \quad \mathbf{H} = \mathbf{I}_n - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^T, \quad (2)$$

where $\mathbf{B} \in \mathbb{R}^{n \times d_1}$ and $\mathbf{C} \in \mathbb{R}^{n \times d_2}$ denote two different matrices or vectors, \mathbf{I}_n is the identity matrix, $\mathbf{1}_n$ is a vector of all ones, and \cdot^T is the transposition operation. \mathbf{K} and \mathbf{L} are Gaussian kernels applied to \mathbf{B} and \mathbf{C} , respectively:

$$K_{ij} = \exp\left(-\frac{\|\mathbf{b}_i - \mathbf{b}_j\|_2^2}{2}\right), \quad L_{ij} = \exp\left(-\frac{\|\mathbf{c}_i - \mathbf{c}_j\|_2^2}{2}\right), \quad (3)$$

where vectors \mathbf{b}_i (or \mathbf{b}_j) and \mathbf{c}_i (or \mathbf{c}_j) represent the elements of the i -th (or j -th) row of \mathbf{B} and \mathbf{C} , respectively.

C Implementation Details

Following Ma et al. [17], the entire \mathbf{X} and \mathbf{T} are given during the training, validation, and testing phases. However, only the observed outcomes of the units in the training set are provided during the training phase.

The three hidden layers of the representation layers of UNITE have (128, 64, 64)-dimensions for the Jobs, Amazon⁻, and Amazon⁺ datasets. The

hidden layer of the GSL of the UNITE has a 128 dimension for the Jobs dataset and 64 for the Amazon⁻ and Amazon⁺ datasets. The hidden layers of the GCN layers of the UNITE have (64, 32)-dimensions for the Jobs dataset and (64, 64, 32)-dimensions for the Amazon⁻ and Amazon⁺ datasets. The hidden layers of the prediction networks of the UNITE have (128, 64, 64)-dimensions for the Jobs dataset and (128, 64, 32)-dimensions for the Amazon⁻ and Amazon⁺ datasets.

In addition, we train our models with the GPU of the NVIDIA RTX A5000. The UNITE utilizes the Adam optimizer with 500 training iterations for the Jobs dataset and 2,000 training iterations for the Amazon⁻, and Amazon⁺ datasets. The learning rate is set to $\lambda_{lr} = 0.0005$ for the Jobs dataset and $\lambda_{lr} = 0.001$ for the Amazon⁻, and Amazon⁺ datasets, and the weight decay γ is set to $\gamma = 0.01$ for the Jobs dataset and $\gamma = 0.001$ for the Amazon⁻, and Amazon⁺ datasets.

We use the grid search method to search for hyperparameters by checking the results on the validation set. The train batch size of the UNITE is the full batch of training units for the Jobs and is searched from {128, 256, 512, 1024} for the Amazon⁻, and Amazon⁺ datasets. The β_1 and β_2 of the UNITE are searched from {0.01, 0.05, 0.1, 0.15, 0.2} for the Jobs and {0.1, 0.5, 1.0, 1.5, 2.0} for the Amazon⁻ and Amazon⁺ datasets. The λ of the UNITE is 0.0005 for the Jobs datasets, 0.02 for the Amazon⁻ datasets, and 0.01 for the Amazon⁺ datasets. Dropout is applied for the UNITE (the dropout rate is searched from {0.0, 0.1, 0.5}). Moreover, after 400 iterations, early stopping is applied for the Amazon⁻, and Amazon⁺ datasets to avoid overfitting. The UNITE-KG uses the same hyperparameters as the UNITE, but it sets the $\lambda = 0$.

Here, all the baseline methods use the default hyperparameter or search for hyperparameters from the ranges suggested in the literature. To avoid overfitting, all the baseline methods apply early stopping on the Amazon⁻, and Amazon⁺ datasets.

For the Amazon⁻, and Amazon⁺ datasets, as the y values span a wide range, z-score normalization is applied during training and testing.

D Ablation Experiments

To investigate the importance of different regularization terms, we conduct ablation experiments on the Jobs and Amazon⁻ datasets. Table 2 shows the results of the ablation experiments. We can observe that removing the $HSIC_{\phi_1}$, or $HSIC_{\phi_2}$, or L_0 results in performance degradation for the ITE estimation. This verifies that these regularization terms are important for the ITE estimation of UNITE. In addition, the results on the Jobs dataset show the performance degradation in the interference-free ITE and ATE estimation when UNITE removes the $HSIC_{\phi_2}$. This implies that the $HSIC_{\phi_2}$ is important to estimate the interference-free ITE and ATE estimation.

Table 2. Results of the ablation experiments on the Jobs and Amazon⁻ datasets.

Method	Jobs		Amazon ⁻	
	$\hat{R}^{\text{pol}}(\pi_f)$	ϵ^{ATT}	ϵ^{PEHE}	ϵ^{ATE}
UNITE	0.19 ± 0.03	0.04 ± 0.02	1.57 ± 0.06	0.07 ± 0.03
w/o HSIC $_{\phi_1}$	0.22 ± 0.03	0.08 ± 0.04	1.61 ± 0.08	0.38 ± 0.06
w/o HSIC $_{\phi_2}$	0.21 ± 0.03	0.08 ± 0.04	1.58 ± 0.07	0.07 ± 0.04
w/o L_0	0.21 ± 0.03	0.08 ± 0.05	1.60 ± 0.04	0.05 ± 0.03

References

1. Aronow, P.M., Samii, C.: Estimating average causal effects under general interference, with application to a social network experiment. *Ann. Appl. Stat.* **11**, 1912–1947 (2017)
2. Bhattacharya, R., Malinsky, D., Shpitser, I.: Causal inference under interference and network uncertainty. In: Proceedings of the 35th Uncertainty in Artificial Intelligence Conference, vol. 2019 (2019)
3. Chen, Y., Wu, L., Zaki, M.: Iterative deep graph learning for graph neural networks: Better and robust node embeddings. In: Advances in Neural Information Processing Systems, vol. 33, pp. 19314–19326 (2020)
4. Forastiere, L., Airoldi, E.M., Mealli, F.: Identification and estimation of treatment and interference effects in observational studies on networks. *J. Am. Stat. Assoc.* **116**(534), 901–918 (2021)
5. Gretton, A., Bousquet, O., Smola, A., Schölkopf, B.: Measuring statistical dependence with Hilbert-Schmidt norms. In: Proceedings of the 16th International Conference on Algorithmic Learning Theory, pp. 63–77 (2005)
6. Guo, R., Li, J., Li, Y., Candan, K.S., Raglin, A., Liu, H.: Ignite: a minimax game toward learning individual treatment effects from networked observational data. In: Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence, pp. 4534–4540 (2021)
7. Guo, R., Li, J., Liu, H.: Learning individual causal effects from networked observational data. In: Proceedings of the 13th International Conference on Web Search and Data Mining, pp. 232–240 (2020)
8. He, R., McAuley, J.: Ups and downs: modeling the visual evolution of fashion trends with one-class collaborative filtering. In: Proceedings of the 25th International Conference on World Wide Web, pp. 507–517 (2016)
9. Hudgens, M.G., Halloran, M.E.: Toward causal inference with interference. *J. Am. Stat. Assoc.* **103**(482), 832–842 (2008)
10. Johansson, F., Shalit, U., Sontag, D.: Learning representations for counterfactual inference. In: Proceedings of the 33rd International Conference on Machine Learning, vol. 48, pp. 3020–3029 (2016)
11. LaLonde, R.J.: Evaluating the econometric evaluations of training programs with experimental data. *Am. Econ. Rev.* 604–620 (1986)
12. Li, Q., Wang, Z., Liu, S., Li, G., Xu, G.: Deep treatment-adaptive network for causal inference. *Int. J. Very Large Data Bases*, 1–16 (2022)

13. Lin, X., Zhang, G., Lu, X., Bao, H., Takeuchi, K., Kashima, H.: Estimating treatment effects under heterogeneous interference. In: Koutra, D., Plant, C., Gomez Rodriguez, M., Baralis, E., Bonchi, F. (eds.) ECML PKDD 2023. LNCS, vol. 14169, pp. 576–592. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-43412-9_34
14. Liu, L., Hudgens, M.G.: Large sample randomization inference of causal effects in the presence of interference. *J. Am. Stat. Assoc.* **109**(505), 288–301 (2014)
15. Louizos, C., Welling, M., Kingma, D.P.: Learning sparse neural networks through L_0 regularization. In: Proceedings of the 6th International Conference on Learning Representations (2018)
16. Ma, J., Wan, M., Yang, L., Li, J., Hecht, B., Teevan, J.: Learning causal effects on hypergraphs. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 1202–1212 (2022)
17. Ma, Y., Tresp, V.: Causal inference under networked interference and intervention policy enhancement. In: Proceedings of the 24th International Conference on Artificial Intelligence and Statistics, vol. 130, pp. 3700–3708 (2021)
18. Nabi, R., Pfeiffer, J., Charles, D., Kicman, E.: Causal inference in the presence of interference in sponsored search advertising. *Front. Big Data* **5** (2022)
19. Rakesh, V., Guo, R., Moraffah, R., Agarwal, N., Liu, H.: Linked causal variational autoencoder for inferring paired spillover effects. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 1679–1682 (2018)
20. Raudenbush, S.W., Schwartz, D.: Randomized experiments in education, with implications for multilevel causal inference. *Annu. Rev. Stat. Appl.* **7**(1) (2020)
21. Rubin, D.B.: Randomization analysis of experimental data: the fisher randomization test comment. *J. Am. Stat. Assoc.* **75**(371), 591–593 (1980)
22. Schnitzer, M.E.: Estimands and estimation of COVID-19 vaccine effectiveness under the test-negative design: connections to causal inference. *Epidemiology* **33**(3), 325 (2022)
23. Shalit, U., Johansson, F.D., Sontag, D.: Estimating individual treatment effect: generalization bounds and algorithms. In: Proceedings of the 34th International Conference on Machine Learning, vol. 70, pp. 3076–3085 (2017)
24. Smith, J.A., Todd, P.E.: Does matching overcome LaLonde's critique of nonexperimental estimators? *J. Econometrics* **125**(1–2), 305–353 (2005)
25. Sävje, F., Aronow, P.M., Hudgens, M.G.: Average treatment effects in the presence of unknown interference. *Ann. Stat.* **49**(2), 673–701 (2021)
26. Tchetgen, E.J.T., VanderWeele, T.J.: On causal inference in the presence of interference. *Stat. Methods Med. Res.* **21**(1), 55–75 (2012)
27. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: Proceedings of the 6th International Conference on Learning Representations (2018)
28. Welling, M., Kipf, T.N.: Semi-supervised classification with graph convolutional networks. In: Proceedings of the 4th International Conference on Learning Representations (2016)
29. Ye, Y., Ji, S.: Sparse graph attention networks. *IEEE Trans. Knowl. Data Eng.* (2021)



A New Loss for Image Retrieval: Class Anchor Margin

Alexandru Ghiță and Radu Tudor Ionescu^(✉)

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

Abstract. The performance of neural networks in content-based image retrieval (CBIR) is highly influenced by the chosen loss (objective) function. The majority of objective functions for neural models can be divided into metric learning and statistical learning. Metric learning approaches require a pair mining strategy that often lacks efficiency, while statistical learning approaches are not generating highly compact features due to their indirect feature optimization. To this end, we propose a novel repeller-attractor loss that falls in the metric learning paradigm, yet directly optimizes for the L_2 metric without the need of generating pairs. Our loss is formed of three components. One leading objective ensures that the learned features are attracted to each designated learnable class anchor. The second loss component regulates the anchors and forces them to be separable by a margin, while the third objective ensures that the anchors do not collapse to zero. Furthermore, we develop a more efficient two-stage retrieval system by harnessing the learned class anchors during the first stage of the retrieval process, eliminating the need of comparing the query with every image in the database. We establish a set of three datasets (CIFAR-100, Food-101, and ImageNet-200) and evaluate the proposed objective on the CBIR task, by using both convolutional and transformer architectures. Compared to existing objective functions, our empirical evidence shows that the proposed objective is generating superior and more consistent results.

Keywords: Loss Function · Image Retrieval · Deep Neural Networks

1 Introduction

Content-based image retrieval (CBIR) is a challenging task that aims to retrieve images from a database that are most similar to a query image. The state-of-the-art image retrieval systems addressing the aforementioned task are currently based on deep neural networks [3, 7, 18, 29, 30, 41]. Although neural models obtained impressive performance levels compared to handcrafted CBIR models [27], one of the main challenges that remain to be solved in training neural networks for retrieval problems is the choice of the objective function. Indeed, a well-chosen objective function should enhance the discriminative power of the

learned embeddings, i.e. the resulting features should exhibit small differences for images representing the same object and large differences for images representing different objects. This would implicitly make the embeddings more suitable for image retrieval.

The majority of loss functions used to optimize neural models can be separated into two major categories: statistical learning and metric learning. The most popular category is represented by statistical learning, which includes objective functions such as the cross-entropy loss [24] or the hinge loss [5]. The optimization problem is usually based on minimizing a particular probability distribution. Consequently, objective functions in this category achieve the desired properties as an indirect effect during optimization. Hence, objective functions based on statistical learning are more suitable for modeling multi-class classification tasks rather than retrieval tasks. The second category corresponds to metric learning, which comprises objective functions such as contrastive loss [12], triplet loss [32], and quadruplet loss [4]. Objective functions in this category operate directly in the embedding space and optimize for the desired distance metric. However, they usually require forming tuples of examples to compute the loss, which can be a costly step in terms of training time. To reduce the extra time required to build tuples, researchers resorted to hard example mining schemes [10, 11, 13, 32, 34, 40]. Still, statistical learning remains more time efficient.

In this context, we propose a novel repeller-attractor objective function that falls in the metric learning paradigm. Our loss directly optimizes for the L_2 metric without needing to generate pairs, thus alleviating the necessity of performing the costly hard example mining. The proposed objective function achieves this through three interacting components, which are expressed with respect to a set of learnable embeddings, each representing a class anchor. The leading loss function ensures that the data embeddings are attracted to the designated class anchor. The second loss function regulates the anchors and forces them to be separable by a margin, while the third objective ensures that the anchors do not collapse to the origin. In addition, we propose a two-stage retrieval method that compares the query embedding with the class anchors in the first stage, then continues by comparing the query embedding with image embeddings assigned to the nearest class anchors.

We carry out experiments on three datasets (CIFAR-100 [17], Food-101 [2], and ImageNet-200 [31]) to compare the proposed loss function against representative statistical and deep metric learning objectives. We evaluate the objectives on the CBIR task, by using both convolutional and transformer architectures, such as residual networks (ResNets) [14] and shifted windows (Swin) transformers [21]. Compared to existing loss functions, our empirical results show that the proposed objective is generating higher and more consistent performance levels across the considered evaluation scenarios. Furthermore, we conduct an ablation study to demonstrate the influence of each loss component on the overall performance.

In summary, our contribution is twofold:

- We introduce a novel repeller-attractor objective function that directly optimizes for the L_2 metric, alleviating the need to generate pairs via hard example mining or alternative mining strategies.
- We conduct comprehensive experiments to compare the proposed loss function with popular loss choices on multiple datasets.

2 Related Work

As related work, we refer to studies introducing new loss functions, which are related to our first contribution, and new content-based image retrieval methods, which are connected to our second contribution.

Loss Functions. For retrieval systems based on neural networks, the choice of the objective function is the most important factor determining the geometry of the resulting feature space [35]. We hereby discuss related work proposing various loss functions aimed at generating effective embedding spaces.

Metric learning objective functions directly optimize a desired metric and are usually based on pairs or tuples of known data samples [4, 33, 36, 43]. One of the earliest works on metric learning proposed the contrastive loss [12], which was introduced as a method to reduce the dimensionality of the input space, while preserving the separation of feature clusters. The idea behind contrastive loss is to generate an attractor-repeller system that is trained on positive and negative pairs generated from the available data. The repelling can happen only if the distance between a negative pair is smaller than a margin m . In the context of face identification, another successful metric learning approach is triplet loss [32]. It obtains the desired properties of the embedding space by generating triplets of anchor, positive and negative examples. For each triplet, the proposed objective enforces the distance between the anchor and the positive example to be larger than the distance between the anchor and the negative example, by a margin m . Other approaches introduced objectives that directly optimize the AUC [9], recall [26] or AP [30]. The main issues when optimizing with loss functions based on metric learning are the usually slow convergence [33] and the difficulty of generating useful example pairs or tuples [34]. In contrast, our method does not require mining strategies and, as shown in the experiments, it converges much faster than competing losses. The usual example mining strategies are hard, semi-hard, and online negative mining [32, 40]. In hard negative mining, for each anchor image, we need to construct pairs with the farthest positive example and the closest negative example. This adds an extra computational step at the beginning of every training epoch, extending the training time. Similar problems arise in the context of semi-hard negative mining, while the difference consists in the mode in which the negatives are sampled.

Statistical learning objective functions indirectly optimize the learned features of the neural network. Popular objectives are based on some variation of the cross-entropy loss [6, 8, 19, 20, 37, 38] or the cosine loss [1]. By optimizing such

functions, the model is forced to generate features that are close to the direction of the class center. For example, ArcFace [6] reduces the optimization space to an n -dimensional hypersphere by normalizing both the embedding generated by the encoder, and the corresponding class weight from the classification head, using an Euclidean norm.

Hybrid objective functions promise to obtain better embeddings by minimizing a statistical objective function in conjunction with a metric learning objective [15, 22]. For example, Center Loss [39] minimizes the intra-class distances of the learned features by using cross-entropy in conjunction with an attractor for each sample to its corresponding class center. Another approach [44] similar to Center Loss [39] is based on predefined evenly distributed class centers. A more complex approach [3] is to combine the standard cross-entropy with a cosine classifier and a mean squared error regression term to jointly enhance global and local features.

Both contrastive and triplet loss objectives suffer from the need of employing pair mining strategies, but in our case, mining strategies are not required. The positive pairs are built online for each batch, between each input feature vector and the dedicated class anchor, the number of positive pairs thus being equal to the number of examples. The negative pairs are constructed only between the class centers, thus alleviating the need of searching for a good negative mining strategy, while also significantly reducing the number of negative pairs. To the best of our knowledge, there are no alternative objective functions for CBIR that use dedicated self-repelling learnable class anchors acting as attraction poles for feature vectors belonging to the respective classes.

Content-Based Image Retrieval Methods. CBIR systems are aimed at finding similar images with a given query image, matching the images based on the similarity of their scenes or the contained objects. Images are encoded using a descriptor (or encoder), and a system is used to sort a database of encoded images based on a similarity measure between queries and images. In the context of content-based image retrieval, there are two types of image descriptors. On the one hand, there are general descriptors [29], where a whole image is represented as a feature vector. On the other hand, there are local descriptors [41] where portions of an image are represented as feature vectors. Hybrid descriptors [3] are also used to combine both global and local features. To improve the quality of the results retrieved by learned global descriptors, an additional verification step is often employed. This step is meant to re-rank the retrieved images by a precise evaluation of each candidate image [28]. The re-ranking step is usually performed with the help of an additional system, and in some cases, it can be directly integrated with the general descriptor [18]. In the CBIR task, one can search for visually similar images as a whole, or search for images that contain similar regions [27] of a query image. In this work, we focus on building global descriptors that match whole images. Further approaches based on metric learning, statistical learning, or hand-engineered features are discussed in the recent survey of Dubey et al. [7]. Different from other CBIR methods, we propose a novel two-stage retrieval system that leverages the use of the class anchors learned through the proposed loss function to make the retrieval process more efficient and effective.

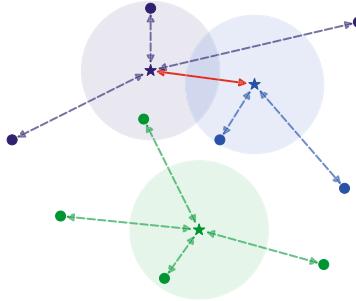


Fig. 1. An example showing the behavior of the attractor-repeller loss components for three classes. The stars represent the class anchors C . Faded circles around class anchors represent the sphere of radius m around each anchor. Solid circles represent feature vectors generated by the encoder f_θ . Dashed arrows between feature vectors and class anchors represent the attraction of the force generated by the attractor \mathcal{L}_A . The solid red arrow between class anchors represents the repelling force generated by the repeller \mathcal{L}_R . Best viewed in color. (Color figure online)

3 Method

Overview. Our objective function consists of three components, each playing a different role in obtaining a discriminative embedding space. All three loss components are formulated with respect to a set of learnable class anchors (centroids). The first loss component acts as an attraction force between each input embedding and its corresponding class anchor. Its main role is to draw the embeddings representing the same object to the corresponding centroid, thus creating embedding clusters of similar images. Each center can be seen as a magnet with a positive charge and its associated embeddings as magnets with negative charges, thus creating attraction forces between anchors and data samples of the same class. The second loss component acts as a repelling force between class anchors. In this case, the class centers can be seen as magnets with similar charges. If brought together, they will repel each other, and if they lie at a certain distance, the repelling force stops. The last component acts similarly to the second one, with the difference that an additional magnet is introduced and fixed at the origin of the embedding space. Its main effect is to push the class centers away from the origin.

Notations. Let $\mathbf{x}_i \in \mathbb{R}^{h \times w \times c}$ be an input image and $y_i \in \mathbb{N}$ its associated class label, $\forall i \in \{1, 2, \dots, l\}$. We aim to optimize a neural encoder f_θ which is parameterized by the learnable weights θ to produce a discriminative embedding space. Let $\mathbf{e}_i \in \mathbb{R}^n$ be the n -dimensional embedding vector of the input image \mathbf{x}_i generated by f_θ , i.e. $\mathbf{e}_i = f_\theta(\mathbf{x}_i)$. In order to employ our novel loss function, we need to introduce a set of learnable class anchors $C = \{\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_t\}$, where $\mathbf{c}_j \in \mathbb{R}^n$ resides in the embedding space of f_θ , and t is the total number of classes.

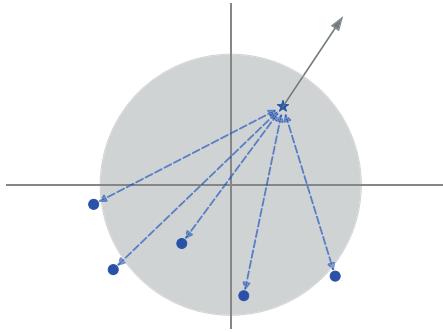


Fig. 2. Contribution of the minimum norm loss \mathcal{L}_N imposed on the class anchors. The blue star represents a class anchor. Solid circles represent embedding vectors generated by the encoder f_θ . Dashed arrows represent the attraction force generated by the attractor \mathcal{L}_A . The solid gray line represents the direction in which the anchor is pushed away from the origin due to the component \mathcal{L}_N . Best viewed in color.

Loss Components. With the above notations, we can now formally define our first component, the attractor loss \mathcal{L}_A , as follows:

$$\mathcal{L}_A(\mathbf{x}_i, C) = \frac{1}{2} \|\mathbf{e}_i - \mathbf{c}_{y_i}\|_2^2. \quad (1)$$

The main goal of this component of the objective is to cluster feature vectors as close as possible to their designated class anchor by minimizing the distance between \mathbf{e}_i and the corresponding class anchor \mathbf{c}_{y_i} . Its effect is to enforce low intra-class variance. However, obtaining low intra-class variance is only a part of what we aim to achieve. The first objective has little influence over the inter-class similarity, reducing it only indirectly. Therefore, another objective is required to repel samples from different classes. As such, we introduce the repeller loss \mathcal{L}_R , which is defined as follows:

$$\mathcal{L}_R(C) = \frac{1}{2} \sum_{y, y' \in Y, y \neq y'} \{\max(0, 2 \cdot m - \|\mathbf{c}_y - \mathbf{c}_{y'}\|)\}^2, \quad (2)$$

where y and y' are two distinct labels from the set of ground-truth labels Y , and $m > 0$ is a margin representing the radius of an n -dimensional sphere around each anchor, in which no other anchor should lie. The goal of this component is to push anchors away from each other during training to ensure high inter-class distances. The margin m is used to limit the repelling force to an acceptable margin value. If we do not set a maximum margin, then the repelling force can push the anchors too far apart, and the encoder could struggle to learn features that satisfy the attractor loss defined in Eq. (1).

A toy example of the attractor-repeller mechanism is depicted in Fig. 1. Notice how the optimization based on the attractor-repeller objective tends to pull data samples from the same class together (due to the attractor), and push

samples from different classes away (due to the repeller). However, when the training begins, all data samples start from a location close to the origin of the embedding space, essentially having a strong tendency to pull the class anchors to the origin. To ensure that the anchors do not collapse to the origin (as observed in some of our preliminary experiments), we introduce an additional objective that imposes a minimum norm on the class anchors. The minimum norm loss \mathcal{L}_N is defined as:

$$\mathcal{L}_N(C) = \frac{1}{2} \sum_{y \in Y} \{\max(0, p - \|\mathbf{c}_y\|)\}^2, \quad (3)$$

where p is the minimum norm that each anchor must have. This objective contributes to our full loss function as long as at least one class anchor is within a distance of p from the origin. Figure 2 provides a visual interpretation of the effect induced by the minimum norm loss. Notice how the depicted class anchor is pushed away from the origin (due to the minimum norm loss), while the data samples belonging to the respective class move along with their anchor (due to the attractor loss).

Assembling the three loss components presented above into a single objective leads to the proposed class anchor margin (CAM) loss \mathcal{L}_{CAM} , which is formally defined as follows:

$$\mathcal{L}_{\text{CAM}}(\mathbf{x}, C) = \mathcal{L}_A(\mathbf{x}, C) + \mathcal{L}_R(C) + \mathcal{L}_N(C). \quad (4)$$

Notice that only \mathcal{L}_A is directly influenced by the training examples, while \mathcal{L}_R and \mathcal{L}_N operate only on the class anchors. Hence, negative mining strategies are not required at all.

4 Experiments

In the experiments, we compare our class anchor margin loss with the cross-entropy and the contrastive learning losses on three datasets, considering both convolutional and transformer models.

4.1 Datasets

We perform experiments on three datasets: CIFAR-100 [17], Food-101 [2], and ImageNet-200 [31]. CIFAR-100 contains 50,000 training images and 10,000 test images belonging to 100 classes. Food-101 is composed of 101,000 images from 101 food categories. The official split has 750 training images and 250 test images per category. ImageNet-200 is a subset of ImageNet-1K, which contains 100,000 training images, 25,000 validation images and 25,000 test images from 200 classes.

Table 1. Retrieval performance levels of ResNet-18 (RN-18), ResNet-50 (RN-50), ResNet-101 (RN-101) and Swin-T models on the CIFAR-100 [17], Food-101 [2] and ImageNet-200 [31] datasets, while comparing the cross-entropy (CE) loss, the contrastive learning (CL) loss, and the proposed class anchor margin (CAM) loss. The best score for each architecture and each metric is highlighted in bold.

	Loss	CIFAR-100			Food-101			ImageNet-200		
		mAP	P@20	P@100	mAP	P@20	P@100	mAP	P@20	P@100
RN-18	CE	.249 _{±.003}	.473 _{±.005}	.396 _{±.005}	.234 _{±.002}	.547 _{±.002}	.459 _{±.001}	.130 _{±.001}	.340 _{±.001}	.262 _{±.001}
	CL	.220 _{±.013}	.341 _{±.010}	.303 _{±.011}	.025 _{±.001}	.042 _{±.004}	.038 _{±.003}	.070 _{±.012}	.139 _{±.017}	.117 _{±.016}
	CAM	.622_{±.005}	.560_{±.007}	.553_{±.007}	.751_{±.001}	.676_{±.003}	.669_{±.003}	.508_{±.004}	.425_{±.004}	.418_{±.005}
RN-50	CE	.211 _{±.006}	.454 _{±.007}	.366 _{±.006}	.158 _{±.004}	.471 _{±.005}	.370 _{±.005}	.088 _{±.003}	.292 _{±.006}	.209 _{±.005}
	CL	.164 _{±.016}	.271 _{±.017}	.240 _{±.016}	.019 _{±.000}	.030 _{±.000}	.028 _{±.001}	.005 _{±.000}	.008 _{±.001}	.007 _{±.000}
	CAM	.640_{±.008}	.581_{±.009}	.578_{±.009}	.765_{±.005}	.697_{±.008}	.697_{±.009}	.543_{±.003}	.472_{±.002}	.468_{±.002}
RN-101	CE	.236 _{±.009}	.482 _{±.008}	.397 _{±.009}	.160 _{±.003}	.479 _{±.008}	.376 _{±.007}	.093 _{±.002}	.299 _{±.002}	.216 _{±.002}
	CL	.034 _{±.028}	.069 _{±.051}	.056 _{±.044}	.014 _{±.002}	.018 _{±.004}	.017 _{±.003}	.006 _{±.001}	.007 _{±.002}	.007 _{±.002}
	CAM	.629_{±.006}	.575_{±.008}	.573_{±.008}	.758_{±.007}	.690_{±.007}	.693_{±.006}	.529_{±.005}	.458_{±.006}	.455_{±.006}
Swin-T	CE	.661 _{±.004}	.808_{±.001}	.770 _{±.001}	.617 _{±.006}	.817 _{±.002}	.777 _{±.003}	.560 _{±.006}	.743 _{±.001}	.707 _{±.003}
	CL	.490 _{±.010}	.629 _{±.007}	.584 _{±.008}	.495 _{±.004}	.667 _{±.001}	.633 _{±.002}	.223 _{±.002}	.397 _{±.001}	.338 _{±.001}
	CAM	.808_{±.004}	.794 _{±.003}	.795_{±.004}	.873_{±.001}	.858_{±.001}	.861_{±.001}	.761_{±.002}	.745_{±.003}	.749_{±.004}

4.2 Experimental Setup

As underlying neural architectures, we employ three ResNet [14] model variations (ResNet-18, ResNet-50, ResNet-101) and a Swin transformer [21] model (Swin-T). We rely on the PyTorch [25] library together with Hydra [42] to implement and test the models.

We apply random weight initialization for all models, except for the Swin transformer, which starts from the weights pre-trained on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) dataset [31]. We employ the Adam [16] optimizer to train all models, regardless of their architecture. For the residual neural models, we set the learning rate to 10^{-3} , while for the Swin transformer, we use a learning rate of 10^{-4} . The residual nets are trained from scratch for 100 epochs, while the Swin transformer is fine-tuned for 30 epochs. For the lighter models (ResNet-18 and ResNet-50), we use a mini-batch size of 512. Due to memory constraints, the mini-batch size is set to 64 for the Swin transformer, and 128 for ResNet-101. Residual models are trained with a linear learning rate decay with a factor of 0.5. We use a patience of 6 epochs in all experiments. Fine-tuning the Swin transformer does not employ a learning rate decay. Input images are normalized to have all pixel values in the range of $[0, 1]$ by dividing the values by 255. The inputs of the Swin transformer are standardized with the image statistics from ILSVRC [31].

We use several data augmentation techniques such as random crop with a padding of 4 pixels, random horizontal flip, color jitter, and random affine transformations (rotations, translations). Moreover, for the Swin transformer, we add the augmentations described in [23].

For the models optimized either with the cross-entropy loss or our class anchor margin loss, the target metric is the validation accuracy. When using

Table 2. Classification and retrieval results while ablating the proposed class anchor initialization and various components of our CAM loss. Results are shown for a ResNet-18 model on CIFAR-100.

\mathcal{L}_R	\mathcal{L}_N	Anchors init	Accuracy	mAP
\times	\times	random	17.46%	0.110
		base vectors	39.24%	0.378
\times	\checkmark	random	17.88%	0.112
		base vectors	40.90%	0.386
\checkmark	\times	random	58.34%	0.558
		base vectors	59.55%	0.570
\checkmark	\checkmark	random	58.75%	0.539
		base vectors	61.94%	0.622

our loss, we set the parameter m for the component \mathcal{L}_R to 2, and the parameter p for the component \mathcal{L}_N to 1, across all datasets and models. Since the models based on the contrastive loss optimize in the feature space, we have used the 1-nearest neighbors (1-NN) accuracy, which is computed for the closest feature vector retrieved from the gallery.

As evaluation measures for the retrieval experiments, we report the mean Average Precision (mAP) and the precision@ k on the test data, where $k \in \{20, 100\}$ is the retrieval rank. For the classification experiments, we use the classification accuracy on the test set. We run each experiment in 5 trials and report the average score and the standard deviation.

4.3 Results

Main Results. We first evaluate the performance of the ResNet-18, ResNet-50, ResNet-101 and Swin-T models on the CBIR task, while using three alternative losses, including our own, to optimize the respective models. The results obtained on the CIFAR-100 [17], Food-101 [2], and ImageNet-200 [31] datasets are reported in Table 1. First, we observe that our loss function produces better results on the majority of datasets and models. Furthermore, as the rank k increases from 20 to 100, we notice that our loss produces more consistent results, essentially maintaining the performance level as k increases.

Ablation Study. In Table 2, we demonstrate the influence of each additional loss component on the overall performance of ResNet-18 on the CIFAR-100 dataset, by ablating the respective components from the proposed objective. We emphasize that the component \mathcal{L}_A is mandatory to make our objective work properly. Hence, we only ablate the other loss components, namely \mathcal{L}_R and \mathcal{L}_N .

In addition, we investigate different class center initialization heuristics. As such, we conduct experiments to compare the random initialization of class centers and the base vector initialization. The latter strategy is based on initializing

class anchors as scaled base vectors, such that each class center has no intersection with any other class center in the n -dimensional sphere of radius m , where n is the size of the embedding space.

As observed in Table 2, the class center initialization has a major impact on the overall performance. For each conducted experiment, we notice a significant performance gain for the base vector initialization strategy. Regarding the loss components, we observe that removing both \mathcal{L}_R and \mathcal{L}_N from the objective leads to very low performance. Adding only the component \mathcal{L}_N influences only the overall accuracy, but the mAP is still low, since \mathcal{L}_N can only impact each anchor's position with respect to the origin. Adding only the component \mathcal{L}_R greatly improves the performance, proving that \mathcal{L}_R is crucial for learning the desired task. Using both \mathcal{L}_R and \mathcal{L}_N further improves the results, justifying the proposed design.

5 Conclusion

In this paper, we proposed a novel loss function based on class anchors to optimize convolutional networks and transformers for object retrieval in images. We conducted comprehensive experiments using four neural models on four image datasets, demonstrating the benefits of our loss function against conventional losses based on statistical learning and contrastive learning. We also performed an ablation study to showcase the influence of the proposed components, empirically justifying our design choices.

In future work, we aim to extend the applicability of our approach to other data types, beyond images. We also aim to explore new tasks and find out when our loss is likely to outperform the commonly used cross-entropy.

References

1. Barz, B., Denzler, J.: Deep learning on small datasets without pre-training using cosine loss. In: Proceedings of WACV, pp. 1360–1369. IEEE (2020)
2. Bossard, L., Guillaumin, M., Van Gool, L.: Food-101 – mining discriminative components with random forests. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8694, pp. 446–461. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10599-4_29
3. Cao, B., Araujo, A., Sim, J.: Unifying deep local and global features for image search. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12365, pp. 726–743. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58565-5_43
4. Chen, W., Chen, X., Zhang, J., Huang, K.: Beyond triplet loss: a deep quadruplet network for person re-identification. In: Proceedings of CVPR, pp. 1320–1329. IEEE (2017)
5. Cortes, C., Vapnik, V.: Support-vector networks. *Mach. Learn.* **20**(3), 273–297 (1995)

6. Deng, J., Guo, J., Xue, N., Zafeiriou, S.: ArcFace: additive angular margin loss for deep face recognition. In: Proceedings of CVPR, pp. 4685–4694. IEEE (2019)
7. Dubey, S.R.: A decade survey of content based image retrieval using deep learning. *IEEE Trans. Circuits Syst. Video Technol.* **32**(5), 2687–2704 (2021)
8. Elezi, I., et al.: the group loss++: a deeper look into group loss for deep metric learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**(2), 2505–2518 (2022)
9. Gajić, B., Amato, A., Baldrich, R., van de Weijer, J., Gatta, C.: Area under the ROC curve maximization for metric learning. In: Proceedings of CVPR, pp. 2807–2816. IEEE (2022)
10. Georgescu, M.I., Duță, G.E., Ionescu, R.T.: Teacher-student training and triplet loss to reduce the effect of drastic face occlusion: application to emotion recognition, gender identification and age estimation. *Mach. Vis. Appl.* **33**(1), 12 (2022)
11. Georgescu, M.I., Ionescu, R.T.: Teacher-student training and triplet loss for facial expression recognition under occlusion. In: Proceedings of ICPR, pp. 2288–2295. IEEE (2021)
12. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: Proceedings of CVPR, vol. 2, pp. 1735–1742. IEEE (2006)
13. Harwood, B., Kumar, V.B., Carneiro, G., Reid, I., Drummond, T.: Smart mining for deep metric learning. In: Proceedings of ICCV, pp. 2821–2829. IEEE (2017)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of CVPR, pp. 770–778. IEEE (2016)
15. Khosla, P., et al.: Supervised contrastive learning. In: Proceedings of NeurIPS, vol. 33, pp. 18661–18673. Curran Associates, Inc. (2020)
16. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: Proceedings of ICLR (2015)
17. Krizhevsky, A.: Learning multiple layers of features from tiny images. Technical report, University of Toronto (2009)
18. Lee, S., Seong, H., Lee, S., Kim, E.: Correlation verification for image retrieval. In: Proceedings of CVPR, pp. 5374–5384. IEEE (2022)
19. Liu, W., Wen, Y., Yu, Z., Li, M., Raj, B., Song, L.: SphereFace: deep hypersphere embedding for face recognition. In: Proceedings of CVPR, pp. 6738–6746. IEEE, Los Alamitos, CA, USA (2017)
20. Liu, W., Wen, Y., Yu, Z., Yang, M.: Large-margin softmax loss for convolutional neural networks. In: Proceedings of ICML, pp. 507–516. JMLR.org (2016)
21. Liu, Z., et al.: Swin Transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of ICCV, pp. 10012–10022. IEEE (2021)
22. Min, W., Mei, S., Li, Z., Jiang, S.: A two-stage triplet network training framework for image retrieval. *IEEE Trans. Multimedia* **22**(12), 3128–3138 (2020)
23. Muller, S.G., Hutter, F.: TrivialAugment: tuning-free yet state-of-the-art data augmentation. In: Proceedings of ICCV, pp. 754–762. IEEE, Los Alamitos, CA, USA (2021)
24. Murphy, K.P.: Machine Learning: A Probabilistic Perspective. The MIT Press, Cambridge (2012)
25. Paszke, A., et al.: PyTorch: an imperative style, high-performance deep learning library. In: Proceedings of NeurIPS, pp. 8024–8035. Curran Associates, Inc. (2019)
26. Patel, Y., Tolias, G., Matas, J.: Recall@k surrogate loss with large batches and similarity mixup. In: Proceedings of CVPR, pp. 7502–7511. IEEE (2022)
27. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: Proceedings of CVPR, pp. 1–8. IEEE (2007)

28. Polley, S., Mondal, S., Mannam, V.S., Kumar, K., Patra, S., Nürnberg, A.: X-vision: explainable image retrieval by re-ranking in semantic space. In: Proceedings of CIKM, pp. 4955–4959. Association for Computing Machinery, New York, NY, USA (2022)
29. Radenović, F., Tolias, G., Chum, O.: Fine-tuning CNN image retrieval with no human annotation. *IEEE Trans. Pattern Anal. Mach. Intell.* **41**(7), 1655–1668 (2019)
30. Revaud, J., Almazán, J., Rezende, R.S., Souza, C.R.d.: Learning with average precision: training image retrieval with a listwise loss. In: Proceedings of ICCV, pp. 5107–5116. IEEE (2019)
31. Russakovsky, O., et al.: ImageNet large scale visual recognition challenge. *Int. J. Comput. Vision* **115**, 211–252 (2015)
32. Schroff, F., Kalenichenko, D., Philbin, J.: FaceNet: a unified embedding for face recognition and clustering. In: Proceedings of CVPR, pp. 815–823. IEEE (2015)
33. Sohn, K.: Improved deep metric learning with multi-class N-pair loss objective. In: Proceedings of NIPS, vol. 29. Curran Associates, Inc. (2016)
34. Suh, Y., Han, B., Kim, W., Lee, K.M.: Stochastic class-based hard example mining for deep metric learning. In: Proceedings of CVPR, pp. 7244–7252. IEEE (2019)
35. Tang, Y., Bai, W., Li, G., Liu, X., Zhang, Y.: CROLoss: towards a customizable loss for retrieval models in recommender systems. In: Proceedings of CIKM, pp. 1916–1924. Association for Computing Machinery, New York, NY, USA (2022)
36. Balntas, V., Riba, E., Ponsa, D., Mikolajczyk, K.: Learning local feature descriptors with triplets and shallow convolutional neural networks. In: Proceedings of BMVC, pp. 119.1–119.11. BMVA Press (2016)
37. Wang, F., Cheng, J., Liu, W., Liu, H.: Additive margin softmax for face verification. *IEEE Sig. Process. Lett.* **25**(7), 926–930 (2018)
38. Wang, H., et al.: CosFace: large margin cosine loss for deep face recognition. In: Proceedings of CVPR, pp. 5265–5274. IEEE (2018)
39. Wen, Y., Zhang, K., Li, Z., Qiao, Yu.: A discriminative feature learning approach for deep face recognition. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9911, pp. 499–515. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46478-7_31
40. Wu, C.Y., Manmatha, R., Smola, A.J., Krähenbühl, P.: Sampling matters in deep embedding learning. In: Proceedings of ICCV, pp. 2859–2867. IEEE (2017)
41. Wu, H., Wang, M., Zhou, W., Li, H.: Learning deep local features with multiple dynamic attentions for large-scale image retrieval. In: Proceedings of ICCV, pp. 11416–11425. IEEE (2021)
42. Yadan, O.: Hydra - a framework for elegantly configuring complex applications. Github (2019). <https://github.com/facebookresearch/hydra>
43. Yu, B., Tao, D.: Deep metric learning with triplet margin loss. In: Proceedings of ICCV, pp. 6489–6498. IEEE (2019)
44. Zhu, Q., Zhang, P., Wang, Z., Ye, X.: A new loss function for CNN classifier based on predefined evenly-distributed class centroids. *IEEE Access* **8**, 10888–10895 (2019)



Personalized EDM Subject Generation via Co-factored User-Subject Embedding

Yu-Hsiu Chen, Zhi Rui Tam[✉], and Hong-Han Shuai^(✉)

National Yang Ming Chiao Tung University, Hsinchu, Taiwan
{yhchen.cm06g,ray.eed08g}@nctu.edu.tw, hhshuai@nycu.edu.tw

Abstract. This paper introduces the Co-Factored User-Subject Embedding based Personalized EDM Subject Generation Framework (COUPES), a model for creating personalized Electronic Direct Mail (EDM) subjects. COUPES adapts to individual content and style preferences using a dual-encoder structure to process product descriptions and template features. It employs a soft template-based selective encoder and matrix co-factorization for nuanced user embeddings. Experiments show that COUPES excels in generating engaging, personalized subjects and reconstructing recommendation ratings, proving its effectiveness in personalized marketing and recommendation systems.

Keywords: Summarization · Recommendation · Tensor co-factorization

1 Introduction

In the competitive landscape of e-commerce advertising, Email Direct Marketing (EDM) has proven its mettle, contributing significantly to the conversion of prospects into paying customers and enhancing open/conversion rates. With an estimated revenue exceeding \$10 billion¹, personalizing EDM subject lines is a critical lever for capturing user attention, with personalized subjects boosting open rates by an impressive 37% over non-personalized ones².

Despite the effectiveness of personalized subjects, the prevailing practice involves manual creation by promotion specialists, a method that does not scale well with the voluminous and dynamic nature of e-commerce. With the advance of deep learning in natural language processing, studies have spurred research into automating the generation of engaging headlines. For instance, some researchers propose to inject specific styles, such as humor and romance, into news headlines or generate interrogative ones to attract readers' attention [5, 10, 22, 29, 30], disregarding the diverse style preferences of users.

Addressing this, Personalized Headline Generation (PHG) has surfaced as a novel research direction, aiming to tailor content to user-specific information inferred from click history [1, 6, 21]. Yet, this approach presents a formidable

¹ <https://www.statista.com/statistics/812060/email-marketing-revenue-worldwide/>.

² <http://bit.ly/2kc7OD7>.

challenge: the generation of personalized headlines without explicit annotations for personal text styles. Current methods often default to predefined rules or human-curated word lists, which fail to accommodate the breadth of individual user styles, particularly for new users with sparse interaction records [13, 19].

Our work seeks to transcend these limitations by introducing a sophisticated approach that does not rely on predefined style classifications or representative word lists. We propose the Co-Factored Embeddings based Personalized EDM Subject Generation Framework, a model designed to navigate the complexities of personalized subject generation. This framework embodies two encoders to process description and template features, a bi-directional selective encoder for optimal information extraction, and a matrix factorization network for crafting user embeddings that reflect individual style preferences. These elements converge in an RNN decoder that generates EDM subjects aligned with user preferences, without the constraints of known styles or fixed word lists.

Furthermore, our approach harnesses user embeddings from browsing histories to enable collaborative filtering, recommending items and generating titles for products. We substantiate our model’s efficacy with a newly compiled dataset containing 17,617 products with descriptions and 278,876 ratings from 21,379 users, demonstrating state-of-the-art performance in personalized subject generation. This represents not only a leap in the automation of EDM but also a new frontier in personalized digital marketing strategies.

The main contributions of our work are summarized as follows:

- We introduce the Co-Factored Embeddings based Personalized EDM Subject Generation Framework, a Seq2Seq-based model tailored for generating personalized EDM subjects. This represents the first endeavor, to our knowledge, to apply a data-driven approach specifically for this purpose, marking a significant departure from manual curation methods.
- Our model innovatively applies matrix factorization to infer users’ preferences from click-through data, effectively utilizing this information not only for subject generation but also to enable a collaborative filtering mechanism within a recommendation system.
- Extensive experiments demonstrate that our framework significantly outperforms benchmarks in generating relevant and personalized subjects as measured by both qualitative and quantitative metrics. Furthermore, our approach shows adeptness in recommending items that align closely with users’ preferences, showcasing personalization and recommendation.

2 Related Work

Text summarization techniques, fundamental in natural language processing, are widely-used in a variety of applications [2, 8, 28]. The methods are broadly classified into extractive and abstractive methods. Extractive methods focus on selecting and combining sentences from source articles into cohesive paragraphs [9, 18]. In contrast, abstractive methods, such as those developed by [4], rewrite summaries, often producing content not explicitly present in the source text. The

advent of sequence-to-sequence (Seq2Seq) frameworks [7, 17] has marked a significant advancement in abstractive summarization. Techniques like the pointer network [24] and the coverage mechanism [20] enhance detail accuracy and minimize word repetition. Meanwhile, template-based summarization utilizes predefined templates for abstractive summarization. For instance, soft template-based models [3, 26] offer a more flexible approach by selecting soft templates from specific training articles. However, the integration of user preferences in subject generation remains an underexplored area. Studies like [13, 14] address this by generating reviews based on user preferences, but often require predefined word lists and fail to capture the diversity of user styles.

Recent research in Personalized Headline Generation (PHG) attempts to inject specific styles into headlines, but often overlooks the diverse and individualized style preferences of users [5, 10, 22, 29, 30]. To mitigate the gap, several works advocate for tailoring content to user-specific information inferred from click history [1, 6, 21]. For instance, [1] constructs a new dataset from Microsoft News user logs, which contains user-specific titles based on individual reading interests and news content. They further address personalized news headline generation by proposing a framework that learns user preferences from behavioral data to personalize text generation. Despite advancements, creating personalized headlines without explicit annotations for personal text styles remains challenging.

3 Proposed Model

Problem Formulation. Suppose we have a corpus D with m article-template-subject triples a - t - s , and each triple contains an article a , a template t and a subject s . Article a consists of l words as $\{a_1, a_2, \dots, a_l\}$, where $a_i \in D$. Template t and subject s consist of $p \leq l, q \leq l$ words as $\{t_1, t_2, \dots, t_p\}$ and $\{s_1, s_2, \dots, s_q\}$ individually, where $t_i, s_i \in D$. In addition, the user preferences are presented by an n -by- m binary matrix \mathbf{R} . $\mathbf{R}_{ij} = 1$ if user i has clicked into the item j . The personalized subject generation task intends to: 1) retrieve template t given article a , and then further summarize a subject \hat{s} given a and t by attending to user u 's preference feature vector, 2) predict the rest of ratings in $\hat{\mathbf{R}}$ given a part of the ratings in \mathbf{R} .

To generate personalized EDM subjects, there are three challenges required to be addressed. First, different users prefer different styles of headlines, while it is difficult to define all styles. One possible solution is to select a subject line among users' clicking history and exploit template-based summarization method to generate a personalized subject. However, this approach is limited since only one clicked subject line is used. Second, to summarize the article, articles are encoded into a latent vector. Therefore, it is necessary to design a loss for disentangling the style and content so that the summarization only changes the style and preserves the content. Third, the clicking history of users may be sparse for new users, which leads to the "cold start" problem. Therefore, to overcome these issues, it requires combining the generation and recommendation elegantly.

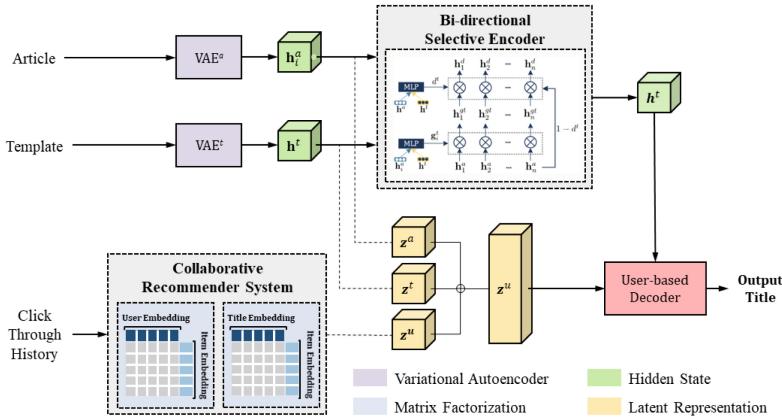


Fig. 1. The user-based encoder-decoder architecture.

Keep the goals in mind, in this paper, we propose a new framework including 5 key modules: **Retrieve and re-rank**, **Variational feature autoencoder**, **Bi-directional selective encoder**, **Collaborative recommender system**, and an **User-based decoder**. The overview model architecture is illustrated as shown in Fig. 1. Specifically, **Retrieve and re-rank** aims to return a few candidate templates from the training corpus and re-rank by semantic relationship to identify the best one. To jointly address the first and second issues, **Variational feature autoencoder** encodes the description to a Gaussian distribution. On the other hand, it encodes template's feature latent to a Gumbel distribution to preserve the categorical nature of templates. Afterward, **Bi-directional selective encoder** mutually selects important information from the source article and template hidden state to generate an article representation of summarization. With the user-item click through history as implicit feedback, **Collaborative recommender system** learns the users' preference representation to address the third issue. Finally, with the generated article, template, user feature latent, and the hidden states fused by **Bi-directional selective encoder**, the **User-based decoder** use an ordinary RNN to generate a personalized subject.

3.1 Retrieve and Re-rank

The module selects the closest candidate templates from the training corpus based on word frequency and semantic distance, assuming similar sentences have similar summary patterns. Utilizing Lucene³, as per the default settings in [3], it

³ <https://lucene.apache.org>.

retrieves similar paragraphs and their summaries as candidate templates, choosing the top 30 search results for each paragraph.

To identify the best template, we calculate the embedding space cosine similarity between two titles, using a Word2Vec embedding [16] trained on Wikipedia. The similarity score s_i between a template \mathbf{z}_i^t and a gold subject \mathbf{z}^s is given by:

$$s_i = (\mathbf{z}_i^t)^T \mathbf{z}^s.$$

We select the subject with the highest score as the template, i.e., $\arg \max_i s_i$.

Notably, unlike the typical ROUGE score evaluation, which only captures word-level similarity, we use Word2Vec similarity [25] to better capture semantic meanings and select our desired template.

3.2 Variational Encoder and Bi-directional Selective Encoder

To better capture the latent representation of the article and template, we use Seq2seq Variational Autoencoder (VAE) [11] to derive the encoded representations of articles and templates. For a dataset $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ containing N independent and identically distributed (i.i.d.) samples of a variable \mathbf{x} alongside an unobserved continuous random variable \mathbf{z} , our objective is to generate new data \mathbf{x} from the latent variable \mathbf{z} . These latent variables are assumed to arise from a prior distribution $p_\theta(\mathbf{z})$, while the data \mathbf{x} are assumed to be generated from a likelihood distribution $p_\theta(\mathbf{x}|\mathbf{z})$. Given that our inputs are sequences of words, we adopt a sequential variant of the Variational Autoencoder (VAE), specifically a variational Seq2Seq model. For the j th timestep of the n th data point, with input $x^{(n)}$ and target $y^{(n)}$, we formulate the loss function to maximize the variational lower bound, incorporating both the reconstruction loss and a regularization term derived from KL divergence, i.e.,

$$\mathcal{L}_j^{(n)}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \underbrace{\mathbb{E}_{z \sim q_\phi(z|x^{(n)})} \left[\log p_\theta(y^{(n)} | z) \right]}_{\text{Reconstruction loss}} - \underbrace{D_{\text{KL}}(q_\phi(z|x^{(n)}) \| p(z))}_{\text{Regularization term}},$$

where q_ϕ and p_θ correspond to the encoder and decoder networks with weights ϕ and θ respectively. $q_\phi(\mathbf{z}|\mathbf{x}_i)$ is the variational distribution to approximate the true posterior $p_\theta(\mathbf{z}|\mathbf{x}_i)$. Here, BiLSTM cells are used within both encoder and decoder of the VAEs, allowing us to obtain the latent encodings \mathbf{z}^a and \mathbf{z}^t for articles and templates, respectively.

Moreover, our work utilizes a Bi-directional selective encoder for personalized summarization, merging template patterns with article sentences. We introduce a Template-to-Article gate and an Article-to-Template gate to filter and weigh the article's BiLSTM hidden states \mathbf{h}_i^a using the template representation \mathbf{h}^t , producing a gated vector \mathbf{g}_i^t :

$$\mathbf{g}_i^t = \sigma(\mathbf{W}_{ah}\mathbf{h}_i^a + \mathbf{W}_{th}\mathbf{h}^t + \mathbf{b}_a),$$

$$\mathbf{h}_i^{gt} = \mathbf{h}_i^a \odot \mathbf{g}_i^t,$$

and a confidence score d^t to compute the final representation \mathbf{h}_i^d :

$$\mathbf{h}_i^d = d^t \mathbf{h}_i^{gt} + (1 - d^t) \mathbf{h}_i^a.$$

3.3 User-Subject Co-factor System

To capture user preferences for personalization, we employ a collaborative recommender system for generating user embeddings. While a direct approach might select subject lines from users' click histories or use template-based summarization for personalization as suggested in [6], it may not effectively capture the preferences of users with sparse click histories. We incorporate two matrix factorization (MF) modules within a co-factor model. The first MF module recommends items based on a user's click-through history, treating it as implicit feedback. The second MF module recommends titles to products, ensuring a product's relevance to its subject line.

For training with implicit feedback, we utilize negative sampling, randomly selecting negative instances to enhance efficiency. Given M users and N items, the user-item interaction matrix $\mathbf{R} \in \mathbb{R}^{M \times N}$, derived from implicit feedback, is defined as:

$$R_{ui} = \begin{cases} 1, & \text{if interaction (user } u, \text{ item } i\text{) is observed;} \\ 0, & \text{otherwise.} \end{cases}$$

Here, $R_{ui} = 1$ indicates an interaction between user u and item i , while $R_{ui} = 0$ does not imply a negative preference but possibly unseen items. Learning from such data poses challenges due to the partial insight it provides into user preferences, with unobserved entries potentially representing missing data and observed ones indicating interest.

Matrix Factorization (MF) maps both users and items into a joint latent feature space of d dimension such that interactions are modeled as inner products in that space. Let \mathbf{p}_u and \mathbf{q}_i be the factor vector for user u and item i , respectively, matrix factorization method estimates an interaction r_{ui} as the inner product of \mathbf{p}_u and \mathbf{q}_i :

$$\hat{R}_{ui} = \mathbf{p}_u^T \mathbf{q}_i = \sum_{k=1}^K p_{uk} q_{ik},$$

where K denotes the dimension of the latent space. The loss function of the first matrix factorization model can be expressed as follows:

$$\mathcal{L}_{mf} = \sum_{(u,i) \in \mathcal{R} \cup \mathcal{R}^-} (r_{ui} - \mathbf{p}_u^T \mathbf{q}_i)^2 + \lambda_p \sum_u \|\mathbf{p}_u\|^2 + \lambda_q \sum_p \|\mathbf{q}_i\|^2,$$

where \mathcal{R} is the set of observed click-through interactions, and \mathcal{R}^- is the set of negative instances, which is sampled from unobserved interactions. The last two terms are l_2 regularization terms.

Similarly, we use the same method to construct the loss function of the second MF recommending preferable titles to a certain item. Let \mathbf{p}_t and \mathbf{q}_i denote the

latent vector for title t and item i , respectively; MF estimates an interaction r_{ui} as the inner product of \mathbf{p}_t and \mathbf{q}_i , i.e., $\hat{y}_{ti} = \mathbf{p}_t^T \mathbf{q}_i = \sum_{k=1}^K p_{tk} q_{ik}$. The loss function of title matrix factorization then becomes:

$$\mathcal{L}_{tmf} = \sum_{(u,i) \in \mathcal{R} \cup \mathcal{R}-} (r_{ti} - \mathbf{p}_t^T \mathbf{q}_i)^2.$$

Subsequently, we obtain the user preference latent representation \mathbf{p}_u and concatenate it with the template latent \mathbf{z}^t and article latent \mathbf{z}^a to form a comprehensive personalized representation \mathbf{z}^u , encompassing article, template, and user features.

3.4 User-Based Decoder

Following the encoder’s selection of key information, a decoder RNN generates the subject line, initiated with the final personalized representation \mathbf{z}^u as its starting state. At each time step t , the decoder updates its state based on the previous word and state:

$$\mathbf{h}_t^c = \text{RNN}(\mathbf{w}^t - 1, \mathbf{h}^{ct-1}).$$

Using concatenate attention, the context vector \mathbf{c}^t is computed, and combined with the decoder’s state into \mathbf{h}_t^o , which is then used to predict the next word in the sequence. The model is trained by minimizing the negative log-likelihood of the generated sequence compared to the gold standard summary:

$$\mathcal{L}_s = -\frac{1}{M} \sum_{i=1}^M \sum_{j=1}^L \log p(\mathbf{w}_j^{*(i)} | \mathbf{w}_{j-1}^{(i)}, \mathbf{x}^{a(i)}, \mathbf{x}^{t(i)}),$$

where M is the number of sequences, and \mathbf{x}^a and \mathbf{x}^t represent the article and template.

The total loss of the proposed model is shown as follows:

$$\mathcal{L} = \mathcal{L}_{cycle}^a + \mathcal{L}_{cycle}^t + \mathcal{L}_{mf} + \mathcal{L}_{tmf} + \mathcal{L}_s.$$

The first two terms \mathcal{L}_{cycle}^a and \mathcal{L}_{cycle}^t denote the loss of Seq2Seq VAE, \mathcal{L}_{mf} and \mathcal{L}_{tmf} is the matrix factorization loss, and \mathcal{L}_s is the summarization loss.

4 Experimental Results

Dataset. Due to the lack of public dataset for personalized EDM, we collect a new one from the well-known travel experience e-commerce platform in Asia that sales tour packages. The raw data contains the paired tuple (article, subject) of tour package DMs, together with the ID list of users who click the subject. Statistics are summarized in Table 1. Specifically, there are 17617 products associated with the EDM subject and a short paragraph of the product description.

Table 1. Dataset statistics.

# products	17617	# title	17617
# users	21379	# word/title	12.8
# item	5365	#word/article	155.2
# item/user	13.04	# vocabulary	71625

Each description contains 885 Traditional Chinese characters on average and is tokenized by CKIP [23].

For the user clicking histories, we collect the data from December 2018 to July 2019, which contains 26,662,557 user-item interaction with 1,271,297 users and 9,702 items. We observe that users interact with items sequentially between a small period. This is usually the situation that they are first attracted by the subject of a tour package and continue to browse related packages, which may have the same tour destination with the first one. To approximately collect items whose subjects are most attractive to the user, we simply take the first item viewed by the user for each viewing sequence as the ground truth. After filtering out users with less than 10 items viewed and items without descriptions, there are 278,876 records with 21,379 users and 5,365 items. For summarization task, we randomly split the dataset into 14095 products for training, 1761 for testing and 1761 for validation. The input vocabularies are collected from the training data, which have 71625 words in total. For recommendation task, we randomly select 80% of each user’s records for training, and the remaining data are for testing.

Evaluation Metrics. For relevance between generated and reference sentences, we adopt ROUGE score, which is a commonly-used metric for text summarization. We report the recall, precision and F-measure, where F-measure is the geometric mean of the precision and recall of ROUGE-1, ROUGE-2, and ROUGE-L. Moreover, the performance of the recommender system is evaluated by Recall@K and NDCG@K. Recall@K indicates the percentage of the recommended items among items relevant to the user. NDCG@K is short for Normalized Discounted Cumulative Gain at K, which takes the position of the recommended items’ order into account.

Baselines. Here, we compare our model with several methods which are popular in text summarization as follows:

- *Lead-1* is an extractive approach which selects the first sentence in review as a summary.
- *S2S+Att* is a sequence-to-sequence model with attention implemented by the OpenNMT [12].
- *BiSET* [26] adopts a selective network to bidirectionally select important information from template and article.
- *TemPEST* [6] uses selective network to bidirectionally select important information from template and article, and also adopts users’ click-through titles

Table 2. The recommendation performance comparison of all methods in terms of $Precision@k$, $Recall@k$, and $NDCG@k$. The best performance is marked as **bold**.

Models	Precision@5	Precision@10	Recall@5	Recall@10	NDCG@5	NDCG@10
CDAE	0.021	0.018	0.061	0.036	0.022	0.020
GATE	0.058	0.047	0.110	0.161	0.068	0.053
TemPEST	0.069	0.054	0.119	0.189	0.075	0.063
Ours	0.253	0.173	0.125	0.166	0.306	0.231

to perform personalization, which is the state-of-the-art personalize summarization model.

- *COUPES* is the proposed model. We use two-layer BiLSTM for both TSE and USE network, and the hidden state size of 500. The learning rate and dropout rate are set to 0.001 and 0.3 respectively.

For the recommendation baselines, since the goal of CREPES is to improve the clicking rate of users, we evaluate the recommender system with CREPES and the following baselines.

- *CDAE* [27] is short for collaborative denoising autoencoder, which uses denoising autoencoder to learn latent representation from user-item feedback.
- *GATE* is the gated attentive autoencoder [15] with source articles input.
- *TemPEST* [6] uses selective network to bidirectionally select important information from template and article, and also adopts users’ click-through titles to perform personalization, which is the state-of-the-art personalize summarization model.
- *COUPES* is the proposed model.

4.1 Quantitative Results

Recommendation Results. Since the goal of our model is to improve the clicking rate of users, we evaluate the recommender system with our model and the above baselines. Table 2 compares our model with baselines in terms of $Precision@k$, $Recall@k$, and $NDCG@k$. our model outperforms TemPEST, which indicates that the user-specific article representation generated by our model better captures the feature of the item description than the fused embeddings in TemPEST. Moreover, in terms of $Precision@k$ and $NDCG@k$, our model outperforms other baseline models a lot, which demonstrates that the recommender mechanism truly improves the recommendation results.

Summarization Results. Table 3 presents the results of ROUGE scores, which manifests that our model performs comparable results in terms of ROUGE-2 score and outperforms in ROUGE-L.

For ROUGE score, ROUGE-L scores outperform other baselines in this task, while the ROUGE-2 scores performs comparable results. This indicates that the generated sentence contains important words in the reference subject without

Table 3. ROUGE scores of all methods on the test set. R in the table denotes ROUGE, R is the Recall, P is the Precision. The best performance is marked as **bold**.

Models	R-1 F1(R/P)	R-2 F1(R/P)	R-L F1(R/P)
Lead-1	48.05 (47.91/49.13)	4.13 (3.93/4.68)	6.80 (6.67/7.89)
S2S+attn	56.91 (56.79/57.47)	3.92 (3.96/4.00)	7.91 (7.80/8.47)
BiSET	55.54 (55.61/56.01)	4.70 (4.81/4.72)	8.70 (8.78/9.15)
TemPEST	65.47 (65.34/66.32)	3.44 (3.43/3.86)	8.33 (8.21/9.17)
Ours	46.38 (46.26/47.12)	4.70 (4.72/4.85)	9.24 (9.13/9.96)

directly copy that since ROUGE-1 and ROUGE-2 is effected by the copying of the real answer. Moreover, since the ROUGE-1 score can directly reflects the performance of copying the original ground truth. The ROUGE-1 score of the purposed model is comparatively low in this task. Note that all ROUGE-1 scores are pretty high since subjects contain a lot of proper nouns like place names to describe the products. Therefore, the extractive method *Lead-1* performs relatively well comparing to other summarization datasets.

Since the evaluation metric has only a gold summary with a certain style. Imagine that we simply rewrite subjects from the ground truth answer and make them transfer to another style preserving the informativeness. The ROUGE metric cannot capture this kind of information well and hence the scores will decrease. In short, our model attempts to strike a balance between the personalized styling and the faithfulness.

4.2 Effect of Template

Our research also explores the impact of template latent factors on final results. Experiments indicate that while templates play a minor role in enhancing quantitative measures like ROUGE scores, they are significant in controlling the output’s stylistic aspects. A notable impact is the presence of parentheses in the results. The dataset includes samples with and without parentheses in sentences. Selecting a template with parentheses tends to strongly influence the outcome, leading to results that adhere to this format. As illustrated in Table 4, we present two cases using different templates. In each case, the first template, chosen by our model described earlier, contrasts with a second, randomly selected template from our dataset. The comparison in 4 demonstrates varying writing styles and tones. The first template, marked by an exclamation point and parentheses, tends to produce more exaggerated results. Consequently, the subjects generated mirror this style, featuring exclamatory sentences with parentheses. Conversely, the second template is more informative and lacks parentheses, resulting in smoother, shorter, and more straightforward subjects.

Table 4. Example of sample subject lines generated by our model conditioned on different related templates. The Chinese descriptions on top are the real inputs of this dataset and translated to the English descriptions.

Article	這個武士培訓計劃將幫助您實現童年的夢想。穿上真正的武士服裝，練習如何正確使用武士刀。有各式各樣的服裝選擇。培訓課程結束後將頒發紀念證書。磨 <u>F</u> 你的精神，準備好掌握一些武士技巧！行程特色：接受正式武士戰鬥訓練。穿上正統武士服裝與武器。受訓後可獲得武士培訓證書。使用對象：孩童：06-15歲成人：16-99 <u>F</u> 。使用期限：依照旅客訂購日期 This warrior training program can help you realize your dream in the childhood. Put on real warrior clothes, and learn to use the samurai knife properly. There are various choices of clothes. You will receive a certificate after the training program. Sharpen your spiritual senses and ready to grasp some warrior skills! Highlights: take real warrior combating training course, put on real warrior clothes and gears, get an certificate after training. Children from 6 to 15, adults from 16 to 99. Expiration Date: by purchase.
Template 1	【東京時代道場】侍！成 <u>F</u> 武士・學習武士道體驗 [Tokyo Era Dojo] Slash! Way To Become A Warrior. The Bushido Training Experience.
Subject 1	【東京武士體驗】舉起武士刀・接受訓練體驗成 <u>F</u> 真正的武士吧 [Tokyo Bushido Experience] Raise Your Weapons! Get Training And Be A Real Warrior!
Template 2	一日 <u>F</u> : 埃斯坦西亞克里斯蒂娜牧場 One Day Tour: Estancia Cristina
Subject 2	東京武士體驗・舉起武士刀的武士體驗 Tokyo Bushido Experience An experience to raise your samurai knife

5 Conclusions and Future Work

In this study, we developed a model for generating user-specific EDM subjects and providing personalized recommendations. Our Co-Factored Embeddings based Personalized EDM Subject Generation Framework enhances personalization for users with limited interaction histories. Our results show it effectively creates relevant and preferred subject lines and recommends items aligning with user preferences. In the future, we plan to focus on refining the model to better interpret complex user behaviors and integrating real-time data for dynamic personalization.

References

1. Ao, X., Wang, X., Luo, L., Qiao, Y., He, Q., Xie, X.: PENS: a dataset and generic framework for personalized news headline generation. In: ACL/IJCNLP (2021)
2. Bražinskas, A., Lapata, M., Titov, I.: Unsupervised opinion summarization as copycat-review generation. In: ACL (2020)
3. Cao, Z., Li, W., Li, S., Wei, F.: Retrieve, rerank and rewrite: soft template based neural summarization. In: ACL (2018)

4. Cao, Z., Wei, F., Li, W., Li, S.: Faithful to the original: fact aware neural abstractive summarization. In: AAAI (2018)
5. Chen, C.Y., Wu, D., Ku, L.W.: HonestBait: forward references for attractive but faithful headline generation. In: Rogers, A., Boyd-Graber, J., Okazaki, N. (eds.) Findings of ACL (2023)
6. Chen, Y.H., Chen, P.Y., Shuai, H.H., Peng, W.C.: Tempest: soft template-based personalized EDM subject generation through collaborative summarization. In: AAAI (2020)
7. Chopra, S., Auli, M., Rush, A.M.: Abstractive sentence summarization with attentive recurrent neural networks. In: NAACL (2016)
8. Gao, S., et al.: Dialogue summarization with static-dynamic structure fusion graph. In: ACL (2023)
9. Jadhav, A., Rajan, V.: Extractive summarization with swap-net: sentences and words from alternating pointer networks. In: ACL (2018)
10. Jin, D., Jin, Z., Zhou, J.T., Orii, L., Szolovits, P.: Hooks in the headline: learning to generate headlines with controlled styles. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 5082–5093. Association for Computational Linguistics, Online (Jul 2020). <https://doi.org/10.18653/v1/2020.acl-main.456>, <https://aclanthology.org/2020.acl-main.456>
11. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. Statistics (2014)
12. Klein, G., Kim, Y., Deng, Y., Senellart, J., Rush, A.M.: OpenNMT: open-source toolkit for neural machine translation. In: ACL (2017)
13. Li, J., Li, H., Zong, C.: Towards personalized review summarization via user-aware sequence network. In: AAAI (2019)
14. Liu, T., Li, H., Zhu, J., Zhang, J., Zong, C.: Review headline generation with user embedding. In: China National Conference on Chinese Computational Linguistics (2018)
15. Ma, C., Kang, P., Wu, B., Wang, Q., Liu, X.: Gated attentive-autoencoder for content-aware recommendation. In: WSDM (2019)
16. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: ICLR (2013)
17. Nallapati, R., Zhou, B., dos Santos, C., Gulcehre, C., Xiang, B.: Abstractive text summarization using sequence-to-sequence RNNs and beyond. In: The special interest group on natural language learning (2016)
18. Narayan, S., Cohen, S.B., Lapata, M.: Ranking sentences for extractive summarization with reinforcement learning. In: NAACL (2018)
19. Ni, J., McAuley, J.: Personalized review generation by expanding phrases and attending on aspect-aware representations. In: ACL (2018)
20. See, A., Liu, P.J., Manning, C.D.: Get to the point: summarization with pointer-generator networks. In: ACL (2017)
21. Song, Y.Z., Chen, Y.S., Wang, L., Shuai, H.H.: General then Personal: decoupling and Pre-training for Personalized Headline Generation. Transactions of the Association for Computational Linguistics (2023)
22. Song, Y.Z., Shuai, H.H., Yeh, S.L., Wu, Y.L., Ku, L.W., Peng, W.C.: Attractive or faithful? Popularity-reinforced learning for inspired headline generation. AAAI **34**(05), 8910–8917 (2020)
23. Sproat, R., Emerson, T.: The first international Chinese word segmentation bake-off. In: The Special Interest Group of the Association for Computational Linguistics (2003)
24. Vinyals, O., Fortunato, M., Jaitly, N.: Pointer networks. In: NIPS (2015)

25. Vulić, I., Moens, M.F.: Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. In: The ACM Special Interest Group on Information Retrieval (2015)
26. Wang, K., Quan, X., Wang, R.: BiSET: bi-directional selective encoding with template for abstractive summarization. In: ACL (2019)
27. Wu, Y., DuBois, C., Zheng, A.X., Ester, M.: Collaborative denoising auto-encoders for top-n recommender systems. In: WSDM (2016)
28. Yi-Ting, C., Song, Y.Z., Chen, Y.S., Shuai, H.H.: Beyond detection: a defend-and-summarize strategy for robust and interpretable rumor analysis on social media. In: EMNLP (2023)
29. Zhan, J., Gao, Y., Bai, Y., Liu, Q.: Stage-wise stylistic headline generation: style generation and summarized content insertion. In: IJCAI (2022)
30. Zhang, R., et al.: Question headline generation for news articles. In: CIKM (2018)



Spatial-Temporal Bipartite Graph Attention Network for Traffic Forecasting

Dimuthu Lakmal^(✉), Kushani Perera, Renata Borovica-Gajic,
and Shanika Karunasekera

University of Melbourne, Melbourne, VIC 3010, Australia

dkariyawasan@student.unimelb.edu.au,

{kushani.perera,renata.borovica,karus}@unimelb.edu.au

Abstract. Accurate traffic forecasting is pivotal for an efficient data-driven transportation system. The intricate nature of spatial-temporal dependencies and non-linearity present in traffic data has posed a significant challenge to the modeling of accurate traffic forecasting systems. Lately, there has been a significant effort to develop complex Spatial-Temporal Graph Neural Networks (STGNN) that predominantly utilize various Graph Neural Networks (GNN) and attention-based encoder-decoder architectures due to their ability to capture non-linear dependencies in spatial and temporal domains effectively. However, conventional GNNs limit explicit propagation of past information among nodes, while attention-based models such as transformers do not support finer-grained attention score distribution. In this study, we address the aforementioned issues and introduce a novel STGNN namely, Spatio-Temporal Bipartite Graph Attention Network (STBGAT) that allows explicit modeling of past information propagation among nodes. Further, we present a heterogeneous cross-attention mechanism in a transformer to compute finer-grained feature-wise attention distribution enabling the model to capture richer and more expressive temporal dependencies. Our experiments reveal that the proposed architecture outperforms the state-of-the-art approaches proposed in recent literature.

Keywords: Graph Attention Network · Traffic Forecasting · Transformers · Spatial Graph Attention Networks

1 Introduction

Transportation systems have become complex with the rapid growth of infrastructure and people's needs. Thus, relevant stakeholders continuously invest in implementing intelligent transportation systems (ITS) aiming for more efficient, accurate, and data-driven traffic management solutions. Accurate and real-time traffic condition forecasting is one of the core components of ITS. Traffic condition forecasting systems are designed to predict future traffic conditions given the historical traffic condition observations. Particularly, we focus on forecasting

traffic flow, one of the main traffic condition measurements. Typically, traffic flow in a specific road section is influenced by not only its own historical traffic conditions, but also the traffic conditions in adjacent connected road sections. *Hence, it is crucial to consider the propagation of traffic information through the spatial structure of the road network when forecasting traffic flow. Moreover, intricate temporal dynamics in road networks have made long-term (30~60 min) traffic forecasting even more challenging [28].*

To address the aforementioned challenges, researchers have formulated traffic flow forecasting as a spatial-temporal graph modeling problem and proposed various types of Spatial-Temporal Graph Neural Networks (STGNN) [12, 29]. Even though recent efforts have attained substantial improvements [9] in accuracy compared to early versions of STGNNs [27], they are not sophisticated enough to effectively discover and leverage intricate spatial and temporal dependencies. This study primarily focuses on two major deficiencies of existing traffic forecasting STGNNs. First, the effects of traffic conditions on roads take time to gradually propagate to their adjacent roads through the network. However, existing approaches fail to ascertain how the traffic flow of a specific road at a given time is impacted by previous traffic conditions on adjacent roads. Second, the majority of existing approaches relied on raw historical observations as input features and have not included and assessed alternative feature sequences, such as averaged traffic flow sequences which could reveal more temporal and spatial dependencies [29].

To address these shortcomings, we present a novel spatial-temporal graph neural network (STBGAT), that consists of a bipartite graph attention network and a transformer with a heterogeneous cross-attention mechanism (Source code is available here: <https://github.com/DimuthuLakmal/STBGAT>). We conducted a comprehensive set of experiments using five different traffic datasets to evaluate the performance of the proposed model. Those experiments revealed that STBGAT significantly outperforms the current state-of-the-art models. The main contributions of this study can be summarized as follows:

- We propose a novel Bipartite Graph Attention Network for past neighborhood information propagation towards center nodes. This mechanism ensures the impact of the previous traffic conditions on adjacent roads is explicitly considered.
- We introduce a heterogeneous cross-attention mechanism in the transformer model which enables the decoder to assign separate feature-wise attention scores to the encoder outputs. This architecture allows for the integration of multiple encoders, each handling different input sequences. It will alleviate the impact of noise and missing values in each feature sequence while revealing more temporal and spatial dependencies.

2 Related Work

Prior to recent advancements in Graph Neural Networks (GNN), researchers have widely adopted classic statistical time series algorithms and machine learning models to make predictions [21, 25]. However, these models are only capable of

analyzing temporal dynamics in traffic data leaving spatial correlations unused. In contrast, Spatial-Temporal Graph Neural Networks (STGNN) have significantly improved accuracy by efficiently capturing and modeling both temporal and spatial dynamics in road networks [24, 27]. Since the introduction of STGCN architecture by Yu et al. [27], various highly complex STGNN architectures have been proposed in the literature attaining substantially improved accuracy [9, 29].

Various graph neural network architectures have been adopted as the spatial module in STGNNs, ranging from recurrent GNN to Graph Attention Neural Networks (GAT) [7, 12, 17]. GCN [11] is one of the prominent works in the Graph Neural Network domain that led to rapid success in STGNNs [26]. The superiority in efficiency, flexibility, and accuracy of GCN and its variants over prior GNN architectures have resulted in wide adoption of GCNs in STGNN models. On the other hand, Graph Attention Network (GAT) [23] outperforms GCNs as it uses an attention mechanism in the data propagation process within the graph. In this study, we develop the proposed bipartite graph by extending the default GAT implementation.

Further, various architectures have been proposed for the temporal module in STGNNs. There are three frequently used neural network architectures in the temporal module: 1) RNN-based, 2) CNN-based, and 3) Attention-based [1, 14, 16]. Compared to the other two, the attention mechanism has emerged as a highly compelling approach in temporal sequence modeling. We develop the proposed temporal module based on a transformer which is an encoder-decoder architecture relying on an attention mechanism [22]. None of the recent STGNN approaches have proposed explicit modeling of past information propagation from neighbors due to the additional complexity it imposes on these models. Further, only a few attempted to incorporate features beyond raw traffic flow values as inputs [5], and these attempts were insufficient to distinctly discern the significance of each feature in making predictions.

3 Definitions and Problem Statement

3.1 Definitions

Traffic Road Network. We represent a traffic road network with a directed graph $G = (V, E, A)$ where $V = v_1, \dots, v_N$ is a set of N nodes representing traffic sensors; E is a set of edges among nodes; $A \in R_{N \times N}$ is a weighted adjacency matrix representing connectivity among nodes and edge weights between any of two connected nodes.

Traffic Flow Matrix. $X_t \in N \times C$ denotes the traffic condition feature matrix at time step t . N represents the number of nodes in the network and C represents the number of traffic condition-related features including traffic flow value associated with each individual node. $X'_t \in N \times 1$ denotes the traffic flow matrix at time step t .

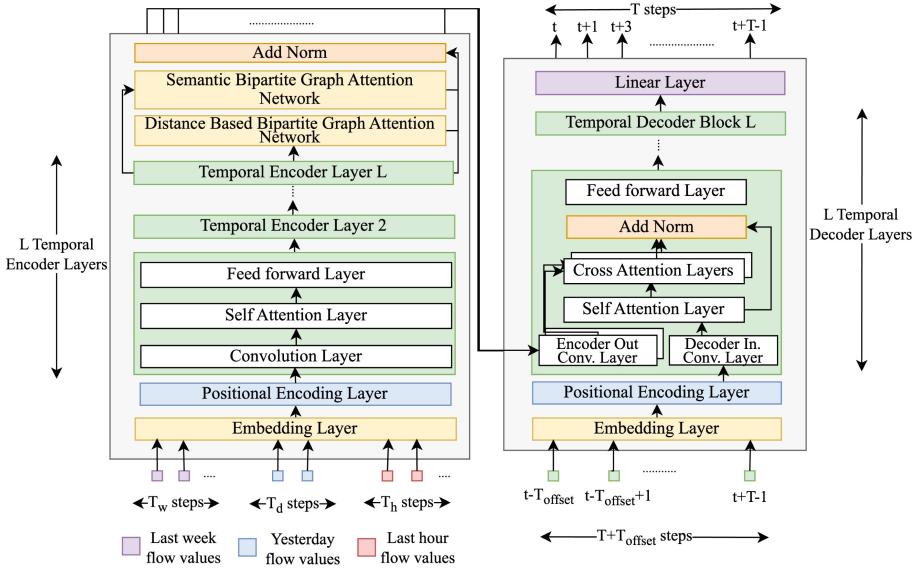


Fig. 1. Overall architecture of STBGAT

3.2 Problem Statement

Given the historical observation spanning a specific time frame, the problem is to establish a mapping function that can output the sequence of future traffic flow values having a predefined length. Let us assume the number of historical observations ending at the current step t is T_1 and the length of the prediction sequence is T_2 . Then the mapping function f can be formally expressed as,

$$[X_{(t-T_1+1)}, X_{(t-T_1+2)}, \dots, X_t; G] \xrightarrow{f} [X'_{(t+1)}, X'_{(t+2)}, \dots, X'_{(t+T_2)}; G]$$

4 Methodology

4.1 Data Inputs and Data Preprocessing

This section focuses on briefing data inputs and the non-trivial data preprocessing steps we followed in this study.

If the target sequence is the traffic flow in the next hour from the current time, the most recent available sequence will be the traffic flow from the last hour, and it is incorporated as a part of the input sequence to the encoder. Several past studies have utilized additional sequences of historical observations as inputs, where these sequences may closely resemble either the pattern of the target sequence or the latest historical observation sequence [19]. We identified two effective periodic sequence patterns that tend to yield better results across all the datasets that we experimented with. The first is the sequence of traffic flow from the same hour as the hour just preceding the target hour, but on the

previous date. The second effective periodic sequence pattern is from the same hour as the hour just preceding the target hour, but on the same day in last week. We concatenate these three shorter sequences into one single sequence and use as an input sequence to the model. Additionally, we use another input sequence to the model, referred to as a representative input sequence, representing averaged behavior corresponding to the time duration of the raw input sequence described earlier. Incorporating repetitive and representative sequence patterns could benefit the model in two ways. First, it helps the model to identify long-term and short-term trends. Second, it helps to mitigate the impact of missing values in shorter sequences. The total length of each encoder sequence can be defined as $T_e = T_{last_week} + T_{last_day} + T_{last_hour}$.

To construct the representative input sequence, it is required to determine the average behavior at each weekly time index. All traffic flow datasets we tested in this study consist of 12 traffic flow values per hour, totaling 2016 per week. Therefore, we can assign a weekly time index for each traffic flow record. When calculating the average traffic flow at a particular weekly time index, we judiciously apply a rule-based filter to remove traffic flow records with noises.

We redefined the connectivity within the sensor network in certain datasets for more efficient and accurate flow of information among nodes. We introduced two types of connectivity producing two different connectivity graphs namely: distance-based bipartite graph and semantic bipartite graph. The distance-based graph is defined based on the assumption that two sensors in close geographic proximity to each other exhibit significant correlations between their recorded traffic flow values. To accommodate this assumption, we calculated edge attributes based on the shortest distance between nodes using Dijkstra's algorithm [10]. Then we dropped edges that exceeded a predefined distance threshold.

We defined a second graph based on the time series semantics among nodes. This helps to identify nodes that have similar behavior, but are not connected in the geographically connected graph defined above. For instance, in a scenario where two sensor nodes are located near two different schools, but are physically distant from each other, it may be important to propagate information between those two sensor nodes to identify common short-term temporal behaviors. For each weekly index described above, we picked a certain number of most similar nodes for every node in the graph. Then, a single global semantic graph is constructed assigning the set of nodes as neighbors of each node in the graph, which have the highest number of short-term similar behaviors with each center node. This calculation is based on the semantic distances among representative time series of nodes, each of which consists of 12 time steps. Dynamic Time Warping (DTW) algorithm is used to measure the similarity between two sequences [3].

4.2 Encoder Decoder Architecture

In this section, we brief the overall architecture of STBGAT model depicted in Fig. 1. The model follows transformer encoder-decoder architecture with some optimization done focusing on the traffic forecasting problem. The model can accommodate multiple encoders at once to facilitate feature extraction from

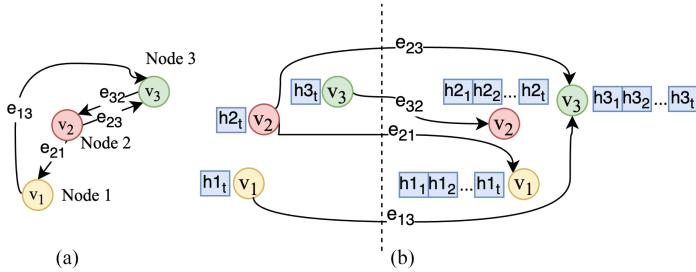


Fig. 2. Formation of bipartite graph. Figure 2a shows the example base graph with three nodes for the bipartite graph depicted in Fig. 2b.

multiple types of input sequences. In this study, we employ two encoders to process the two input sequences described in Sect. 4.1. The embedding layer maps the inputs to a higher dimensional vector space. The positional embedding layer fuses time step information and makes sure that the self-attention module is aware of the positional information.

The output of the positional encoding layer is then processed through a temporal encoder layer stack comprised of L number of layers. Each encoder layer includes a convolution layer similar to the one used in the temporal trend-aware multi-head self-attention layer described in [4]. The processed output of the convolution layer is then fed into a self-attention layer followed by a feed-forward layer. The output of the last temporal encoder layer is subsequently passed into the spatial module that consists of two bipartite graph attention networks enabling information propagation among nodes. Bipartite graphs are constructed as explained in Sect. 4.1. Finally, the output from bipartite GATs and the output from the final temporal encoder layer are summed together to produce the final encoder output. The decoder uses the output from the encoder as the historical context in the process of cross-attention calculation.

The embedding layer, positional encoding layer, self-attention layer, and feed-forward layer of the decoder are similar to the ones used in the encoder. Further, the decoder input convolution layer is also similar to the convolution layer used in the encoder layer. We introduce a heterogeneous feature-wise cross-attention mechanism as described in Sect. 4.4. Latent vector output given by the last decoder layer L is projected to an output with a single dimension by the linear layer of the decoder.

4.3 Bipartite Graph Attention Layer

In this section, we explain one of the main contributions of this paper that solves past information propagation problem described in Sect. 1. Traffic flow recorded at a sensor at a specific time step is influenced not only by the traffic flow on neighboring roads at the exact time step but also by previous traffic conditions on neighboring roads. The dependency on traffic conditions during previous time

steps arises due to the propagation delay between different parts of the road network [9]. For instance, if a road accident occurs on a road section, then the traffic conditions of neighboring road segments will gradually adjust over time to accommodate the impact of the accident.

It can be argued that as the input sequences pass through a set of temporal encoder layers before reaching the spatial module, the latent vector output at each time step comprises a certain level of information about its preceding temporal context. Nonetheless, our study reveals that this implicit representation of the temporal context alone is not sufficient to effectively address the propagation delay in road networks. Instead, we propose to use a bipartite graph G_t that consists of two sets of nodes, (u_t, v_{T_e}) where u_t represents nodes consisting of outputs of temporal encoder module at time step t , and each node in v_{T_e} consists of concatenated output of temporal encoder from time step $t = 1$ to $t = T_e$. The edge attributes of the bipartite graph are equal to the corresponding edge attributes in the regular graph. The formulation of the bipartite graph from a regular graph is shown in Fig. 2. The implementation of Bipartite GAT can be outlined in three equations from Eq. 1 to Eq. 3.

$$e_{ij} = \text{LeakyReLU}(a^T (W h_i || W h_j || W_E E_{ij})) \quad (1)$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in N_i} \exp(e_{ik})} \quad (2)$$

$$h_i^{imm} = \|\sum_{k=1}^K \sigma(\sum_{j \in N_i} \alpha_{ij}^k W^k h_j) \quad (3)$$

In Eq. 1, W, W_E and a denote learnable weights while E_{ij} denotes edge attribute associated with the edge connecting nodes i and j . Node i belongs to u_t and node j belongs to v_{T_e} . h_i represents the value of the central node i while h_j represents the value of neighbor node j . α_{ij} in Eq. 2 is the attention score calculated for neighbor node j . The final output for central node i is derived using Eq. 3. A single GAT layer consists of K heads. Thus, the final output h_i^{imm} is formed by either a concatenating or averaging of the outputs generated by K heads. Equation 3 shows only the concatenation operation over K heads.

4.4 Heterogeneous Cross Attention Layers

In this section, we present the second contribution of this paper that enables integrating multiple feature sequences. The transformer decoder consists of a cross-attention component that calculates attention values for elements in the encoder output sequence with respect to the decoder input sequence. This mechanism allows the decoder to focus on relevant information in the encoder output when generating the decoder output. The naive implementation of cross-attention only accepts the encoder output sequence as a single sequence and is not granular enough to calculate feature-wise attention. In contrast, we propose a heterogeneous cross-attention mechanism that is capable of calculating

Table 1. Prediction accuracy results (PEMS04-08)

		VAR	SVR	LSTM	DC-RNN	ST-GCN	GMAN	AST-GNN	PDFFormer	PDFFormer(L)	ST-BGAT
PEMS04	MAE	23.75	28.66	26.81	23.65	22.27	19.14	18.60	18.39	18.40	18.17
	RMSE	36.66	44.59	40.74	37.12	35.02	31.60	31.03	30.01	30.25	28.23
	MAPE	18.10	19.15	22.33	14.75	13.87	13.19	12.63	12.13	12.23	12.02
PEMS07	MAE	101.20	32.97	29.71	23.63	22.90	20.97	20.62	19.83	N/A	18.34
	RMSE	155.14	50.15	45.32	36.51	35.44	34.10	34.02	32.87	N/A	30.86
	MAPE	39.69	15.43	14.14	12.28	11.98	9.05	8.86	8.53	N/A	7.63
PEMS08	MAE	22.32	23.25	22.19	18.19	17.84	15.31	13.29	13.58	12.51	12.39
	RMSE	33.83	36.15	33.59	28.18	27.12	24.92	23.33	23.51	22.10	21.02
	MAPE	14.47	14.71	18.74	11.24	11.21	10.13	9.03	9.05	8.55	8.43

distinct attention distributions for different feature sequences. This mechanism allows for more precise modeling of temporal dynamics and relationships present in different feature sequences. STBGAT produces two separate encoder output sequences in parallel using the two input sequences described in Sect. 4.1. Following this, two cross-attention distributions are generated based on encoder output sequences. The final output of the cross-attention layer will be calculated according to the Eq. 4. In Eq. 4, LayerNorm refers to layer normalization. $X_{self-attn}$ refers to self-attention output calculated over decoder input. x_f is the encoder output calculated for the sequence of feature type f .

$$h_{cross} = \text{LayerNorm}(X_{self-attn} + \sum_{f \in F} \text{CrossAttn}(x_f)) \quad (4)$$

5 Experiments

5.1 Experiment Setup

We evaluate our model on two groups of datasets that are widely used in the literature. The first group consists of three datasets; PEMS04, PEMS07, and PEMS08 [2] while the second group consists of two datasets; PEMS-BAY and METR-LA [8].

We evaluate STBGAT against a variety of baselines proposed in the literature on the aforementioned two dataset groups. Tested baseline models are listed below.

- **PEMS04-08:** VAR [18], SVR, LSTM [6], DCRNN [15], STGCN [27], GMAN [29], ASTGNN [4], PDFFormer [9]
- **PEMS-BAY, METR-LA:** VAR [18], SVR, FC-LSTM [6], DCRNN [15], STGCN [27], GMAN [29], STGM [13], STEP [20]

The default PDFormer only supports input sequences with a maximum length of 12. However, to ensure a fair comparison, we adapted the default PDFormer to

Table 2. Prediction accuracy results (PEMS-BAY, METR-LA)

		VAR	SVR	LSTM	DC-RNN	ST-GCN	GMAN	STGM	STEP	ST-BGAT
PEMS -BAY	MAE	2.93	3.28	2.37	2.07	2.49	1.86	1.86	1.79	1.75
	RMSE	5.44	7.08	4.96	4.74	5.69	4.32	4.37	4.20	3.60
	MAPE	6.50	8.00	5.70	4.90	5.79	4.37	4.34	4.18	4.01
METR -LA	MAE	6.52	6.72	4.37	3.60	4.59	3.44	3.23	3.37	3.94
	RMSE	10.11	13.76	8.69	7.60	9.40	7.35	7.10	6.99	7.25
	MAPE	15.80	16.70	14.00	10.50	12.70	10.07	9.39	9.61	9.79

accept a 36-length input sequence that includes repetitive patterns. It is referred to as PDFormer(L) in Table 1. In contrast, other recent architectures STEP and ASTGNN support longer sequences by default.

Three evaluation matrices are used namely, Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE). Masked versions of MAE, RMSE, and MAPE matrices are used to alleviate the effect of missing values in the dataset. Experiments on each model are repeated 3 times and we report mean values for said matrices.

The test results of the STBGAT model presented in Table 1 and Table 2 are generated by passing input sequences consisting of repetitive patterns. The reported error values are the averaged error values computed across the entire prediction sequence length.

5.2 Comparison of Performance

The overall performance of baseline models and STBGAT is summarized in Table 1 and Table 2. It is important to highlight that the PDFormer(L) model fails to run on PEMS07 dataset due to memory overflow which suggests that it is not suitable for large road networks (tested on 128GB of RAM). The best results for each metric reported in each dataset are highlighted in bold. Our model outperforms all baselines in every performance metric across four datasets; PEMS07, PEMS08, PEMS04, and PEMS-BAY. However, the STBGAT model exhibits lower performance on the METR-LA dataset, particularly in MAE metric. We discover that this is attributed to the high discrepancy between train and test data distributions. Moreover, STBGAT model significantly outperforms all baselines in terms of RMSE metric. This metric is a useful indicator of performance when large errors between ground truth and predicted values are undesirable. Hence, it suggests that STBGAT can better approximate sudden fluctuations in traffic flow.

The most notable observation across the experiments is the substantial performance enhancement achieved by spatial-temporal models in comparison to temporal prediction models alone. This accentuates the importance of discover-

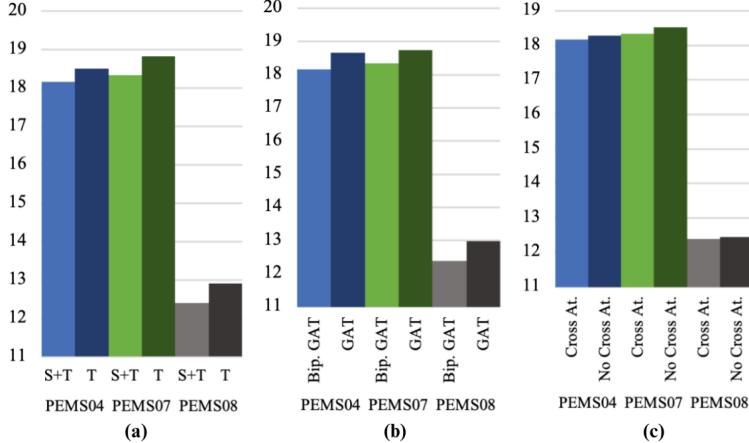


Fig. 3. Ablation study results (MAE Values)

ing and exploiting spatial dependencies in road network graphs. LSTM exhibits the best performance among temporal prediction models leveraging its ability to identify long temporal dependencies compared to other temporal models. Based on our experiments, the spatial module can be considered as an enhancement to improve the accuracy of the temporal module of spatial-temporal models. Hence, having a comprehensive temporal module is also important to have better performance. DCRNN and STGCN models consist of RNN and convolution components in the temporal module that could hinder performance in modeling temporal dependency effectively. In contrast, GMAN, ASTGNN, STEP, STGM, and PDFormer use various attention mechanisms that help achieve superior performance over traditional RNN-based models. In addition to the performance enhancements achieved through the two novel concepts presented here, two other existing concepts contributed to STBGAT's performance. First, STBGAT use CNN layers in temporal module for local context recognition which is also used in ASTGNN, PDFormer and STGM. Second, STBGAT effectively harnesses both short-term and long-term spatial dependencies as it uses two types of bipartite graphs in the encoder. A comparable approach is also utilized in PDFormer and STGM models. The contribution of the two novel concepts presented in this paper is discussed in the next section.

5.3 Ablation Study

Additional experiments are carried out to investigate the effectiveness of different components of STBGAT using PEMS04, PEMS07 and PEMS08 datasets. We first study the prediction capability of the temporal module by training the model without the spatial component of the model. We then test the impact and contribution of the spatial component to predictions. The evaluation results are

depicted in Fig. 3a. Results suggest that the temporal module plays a critical role in the prediction task. The model with only the temporal module outperforms the majority of the baselines except for ASTGNN and PDFormer in every performance metric. These results also indicate the importance of information propagation within the graph, particularly in PEMS07 and PEMS08 datasets.

Next, we assess the performance enhancement in the spatial module achieved by bipartite GAT in comparison to conventional GAT. The results of this experiment are presented in Fig. 3b. A substantial improvement can be observed when utilizing the proposed bipartite GAT in contrast to conventional GAT.

Finally, an experiment is conducted to evaluate the impact of the heterogeneous cross-attention mechanism compared to the traditional cross-attention mechanism. The results of this experiment are presented in Fig. 3c. According to the experiment, the effect of the heterogeneous cross-attention mechanism on the performance is not as pronounced as in bipartite GAT. Nevertheless, it contributes to enhancing the overall performance of the model compared to the conventional cross-attention mechanism.

6 Conclusion and Future Works

In this paper, we introduce a novel spatial-temporal graph neural network architecture for traffic forecasting that outperforms the latest state-of-the-art baselines across four real-world datasets. We proposed two novel concepts in this paper; bipartite graph attention network and heterogeneous cross-attention mechanism. The first concept enhances the spatial information propagation while the second concept improves the temporal dependency analysis of the model. The ablation study demonstrates the effectiveness of these two novel concepts in modeling spatial and temporal dynamics. As future work, the model can be extended and utilized in various downstream tasks in domains such as traffic analysis and social media.

Acknowledgements. This research was supported by The University of Melbourne’s Research Computing Services and the Petascale Campus Initiative.

References

1. Bai, L., Yao, L., Kanhere, S.S., Wang, X., Liu, W., Yang, Z.: Spatio-temporal graph convolutional and recurrent networks for citywide passenger demand prediction. In: Proceedings of the 28th ACM CIKM, pp. 2293–2296 (2019)
2. Chen, C., Petty, K., Skabardonis, A., Varaiya, P., Jia, Z.: Freeway performance measurement system: mining loop detector data. TRR **1748**(1), 96–102 (2001)
3. Giorgino, T.: Computing and visualizing dynamic time warping alignments in r: the dtw package. J. Stat. Softw. **31**, 1–24 (2009)
4. Guo, S., Lin, Y., Wan, H., Li, X., Cong, G.: Learning dynamics and heterogeneity of spatial-temporal graph data for traffic forecasting. IEEE TKDE **34**(11), 5415–5428 (2021)
5. He, H., Ye, K., Xu, C.Z.: Multi-feature urban traffic prediction based on unconstrained graph attention network. In: 2021 IEEE BigData, pp. 1409–1417 (2021)

6. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
7. Huang, R., Huang, C., Liu, Y., Dai, G., Kong, W.: Lsgcn: Long short-term traffic prediction with graph convolutional networks. In: IJCAI, vol. 7, pp. 2355–2361 (2020)
8. Jagadish, H.V., Gehrke, J., Labrinidis, A., Papakonstantinou, Y., Patel, J.M., Ramakrishnan, R., Shahabi, C.: Big data and its technical challenges. *Commun. ACM* **57**(7), 86–94 (2014)
9. Jiang, J., Han, C., Zhao, W.X., Wang, J.: Pdformer: propagation delay-aware dynamic long-range transformer for traffic flow prediction. arXiv preprint [arXiv:2301.07945](https://arxiv.org/abs/2301.07945) (2023)
10. Johnson, D.B.: A note on dijkstra’s shortest path algorithm. *JACM* **20**(3), 385–388 (1973)
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. Kong, X., Xing, W., Wei, X., Bao, P., Zhang, J., Lu, W.: Stgat: spatial-temporal graph attention networks for traffic flow forecasting. *IEEE Access* **8**, 134363–134372 (2020)
13. Lablack, M., Shen, Y.: Spatio-temporal graph mixformer for traffic forecasting. *Expert Syst. Appl.* **228**, 120281 (2023)
14. Li, W., Wang, X., Zhang, Y., Wu, Q.: Traffic flow prediction over muti-sensor data correlation with graph convolution network. *Neurocomputing* **427**, 50–63 (2021)
15. Li, Y., Yu, R., Shahabi, C., Liu, Y.: Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. arXiv preprint [arXiv:1707.01926](https://arxiv.org/abs/1707.01926) (2017)
16. Li, Y., Moura, J.M.: Forecaster: a graph transformer for forecasting spatial and time-dependent data. In: ECAI 2020, pp. 1293–1300. IOS Press (2020)
17. Lu, Z., Lv, W., Cao, Y., Xie, Z., Peng, H., Du, B.: Lstm variants meet graph neural networks for road speed prediction. *Neurocomputing* **400**, 34–45 (2020)
18. Lütkepohl, H.: Vector autoregressive models. *Handbook of research methods and applications in empirical macroeconomics* 30 (2013)
19. Roy, A., Roy, K.K., Ali, A.A., Amin, M.A., Rahman, A.M.: Unified spatio-temporal modeling for traffic forecasting using graph neural network. In: 2021 IJCNN, pp. 1–8. IEEE (2021)
20. Shao, Z., Zhang, Z., Wang, F., Xu, Y.: Pre-training enhanced spatial-temporal graph neural network for multivariate time series forecasting. In: Proc. 28th ACM SIGKDD Conf. Know. Disc. Data Min., pp. 1567–1577 (2022)
21. Tian, Y., Zhang, K., Li, J., Lin, X., Yang, B.: Lstm-based traffic flow prediction with missing data. *Neurocomputing* **318**, 297–305 (2018)
22. Vaswani, A., et al.: Attention is all you need. *Adv. NIPS* **30** (2017)
23. Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y., et al.: Graph attention networks. *Stat* **1050**(20), 10–48550 (2017)
24. Wang, X., et al.: Traffic flow prediction via spatial temporal graph neural network. In: Proc. web conf. 2020, pp. 1082–1092 (2020)
25. Williams, B.M., Hoel, L.A.: Modeling and forecasting vehicular traffic flow as a seasonal arima process: theoretical basis and empirical results. *J. Trans. Eng.* **129**(6), 664–672 (2003)
26. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. *IEEE TNNLS* **32**(1), 4–24 (2020)
27. Yu, B., Yin, H., Zhu, Z.: Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting

28. Yu, H., Wu, Z., Wang, S., Wang, Y., Ma, X.: Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks. *Sensors* **17**(7), 1501 (2017)
29. Zheng, C., Fan, X., Wang, C., Qi, J.: Gman: a graph multi-attention network for traffic prediction. In: Proc. of the AAAI Conf. on Art. Intell., vol. 34, pp. 1234–1241 (2020)



CMed-GPT: Prompt Tuning for Entity-Aware Chinese Medical Dialogue Generation

Zhijie Qu^(✉), Juan Li, Zerui Ma, and Jianqiang Li

Beijing University of Technology, Beijing 100124, China
quzhijie@mails.bjut.edu.cn

Abstract. Medical dialogue generation relies on natural language generation techniques to enable online medical consultations. Recently, the widespread adoption of large-scale models in the field of natural language processing has facilitated rapid advancements in this technology. Existing medical dialogue models are mostly based on BERT and pre-trained on English corpora, but there is a lack of high-performing models on the task of Chinese medical dialogue generation. To solve the above problem, this paper proposes CMed-GPT, which is the GPT pre-training language model based on Chinese medical domain text. The model is available in two versions, namely, base and large, with corresponding perplexity values of 8.64 and 8.01. Additionally, we incorporate lexical and entity embeddings into the dialogue text in a uniform manner to meet the requirements of downstream dialogue generation tasks. By applying both fine-tuning and p-tuning to CMed-GPT, we lowered the PPL from 8.44 to 7.35. This study not only confirms the exceptional performance of the CMed-GPT model in generating Chinese biomedical text but also highlights the advantages of p-tuning over traditional fine-tuning with prefix prompts. Furthermore, we validate the significance of incorporating external information in medical dialogue generation, which enhances the quality of dialogue generation.

Keywords: Chinese medical dialogue · Pre-trained language model · P-tuning

1 Introduction

In the context of the current global health crisis, telemedicine's role as a supplement to traditional healthcare has grown in significance. Telemedicine can not only help address the imbalance in the distribution of medical resources but also alleviate the problem of resource scarcity. Additionally, it can enhance medical efficiency and convenience, facilitate real-time communication between patients and doctors, reduce treatment duration, and improve overall medical efficiency. Due to the rapid advancement of Artificial Intelligence (AI) systems, an increasing number of medical researchers are eager to advance intelligent AI dialogue systems into virtual medical practitioners. These virtual medical practitioners can engage in patient consultations, understand their medical conditions and complete medical histories, and offer well-informed clinical recommendations. Consequently, in recent years, He [1], Liu [2], Li [3], Wei [4], Xu [5] and

others have proposed task-oriented medical dialogue models to play the role of virtual medical practitioners and engage in one-on-one consultations with patients. However, the presence of specialized phrases and formal medical expressions in Chinese medical conversations makes medical dialogue systems more difficult to implement than task-oriented dialogue systems (TDS).

Pre-training followed by fine-tuning has emerged as the prevailing approach for transferring the capabilities of large models to downstream tasks within the NLP domain [6], including medical dialogue generation. In the biomedical field, researchers have begun investigating the application of pretrained language models. Unfortunately, general pre-trained models perform poorly in the medical domain [7, 8], likely due to the presence of a significant number of domain-specific terms that are difficult for these models to comprehend. To enhance pretrained model performance in the medical domain, a series of expert models based on medical datasets have emerged. Even without fine-tuning, PubMedBERT [8], BioBERT [9], SciBERT [10], and DilBERT [11] models outperform BERT in downstream tasks. This represents an outstanding development, as researchers appear to have identified the future research direction for telemedicine: medical pre-training models and fine-tuning.

However, most prior research has focused on BERT-pretrained models and English medical dialogue datasets, leading to underwhelming performance in the generation of Chinese medical dialogues. After analyzing the causes, the main problems with the existing methods are as follows: Regarding medical dialogue datasets, most of the dialogue datasets are in English not Chinese. Regarding models, BERT, a representative bidirectional language model with its bidirectional attention mechanism, may not be as suitable as unidirectional models like GPT for natural language generation (NLG) tasks. Regarding medical domain knowledge, it is more specialized compared to other domains, characterized by complex, less frequent vocabulary, and abundant technical jargon. Undoubtedly, these three factors pose significant challenges for the models.

This paper makes the following major contributions to solving the above problems:

1. We collect medical dialogue datasets in Chinese and proceed with pre-training of GPT on the Chinese medical dialogue datasets to develop a Chinese generalized pre-trained language model, namely CMed-GPT, for the biomedical domain.
2. To enhance the model's understanding of medical dialogue, we executed fine-tuning and p-tuning on the pre-trained CMed-GPT model employing downstream medical dialogue data. This enhanced the model's comprehension while simultaneously reducing the number of parameters during the fine-tuning stage.
3. To enable the model to better understand the entities in the medical domain, we combine lexical and entity embeddings with dialog history to obtain the entity-aware medical dialog generation models KB-FT-CMed-GPT and KB-PT-CMed-GPT. The results demonstrate the effectiveness of our method.

The paper is organized as follows: the related works on dialogue generation is shown in Sect. 2. Datasets are shown in Sect. 3. Details of the proposed Chinese medical dialogue generation model is introduced in Sect. 4. Experimental and results are presented in Sect. 5. Finally, conclusion is given in Sect. 6.

2 Related Work

For recent research on dialogue generation, pre-training on large-scale datasets and fine-tuning on downstream tasks have proved to be a successful paradigm and have become the standard models. Two of the typical models are BERT and GPT.

BERT is a bidirectional encoder representation language model based on the transformer architecture. In subsequent studies, the improvement points of BERT are mainly focused on two parts: the construction of large-scale corpus data [1] and model pre-training [12]. In the field of biomedicine, scholars have proposed several approaches to enhance the performance of BERT models by pre-training them with English medical domain datasets. These approaches include the following three ideas: 1) Continued pre-training using medical domain data, such as BioBERT [9] and DiLBERT [11]; 2) Pre-training from scratch using medical domain data, such as SciBERT [10] and PubMedBERT [8]; 3) Pre-training based on self-supervised tasks specifically designed for the medical domain, such as MC-BERT [13], SMedBERT [14] and BERT-MK [15]. While various biomedical pre-trained language models based on BERT have achieved great success in natural language understanding and classification tasks, few scholars have worked on the task of generating BERT models due to its Transformer encoder model architecture that limits NLG.

GPT is an autoregressive language model based on Transformer’s decoder. Compared to the BERT, GPT has significantly larger training corpora and model parameters such as GPT-2 [16] and GPT-3 [17]. The modeling structure of GPT makes it outstanding for generative tasks. With the emergence of ChatGPT, a new fine-tuning paradigm based on pre-trained language models, p-tuning, has become a major mainstream approach to improve the performance of GPT on downstream tasks. For domain-specific generative tasks, the inclusion of prompt allows the model to generate text that is more consistent with the user’s intent. In the field of biomedicine, while GPT is well-suited for medical dialogue generation, there are very few pretrained language models based on GPT for medical text generation. Previous work on pretraining GPT in the biomedical literature is DARE [18]. However, they pretrained GPT on an extremely limited dataset consisting of only 0.5 million PubMed abstracts and used it solely for data augmentation in relation extraction tasks. Generative pre-trained language modeling for Chinese is a research gap. With the increase of Chinese medical dialogue datasets in recent years, it is a matter of concern how the GPT model performs in the field of Chinese medical dialogue generation.

Therefore, we propose the CMed-GPT specifically designed for Chinese medical dialogue scenarios. During the pre-training phase of the model, we continue to train the GPT model using Chinese medical datasets. In the fine-tuning phase, we perform p-tuning of CMed-GPT by integrating lexical and entity information into the dialogue text in a uniform manner. This approach not only further demonstrates the effectiveness of domain-specific pretraining on improving model performance but also significantly reduces the parameter count compared to fine-tuning. Furthermore, the uniform incorporation of lexical and entity information in this paper, as opposed to the discrete approach, enhances the model’s understanding ability and generates higher-quality dialogue responses that better align with user intent.

3 Datasets

Datasets are crucial for pre-trained language models. This section aims to provide a comprehensive description of the datasets utilized in this study. The datasets used for model pre-training include Chinese medical dialogue datasets and medical books.

Chinese Medical Dialogue Datasets. The dataset consists of seven single and multi-turn medical dialogue datasets from different sources. It includes five multi-turn medical dialogue datasets labeled with entities such as symptoms, diseases, drugs, examinations, etc., as well as two medical dialogue datasets without entity labeling. The above datasets were obtained from online consulting medical websites, smart conversation clinic competitions and previous studies. The data is shown in Table 1:

Table 1. Chinese medical dialogue datasets.

Dataset	Diseases	Dialogues	Utterances	Tokens	Entities
CHIP-MDCFNPC	–	8000	247,520	4259,819	5494,944
CMDD	4	2064	86,874	868,738	651,553
cMQA-master	–	260,000	520,000	67,080,000	–
IMCS-IR	6	3052	122,080	1,599,248	1,903,227
MedDG	12	17,864	385,862	6829,764	4692,086
ChunYu	15	12,842	317,197	3,362,292	4091,846
COVID-DDC	1	1088	9465	405,128	–

Chinese Medical Book Text Dataset. To enhance the comprehension capabilities of pre-trained models, this paper additionally incorporates open-source medical book text data. The datasets were used to further pre-train existing models, resulting in a pre-trained Chinese language model for the medical field. The book text encompasses pharmacology, diagnostics, pathology, etc., totaling approximately 655.8MB of Chinese biomedical text and roughly 11.2 million tokens.

4 Method

4.1 Pre-training Model

Model Structure. We primarily continue pretraining on GPT2-Chinese¹, which follows the structure of a Transformer decoder [19] (GPT structure) and employs a multi-head unidirectional attention mechanism. Given the dialogue history $X = X_1, X_2, X_3 \dots X_i \dots X_k$, where X_i is either a doctor's or a patient's utterance. The utterance $X_i = [u_1^i, u_2^i \dots u_j^i \dots u_s^i]$ has s tokens, when predicting the token u_j^i , the model will mask the tokens after u_j^i . The model structure (see Fig. 1).

¹ <https://github.com/Morizeyao/GPT2-Chinese>.

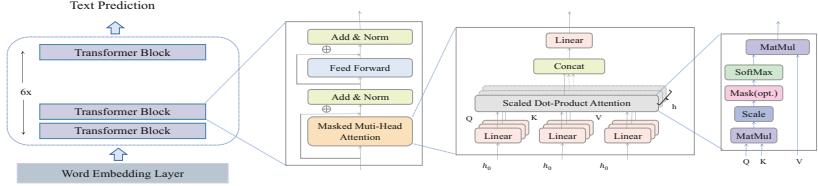


Fig. 1. Pre-trained language model structure.

As shown in Fig. 1, the model structure is divided into three layers, including the word embedding layer, transformer block layer, and text prediction layer. The GPT model architecture consists of six transformer block layers, with its core component being the multi-head attention structure. The input matrix h_0 is transformed using linear transformation matrices $WQ/WK/WV$ to obtain Q , K , and V , which are used to compute the output of the self-attention mechanism.

$$\text{head}_i(Q, K, V) = \text{soft max} \left(\frac{Q_i K_i^T}{\sqrt{d_k}} \right) V \quad (1)$$

Multi-head Attention consists of multiple self-attention layers:

$$\text{Multihead}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, \dots, \text{head}_h)W \quad (2)$$

Model Training Objectives. Given a dialogue history $X = X_1, X_2, X_3 \dots X_i \dots X_k$, the objective of training the model is to minimize the following likelihood function:

$$\min \left(-\frac{1}{K} \sum_{i=1}^K \sum_{j=1}^S \log P(u_j^i | u_1^i \dots u_{j-1}^i) \right) \quad (3)$$

The final loss is:

$$\text{Loss} = \text{CrossEntropyLoss}(H) \quad (4)$$

where H represents the vector after passing through the text prediction layer.

In the following, fine-tuning, p-tuning, and the model with lexical and entity embeddings, the training objectives for these models all involve the next token prediction task, and the formulas are as shown in Eq. 4.

4.2 Medical Dialogue Generation Model

Fine-Tuning. Fine-tuning is an effective method for transferring the capabilities of pretrained models to downstream tasks. Therefore, this paper incorporates fine-tuning as one of the techniques to evaluate the performance of CMed-GPT. Prior studies have extensively employed fine-tuning on models such as BERT or GPT [20–22], and this paper refrains from delving further into this topic.

P-Tuning. Compared to fine-tuning, p-tuning [23, 24] necessitates a smaller amount of data, fewer model parameters, and thus, reduces the demand on GPU memory. This renders it an essential technique for expeditiously transferring pretrained models to downstream tasks.

For p-tuning, we keep the parameters φ of the pre-training fixed and add discrete prefix prompt $\text{Prefix} = \{p_1, p_2 \dots p_m\}$, before the dialogue text to obtain $Z = [\text{Prefix}; X; Y]$. The goal of the model fine-tuning is to maximize the likelihood of Y , $P_{\varphi\theta_p} = P(Y|[\text{Prefix}; X])$, where the parameter $\theta_p \ll \varphi$. Assuming that the pre-trained model has a vocabulary size of V , V_p prompt tokens, and a hidden layer dimension of H , the model’s word embedding matrix would be of size $E \in V \times H$, and the prompt’s word embedding matrix would also be of size $E_p \in V_p \times H$. Considering the input $X = X_1, X_2, X_3 \dots X_i \dots X_k$ comprising utterances $X_i = [u_1^i, u_2^i \dots u_j^i \dots u_s^i]$, where $u_j^i \in V$, and the prompt prefix $\text{Prefix} = \{p_1, p_2 \dots p_m\}$, where $p_m \in V_p$. During the training process, the prompt tokens are concatenated with the input text before forwarding it. This concatenation transforms the input X into $\text{Prefix}, X_1, X_2, X_3 \dots X_i \dots X_k$. The loss is then computed based on this modified input text. It is important to note that the model’s parameter gradients are set to false, while only the prompt parameter gradients are set to true. As a result, during the backward pass, the gradients for the model parameters are set to 0, while only the prompt parameters are updated.

Lexical and Entity Embeddings. In the medical domain, there are many specialized and less common entities (long-tail entities) that are relatively rare in general corpora. This rarity can lead to inaccurate word vector representations for these entities. By enhancing the representation of entities, it helps the model better understand the context of the conversation and improves the accuracy of the model’s responses [25]. In this paper, we represent the entity positions using one-hot vectors. Consequently, the entity information E_i is encoded as an entity vector.

$$E_i = e_1^i, \dots, e_s^i \quad (5)$$

where s represents the length of a utterance, $e_j^i = 0|1$.

In addition to one-hot encoding entity positions, we also enhance the text representation using lexical embedding. In general, entities in medical text are predominantly nouns (n), while the severity of symptoms is primarily adjectives (adj). These lexical have significant importance in medical text. Therefore, we use “jieba” for lexical tagging in the text, identifying nouns, adjectives, and verbs and assigning them numerical labels, such as nouns: 0, adjectives: 1, verbs: 2. These labels are encoded into vectors and introduced into the text vectors. The modified model structure is shown in Fig. 2. Compared to the original structure, we use a positional encoding-like approach to embed lexical and entity encodings within the input embedding. This modification is intended to enhance the model’s ability to comprehend medical text.

Assuming the original word vector representation is E_w and the position vector is E_p the input vector for the block is $E = E_w + E_p$, After adding the lexical vector E_t and the entity vector E_e :

$$E = E_w + E_p + E_t + E_e \quad (6)$$

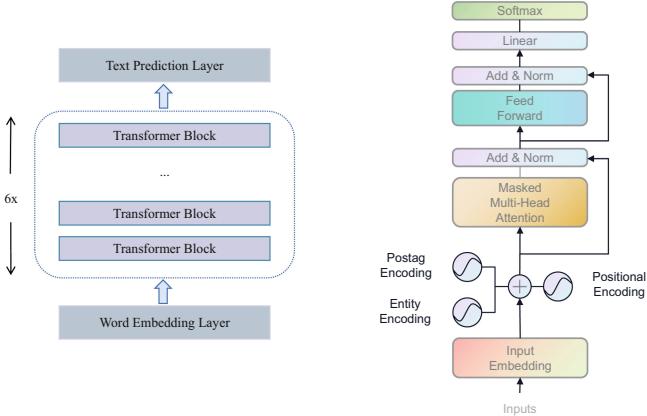


Fig. 2. Model structure for fusion entity and lexical embeddings.

5 Experiments

In this section, we present the experimental setup and related experimental results for the pre-training model CMed-GPT and fine-tuning, p-tuning.

5.1 Experimental Setting

The model utilizes all the data from Sect. 3 for pre-training, excluding CMDD and IMCS-IR, which is divided into training and test sets at a ratio of 100:1. The AdamW [26] training optimizer was employed with β_1 and β_2 values set to 0.9 and 0.95, respectively. Weight decay was set at 0.1, and gradient norm clipping was performed at 0.5. A warmup cosine scheduler was used with an initial learning rate of 0.0001, a warmup step of 2000, followed by a cosine decay to a minimum learning rate of 5e-6. The minimum learning rate was maintained after 100,000 steps and continued training until completion, totaling 3 epochs. The maximum text length was set to 512, and if the text exceeded this length, it was truncated from the end to 512. The batch size was set at 32. The entire model was trained using float16 mixed precision, which accelerates training speed while maintaining accuracy compared to float32. Lastly, the base model was trained for 20 days on a 4-card machine with NVIDIA TITAN RTX 24GB, while the large model was trained for 30 days.

To evaluate the model's performance in downstream tasks, we conducted p-tuning/fine-tuning using CMDD and IMCS-IR2 data. The dataset was partitioned into training and test sets, with a ratio of 8:2. The hyperparameters for fine-tuning remained consistent with the pretraining phase, except for a modification in the initial learning rate, set to 5e-5. The model underwent training for 6 epochs. For p-tuning, different quantities of prompt tokens {1, 25, 50, 75, 100} were employed in experimental trials.

5.2 Experimental Results

Pre-training Model. The pretrained language model for the Chinese medical domain proposed in this paper hasn't found an equivalent pretrained model. Here, we compared

it with four other models: Bert-base-chinese (see footnote 2), Chinese-GPT2-base (see footnote 3), Chinese-GPT2-large (see footnote 3), and Medbert (see footnote 4). Bert-base-chinese: A BERT model pretrained on general Chinese text data; Chinese-GPT2: A GPT model pretrained on general Chinese text data; Medbert: A BERT model pretrained on Chinese medical domain text data.

Given the nature of the task being a language modeling task, perplexity (PPL) was employed as the metric for evaluating the model’s performance. The formula for calculating PPL is as follows:

$$PPL = e^{loss}, e \approx 2.78 \quad (7)$$

The results are shown in the Table 2, and it was observed that general models, whether BERT or GPT2, perform poorly on Chinese medical text, with PPL values greater than 20. However, Medbert, which has been fine-tuned on medical text, outperforms general BERT models, further emphasizing the importance of continued training on medical domain text.

Table 2. PPL of CMed-GPT compared to other models in the medical domain test set

Model	Parameters	Test PPL	Size
Bert-base-chinese ²	6 layers, 12 head, 768 hidden size	22.63	110M
Chinese-GPT2-base ³	6 layers, 12 head, 768 hidden size	20.82	110M
Chinese-GPT2-large (see footnote 3)	12 layers, 12 head, 768 hidden size	18.64	200M
Medbert ⁴	6 layers, 12 head, 768 hidden size	9.37	110M
CMed-GPT-base(Ours)	6 layers, 12 head, 768 hidden size	8.64	110M
CMed-GPT-Large(Ours)	12 layers, 12 head, 768 hidden size	8.01	200M

In addition, CMed-GPT-base also outperforms an equivalently sized Medbert. This can be attributed to the inherent suitability of unidirectional language models for generation tasks. Furthermore, increasing the model size (from 6 to 12 layers) leads to a noticeable improvement in performance (Fig. 3).

Fine-Tuning/P-Tuning. In the fine-tuning and p-tuning phases, we utilized CMDD dataset and MCS-IR dataset. The results are shown in Table 3, where CMed-GPT-base is the pre-trained model, KB-FT-CMed-GPT is the CMed-GPT-base post-fine-tuning model and KB-PT-CMed-GPT is the CMed-GPT-base post-p-tuning model. It was observed that after fine-tuning/p-tuning on the downstream dialog task datasets, the PPL decreased from above 8 to below 8, indicating an improvement of approximately 10%. Additionally, p-tuning achieves comparable or slightly superior results compared to fine-tuning. This suggests that the prefix prompts used in p-tuning had a significant

² <https://huggingface.co/bert-base-chinese>.

³ <https://github.com/Morizeyao/GPT2-Chinese>.

⁴ <https://github.com/trueto/medbert>.

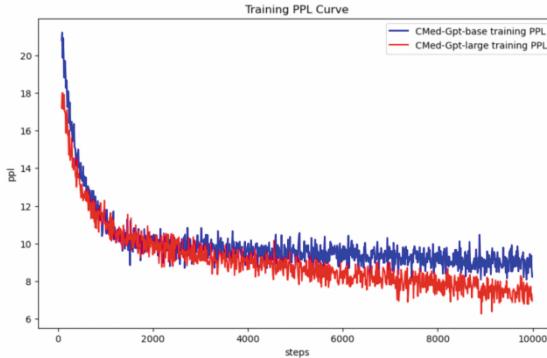


Fig. 3. Train PPL curves for CMed-Gpt-base/large

impact. Compared to fine-tuning, p-tuning required fewer parameters and less time for training.

Table 3. CMed-GPT-base fine-tuning/p-tuning results on CMDD and IMCS-IR.

Dataset	CMed-GPT-base	KB-FT-CMed-GPT	KB-PT-CMed-GPT
CMDD	8.44	7.68	7.58
IMCS-IR	8.79	7.75	7.62

In the p-tuning phase, we experimented with different numbers of prompt tokens to assess their impact on the results. For this experiment, we only used the CMDD dataset, and the results are as follows:

Table 4. PPL of the model with different prompt token.

Dataset	1	25	50	75	100
CMDD	8.32	7.85	7.68	7.69	7.93

Table 5. PPL of the model after lexical/entity embedding for different datasets.

Dataset	P-tuning	Lexical	Entity	Lexical and entity	Entity splicing method ⁵
CMDD	7.58	7.5	7.46	7.35	7.89
IMCS-IR	7.62	7.58	7.53	7.43	7.93

⁵ <https://arxiv.org/abs/2212.06049>.

Based on the Table 4, it is evident that the best result is achieved at $v = 50$, suggesting that the addition of prompts does not always lead to performance improvements for the model. This phenomenon is likely influenced by the volume of tuning data. When there is a large amount of data for tuning, more parameters are needed to fine-tune and fit the data. However, when the tuning data is limited, an appropriate number of prompt tokens can improve model performance (Table 5).

Entity-Aware Medical Dialogue Generation. The introduction of entity and lexical enhanced GPT models has been widely discussed in other papers as well. However, the approach of incorporating entities often involves identifying them and discretely appending them to the input text⁵. This results in discrete entity vectors that cannot uniformly represent each token. In contrast to other papers, we integrate entity and lexical vectors with the text embedding and positional embedding in an encoded manner, similar to the transformer input format. The uniformly spliced matrix is fed into our model after p-tuning. Experimental results indicate that incorporating entity and lexical information further improves the model’s performance.

The inclusion of lexical and entity vectors in medical text has strengthened the representation of certain important words in the text. This makes it easier for the model to generate accurate responses during subsequent conversations.

6 Conclusion

In this paper, we first conducted a comprehensive compilation and analysis of existing Chinese medical dialogue datasets and medical book data. Subsequently, using this valuable collection of Chinese medical text data, we successfully pretrained a powerful Chinese medical pre-trained language model, named CMed-GPT. This model not only possesses extensive medical knowledge but also has the ability to understand and generate medical dialogue text.

To address the common long-tail entity problem in medical text, we propose an innovative approach that effectively integrates lexical and entity information during model training. Additionally, we devised discrete pseudo-token prefixes, which were appended to the text data. Through a series of rigorous experiments, we have concluded that our proposed CMed-GPT, including both the base and large versions, exhibits outstanding performance in the medical domain.

Furthermore, our CMed-GPT model can be easily applied to various downstream tasks in the medical field. By performing simple p-tuning, which involves adjusting a small number of parameters, the model’s performance on specific tasks can be significantly improved. This provides a powerful and efficient tool for medical information processing.

To summarize, this study not only successfully constructs a pre-trained language model CMed-GPT for the Chinese medical domain, but also proposes an effective method for entity information incorporation and demonstrates its excellent performance in medical text processing tasks. In the future, we will continue to improve the model and explore more application areas to better meet the needs of medical information processing and contribute to advancements in the medical field.

References

1. He, X., et al.: MedDialog: Two large-scale medical dialogue datasets (2020). arXiv preprint [arXiv:2004.03329](https://arxiv.org/abs/2004.03329)
2. Liu, W., Tang, J., Qin, J., Xu, L., Liang, X.: MedDG: A large-scale medical consultation dataset for building medical dialogue system (2020). arXiv preprint [arXiv:2010.07497](https://arxiv.org/abs/2010.07497)
3. Li, D., et al.: Semi-supervised variational reasoning for medical dialogue generation. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 544–554. Association for Computing Machinery, New York (2021)
4. Wei, Z., et al.: Task-oriented dialogue system for automatic Diagnosis. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, pp. 201–207. Association for Computational Linguistics, Melbourne (2018)
5. Xu, L., Zhou Q., Gong, K., Liang, X., Tang, J., Lin, L.: End-to-end knowledge-routed relational dialogue system for automatic diagnosis. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 7346–7353. Association for the Advancement of Artificial Intelligence (2019)
6. Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.R.: GLUE: a multi-task benchmark and analysis platform for natural language understanding. In: Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, pp. 353–355. Association for Computational Linguistics, Brussels (2018)
7. Peng, Y., Yan, S., Lu, Z.: Transfer learning in biomedical natural language processing: an evaluation of BERT and ELMo on ten benchmarking datasets. In: Proceedings of the 18th BioNLP Workshop and Shared Task, pp.58–65. Association for Computational Linguistics, Florence (2019)
8. Gu, Y., et al.: Domain-specific language model pretraining for biomedical natural language processing. ACM Trans. Comput. Healthc. **3**(1), 1–23 (2022)
9. Lee, J., et al.: BioBERT: a pre-trained biomedical language representation model for biomedical text mining. J. Leukoc. Biol. **36**(4), 1234–1240 (2020)
10. Beltagy, I., Lo, K., Cohan, A.: SciBERT: a pretrained language model for scientific text. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pp.3615–3620. Association for Computational Linguistics, Hong Kong (2019)
11. Roitero, K., et al.: DiLBERT: cheap embeddings for disease related medical NLP. IEEE Access **9**(9), 2169–3536 (2021)
12. Liu, Y., et al.: RoBERTa: A robustly optimized BERT pretraining approach (2019). arXiv preprint [arXiv:1907.11692](https://arxiv.org/abs/1907.11692)
13. Zhang, N., Jia, Q., Yin, K., Dong, L., Gao, F., Hua, N.: Conceptualized Representation Learning for Chinese Biomedical Text Mining (2020). arXiv preprint [arXiv:2008.10813](https://arxiv.org/abs/2008.10813)
14. Zhang, T., Cai, Z., Wang, C., Qiu, M., Yang, B., He, X.: SMedBERT: a knowledge-enhanced pre-trained language model with structured semantics for medical text mining. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pp.5882–5893. Association for Computational Linguistics (2021)
15. He, B., et al.: BERT-MK: integrating graph contextualized knowledge into pre-trained language models. In: Findings of the Association for Computational Linguistics: EMNLP 2020, pp.2281–2290. Association for Computational Linguistics (2020)
16. Radford, A., et al.: Language models are unsupervised multitask learners. GPT-2 OpenAI blog (2019)

17. Brown, T.B., et al.: Language models are few-shot learners. In: Proceedings of the 34th International Conference on Neural Information Processing Systems, pp.1877–1901. Curran Associates Inc, Red Hook (2020)
18. Papanikolaou, Y., Pierleoni, A.: DARE: Data augmented relation extraction with GPT-2 (2020). arXiv preprint [arXiv:2004.13845](https://arxiv.org/abs/2004.13845)
19. Vaswani, A., et al.: Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp.6000–6010. Curran Associates Inc., Red Hook (2017)
20. Loshchilov, I., Hutter, H.: Decoupled weight decay regularization. In: International Conference on Learning Representations (2017)
21. Peng, X., et al.: Fine-Tuning a transformer-based language model to avoid generating non-normative text (2020). arXiv preprint [arXiv:2001.08764v1](https://arxiv.org/abs/2001.08764v1)
22. Davier, M.V., Training optimus prime, M.D.: Generating medical certification items by Fine-Tuning OpenAI’s gpt2 transformer model (2019). arXiv preprint [arXiv:1908.08594](https://arxiv.org/abs/1908.08594)
23. Tsai, D.C.L., et al.: Short answer questions generation by Fine-Tuning BERT and GPT-2. In: 29th International Conference on Computers in Education Conference, pp. 509–515. Asia-Pacific Society for Computers in Education (2021)
24. Li, X., Liang, P.: Prefix-Tuning: optimizing continuous prompts for generation. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pp.4582–4597. Association for Computational Linguistics (2021)
25. Lester, B., et al.: The power of scale for parameter-efficient prompt tuning. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp.3045–3059. Association for Computational Linguistics (2021)
26. Cui, L., et al.: Knowledge enhanced fine-tuning for better handling unseen entities in dialogue generation. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp.2328–2337. Association for Computational Linguistics (2021)



MvRNA: A New Multi-view Deep Neural Network for Predicting Parkinson’s Disease

Lin Chen, Yuxin Zhou, Xiaobo Zhang^(✉), Zhehao Zhang, and Hailong Zheng

School of Computing and Artificial Intelligence, Southwest Jiaotong University,
Chengdu, China

{chenlynn,yuzs,zhzhang,zhl}@my.swjtu.edu.cn, zhangxb@swjtu.edu.cn

Abstract. Magnetic Resonance Imaging (MRI) is a critical medical diagnostic tool that assists experts in precisely identifying lesions. However, due to its high-dimensional nature, it requires substantial storage resources. If only one MRI slice were to be used, a significant amount of information might be lost. To address these issues, we propose segmenting 3D MRI data and training these slices separately. We propose a new Multi-view Learning neural network based on ResNet and an Attention mechanism, called MvRNA. ResNet18 is selected as the backbone network, and the Squeeze-and-Excitation network is applied between blocks to extract features from slices. Additionally, we propose a new BWH (Basic Block with Hybrid Dilated Convolution) module to capture a broader range of receptive fields, thus acquiring additional spatial features. We obtained data from Parkinson’s Progression Markers Initiative (PPMI) and applied our method to distinguish between Healthy Control, Prodromal, and Parkinson’s disease patients. The experimental results demonstrate that our method achieved an accuracy of 81.84%.

Keywords: Parkinson’s disease (PD) · Multi-view Learning · 3D MRI · ResNet · Squeeze-and-Excitation Attention · Hybrid Dilated Convolution

1 Introduction

Parkinson’s disease (PD) is a prevalent neurodegenerative condition. Its primary pathological feature involves the degeneration and demise of dopaminergic neurons in the substantia nigra of the midbrain, resulting in a notable reduction in dopamine content within the striatum. The intricate nature of PD leads to over 90% of patients enduring a period exceeding a year before receiving an official diagnosis. This diagnostic delay brings significant harm to patients, emphasizing the critical importance of timely detection and treatment. In recent years, with the advancement of deep learning, computer-aided diagnoses [1] have also

L. Chen and Y. Zhou contribute equally to this work.

made significant progress. The combination of machine learning techniques and medical knowledge offers a promising direction for neurological disease research.

Numerous novel neural network algorithms [2,3] have been proposed for classifying medical subjects in various medical scenarios and have achieved commendable results. However, as research has evolved, a critical issue has garnered attention: a single two-dimensional medical image slice loses a significant amount of contextual information, which is detrimental to volume-level medical research. To model three-dimensional spaces and acquire more complex feature representations, 3D convolutional neural networks (CNNs) have been introduced [4]. Nevertheless, it comes with new limitations, such as increased requirements for computing resources and time due to more intricate structures.

As a distinct learning paradigm, multi-view learning [5] involves acquiring data characteristics from various perspectives. These unique viewpoints offer complementary and consistent information. Specifically, when data is missing from one perspective, other views can act as supplements to alleviate the problem of feature loss [6]. To simultaneously reduce the heavy usage of computational resources and capture as much multi-dimensional information from the images as possible, inspired by multi-view learning, we position 3D MRI images within a three-dimensional coordinate system. We slice the original images along the x, y, and z axes separately and achieve our goal through a split-and-merge strategy. Slices from different axes are sent separately to the network for training, and we concatenate the obtained features to enable the prediction of PD. To enhance the model's sensitivity to the data, we integrate Squeeze-and-Excitation (SE) [7] blocks to improve the identification of channel features. In more detail, this process captures important features between slices. Furthermore, we have made improvements to the basic blocks and enhanced the spatial feature extraction capabilities by introducing hybrid dilated convolutions [8].

In summary, the main contributions of this work are summarized as follows:

- We propose a split-merge strategy for 3D MRI data. This strategy preserves all data information while minimizing computational resource usage.
- MvRNA, a new Multi-view learning neural network based on ResNet and Attention mechanism is introduced. This new framework can learn meaningful feature representations that are less noise-sensitive.
- We improved the basic block and introduced a basic block with hybrid dilated convolutions (BWH), which enhances the perception of image spatial features.
- We conducted a series of processing steps on the PPMI dataset [9], and numerous experiments have demonstrated that our method significantly enhances the accuracy of PD prediction.

2 Related Work

In this section, we will introduce Convolutional Neural Networks, the concept of Multi-view Learning, Attention Mechanisms with a particular focus on the Squeeze-and-Excitation network, and Hybrid Dilated Convolutions.

Convolutional Neural Networks (CNNs). With the development of machine learning, more and more CNNs have been proposed. AlexNet [10], VGG [11], and GoogleNet [12] have achieved good results on image prediction tasks. Utilizing CNNs, [13,14] have effectively employed 2D slices from 3D images to classify Healthy Control (HC) and PD. In order to train the deep network more easily, ResNet [15] was proposed, which introduced residual connections to solve the problem of gradient disappearance. Building upon this concept, an efficient ResNet-50-based CNN model for pneumonia prediction using medical images was proposed [16]. Res2Net [17] was developed by constructing hierarchical residual connections within a single residual block, thereby achieving multi-scale feature representation at a fine-grained level and expanding the receptive field of each network layer.

Multi-view Learning. Multi-view learning [5] is a machine learning paradigm that leverages information from multiple perspectives or data sources to enhance the generalization ability of models. Given that different views often contain complementary information, merging them results in a more accurate feature representation. In the context of medical data, Masked Multi-view with Swin Transformers (SwinMM) [18] has been developed to extract hidden multi-view information from 3D medical datasets. Additionally, the introduction of Multi-scale attention (MSA) [19] has expanded the horizons of multi-view imaging integration, enabling robust medical image segmentation by collecting global correspondences of multi-scale feature representations. A feature representation method aimed at enhancing multimodal MRI data through the integration of multi-view information was proposed [20], a multiple kernel learning algorithm was employed to combine multiple features for enhancing prediction performance in tasks related to mild cognitive impairment prediction.

Attention Mechanism. Attention mechanisms have gained widespread application in the domain of deep neural networks, empowering networks to dynamically focus on crucial elements while disregarding less relevant details. This innovative approach has shown success in PD prediction tasks, as evidenced by [5,21]. A novel network architecture [22] combined BAM [23] and SA [24] blocks with a 3D ResNet18 backbone network to achieve PD prediction, highlighting the potential of integrating attention mechanisms for enhanced model performance. One noteworthy attention mechanism that prioritizes channel importance is the Squeeze-and-Excitation Network (SENet) [7], consisting of two key components: Squeeze and Excitation. SENet successfully enhances model performance with minimal computational cost. Additionally, [25] employed SENet as a bridge between the convolutional layer and the transition layer to design a novel brain tumor prediction network model based on DenseNet-201. [26] introduced the 3D SE-DenseNet for the prediction of hepatocellular carcinoma grading, utilizing enhanced clinical MRI data from two different clinical centers. The combination of SENet with CNN, as demonstrated by [27,28], has significantly improved prediction accuracy.

Hybrid Dilated Convolution (HDC). As we all know, with an increase in the number of network layers, deeper features are captured, but concurrently, the resolution decreases. Dilated convolution [29] preserves the resolution by using zero-padding in the convolution kernel, but it can result in a gridding effect. To tackle this issue, hybrid dilated convolution (HDC) [8] was introduced. By overlapping dilated convolutions with different rates, it becomes possible to capture more contextual information while retaining image resolution. [30] proposed a neural network with the HDC for pixel-level crack detection on concrete bridges, which effectively improved the detection accuracy of blurred cracks. [31] improved the decoding process using HDC, obtaining more receptive fields and obtaining higher Dice scores.

3 Methods

In this section, we introduce the MvRNA framework. First of all, we position 3D MRI in a three-dimensional coordinate system and slice the images along the x, y, and z axes to obtain a multi-view dataset. Subsequently, we employ ResNet18 as the basic network. Combined with HDC, we introduce a novel BWH feature extraction module. This module not only preserves medical image resolution but also expands the receptive field. Additionally, we incorporate the SE block into our framework, enhancing feature representation capability. Finally, we concatenate features of varying dimensions to achieve precise PD prediction. The structure of MvRNA is shown in Fig. 1.

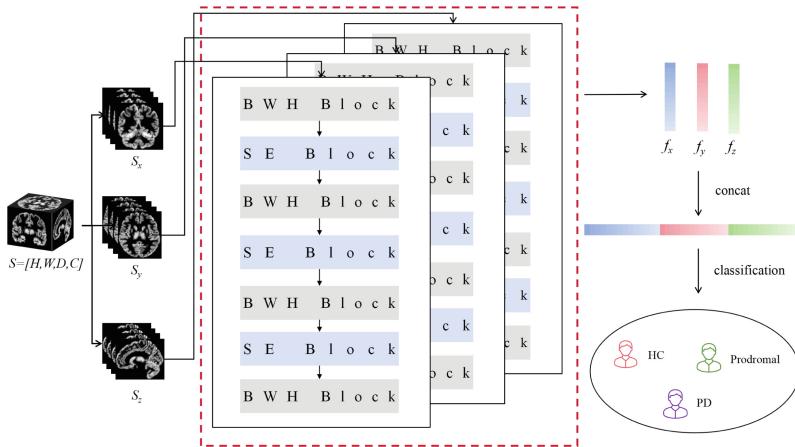


Fig. 1. The overall structure of MvRNA framework.

3.1 Data Representation Based on Multiple Views

While 3D CNN can perform feature extraction while preserving the original data structure, it involves a substantial computational burden. Hence, we propose a split-merge strategy to address this issue. Splitting involves dividing the original 3D MRI into 2D slice sets from different angles, and these sets form the multi-view dataset. Each preprocessed MRI data is represented in the [x, y, z] format. Therefore, we position the medical image within a three-dimensional coordinate system and perform slicing operations along the x-axis, y-axis, and z-axis, respectively.

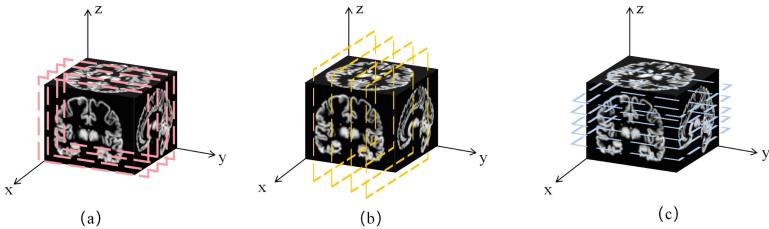


Fig. 2. Schematic diagram of dimensional splitting of 3D data along the x-axis(a), y-axis(b) and z-axis(c) respectively.

As shown in Fig. 2, the dotted box illustrates the partition plane. We define a single original image as $S = (H, W, D, C)$, where H, W, D, and C respectively represent the height, width, depth, and channels ($C = 1$). The split-merge strategy is depicted in Eq. 1.

$$S_i = \{(H', W', C') \mid C' = \text{seg}(i), i \in \{x, y, z\}\} \quad (1)$$

$$S = S_x \cup S_y \cup S_z \quad (2)$$

where S_i represents the new data obtained after slicing along a single axis i , H' , W' , and C' are the updated height, width, and number of channels respectively, and $\text{seg}(\cdot)$ represents segmentation.

Taking the x-axis division as an example (Fig. 2(a)), the original 3D data has dimensions of [113, 137, 113, 1]. Following the x-axis segmentation, we obtain S_1 , a 2D dataset with 113 channels and dimensions of 113 in height and 137 in width, which can be expressed as $S_x = [113, 137, 113]$. The segmentation for the y-axis (Fig. 2(b)) and z-axis (Fig. 2(c)) follows the same process as for the x-axis. To preserve information integrity and avoid potential issues arising from dimensionality splitting, datasets from all three perspectives are concatenated to create a multi-view representation dataset, referred to as ‘merge’, refer to Eq. 2.

3.2 ResNet18 with BWH

ResNet18 preserves shallow features in deeper network layers through the use of skip connections. As convolutional layers are added, deeper features are captured, but the resolution of the image decreases. To address this issue, dilated convolution was introduced. However, dilated convolution can introduce a gridding effect. To mitigate this, [8] proposed hybrid dilated convolution. This method involves the continuous use of multiple dilation rates that meet specific conditions for dilated convolution, thereby achieving full pixel utilization. Inspired by this, we introduced a novel feature extraction module: the Basic block with Hybrid dilated convolution (BWH). We embed the HDC block between two basic blocks to extract additional spatial features before each downsampling step, particularly without significantly increasing model parameters or reducing image resolution. The specific structure is illustrated in Fig. 3. Assuming the size of the convolution kernel is k , the feature f_h after applying HDC can be obtained using Eq. 3.

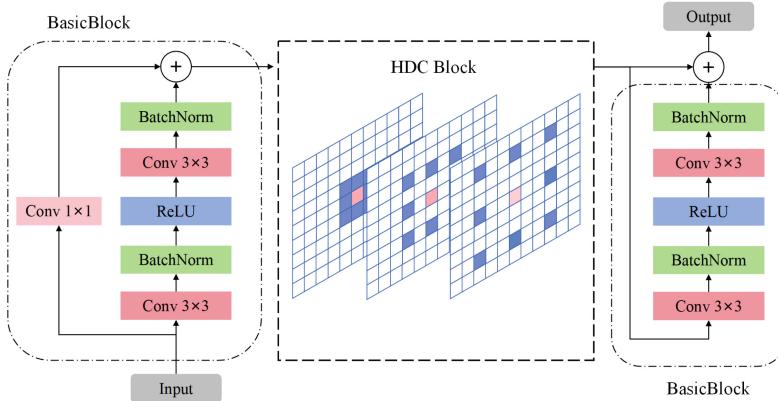


Fig. 3. Structure diagram of BasicBlock with HDC block (BWH).

$$f_h(x; k, r) = \frac{1}{k} \sum_{i=1}^k h(x, r_i) \quad (3)$$

where x represents the feature vector of the input, $r=[r_1, \dots, r_k]$ is the expansion rate list and $h(\cdot)$ represents the operation of HDC.

The features obtained through the basic block can be calculated using Eq. 4.

$$f_b(x; W_i) = g(W_i \cdot x) + x \quad (4)$$

where W_i represents the weight parameters of the convolution, and $g(\cdot)$ represents the convolution operation and activation function. Finally, we can obtain

the feature representation f_{bwh} after the BWH block by using Eq. 5.

$$f_{bwh} = f_b(f_h(f_b(x))) \quad (5)$$

3.3 Channel Attention Implemented Using SENet

Since we sliced a 3D subject into a 2D slide set, the channel dimension of the image is converted from the original 1D to the number of slices generated by the partition plane.

As described in Sect. 3.1, the dimension of the image is transformed from 1 dimension to 113 dimensions. Therefore, it is crucial to extract features across channels. SENet is an attention mechanism specifically designed for channels, allowing it to adaptively focus on features in channels beneficial for classification while ignoring unimportant channels. As a result, we embed the SE block in the downsampling process, as shown in Fig. 1, and combine it with the proposed BWH to achieve efficient feature extraction from both spatial and channel perspectives.

The feature extraction process described above is shown in Algorithm 1.

Algorithm 1. MvRNA Feature Extraction Algorithm.

Require: 3D MRI X with dimension [H, W, D, C]
where H:height, W:width, D:depth, C:channel

Ensure: Extracted features f .

- 1: **for** dimension $\in X$ **do**:
- 2: Transform X , convert dimension to channel.
- 3: **for** single-view $\in X$ **do**:
- 4: Convolution, Normalization;
- 5: ReLU, Max Pooling;
- 6: **while** n < num(block) **do**:
- 7: **while** n < num(BasicBlock) **do**:
- 8: Extract features with BWH block;
- 9: **end while**
- 10: Extract features with SE block;
- 11: **end while**
- 12: **end for**
- 13: **end for**
- 14: **return** f_i ;
- 15: Concatenate feature vectors $f_i \rightarrow f$;
- 16: **return** f

4 Experiments and Results

4.1 Dataset

The T1-weighted MRI dataset is acquired from the PPMI database (www.ppmi-info.org/data) for this study. We collect a total of 348 instances from the website

to create the dataset. The dataset comprises 124 instances from healthy control, 77 from individuals with prodromal Parkinson’s disease, and 147 from diagnosed Parkinson’s disease patients.

Each 3D MRI data includes dimensions of height, width, and depth and is formatted as NiFTI files. Since the data come from different research centers, they need to be normalized into a unified space, image registration operation is performed on the raw data using MATLAB. All the data are normalized to the MNIPD25-T1MPRAGE-1 mm template space [32]. After image registration, each image can be represented as [113, 137, 113, 1], where each parameter corresponds to height, width, depth, and channel, respectively. We divide all MRI images downloaded from PPMI into training set, validation set and test set and divide them according to the ratio of 8:1:1. Each MRI data instance is transformed along different axes to generate various view sets. Specifically, segmenting 3D MRI data along the x-axis results in sagittal (SAG) channels with 113 channels, along the y-axis results in coronal (COR) channels with 137 channels, and along the z-axis results in an axial (AXI) channel set with 113 channels. Comprehensive statistical results are provided in Table 1 to describe this data. It’s important to note that the channel data listed in Table 1 only pertains to a single MRI data instance.

Table 1. Segmentation of a single 3D MRI.

Class	<i>SAG Channel</i>	<i>COR Channel</i>	<i>AXI Channel</i>
HC	113	137	113
Prodromal	113	137	113
PD	113	137	113

4.2 Experimental Settings

The experiments were conducted using Python on a GPU server (equipped with an Intel(R) Xeon CPU E5-2620 V4 @2.10 GHz, 128 GB RAM, and an NVIDIA TITAN Xp GPU with 12 GB memory). In our experiments, Experiments are conducted in 20 iterations, and due to the complexity of the data, the batch size is set to 8. Furthermore, the most effective number of epochs is 100, and the best learning rate is 1e-6. Additionally, we utilize the Adam optimizer to expedite the model’s convergence. We set the weight decay to 5e-4, a value determined through experimentation. Additionally, we selected Precision, Recall, F1-score and Accuracy as model evaluation metrics.

4.3 Experimental Results and Analysis

In this section, we introduce the comparison models and the comparison results between each model and our proposed method under the same experimental conditions.

Our model is compared with 3D VGG16, 3D DenseNet121, and 3D ResNet18 equipped with BAM and SA. The models share the same parameter settings; however, due to the complexity of the data and limitations in computational resources, the batch size for 3D VGG16 is set to 4. The changes in training loss and validation loss for all models throughout the entire training process are shown in Fig. 4 and Fig. 5, respectively.

Table 2 displays the comparative performance of our proposed model and the baseline, utilizing the same dataset. The figures presented in the Table 2 represent the average values of the performance metrics and the best performance indicators have been bolded. Our experimental results yielded an accuracy of 81.84%, showcasing the best prediction performance when compared to the baseline model. Moreover, our method attains a precision of 0.85, recall of 0.79, and F1-score of 0.81, all of which represent the best performance.

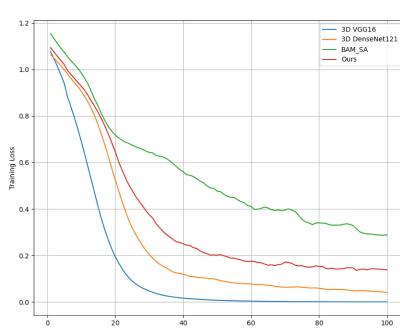


Fig. 4. The training loss of all models.

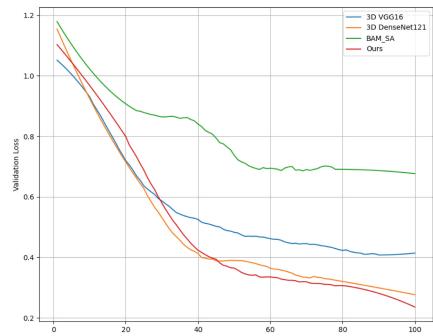


Fig. 5. The validation loss of all models.

Table 2. Comparison of results between different models.

Model	Precision	Recall	F-score	Accuracy
3D VGG16	0.83	0.74	0.74	78.42%
3D DenseNet121	0.83	0.76	0.76	77.88%
3D ResNet18 + BAM + SA	0.82	0.79	0.79	81.06%
MvRNA(ours)	0.85	0.79	0.81	81.84%

4.4 Ablation Experiment

To verify the effectiveness of adding the HDC (Hierarchical Dilated Convolution) module for the task of predicting Parkinson’s disease, an ablation experiment was conducted. The results, as shown in Table 3, indicate that after incorporating the HDC module, the model achieved better results across various metrics due to the increased range of the receptive field. This demonstrates the efficacy of the HDC module in enhancing the performance of Parkinson’s disease prediction algorithms.

Table 3. Comparison of results between different models.

Model	<i>Precision</i>	<i>Recall</i>	<i>F-score</i>	<i>Accuracy</i>
MvRNA(without HDC)	0.79	0.75	0.75	76.76%
MvRNA	0.85	0.79	0.81	81.84%

5 Conclusion

We present a novel deep neural network for predicting PD. Firstly, we perform axial segmentation of 3D MRI images to generate a multi-view dataset. Secondly, we employ ResNet18 as the base network and introduce the novel BWH feature extraction module in combination with HDC. This module not only preserves the resolution of medical images but also extends the receptive field. Furthermore, we integrate the SE block into our framework to enhance feature representation. Finally, we combine these features to achieve accurate PD prediction. The results of our experiments demonstrate that the novel MvRNA framework maximizes the utilization of all feature information in 3D MRI data while reducing computational load. This framework significantly enhances the accuracy of Parkinson’s disease recognition. It’s important to note that, unlike other studies, our approach addresses the multi-categorization problem associated with Parkinson’s disease.

Acknowledgements. This work was supported by the National Natural Science Foundation of China (No. 61976247), the Key Research and Development Program in Sichuan Province of China (No. 2023YFS0404), the Central guiding Local Science and Technology Development Fund (No. 23ZYZZTS0189), the Fundamental Research Funds for the Central Universities (Nos. 2682022KJ045 and 2682023ZTPY081), and the Open Research Fund Program of Data Recovery Key Laboratory of Sichuan Province (No. DRN2203).

References

1. Dmitriev, K., Marino, J., Baker, K., Kaufman, A.E.: Visual analytics of a computer-aided diagnosis system for pancreatic lesions. *IEEE Trans. Visual Comput. Graphics* **27**(3), 2174–2185 (2021). <https://doi.org/10.1109/TVCG.2019.2947037>
2. Han, L., Kamdar, M.R.: MRI to MGMT: predicting methylation status in glioblastoma patients using convolutional recurrent neural networks. In: Pacific Symposium on Biocomputing Pacific Symposium on Biocomputing, vol. 23, p. 331 (2018)
3. Tang, Z., Xu, Y., Jin, L., Aibaidula, A., Shen, D.: Deep learning of imaging phenotype and genotype for predicting overall survival time of glioblastoma patients. *IEEE Trans. Med. Imaging* **PP**(99), 1 (2020)
4. Thuseethan, S., Rajasegarar, S., Yearwood, J.: Detecting micro-expression intensity changes from videos based on hybrid deep CNN. In: Yang, Q., Zhou, Z.-H., Gong, Z., Zhang, M.-L., Huang, S.-J. (eds.) PAKDD 2019. LNCS (LNAI), vol. 11441, pp. 387–399. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-16142-2_30
5. Xu, C., Tao, D., Xu, C.: A survey on multi-view learning. *Computer Science* (2013)
6. Zhou, W., Wang, H., Yang, Y.: Consensus Graph Learning for Incomplete Multi-view Clustering. In: Yang, Q., Zhou, Z.-H., Gong, Z., Zhang, M.-L., Huang, S.-J. (eds.) PAKDD 2019. LNCS (LNAI), vol. 11439, pp. 529–540. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-16148-4_41
7. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7132–7141 (2018). <https://doi.org/10.1109/CVPR.2018.00745>
8. Wang, P., et al.: Understanding convolution for semantic segmentation. In: 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1451–1460 (2018). <https://doi.org/10.1109/WACV.2018.00163>
9. Marek, K., Jennings, D., Lasch, S., Siderowf, A., Taylor, P.: The Parkinson progression marker initiative (PPMI). *Prog. Neurobiol.* **95**(4), 629–635 (2011)
10. Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, vol. 25, no. 2 (2012)
11. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv e-prints (2014)
12. Szegedy, C., et al.: Going deeper with convolutions. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–9 (2015). <https://doi.org/10.1109/CVPR.2015.7298594>
13. Zhu, S.: Early diagnosis of Parkinsons disease by analyzing magnetic resonance imaging brain scans and patient characteristics (2022)
14. Erdaş, Ç.B., Sümer, E.: A deep learning method to detect Parkinson's disease from MRI slices. *SN Comput. Sci.* **3**(2), 1–7 (2022). <https://doi.org/10.1007/s42979-022-01018-y>
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778 (2016). <https://doi.org/10.1109/CVPR.2016.90>
16. Chhabra, M., Kumar, R.: An efficient ResNet-50 based intelligent deep learning model to predict pneumonia from medical images. In: 2022 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS), pp. 1714–1721 (2022). <https://doi.org/10.1109/ICSCDS53736.2022.9760995>

17. Gao, S.H., Cheng, M.M., Zhao, K., Zhang, X.Y., Yang, M.H., Torr, P.: Res2Net: a new multi-scale backbone architecture. *IEEE Trans. Pattern Anal. Mach. Intell.* **43**(2), 652–662 (2021). <https://doi.org/10.1109/TPAMI.2019.2938758>
18. Wang, Y., et al.: SwinMM: masked Multi-view with Swin transformers for 3D medical image segmentation. In: Greenspan, H., et al. (ed.) MICCAI 2023. MICCAI 2023. LNCS, vol. 14222. Springer, Cham. (2023). https://doi.org/10.1007/978-3-031-43898-1_47
19. Liu, D., Gao, Y., Zhangli, Q., Yan, Z., Zhou, M., Metaxas, D.: Transfusion: multi-view divergent fusion for medical image segmentation with transformers. In: Wang, L., Dou, Q., Fletcher, P.T., Speidel, S., Li, S. (eds.) MICCAI 2022. LNCS, vol. 13435. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-16443-9_47
20. Liu, J., Pan, Y., Wu, F.X., Wang, J.: Enhancing the feature representation of multi-modal MRI data by combining multi-view information for MCI classification. *Neurocomputing* **400**, 322–332 (2020)
21. Xue, Z., Zhang, T., Lin, L.: Progress prediction of Parkinson’s disease based on graph wavelet transform and attention weighted random forest. *Expert Syst. Appl.* **203**, 117483 (2022)
22. Zhang, Y., Lei, H., Huang, Z., Li, Z., Liu, C.M., Lei, B.: Parkinson’s disease classification with self-supervised learning and attention mechanism. In: 2022 26th International Conference on Pattern Recognition (ICPR), pp. 4601–4607 (2022). <https://doi.org/10.1109/ICPR56361.2022.9956213>
23. Park, J., Woo, S., Lee, J.Y., Kweon, I.S.: BAM: Bottleneck attention module (2018)
24. Zhang, Q.L., Yang, Y.B.: SA-Net: shuffle attention for deep convolutional neural networks. In: ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2235–2239 (2021). <https://doi.org/10.1109/ICASSP39728.2021.9414568>
25. Liu, M., Yang, J.: Image classification of brain tumor based on channel attention mechanism. *J. Phys: Conf. Ser.* **2035**(1), 012029 (2021). <https://doi.org/10.1088/1742-6596/2035/1/012029>
26. Zhou, Q., et al.: Grading of hepatocellular carcinoma using 3d SE-DenseNet in dynamic enhanced MR images. *Comput. Biol. Med.* **107**, 47–57 (2019). <https://doi.org/10.1016/j.combiomed.2019.01.026>
27. Linqi, J., Chunyu, N., Jingyang, L.: Glioma classification framework based on SE-ResNeXt network and its optimization. *IET Image Processing* **2**(16), 596–605 (2022)
28. Luo, M., et al.: A multi-granularity convolutional neural network model with temporal information and attention mechanism for efficient diabetes medical cost prediction. *Comput. Biol. Med.* **151**, 106246 (2022). <https://doi.org/10.1016/j.combiomed.2022.106246>
29. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. In: ICLR (2016)
30. Jiang, W., Liu, M., Peng, Y., Wu, L., Wang, Y.: HDCB-Net: a neural network with the hybrid dilated convolution for pixel-level crack detection on concrete bridges. *IEEE Trans. Industr. Inf.* **17**(8), 5485–5494 (2021). <https://doi.org/10.1109/TII.2020.3033170>
31. Zhao, X., et al.: D2A U-NET: automatic segmentation of COVID-19 CT slices based on dual attention and hybrid dilated convolution. *Comput. Biol. Med.* **135**, 104526 (2021)
32. Xiao, Y., Fonov, V., Chakravarty, M.M., Beriault, S., Collins, D.L.: A dataset of multi-contrast population-averaged brain MRI atlases of a Parkinsons disease cohort. *Data Brief* **12**(C), 370–379 (2017)



Path-Aware Cross-Attention Network for Question Answering

Ziye Luo¹, Ying Xiong¹, and Buzhou Tang^{1,2(✉)}

¹ Department of Computer Science, Harbin Institute of Technology, Shenzhen, China
tangbuzhou@gmail.com

² Peng Cheng Laboratory, Shenzhen, China

Abstract. Reasoning is an essential ability in QA systems, and the integration of this ability into QA systems has been the subject of considerable research. A prevalent strategy involves incorporating domain knowledge graphs using Graph Neural Networks (GNNs) to augment the performance of pre-trained language models. However, this approach primarily focuses on individual nodes and fails to leverage the extensive relational information present within the graph fully. In this paper, we present a novel model called Path-Aware Cross-Attention Network (PCN), which incorporates meta-paths containing relational information into the model. The PCN features a multi-layered, bidirectional cross-attention mechanism that facilitates information exchange between the textual representation and the path representation at each layer. By integrating rich inference information into the language model and contextual semantic information into the path representation, this mechanism enhances the overall effectiveness of the model. Furthermore, we incorporate a self-learning mechanism for path scoring, enabling weighted evaluation. The performance of our model is assessed across three benchmark datasets, covering the domains of commonsense question answering (CommonsenseQA, OpenbookQA) and medical question answering (MedQA-USMLE). The experimental results validate the efficacy of our proposed model.

Keywords: Meta Path · Cross-Attention · Self-Learning

1 Introduction

Question answering requires confronting questions that may be ambiguous, vague, or require reasoning to obtain an answer. It is an important challenge in the field of natural language processing. In recent years, pre-trained language models [11, 12] have been remarkably successful for most of the tasks in natural language processing. Language models seem to encode implicit factual knowledge, but they still struggle to accurately capture the domain knowledge used to accomplish complex reasoning. One possible reason is that this domain knowledge is implicit in the natural language text and is not explicitly informed.

In contrast, language models tend to remember explicit co-occurrence relations between words.

To enhance the reasoning ability of models, a structured knowledge graph with inference capabilities has been introduced in previous research [10, 22, 26, 27] and investigated how it can be used to enhance the inference capabilities of Pre-trained language models (PTM). A part of the current research [22, 26, 27] aims at enhancing Q&A model inference by constructing graph encoders to integrate subgraph knowledge. This approach often relies on fusing shallow knowledge between nodes in the GNN and language model and requires the construction of complex network structures. In addition, graph encoders constructed in this way tend to focus more on the information of the nodes than the relation information of the edges. In order to focus on the large amount of relational information in the graph, [7, 19] processed the relational information in the graph by extracting meta-paths from the knowledge graph and constructing different meta-path encoders. However, the interactions between their language model and meta-path encoders are too crude and neither reflects the model’s selection process for inference paths.

When combining knowledge graphs with pre-trained language models, the following three issues need to be considered: (1) How to combine structured information from knowledge graphs with pre-trained language models that deal with unstructured text (2) The feature space of a textual representation is completely different from that of a graphical representation. How to do feature space alignment (3) There are many noisy nodes in the knowledge graph. How do we remove the noise and select the valid information.

For these three problems, we have carried out a study. To address problem one, we employ a form of meta-path to encode relational paths in the knowledge graph. To solve problem two, We introduce a cross-attention mechanism between the text encoder and the path encoder, allowing structured knowledge and contextual semantic information to be passed between the two modalities. How to select the useful information is important. We propose a self-learning mechanism that weights and scores the meta-path representation. The results of the experiments prove the superiority of our proposed model.

Our contributions are as follows:

1. We introduce a cross-attention mechanism that facilitates the interaction between contextual semantic information and path inference information.
2. We put forth a weighted scoring method as part of our self-learning mechanism, offering valuable assistance in selecting pertinent and valid path information from a vast array of available paths.
3. Our approach has yielded remarkable results on prominent commonsense QA datasets, along with the medical QA dataset, demonstrating the effectiveness of our approach.

2 Related Work

Knowledge graphs have been widely used to enhance QA systems due to their ability to provide structured reasoning information. In previous studies, some works have attempted to encode knowledge graph information as unstructured text to directly augment the input information of pre-trained language models, (e.g., Knowledgeable Reader [16], KagNet [10]). Others use the textual representation provided by the language model as instruction information to do inference with the graph encoder represented by GNN (e.g., MHGRN [4]).

PTM+GNN: Some previous works [4, 10, 20] try to use PTM+GNN approach for bi-directional inference. QA-GNN [26] proposes to use the output vectors of the language model as graph neural network nodes so that they can pass information to the graph encoder and guide the convolution. However, their method uses pooled representation vectors, which limits the information exchange and propagation, and the information propagation is only from LM to GNN. GreaseLM [27] proposes a bidirectional information propagation mechanism to fuse the information from GNN and LM through an MLP, but such an approach is too crude to screen the information. JointLK [22] proposes a bidirectional attention and dynamic cropping mechanism, which achieves good results. However, the space complexity of their approach is too high and the computation is too time-consuming.

PTM+Meta-Path: GNN-based graph encoders tend to be node-centric during inference, downplaying the relational information in the graph. To overcome this drawback, relationship-centric meta-path encoders have emerged, and good results have been achieved in SAFE [7] by encoding meta-paths with unique one-hot embedding and scoring them by MLP. Their method has good interpretability but does not exclude noise paths and fails to exchange information between the two modalities. QAT [19] improves the lack of interaction by feeding the meta-path representation and the text representation into the Transformer directly, but their interaction is brute and dependent on the performance of the Transformer.

3 Task Definition

The commonsense QA or medical QA task described in this paper is a multiple-choice problem. First, a natural language question Q and n options c_1, c_2, \dots, c_n are given, and we are asked to choose the most sensible answer from the given n options as the answer. To obtain background knowledge, we need to introduce domain knowledge graphs. A general knowledge graph can be represented as a multi-relational graph $G = (V, R, E)$, where V is the set of all concept nodes, R is the set of relation types, and $E \subseteq \{V \times R \times V\}$ is the set of edges that connect two concept nodes in V .

4 Method

In this section, We present the overall structure of our model. The model structure is shown in Fig. 1. The model consists of four components: text encoder, path encoder, PACA module, and self-learning module.

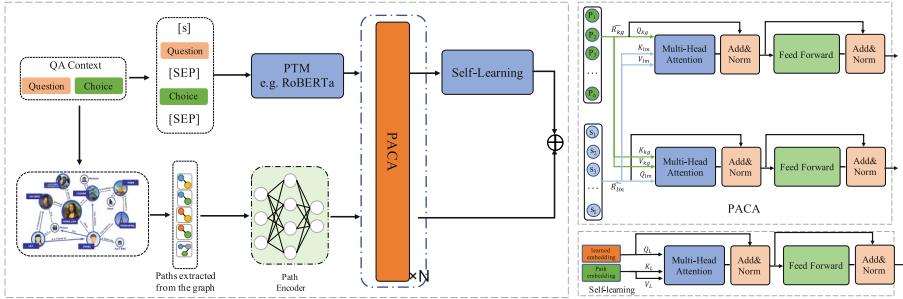


Fig. 1. Overall structure of the PCN (left), detailed structure of PACA (top right), detailed structure of Self-Learning (lower right). Initially, we extract the graph path and employ an MLP for encoding (§4.1). Simultaneously, the domain Pre-trained LM (PTM) encodes the QA content (§4.1). Subsequently, the PACA module plays a pivotal role by integrating these two information representations, facilitating the exchange of modal information and enabling reasoning capabilities (§4.2). Lastly, the self-learning module incorporates the path representation, enabling the selection and scoring of valid paths (§4.3).

4.1 Text Encoder and Path Encoder

Text Encoder. We obtain the representation of the question-choice content based on the pre-trained language model. Given the sequence of tokens $[q; c_i]$ of the input question-choice pair, we obtain the textual representation of the pair via the domain pre-trained language model, denoted as:

$$R_{lm} = f_{enc}(text[q; c_i]) \quad (1)$$

where f_{enc} denotes the output of the language model and $[;]$ represents the concatenation of the question text and the answer text.

Path Encoder. We follow the data processing method of QAT [19] to extract the meta-paths from the knowledge graph. Specifically, we first link the entities in the question texts and answer texts to nodes on the knowledge graph. Based on the linked concepts in the question and each answer candidate, we further extract their neighboring nodes from G and the relational edges connecting nodes to compose a subgraph G^{q, c_i} . We extract all the paths from the entity nodes in

the question to the ones in the answer and encode the edge features with one-hot vectors. Specifically, one-hop edges and two-hop paths are encoded as:

$$P_1 = g_{\theta_1}([\phi(h), r, \phi(t), \delta_{h,t}]); \quad P_2 = g_{\theta_2}([\phi(h), r_1, \phi(v_1), r_2, \phi(t), \delta_{h,t}]) \quad (2)$$

Where $g_{\theta_1}(\cdot)$ and g_{θ_2} are MLP. $\delta_{h,t} = f_t - f_h$, where f_t, f_h are the tail and head node features, respectively. $\phi(\cdot) : V \rightarrow T_v$ is a one-hot encoder that maps a node $v \in V$ to a node type in T_v .

In our experiments, we use meta-paths with two hops and less, since paths with three hops and more tend to be irrelevant and rare. These paths are denoted as:

$$R_{KG} = \bigcup_{k=1}^2 \{p_i | p_i \in P_k\} \quad (3)$$

4.2 Path-Aware Cross-Attention

Theoretically, integrating the knowledge of meta-paths and QA texts based on the contextual semantic information provided by the pre-trained language model and the inference knowledge provided by the meta-paths can provide more accurate answers. However, the data formats and data quality of these two data sources are different and need to be transformed and standardized before they can be fused. To distinguish between the two source datasets, we add learnable embedding representations to the output representations of the two encoders:

$$\hat{R}_{lm} = R_{lm} + e_{lm}; \quad \hat{R}_{kg} = R_{kg} + e_{kg}; \quad (4)$$

where both e_{lm} and e_{kg} are learnable embedding representations. In order to make inference information and contextual information fully integrated, we propose a path-aware cross-attention mechanism. The detail of the path-aware cross-attention (PACA for short) structure is shown in Fig. 1:

We perform a linear transformation on the text representation, utilizing the multiplication of $W_{Q_{lm}}$, $W_{K_{lm}}$, and $W_{V_{lm}}$, resulting in the derivation of three distinct vector representations, namely Q_{lm} , K_{lm} , and V_{lm} .

$$Q_{lm} = \widehat{R_{lm}} W_{Q_{lm}}; \quad K_{lm} = \widehat{R_{lm}} W_{K_{lm}}; \quad V_{lm} = \widehat{R_{lm}} W_{V_{lm}}; \quad (5)$$

Likewise, the meta-path representation $\widehat{R_{kg}}$ undergoes multiplication with $W_{Q_{kg}}$, $W_{K_{kg}}$, and $W_{V_{kg}}$ to yield three distinct vectors, namely Q_{kg} , K_{kg} , and V_{kg} , representing the meta-path information.

We employ the query vector Q_{lm} to perform multi-headed attention in conjunction with the meta-path key vector K_{kg} . Subsequently, the attention weights and the value vector V_{kg} are combined through weighting and summation processes to derive the conclusive contextual representation Z_{lm} , which is followed by an LN layer and a Skip connection to get $\widehat{Z_{lm}}$. This representation not only

encompasses the inherent contextual semantic information but also captures the inference logic of the paths, thereby enriching the overall comprehension.

$$Z_{lm} = \text{MultiHead}(Q_{lm}, K_{kg}, V_{kg}); \quad \widehat{Z_{lm}} = \text{FFN}(\text{LN}(Z_{lm})) + Z_{lm} \quad (6)$$

In the same manner, we obtain the representation of the path:

$$Z_{kg} = \text{MultiHead}(Q_{kg}, K_{lm}, V_{lm}); \quad \widehat{Z_{kg}} = \text{FFN}(\text{LN}(Z_{kg})) + Z_{kg} \quad (7)$$

We use multiple layers of PACA to enhance the deep interaction between the two modalities, and the experiments in Table 3 show that a certain number of layers of PACA enhances the information propagation between the two modalities.

4.3 Self-learning Based Path Scoring Method

The path-aware cross-attention-based mechanism provides better information interaction capability. However, not all of the extracted meta-paths inherently possess valid inference information. To address this, we introduce a self-learning-based scoring approach, enabling the evaluation and scoring of the extracted paths based on their reliability.

As depicted in Fig. 1, we establish a one-dimensional learnable vector, which serves as the query vector. Additionally, we employ linear transformations to the path vector to derive the key and value vectors. Through the utilization of the attention mechanism, the query vector and key vector collectively calculate the attention weights. Subsequently, the value vector is weighted and consolidated based on the attention weights, resulting in an aggregated representation.

$$Q_L = E_L W_Q; \quad K_L = Z_{kg} W_K; \quad V_L = Z_{kg} W_V \quad (8)$$

$$Z'_{kg} = \text{MHA}(Q_L, K_L, V_L) \quad (9)$$

$$S_{q,c_i}^{ph} = \text{MLP}_1(\text{FFN}(\text{LN}(Z'_{kg})) + Z'_{kg}) \quad (10)$$

where S_{q,c_i}^{ph} is the path scoring of the question-answer pair (q, c_i) and E_L is the learnable vector. W_Q , W_K , W_V are three linear transformation matrices.

4.4 Learning and Inference

The scoring of a question-option pair encompasses two components: the language model score and the meta-path score. In Sect. 4.3, we elaborate on the derivation of the meta-path scores. As for the language model score, it is computed using the following approach:

$$S_{q,c_i}^{lm} = \text{MLP}_1(\text{MLP}_2(h_{cls})) \quad (11)$$

where $h_{<\text{cls}>}$ is the first token representation in Z_{lm} , that is, the representation of $<\text{cls}>$. Ultimately, the scoring of a question-option pair is represented as:

$$S_{q,c_i} = S_{q,c_i}^{lm} + S_{q,c_i}^{ph} \quad (12)$$

For a question-option pair, if it is labeled as positive, the final score is assigned a value of 1; otherwise, it is set to 0. To obtain the final probability, we apply softmax normalization across all question-choice pairs. The model is then trained end-to-end using the cross-entropy loss, ensuring comprehensive optimization.

During the inference stage, the option c_i with the highest score among all available options is selected as the predicted true option.

5 Experiment

5.1 Dataset

We evaluate our model on three multiple-choice question-answering datasets across two domains: CommonsenseQA [23] and OpenBookQA [15] as common-sense QA benchmarks, and MedQA-USMLE [8] as a clinical QA benchmark.

CommonsenseQA is a 5-way multiple choice QA task that requires reasoning with commonsense knowledge, containing 12,102 questions. The test set of CommonsenseQA is not publicly available, and model predictions can only be evaluated once every two weeks via the official leaderboard. Hence, we perform main experiments on the in-house (IH) data split used in [23].

OpenBookQA is a 4-way multiple choice QA task that requires reasoning with elementary science knowledge, containing 5,957 questions. We use the official data split from [15].

MedQA-USMLE is a 4-way multiple-choice question-answering dataset, which requires biomedical and clinical knowledge. The questions are originally from practice tests for the United States Medical License Exams (USMLE). The dataset contains 12,723 questions. We use the original data splits from [8].

5.2 Baseline Models

We compare our model with the following baseline methods, including a fine-tuned PTM, 4 PTM+GNN Encoder, and 2 latest PTM+Meta-Path Encoder.

Fine-Tuned PTM. Without adding any external knowledge, we directly fine-tune the PTM. For the commonsense QA task, we use language models such as RoBERTa-L [12]. For medical Q&A, we use Sap-BERT [11] to compare various methods.

PTM+GNN. Joint scoring using GNN and PTM. That is, LM is used as a text encoder, and GNN or RN [20] is used as a KG encoder. We compare several representative methods: (1) Relationship network (RN) (2) QA-GNN [26] (3) GreaseLM [27] (4) JointLK [22].

PTM+Meta-Path. Joint scoring using Path Encoder and PTM. That is, encoding meta-paths and interacting with pre-trained language models using different inference methods. We compare two representative methods: (1) SAFE [7] (2) QAT [19].

5.3 Main Result

Table 1. Performance comparison on CommonsenseQA in-house split and test accuracy on OpenBookQA. For CommonsenseQA, we follow the data division method of Lin et al. (2019) and report the in-house Dev (IHdev) and Test (IHtest) accuracy (mean and standard deviation of four runs).

Method	CommonsenseQA		OpenBookQA
	IHdev-Acc. (%)	IHtest-Acc. (%)	Test-Acc. (%)
RoBERTa-Large(w/o KG)	73.1(± 0.5)	68.7(± 0.6)	64.8(± 2.4)
RN [20]	74.6(± 0.9)	69.1(± 0.2)	65.2(± 1.2)
QAGNN [26]	76.5(± 0.2)	73.4(± 0.9)	67.8(± 2.8)
GreaseLM [27]	78.5(± 0.5)	74.2(± 0.4)	-
JointLK [22]	77.9(± 0.3)	74.4(± 0.8)	70.3(± 0.8)
SAFE [7]	-	74.5($\pm \text{NA}$)	-
QAT [19]	79.5(± 0.4)	75.4(± 0.3)	71.2(± 0.8)
PCN(Ours)	79.1(± 0.6)	75.6(± 0.7)	71.5(± 1.7)

CommonsenseQA and OpenBookQA. The experimental results in Table 1 show that the proposed model approach performs best on CommonsenseQA (IHtest) and OpenBookQA datasets, verifying the effectiveness of the model under Commonsense question answering. Our proposed PCN, compared to the vanilla pre-trained language model RoBERTa-Large, has improved by 6.9% in CommonsenseQA and 6.7% in OpenBookQA, demonstrating that meta-paths can integrate effective inference information to enhance the performance of PTM on commonsense question answering task. Additionally, among all PTM+GNN approaches, JointLK performs best thanks to the excellent interaction and graph pruning techniques. However, our PCN outperforms them by 1.2% on CSQA and OBQA, respectively, illustrating the superiority of meta-paths as auxiliary inference information. Our approach also performs best among all PTM+Path Encoder models. QAT has achieved the best result among PTM+Path Encoder approaches in the past with its excellent path encoder. Our approach adds path-aware cross-attention and self-learning mechanisms to the path representation of QAT and achieves the best results. This fully demonstrates the superiority of our approach.

MedQA-USMLE. To evaluate the domain generalizability of our method, we carry out experiments on MedQA-USMLE with results shown in Table 2. The table demonstrates that our method surpasses previous methods in achieving the best results on this dataset. This result illustrates the excellent domain generalization ability of our method.

Table 2. Test accuracy comparison on MedQA-USMLE.

Method	Accuracy (%)
SapBERT-Base [11]	37.2
QA-GNN [26]	38.0
GreaseLM [27]	38.5
QAT [19]	39.3
PCN (Ours)	39.7

6 Analysis

6.1 Ablation Studies

To thoroughly analyze the role of each component in the model, we develop ablation experiments on the test set on OpenBookQA:

We find that the final performance of the model is reduced by at least 2.4% when either path or text is removed from the attention of the module. When we completely break the interaction between these two modules, the model’s score decreases significantly, to 69.4%. When the self-learning module is removed, the prediction accuracy drops by 1.6%. In order to verify that the meta-path works, we remove the meta-path encoder and the subsequent interaction module, and the model’s performance is sharply reduced by 8%. We also conduct experiments on the number of layers of interaction to find the most appropriate interaction depth and find that the best performance is achieved when the number of layers is 2, which is roughly in line with previous experiments [5, 26].

6.2 Model Interpretability

To explore the specific inference process of our proposed PACA module and Self-Learning module, we analyse the Attention weights in the modules. A concrete example is illustrated in Fig. 2.

Table 3. Ablation study of our model components, using the OpenBookQA test set. We choose one of the four random seeds for the ablation experiment.

Model	Test Acc.(%)	PACA Layers	Test acc
PCN (ours)	73.0	L = 1	71.6
w/o meta-path	64.8	L = 2	73.0
w/o PACA	69.4	L = 3	70.6
w/o PACA (Path)	70.2	L = 4	70.6
w/o PACA (Text)	70.6		
w/o self-learning	71.4		

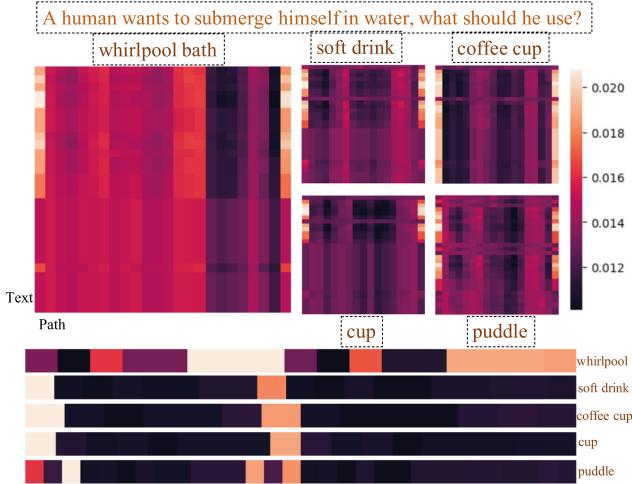


Fig. 2. Attention visualisation in PACA and Self-Learning. (above) The Attention matrix for text and paths in PACA. (below) The Attention matrix of the self-learning vector and paths

As the diagram shows, for the question “A human wants to submerge himself in water, what should he use?”, our model has a larger bright range for the correct option “whirlpool bath”. This means for the correct option the model finds more correlations and supporting details between the two representations, text and path, which enriches the inference process and enhances the robustness of the model.

6.3 Quantitative Analysis

To further analyse the capabilities of the model, we explore (1) what types of problems the model’s performance improvement is reflected in (2) whether the model can solve more complex problems (e.g. problems containing negation and complex long sentences). We compare the best results of the previous meta-paths QAT with our model, which are shown in Table 4.

Negation. To investigate the model’s reasoning ability in negative sentences, we extract negative sentences from the test set in commonsenseQA. We treat sentences containing negative terms as negative sentences. Based on the results in Table 4, our PCN outperforms the previous PTM+GNN method in reasoning about negation. Although we were slightly less effective than QAT on dev, we perform on par with QAT on negation, indicating that our model has a stable judgment ability for negative samples. We conjecture that it is the PACA mechanism that allows the model to understand negation in sentences well.

Question with Fewer/More Entities. The complexity of a problem can usually be measured according to the number of entities extracted from the problem.

Table 4. Performance on questions with negative words and fewer/more entities. The questions are retrieved from the CommonsenseQA IDev set. We choose one of the four random seeds for the experiment.

Method	Number	PTM+GNN	QAT	Ours
ITese-Acc (Overall (%))	1221	77.9	79.8	79.7
Questions w/negation	133	75.9	79.0	79.0
Questions w/<7 entities	723	76.2	79.9	79.6
Questions w/>7 entities	498	77.9	79.7	79.8

Following the NER method in QAGNN [26], we obtain the number of entities in the problem, use a threshold of 7 entities, and divide them into two categories: containing fewer entities (<7) and more entities (>7). As Table 4 shows, our method showcases notable enhancements in scenarios involving both fewer and greater than 7 entities. Furthermore, when compared to QAT, our approach exhibits a slight improvement in cases where the entity count exceeds 7. This shows that our method is robust and better at removing noisy information when faced with complex problems.

7 Conclusion

We introduce a novel QA model that leverages a path-aware cross-attention mechanism to facilitate inference. Our model incorporates meta-paths, which possess rich inference information, as auxiliary information. Through multiple layers of cross-attention, the textual representation obtains inference information about the relationship, while the contextual semantics is encoded into the meta-path. Additionally, we introduce a self-learning scoring mechanism that dynamically assigns weights to the meta-paths. We perform a series of exploratory experiments on three distinct datasets encompassing two domains. The experimental findings strongly validate the success and effectiveness of our proposed approach.

Acknowledgements. We thank the reviewers for their insightful comments and valuable suggestions. This study is partially supported by National Key R&D Program of China (2021ZD0113402), National Natural Science Foundation of China (62276082), Major Key Project of PCL (PCL2021A06), Shenzhen Soft Science Research Program Project (RKG20220705152815035), Shenzhen Science and Technology Research and Development Fund for Sustainable Development Project (GXWD20231128103819001, No. KCXFZ20201221173613036) and the Fundamental Research Fund for the Central Universities (HIT.DZJJ.2023117).

References

1. Alsentzer, E., et al.: Publicly available clinical BERT embeddings. In: Proceedings of the 2nd Clinical Natural Language Processing Workshop (2019)
2. Chen, D., Li, Y., Yang, M., Zheng, H.T., Shen, Y.: Knowledge-aware textual entailment with graph attention network. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management (2019)
3. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding (2019)
4. Feng, Y., Chen, X., Lin, B.Y., Wang, P., Yan, J., Ren, X.: Scalable multi-hop relational reasoning for knowledge-aware question answering. In: EMNLP (2020)
5. Guan, X., Cao, B., Gao, Q., Yin, Z., Liu, B., Cao, J.: CORN: co-reasoning network for commonsense question answering. In: Proceedings of the 29th International Conference on Computational Linguistics, Gyeongju, Republic of Korea, October 2022. International Committee on Computational Linguistics (2022)
6. Gururangan, S., et al.: Don't stop pretraining: adapt language models to domains and tasks. In: Proceedings of ACL (2020)
7. Jiang, J., Zhou, K., Wen, J.R., Zhao, X.: *great truths are always simple*: a rather simple knowledge encoder for enhancing the commonsense reasoning capacity of pre-trained models. In: Findings of the Association for Computational Linguistics (2022)
8. Jin, D., Pan, E., Oufattolle, N., Weng, W.H., Fang, H., Szolovits, P.: What disease does this patient have? A large-scale open domain question answering dataset from medical exams. *Appl. Sci.* **11**(14), 6421 (2021)
9. Lee, J., et al.: BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* **36**(4), 1234–1240 (2020)
10. Lin, B.Y., Chen, X., Chen, J., Ren, X.: KagNet: knowledge-aware graph networks for commonsense reasoning. In: EMNLP-IJCNLP 2019 (2019)
11. Liu, F., Shareghi, E., Meng, Z., Basaldella, M., Collier, N.: Self-alignment pretraining for biomedical entity representations. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (2021)
12. Liu, Y., et al.: ROBERTa: a robustly optimized BERT pretraining approach. arXiv preprint [arXiv:1907.11692](https://arxiv.org/abs/1907.11692) (2019)
13. Lv, S., et al.: Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. In: AAAI 2020 (2020)
14. Lv, S., et al.: Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. *Proc. AAAI Conf. Artif. Intell.* **34**(05), 8449–8456 (2020)
15. Mihaylov, T., Clark, P., Khot, T., Sabharwal, A.: Can a suit of armor conduct electricity? A new dataset for open book question answering. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (2018)
16. Mihaylov, T., Frank, A.: Knowledgeable reader: enhancing cloze-style reading comprehension with external commonsense knowledge. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (2018)
17. Mihaylov, T., Frank, A.: Knowledgeable reader: enhancing cloze-style reading comprehension with external commonsense knowledge. In: ACL (2018)
18. Pan, X., et al.: Improving question answering with external knowledge. In: Proceedings of the 2nd Workshop on Machine Reading for Question Answering, November 2019

19. Park, J., et al.: Relation-aware language-graph transformer for question answering (2022)
20. Santoro, A., et al.: A simple neural network module for relational reasoning. In: Guyon, I., et al. (eds.) *Advances in Neural Information Processing Systems*, vol. 30 (2017)
21. Schlichtkrull, M., Kipf, T.N., Bloem, P., Van Den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: *The Semantic Web: 15th International Conference, ESWC 2018* (2018)
22. Sun, Y., Shi, Q., Qi, L., Zhang, Y.: JointLK: joint reasoning with language models and knowledge graphs for commonsense question answering. In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (2022)
23. Talmor, A., Herzig, J., Lourie, N., Berant, J.: CommonsenseQA: a question answering challenge targeting commonsense knowledge. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (2019)
24. Wang, P., Peng, N., Ilievski, F., Szekely, P., Ren, X.: Connecting the dots: a knowledgeable path generator for commonsense question answering. In: *Findings of the Association for Computational Linguistics, EMNLP 2020*, November 2020 (2020)
25. Yang, A., et al.: Enhancing pre-trained language representations with rich knowledge for machine reading comprehension. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (2019)
26. Yasunaga, M., Ren, H., Bosselut, A., Liang, P., Leskovec, J.: QA-GNN: reasoning with language models and knowledge graphs for question answering. In: *North American Chapter of the Association for Computational Linguistics (NAACL)* (2021)
27. Zhang, X., et al.: GreaseLM: graph reasoning enhanced language models. In: *International Conference on Learning Representations* (2021)



StyleAutoEncoder for Manipulating Image Attributes Using Pre-trained StyleGAN

Andrzej Bedychaj^(✉), Jacek Tabor, and Marek Śmieja

Jagiellonian University, Kraków, Poland
andrzej.bedychaj@student.uj.edu.pl, marek.smieja@uj.edu.pl

Abstract. Deep conditional generative models are excellent tools for creating high-quality images and editing their attributes. However, training modern generative models from scratch is very expensive and requires large computational resources. In this paper, we introduce StyleAutoEncoder (StyleAE), a lightweight AutoEncoder module, which works as a plugin for pre-trained generative models and allows for manipulating the requested attributes of images. The proposed method offers a cost-effective solution for training deep generative models with limited computational resources, making it a promising technique for a wide range of applications. We evaluate StyleAE by combining it with StyleGAN, which is currently one of the top generative models. Our experiments demonstrate that StyleAE is at least as effective in manipulating image attributes as the state-of-the-art algorithms based on invertible normalizing flows. However, it is simpler, faster, and gives more freedom in designing neural architecture.

1 Introduction

Generative models, such as generative adversarial networks (GAN) [10], variational AutoEncoders (VAE) [20], diffusion models [23], and flow-based generative models [8], have gained popularity due to their ability to create high-quality images, videos, and texts. These models are trained using supervised or unsupervised learning techniques on large datasets. They have transformed the field of artificial intelligence and are used to create innovative applications across various research fields [9, 13, 22, 24, 26, 28].

StyleGAN [15–17] is one of the most popular generative models used for creating high-quality images, known for its ability to control various aspects of the generated image such as pose, expression, and style. However, the latent space of StyleGAN is highly entangled [16], meaning that the different attributes of the generated image are not easily separable. This makes it challenging to manipulate individual attributes without affecting others, limiting the controllability of the generated images. Furthermore, manipulating the latent space of StyleGAN is a challenging task due to the complex and high-dimensional nature of the model, which limits its practical applications.



Fig. 1. Single-attribute manipulation with StyleAE in StyleGAN’s latent space.

There are various methods for simplifying the latent space of generative models. Unsupervised methods include algorithms such as Interface GAN [24], GANSpace [11] and InfoGAN [5], which aim to learn a disentangled representation of the data without requiring any labeled data. Supervised methods like PluGeN [25, 29] or StyleFlow [3], on the other hand, require labeled data and involve training an auxiliary model to predict a particular attribute, such as the pose or shape of the generated image. These approaches are essential for improving the practical applications of generative models and making them more useful for a wider range of applications.

While flow-based models such as StyleFlow and PluGeN have shown promising results in disentangling the latent space of StyleGAN, they also have some limitations. One significant limitation is the difficulty in scaling these models to high-resolution images due to their computationally intensive nature [18]. Moreover, invertible models require a large amount of training data to learn the complex data distributions [14], which may be challenging to obtain in some cases. Finally, flow-based models can be sensitive to hyperparameters [27], making them difficult to develop optimal performance.

This paper presents a novel approach, called StyleAE, for modifying image attributes using a combination of AutoEncoders with StyleGAN. StyleAE simplifies the latent space of StyleGAN so that the values of target attributes can be effectively changed. While StyleAE achieves at least as good results as the state-of-the-art flow-based methods, it is computationally more efficient and does not require so large amount of training data, which makes it a practical approach for various applications (see Fig. 1 for sample results).

We conducted an assessment of StyleAE through extensive experiments on datasets containing images of human and animal faces, benchmarking it against state-of-the-art flow-based models. Our findings demonstrate that StyleAE’s effectiveness in manipulating the latent space of StyleGAN is on par with that of flow-based models. Our research provides crucial insights into the unique strengths and limitations of StyleAE model, highlighting the potential of our algorithm to improve the effectiveness of latent space manipulation for StyleGAN and other generative models. Furthermore, our approach exhibits superior time complexity, making it a more feasible solution for a wide range of applications.

2 Related Work

Conditional VAE (cVAE) introduced label information integration into generative models but lacks assured latent code and label independence, impacting generation quality; on the other hand, Conditional GAN (cGAN) produces

higher-quality outputs but involves more complex training and falls short in manipulating existing examples [13, 19].

Fader Networks [21] address this limitation by combining cVAE and cGAN components, utilizing an encoder-decoder architecture and a discriminator predicting attributes. However, Fader Networks struggle with attribute disentanglement, and their training is more challenging than standard GANs. CAGlow [22] takes a different approach, using Glow for conditional image generation based on joint probabilistic density modelling. However, it does not reduce data dimensionality, limiting its applicability to more complex data. Competitive approaches like HifaFace [9], Pie [26], and GANSpace [11] have also been explored.

InterFaceGAN [24] and Hijack-GAN [28] manipulate facial semantics through linear models and a proxy model for latent space traversal. Recent approaches focus on manipulating latent codes of pre-trained networks, where data complexity is less restrictive, making flow models applicable. StyleFlow [3] and PluGeN [29] use normalizing flow modules on GAN latent spaces, employing conditional CNF and NICE, respectively. While StyleFlow is tailored for StyleGAN, PluGeN achieves important results across various models and domains. Further extensions of PluGeN on face images (dubbed PluGeN4Faces) [25] allowed for a significant improvement in the disentanglement between the attributes of the face and the identity of the person.

Our work takes a distinct approach, achieving results comparable to these methods while providing superior attribute decomposition and structural consistency for image datasets, coupled with significantly improved computational efficiency.

3 Methodology

In this section, we give a description of the proposed StyleAE approach to disentangling the latent space of generative models. Before that, we recall the StyleGAN and AutoEncoder architectures, which are the main building blocks of StyleAE .

3.1 Preliminaries

StyleGAN [15–17]: is a cutting-edge generative model lauded for its capacity to produce high-quality, realistic images. Its architecture comprises two key elements. Initially, the latent vector z , generated from a standard Gaussian distribution $\mathcal{N}(0, I)$, undergoes mapping to the style space vector w through a series of fully connected layers. Subsequently, this style vector is injected into the following convolutional blocks of the synthesis network, progressively crafting the image in the desired resolution.

While the latent vector z serves as the foundation for image creation, manipulating the image via the style vector w is notably more convenient. As the replicated style vector influences various blocks of the synthesis network, it enables

the control of the image’s style at different abstraction levels, offering users versatile means to manipulate generated images. However, achieving control over specific attributes without impacting others requires disentangling the style space. In the subsequent sections, we detail how we employ an AutoEncoder to modify the latent space of StyleGAN, enhancing its separability and controllability.

AutoEncoder: architecture comprises an encoder function, mapping input data to a compressed representation in the latent space, and a decoder function, mapping this representation back to the original input space. This is represented as:

$$z = \mathcal{E}(x) \text{ and } x' = \mathcal{D}(z), \quad (1)$$

where \mathcal{E} is the encoder, \mathcal{D} is the decoder, x is the input data, z is the latent space representation, and x' is the reconstructed output.

Training involves minimizing the reconstruction error, typically mean squared error (MSE) or binary cross-entropy (BCE), between x and x' . If the latent space’s dimensionality is much lower than the input resolution, the AutoEncoder learns essential features in the compressed representation z . In our research, as we do not emphasize image compression, we do not reduce the dimensionality.

3.2 StyleAutoEncoder

Our goal is to simplify and enhance the manipulation of image attributes by modifying the latent space of StyleGAN using an AutoEncoder. To this end, we develop StyleAE, an AutoEncoder plugin to StyleGAN, which allows for convenient manipulation of the requested attributes and preserving the quality of StyleGAN, see Fig. 2.

Structure of the Target Space: We assume that every instance $x \in X$ is associated with a K dimensional vector of labels $y = (y_1, \dots, y_K)$. The labels can represent a combination of discrete and continuous features. Our objective is to find a representation of images, where each label is encoded as a separate coordinate. More precisely, let the k -th latent variable c_k correspond to the attribute y_k . By modifying the value of c_k , we would like to change the value of k -th attribute in the image. Since labels do not fully describe the image, additional M variables (s_1, \dots, s_M) are included to encode the remaining information. Therefore, the latent vector of our new target space is defined as $(c_1, \dots, c_K, s_1, \dots, s_M)$.

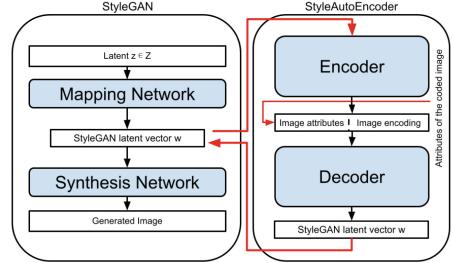


Fig. 2. Architecture design of StylebreakAE. StyleAE maps the style code w of the pre-trained StyleGAN into a target space, where labelled attributes are modelled by individual coordinates.

To construct such a target space, we operate on the representation given by a pre-trained generative model. Although our approach theoretically applies to arbitrary generative models, we consistently use StyleGAN as a base model and fix our attention to synthesis network $G : W \rightarrow \mathbb{R}^N$, which maps style vectors to the images. We focus on finding a mapping between the style space W and the target space (C, S) , where $C = (C_1, \dots, C_k)$ describes labelled attributes and $S = (S_1, \dots, S_M)$ denotes the remaining variables.

Loss Function: To establish an approximately invertible mapping, we use an AutoEncoder-inspired neural network dubbed StyleAE. More precisely, the encoder $\mathcal{E} : W \rightarrow (C, S)$ focuses on retrieving the attributes from the style vector while the decoder $\mathcal{D} : (C, S) \rightarrow W$ is used to recover the input data. StyleAE applies the cost function, which consists of two components: attribute loss and image loss.

To explain the details behind our loss, let w be the style vector representing the image x with attributes y , $(c, s) = \mathcal{E}(w)$ be the target representation, and $\hat{w} = \mathcal{D}(c, s)$ be the recovered style code. The image loss

$$d_I(x, G(\hat{w})) = \|x - G(\hat{w})\|^2 \quad (2)$$

aims at reconstructing the image from AutoEncoder representation while the attribute loss

$$\sum_{k=1}^K d_A(c_k, y_k), \quad (3)$$

aligns target coordinates c_k with image attributes. While the mean-square error (MSE) is the obvious choice for implementing attribute loss d_A , we found that for binary attributes $y_k \in \{0, 1\}$ an alternative loss can be beneficial. Namely, for positive label $y_k = 1$, we calculate the distance between the value c_k returned by AE and the interval $[y_k, \infty) = [1, \infty)$ as follows:

$$d_A^S(c_k, 1) = \max(0, 1 - c_k) \quad (4)$$

This allows us to encode a different style for a binary value, e.g. different type of facial hair. For negative label $y_k = 0$, we use typical MSE since this corresponds to the absence of an attribute.

3.3 Discussion

Image Editing: Attribute manipulation in an image x involves obtaining the style vector w . While a classification mechanism can assist by tagging generated images with desired attributes, this method was applied to our human facial features dataset, categorized externally using the Microsoft Face API.

However, for datasets lacking pre-tagged samples, a mechanism to retrieve w is essential. Literature suggests various methods [2], often using an iterative approach to approximate the StyleGAN latent vector w for a given image x .

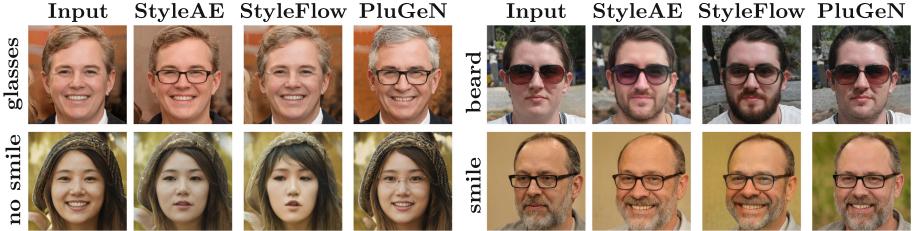


Fig. 3. Examples of attributes modification for all of the tested models on FFHQ dataset. One can observe that StyleAE correctly modifies the requested attributes and is less invasive to the remaining characteristics of the image than the competitive flow-based methods.

This process, though, can be time-consuming and does not guarantee optimal results.

In our experiments, we used the method proposed in [1] to retrieve w for the AFHQv2 dataset, which did not initially come with the required latent vectors for the images.

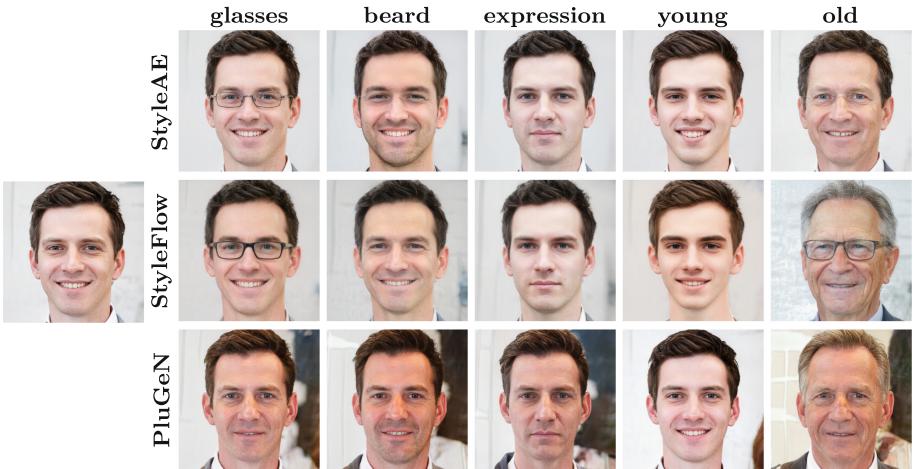


Fig. 4. Attribute modification on a sample image generated from StyleGAN. The generated images by all models exhibit successful changes in the manipulated attributes while maintaining the overall coherence of the image. Our findings indicate that the performance of StyleAE method is comparable to state-of-the-art flow-based models in producing effective attribute manipulation.

Related Models: The proposed simplification of the StyleGAN latent space utilizes an AutoEncoder architecture, offering advantages over flow-based models like PluGeN [29] and StyleFlow [3].

Unlike PluGeN and StyleFlow, which rely on complex architectures for reversible transformations due to challenges posed by flow models, the AutoEn-

coder is simpler and computationally more efficient. With only two components - Encoder and Decoder, AutoEncoders facilitate easier training and demand fewer computational resources than flow-based models. Additionally, AutoEncoders autonomously learn a disentangled data representation without explicit supervision [4], enhancing adaptability. In contrast, flow-based models often require intricate objective functions, posing challenges in optimization. AutoEncoders also excel in scenarios with limited labelled data, being effectively trainable with smaller datasets.

It's crucial to emphasize that StyleAE presents a unique and innovative approach, benefiting from its inherent simplicity and ease. Its architecture, based on AutoEncoders, allows for task-specific cost functions independent of distribution assumptions, contributing to a straightforward and efficient design.

4 Experiments

In this section, we present the results for manipulating image attributes through the proposed StyleAE . We use two publicly available datasets:

Flickr Faces (FFHQ) [16] and Animal Faces (AFHQv2) [6]. Our method is compared to StyleFlow and PluGeN, which represent the current state-of-the-art. We compare our method regarding structural coherence, effectiveness in generating images with requested changes, and time efficiency. In all experiments, we combine the considered methods with the StyleGAN backbone.

4.1 Evaluation Metrics

The goal of attribute manipulation is to accurately modify designated image attributes while preserving other characteristics. To assess accuracy, we use classification accuracy from an independent multi-label face attribute classifier, trained on datasets not used in training the evaluated models.

To evaluate potential impacts on other image features, we employ three metrics: mean square error (MSE), peak signal-to-noise ratio (PSNR), and structural similarity index (SSIM). PSNR, a logarithmic-scale-modified MSE, and SSIM, assessing visible structures, offer insights, with higher values indicating better performance. Additionally, we calculate perceptual MSE (p-MSE) on image embeddings from a pre-trained ArcFace model [7] sourced from the Python library [arcface](#).

Table 1. Perceptual MSE describing the distance between embeddings of input and modified images. We utilized an ArcFace model to extract the embedding of each image. One may observe that StyleAE obtains significantly lower MSE than state-of-the-art models. Our qualitative experiments also demonstrate that our method preserves a substantial number of other visual facial features when manipulating just one of them.

Attribute	StyleAE	PluGeN	StyleFlow
man	73.241	56.955	53.999
woman	36.650	28.300	23.507
no glasses	20.532	49.325	32.092
glasses	48.298	52.424	57.537
no beard	26.638	57.297	52.511
beard	4.099	5.106	5.422
no smile	47.496	36.251	36.471
smile	13.697	22.987	17.308

We intentionally avoided using the Frechet Inception Distance (FID) measure due to its unsuitability for attribute manipulation settings. FID primarily compares the distribution of generated images to real ones, which may not precisely reflect changes in specific attributes. As image attribute manipulation alters the distribution of generated data, FID scores can increase, even if image quality and diversity remain constant.

4.2 Models Implementation

StyleAE is a neural network comprising three fully connected layers in the encoder and decoder, each with 512 neurons and PReLU activation. Inputting a 512-dimensional style vector w to the encoder yields a decoded projection \hat{w} from the decoder. The omission of further latent vector compression aligns with flow-based models for a fair comparison.

Training StyleAE with the Adam optimizer at a learning rate of 0.00001 spans 100 epochs. To foster effective learning in both proper reconstruction and desirable attribute organization, we gradually increase the attribute loss weight factor from 0 to 0.3 over the initial 30 epochs.

Baseline comparisons include two popular attribute manipulation plugins: StyleFlow [3] and PluGeN [29]. Both rely on flow-based models: StyleFlow using CNF and PluGeN using NICE. We use publicly available checkpoints for evaluation, avoiding retraining PluGeN or StyleFlow ourselves.

Table 2. Accuracy in modifying consecutive image attributes. We assessed the classifier’s predictive accuracy for a specific attribute, incorporating it in the final phase of vector modification for fair comparison across methods. Results indicate our plugin’s efficiency, comparable to flow-based models in achieving the goal of attribute modification.

Attribute	StyleAE	PluGeN	StyleFlow
man	0.92	0.91	0.95
woman	0.96	0.89	0.91
no glasses	0.74	0.78	0.67
glasses	0.90	0.94	0.88
no beard	0.96	0.95	0.96
beard	0.78	0.55	0.67
no smile	0.99	1.0	0.96
smile	1.0	1.0	0.99

4.3 Manipulation of Facial Features

Setup: In the first experiment, we use the FFHQ dataset, which contains 70 000 high-quality images of resolution 1024×1024 . All considered methods were trained on 10 000 images generated by StyleGAN. Eight attributes of these images were labelled using the Microsoft Face API (gender, pitch, yaw, eye-glasses, age, facial hair, expression, and baldness).

While the previous studies employing flow-based models utilized the Microsoft Face API for evaluating the accuracy of attribute manipulation, we decided to develop our own classification network due to alterations in the licensing of the Microsoft model. Our classifier is based on the ResNet18 architecture [12] and consisted of 8 target classes aligned with Microsoft’s classification system.

Since every method can use different scales to represent the intensity of attributes being modelled, we employed an attribute classifier to apply a minimal modification to obtain the requested value of the attribute. In other words, we gradually modify the attribute until the classifier recognizes the attribute of the generated image with sufficient confidence. If we cannot obtain the requested modification, the classifier returns failure. All the metrics comparing original images with the modified ones, including MSE, PSNR and SSIM, are calculated on minimally modified images.

Results: Sample results of attribute manipulation, presented in Fig. 3 and 4, suggest that StyleAE correctly modifies the requested attributes while preserving the remaining characteristics of the image to a high extent. To support this conclusion with quantitative assessment, we analyze the classification accuracy shown in Table 2 and the remaining metrics describing the difference between the original and modified images, see Tables 1 and 4.

As can be seen from Table 2, in most cases, StyleAE obtains comparable accuracy to PluGeN and better performance than StyleFlow. Closer inspection reveals that it is more accurate at modifying beard attributes, presents very good performance on gender and smile attributes, and slightly lower scores on the glasses feature. Taking into account image differences, reported in Tables 1 and 4, we can observe that StyleAE bet-

Table 3. Average training and inference time. Time required for training and inference on a single NVIDIA GeForce RTX 3080 GPU for each method. Results highlight the substantial speed advantage of our plugin over state-of-the-art flow-based models.

Time	StyleAE	PluGeN	StyleFlow
Training ^a	~15 min	~30 min	~1.5 h
Inference ^b	~20 s	~5 min	~1 h

^a Average time of 1 training epoch.

^b Average time taken by generation of 500 images.

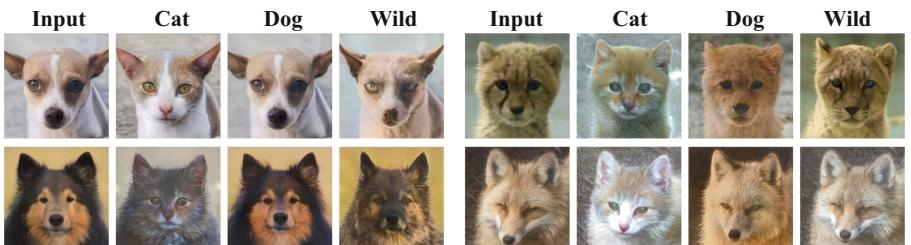
Table 4. Structural reconstruction qua-breaklity measures. MSE and PSNR estimate absolute errors, while SSIM considers perceived changes in structural information. Our results indicate that StyleAE maintains greater structural similarity between modifications and base images compared to state-of-the-art flow-based models.

Attribute	Measure	StyleAE	PluGeN	StyleFlow
man	<i>PSNR</i> \uparrow	17.947	19.445	19.607
	<i>SSIM</i> \uparrow	0.684	0.733	0.709
	<i>MSE</i> \downarrow	0.138	0.130	0.153
woman	<i>PSNR</i> \uparrow	20.485	19.026	19.576
	<i>SSIM</i> \uparrow	0.761	0.733	0.822
	<i>MSE</i> \downarrow	0.105	0.121	0.083
no glasses	<i>PSNR</i> \uparrow	22.810	18.764	17.552
	<i>SSIM</i> \uparrow	0.830	0.733	0.800
	<i>MSE</i> \downarrow	0.075	0.121	0.055
glasses	<i>PSNR</i> \uparrow	18.790	18.210	16.840
	<i>SSIM</i> \uparrow	0.731	0.706	0.698
	<i>MSE</i> \downarrow	0.121	0.125	0.156
no beard	<i>PSNR</i> \uparrow	19.433	20.389	28.463
	<i>SSIM</i> \uparrow	0.763	0.728	0.706
	<i>MSE</i> \downarrow	0.108	0.110	0.051
beard	<i>PSNR</i> \uparrow	21.425	20.259	20.769
	<i>SSIM</i> \uparrow	0.798	0.769	0.713
	<i>MSE</i> \downarrow	0.088	0.109	0.092
no smile	<i>PSNR</i> \uparrow	19.869	19.108	19.463
	<i>SSIM</i> \uparrow	0.750	0.739	0.706
	<i>MSE</i> \downarrow	0.104	0.118	0.105
smile	<i>PSNR</i> \uparrow	22.817	22.474	22.701
	<i>SSIM</i> \uparrow	0.831	0.801	0.833
	<i>MSE</i> \downarrow	0.073	0.089	0.076

ter preserves most of the remaining image characteristics. While the competitive approaches excel at modifying attributes with a significant impact on the image structure (such as gender), StyleAE showcases superior performance in manipulating more subtle attributes (such as facial expressions or an addition of the eyeglasses). One can see the comparison of such modifications in Fig. 3. In the case smile attribute the results are comparable. The outcomes demonstrate that our approach can generate images with the desired changes without considerably altering other aspects of the image, as evident from Fig. 4. This highlights the ability of our method to simplify the latent space and produce more meaningful and controllable images.

Moreover, our method benefits significantly from the simplicity of the AutoEncoder approach. It requires fewer parameters and less complex architecture compared to other models, making it less time-consuming and easier to train. In fact, our model achieved comparable results with state-of-the-art flow-based models with only 100 epochs of training. On the other hand, these models require significantly more complicated setups and longer training times, as reported in respective papers. Furthermore, our method was more efficient in generating the same number of images compared to both models, as shown in Table 3.

Table 5. Examples of attributes modification for AFHQv2 dataset. Pose, shape and fur colour seem to be inherited from the input image. The style transfer is not ideal i.e., the quality of the input image features is not reliably copied. Finer traversing over the requested latent attribute could solve that particular issue.



4.4 Evaluation on Animal Faces

For the assessment of StyleAE additional potential, a qualitative evaluation was performed utilizing the AFHQv2 dataset. This dataset comprises high-quality images featuring animal faces, categorized into specific classes, namely cats, dogs, and wild animals.

Setup: Given the unavailability of suitable classification tools, the training of StyleAE on generated images, similar to the FFHQ dataset, was unfeasible. In order to overcome this obstacle, we applied a projection technique, as presented in [1], to convert real images of animal faces from the AFHQv2 dataset into latent

vectors of StyleGAN. These transformed vectors, marked with labels denoting the original animal class, were employed to train our model. In this specific experiment, the training of StyleAE was conducted for 100 epochs, utilizing the $d_A^S(c_k, 1)$ method described in Eq. (4).

It is essential to highlight that a direct comparison between our results and those of other methods was not feasible in this setting, because previous methods were not trained nor evaluated on the AFHQv2 dataset. Retraining the other models on the reconstructed StyleGAN latent vectors projections dataset would entail potential risks associated with the need to optimize their parameters for our specific task.

Results: In this experiment, we aimed to explore the feasibility of achieving style transfer, specifically in terms of animal type, through the modification of racial attributes.

Our empirical outcomes illustrate the effectiveness of StyleAE plugin. This approach adeptly facilitates the transfer of specific animal classes onto generated animal faces, all while preserving integral structural characteristics like fur color and animal posture, as shown in Table 5. This outcome attests to the robustness and effectiveness of our method in producing images that conform to the desired attributes while retaining essential features.

The generated images display a high level of diversity and realism, highlighting the versatility of our approach. Notably, these results stand out considering the challenges inherent in the animal faces dataset, which encompasses a wide array of shapes and textures.

The results of this experiment suggest that it has the potential to be applied to a variety of image-generation tasks, including those involving complex and diverse datasets.

5 Conclusion

This paper presents StyleAE, a novel method utilizing AutoEncoders to modify StyleGAN latent space efficiently. StyleAE is computationally efficient and capable of generating high-quality images with controllable features across diverse datasets.

Our experiments show that StyleAE achieves comparable attribute modification accuracy to state-of-the-art flow-based models while being less intrusive to other image characteristics. The model's simplicity and time efficiency are key advantages.

Future research could focus on enhancing StyleAE's latent space disentanglement for more precise image control. Exploring advanced optimization methods for model fine-tuning and assessing StyleAE's efficacy in various generative model settings are promising directions.

Acknowledgements. This research has been supported by the flagship project entitled “Artificial Intelligence Computing Center Core Facility” from the Priority Research Area Digi World under the Strategic Programme Excellence Initiative at Jagiellonian University. The work of M. Śmieja was supported by the National Science Centre (Poland), grant no. 2022/45/B/ST6/01117.

References

1. Abdal, R., Qin, Y., Wonka, P.: Image2StyleGAN: how to embed images into the StyleGAN latent space? CoRR abs/1904.03189 (2019)
2. Abdal, R., Zhu, P., Femiani, J., Mitra, N.J., Wonka, P.: CLIP2StyleGAN: unsupervised extraction of StyleGAN edit directions. CoRR abs/2112.05219 (2021)
3. Abdal, R., Zhu, P., Mitra, N.J., Wonka, P.: StyleFlow: attribute-conditioned exploration of StyleGAN-generated images using conditional continuous normalizing flows. CoRR abs/2008.02401 (2020)
4. Cha, J., Thiyyagalingam, J.: Disentangling autoencoders (DAE) (2022)
5. Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., Abbeel, P.: InfoGAN: interpretable representation learning by information maximizing generative adversarial nets (2016)
6. Choi, Y., Uh, Y., Yoo, J., Ha, J.: StarGAN v2: diverse image synthesis for multiple domains. CoRR abs/1912.01865 (2019)
7. Deng, J., Guo, J., Zafeiriou, S.: ArcFace: additive angular margin loss for deep face recognition. CoRR abs/1801.07698 (2018)
8. Dinh, L., Sohl-Dickstein, J., Bengio, S.: Density estimation using real NVP (2016)
9. Gao, Y., et al.: High-fidelity and arbitrary face editing. CoRR abs/2103.15814 (2021)
10. Goodfellow, I.J., et al.: Generative adversarial networks (2014)
11. Härkönen, E., Hertzmann, A., Lehtinen, J., Paris, S.: GANSpace: discovering interpretable GAN controls. CoRR abs/2004.02546 (2020)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CoRR abs/1512.03385 (2015)
13. He, Z., Zuo, W., Kan, M., Shan, S., Chen, X.: AttGAN: facial attribute editing by only changing what you want. IEEE Trans. Image Process. **28**(11), 5464–5478 (2019)
14. Ho, J., Chen, X., Srinivas, A., Duan, Y., Abbeel, P.: Flow++: improving flow-based generative models with variational dequantization and architecture design. CoRR abs/1902.00275 (2019)
15. Karras, T., et al.: Alias-free generative adversarial networks. In: Proceedings of the NeurIPS (2021)
16. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. CoRR abs/1812.04948 (2018)
17. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of StyleGAN. CoRR abs/1912.04958 (2019)
18. Kingma, D.P., Dhariwal, P.: Glow: generative flow with invertible 1×1 convolutions (2018)
19. Kingma, D.P., Rezende, D.J., Mohamed, S., Welling, M.: Semi-supervised learning with deep generative models. CoRR abs/1406.5298 (2014)
20. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint [arXiv:1312.6114](https://arxiv.org/abs/1312.6114) (2013)

21. Lample, G., Zeghidour, N., Usunier, N., Bordes, A., Denoyer, L., Ranzato, M.: Fader networks: manipulating images by sliding attributes. CoRR abs/1706.00409 (2017)
22. Liu, R., Liu, Y., Gong, X., Wang, X., Li, H.: Conditional adversarial generative flow for controllable image synthesis. CoRR abs/1904.01782 (2019)
23. Preechakul, K., Chatthee, N., Wizadwongsa, S., Suwajanakorn, S.: Diffusion autoencoders: toward a meaningful and decodable representation. In: CVPR (2022)
24. Shen, Y., Yang, C., Tang, X., Zhou, B.: InterFaceGAN: interpreting the disentangled face representation learned by GANs. CoRR abs/2005.09635 (2020)
25. Suwała, A., Wójcik, B., Proszewska, M., Tabor, J., Spurek, P., Śmieja, M.: Face identity-aware disentanglement in StyleGAN. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 5222–5231 (2024)
26. Tewari, A., et al.: PIE: portrait image embedding for semantic control. ACM Trans. Graph. **39**(6), 1–14 (2020)
27. Vidal, A., Wu Fung, S., Tenorio, L., Osher, S., Nurbekyan, L.: Taming hyperparameter tuning in continuous normalizing flows using the JKO scheme. Sci. Rep. **13**, 4501 (2023)
28. Wang, H., Yu, N., Fritz, M.: Hijack-GAN: unintended-use of pretrained, black-box GANs. CoRR abs/2011.14107 (2020)
29. Wołczyk, M., et al.: PluGeN: multi-label conditional generation from pre-trained models. In: AAAI 2022 (2022)



SEE: Spherical Embedding Expansion for Improving Deep Metric Learning

Binh Minh Le^{ID} and Simon S. Woo^(✉)^{ID}

Dept. of Computer Science & Engineering, Sungkyunkwan University, Suwon, South Korea
{bmle, swoo}@g.skku.edu

Abstract. The primary goal of deep metric learning is to construct a comprehensive embedding space that can effectively represent samples originating from both intra- and inter-classes. Although extensive prior work has explored diverse metric functions and innovative training strategies, much of this work relies on default training data. Consequently, the potential variations inherent within this data remain largely unexplored, constraining the model’s robustness to unseen images. In this context, we introduce the Spherical Embedding Expansion (SEE) method. SEE aims to uncover the latent semantic variations in training data. Especially, our method augments the embedding space with synthetic representations based on Max-Mahalanobis distribution (MMD) centers, which maximize the dispersion of these synthetic features without increasing computational costs. We evaluated the efficacy of SEE on four renowned standard benchmarks for the image retrieval task. The results demonstrate that SEE consistently enhances the performance of conventional methods when integrated with them, setting a new benchmark for deep metric learning performance across all settings. Particularly, the proposed method reveals its potency, especially when training with a low-dimensional embedding space and a large number of classes.

Keywords: Deep Metric Learning · Max-Mahalanobis Distribution

1 Introduction

Learning to create a semantic embedding space that possesses both discriminative and generalized properties has been extensively studied across a variety of machine learning tasks. Such tasks encompass image retrieval [18], face verification [6], person re-identification [4], few-shot learning [27], and representation learning [12]. Consequently, deep metric learning, facilitated by neural networks, has garnered significant attention. Its objective is to learn an efficient embedding space in which semantically similar sample are pulled close together, while dissimilar ones are pushed far apart. To this end, various training loss functions, which are broadly categorized into pair-based and proxy-based methods, have been proposed.

In addition to refining the loss function, the development of sampling strategies is also pivotal in enhancing performance. Prevailing methods [38] emphasize the mining of hard samples. However, this often results in a biased model, as it overlooks the majority of easy samples [38,42]. To address this critical issue, recent research [8,41,42] has suggested the use of generative adversarial networks or autoencoders to synthesize

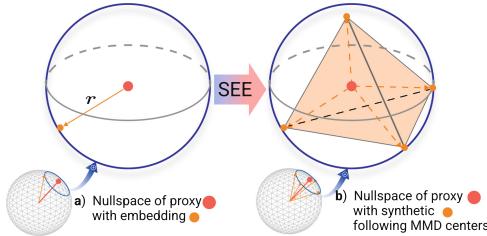


Fig. 1. Motivation of SEE. SEE aims to discover the latent space of training data by synthesizing new embedding vectors ($n_{\text{aug}} = 3$) derived from the nullspace $\mathcal{S}_{||r||}^{d-2}$ of a proxy. The synthesized samples follow the MMD properties that enrich the representation space, benefiting for optimization procedure.

challenging samples using easy ones. Although promising, these approaches have drawbacks, such as model size and optimization issues. Other studies [13, 19] have attempted to synthesize these challenging samples directly from the original embedding, yet they are predominantly constrained to paired-based techniques.

In this paper, we introduce a novel proxy-based synthesis technique in the embedding space of deep metric learning, termed as Spherical Embedding Expansion (SEE). As depicted in Fig. 1, given an embedding and its corresponding proxy anchor, our approach initially explores the proxy’s null space, which is represented as a sphere with a radius of r . This ensures consistent distances of the synthetic samples to the anchor. Subsequently, the synthetic embeddings are generated according to the Max-Mahalanobis distribution (MMD) mean vectors [25] (hereinafter referred to as *MMD centers*), allowing for enabling a comprehensive exploration of the embedding space. Our method is straightforward and seamlessly integrates with existing proxy-based metric learning losses. Notably, implementing our approach neither alters the embedding network architecture nor impacts its training speed. Nonetheless, it enhances overall performance, especially in scenarios with low-dimensional spaces, having a large number of classes. Our contributions in this paper are summarized as follows¹:

- We propose a novel method that augments the embedding space during training by constructing synthetic feature points aligned with MMD centers.
- Through seamless integration, SEE improves proxy-based metric learning losses across numerous backbones and benchmarks without adding parameters.
- SEE excels at densely navigating embedding space, significantly boosting performance, particularly in low-dimensional spaces with datasets that have a large number of training classes.

¹ Our code is available at <https://github.com/leminhbinh0209/Spherical-Expansion>.

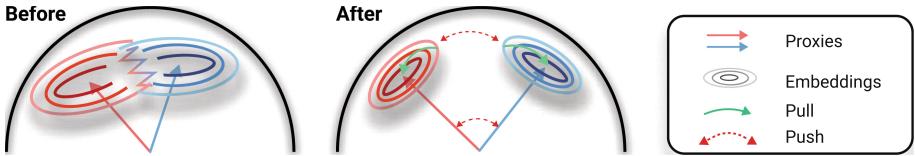


Fig. 2. A schematic representation of our learning objective. **Left:** Training with a constrained dataset can result in under-represented regions that fuse the representations of two distinct classes. **Middle:** SEE enhances intra-class samples, leading to denser clustering within each class while ensuring distinct separations between different classes.

2 Related Works

Deep Metric Learning. Metric learning losses have been developed to enhance the similarity of feature representations within a designated metric space. Initial methodologies employed pair-based loss functions, striving to refine the embedding space by diminishing the distance between samples of the same class (positive pairs) and amplifying the distance for samples from distinct classes (negative pairs) [5]. More recent advancements in pair-based loss functions have considered relationships between pairs [18, 35, 36]. However, these methods encounter extreme computational complexity during training, and the difficulty in drawing sufficient informative tuples, as well as the risk of overfitting [14, 38]. To overcome these limitations, proxy-based approaches have been introduced in recent studies [1, 6, 17]. A proxy is a representative of a class or a subset of the training dataset, and it is optimized simultaneously with the networks. Thus, samples from the same classes are pushed near to their proxies but distant from those of other classes. However, while the objective of metric learning is to produce a well-structured embedding space with fast convergence, it heavily relies on the limited training set and may not encompass the entire spectrum of variations present in the validation set. Our proposed method generates samples to train with augmented information by fully discovering the subspace of proxies in the embedding space.

Sample Generation. Recently, the generation of challenging samples that can potentially augment pair-based metric learning losses has garnered considerable interest of researchers [8, 41, 42]. The central objective of these studies revolves around the creation of hard samples enriched with supplemental semantic information derived from an abundant pool of easy negative samples. Among the pioneering works in this domain are DAML [8] and HTG [41] that both leverage GAN networks. Subsequently, Zheng *et al.* [42] introduced an autoencoder-based hardness-aware deep metric learning framework that generates label-preserving synthetic samples with adjustable degrees of difficulty. Later on, researchers [13, 19] introduced linear synthesis and symmetrical synthesis methods. Notably, while these techniques do not rely on a secondary network, their utility is primarily confined to pair-based metric learning losses. On the other hand, our proposed method supports various proxy-based approaches, which have consistently showcased their superiority over their pair-based counterparts.

3 Method

3.1 Preliminary

Consider a deep neural network, denoted as $f : \mathcal{D} \rightarrow \mathcal{Z}$, which maps an input data space \mathcal{D} to an embedding space \mathcal{Z} belonging to a unit d -dimensional hypersphere \mathbb{S}^{d-1} . Let $y \in \mathcal{Y} = \{1, \dots, C\}$ be the label of an embedding feature z . We define a set of normalized proxies as $\mathbf{w} = \{w_1, w_2, \dots, w_C\}$ and formulate a general proxies-based loss function for metric learning as follows:

$$\mathcal{L}_{\text{ML}} = \mathbb{E}_{(z,y) \sim (\mathcal{Z}, \mathcal{Y})} \ell(z|y, \mathbf{w}). \quad (1)$$

In Eq. 1, the normalized softmax loss [33] and its variations [6, 29, 34] are widely used as classification loss ℓ due to their interpretability and performance.

3.2 Spherical Embedding Expansion

Motivation. Our primary purpose of metric learning is to construct a robust and efficient embedding space for *unseen* samples. A common approach is to apply data augmentation techniques such as Mixup [40]. However, these techniques require forwarding augmented inputs to obtain augmented representations. In contrast, we introduce a plug-and-play module, Spherical Embedding Expansion (SEE), which operates in the embedding space \mathcal{Z} . This method facilitates a more efficient augmentation process by allowing for the forwarding of un-augmented inputs and performing augmentations directly on the output representations. The conceptual illustration of SEE is provided in Fig. 2. In fact, the main motivation of our work is to address the following requirements: *Given an embedding vector z and its corresponding proxy w_y , how can we efficiently synthesize n_{aug} additional embedding vectors z_i^* that satisfy the following conditions:* (1) *The distances between the synthetic vectors and w_y , denoted as $d_{(y,i)}$, remain unchanged.* (2) *The distances between any two synthetic vectors, denoted as $d_{(i,j)}$, are maximized, resulting in optimal dispersion of synthetic vectors in the space.*

The first requirement ensures that the synthetic vectors maintain similar quality to the original input and do not become outliers, or too close to their proxies. The second condition aims to diversify the distribution of the synthetic vectors in the embedding space, enabling the proxies of other classes more challenging and pushing those classes further away from their proxies.

Method. To ensure the first requirement, we define a $\|r\|$ -radius $(d-1)$ -dimensional hypersphere as: $\mathcal{S}_{\|r\|}^{d-2} = \{\mu | \mu \perp w_y \wedge \|\mu\| = \|r\|\}$, where $r = z - \langle w_y, z \rangle \cdot w_y$. This space $\mathcal{S}_{\|r\|}^{d-2}$ represents the null space of w_y , and r is the projection of z onto this defined null space. As a result, for any $\mu \in \mathcal{S}_{\|r\|}^{d-2}$, a synthetic vector formed by $z^* = \langle w_y, z \rangle \cdot w_y + \mu$ will satisfy $d(y, i) = d_y$. In practice, basis of this space can be constructed using Gram-Schmidt process.

To generate a set of synthetic vectors z^* , one approach is to randomly sample n_{aug} vectors μ from the hyper-spherical space $\mathcal{S}_{||r||}^{d-2}$ and translate them to z^* . However, randomly sampling n_{aug} vectors when $n_{\text{aug}} \ll d$ may not efficiently utilize the space. Conversely, if we choose a large value of n_{aug} , it will scale up the mini-batch size and affect computational efficiency. Hence, to fully utilize the space $\mathcal{S}_{||r||}^{d-2}$ while maintaining efficiency, we need to satisfy the second requirement. This requirement aims to maximize the distance between any two synthetic vectors and achieve optimal dispersion in the embedding space. Inspired by the above analysis, we propose the Max-Mahalanobis center sampling method to induce high-density regions in the hyper-spherical space $\mathcal{S}_{||r||}^{d-2}$, where the MMD [25] is a mixture of Gaussian distributions with an identity covariance matrix and K preset centers denoted as $\mu^* = \{\mu_i^*\}_{K}$. The MMD centers are created based on the criterion $\mu^* = \arg \min_{\mu} \max_{i \neq j} \langle \mu_i, \mu_j \rangle$. This criterion aims to maximize the smallest angle between any two centers, resulting in the most dispersion of the centers across the entire hyper-spherical space [25]. Previous work [25] introduced a fixed set of μ^* . However, in our case, the centers vary depending on $r = \mu_1^*$, which is the image of z in the defined null space as illustrated in Fig. 1. Additionally, the set of centers must satisfy the constraint $\mu_i^* \perp w_y$. To overcome this challenge, we propose a novel algorithm outlined in Algorithm 1 to generate optimal expanded vectors following the centers of MMD within the constrained space $\mathcal{S}_{||r||}^{d-2}$. The main difference between Algorithm 1 and the *GenerateOptMeans* algorithm in [25] are the initialization of $\mu_1^* = r/||r||$ vs. $\mu_1^* = e_1$ (one-hot vector), and subsequent MMD centers formalized in lines 2nd – 5th. Next, in the 7th line, centers are re-scaled so that $||\mu_i^*|| = ||r||$. By using Algorithm 1, one can easily prove that the set $\{\mu_i^*\}_{n_{\text{aug}}+1}$ are MMD centers, i.e., with $C = ||r||^2$,

$$\mu_i^{*T} \mu_j^* = \begin{cases} C, & i = j \\ -C/n_{\text{aug}}, & i \neq j \end{cases}, \quad (2)$$

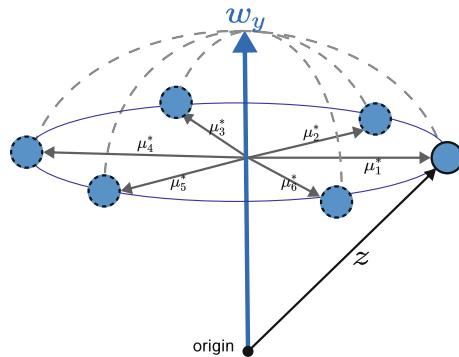


Fig. 3. Illustration of feature point generation. The solid boundary represents the original feature point, while the dotted boundary indicates its synthetic counterparts. For illustration, we depict $\mathcal{S}_{||r||}^{d-2}$ (which represents a hypersphere) as a 2D blue circle.

but they are more flexible than [25] in terms of initialization of μ_1^* . Hence, the *GenerateOptMeans* algorithm in [25] is solely used as a regularization and inapplicable to synthesize new embedding vectors in our case. Consequently, the optimal sampled vectors, as shown in Fig. 3, are produced as (see line 8th of Algorithm 1): $z_i^* = \langle w_y, z \rangle \cdot w_y + \mu_i^*$.

Although the synthetic embedding vectors can diversity its metric space, early applying the expansion can hinder model's optimization. Inspired by curriculum training scheme [2, 16], we selectively apply our method on top-k embedding vector z 's such that its d_y in top-k smallest in one mini-batch, denoted as \mathcal{M}_k , where k is monotonously increasing after epochs. Therefore, at epoch t^{th} , we have the loss function for synthetic vectors as follows:

$$\mathcal{L}_{\text{SEE}} = \mathbb{E}_{(z,y) \sim (\mathcal{Z}, \mathcal{Y}), d_y \in \mathcal{M}_k} \left[\sum_{n_{\text{aug}}} \ell(z_i^* | y, \mathbf{w}) \right]. \quad (3)$$

Overall Objective. Although our approach can help learning model to be more robust by diversely and optimally exploring the embedding space, it is important to note that a metric learning loss still play crucial roles as it utilizes the ground-truth labels for supervised training. The overall training loss for our proposed approach is formulated as follows:

$$\mathcal{L} = \mathcal{L}_{\text{ML}} + \lambda \mathcal{L}_{\text{SEE}}, \quad (4)$$

where λ is a hyper-parameter that balances the contribution of the original embedding and the synthetic vectors. It is important to note that our proposed approach does not require any modification to the loss function. It can be used as a plug-and-play module in the training process, introducing negligible computational cost.

Algorithm 1. Generate optimal synthetic samples following MMD centers.

Require: $z, w_y \in \mathcal{S}^{d-1}$ are embedding vector and its corresponding proxy vector, and the number of expansion samples n_{aug} .

- 1: Initialization: Let $r = z - \langle w_y, z \rangle \cdot w_y$, $\mathbf{V} = \{v_0, v_1, \dots, v_{n_{\text{aug}}+1}\}$, in which $v_0 = w_y$, $v_1 = r / \|r\|$, and $v_{i>1}$ are normalized vectors generated by Gram-Schmidt process sequentially. Let $\mu_1^* = v_1$.
- 2: **for** $k = 2$ to $n_{\text{aug}} + 1$ **do**
- 3: $\mu_k^* = \sum_{i=1}^k \alpha_{ki} v_i$, where
- 4:
$$\begin{cases} \alpha_{k1} = -1/n_{\text{aug}} \\ \alpha_{kj} = -\left(1 + n_{\text{aug}} \cdot \sum_{i=1}^{j-1} \alpha_{ki} \alpha_{ji}\right) / (n_{\text{aug}} \cdot \alpha_{jj}) \\ \alpha_{kk} = \sqrt{1 - \sum_{i=1}^{k-1} \alpha_{ki}^2} \end{cases}$$
- 5: **end for**
- 6: **for** $k = 1$ to $n_{\text{aug}} + 1$ **do**
- 7: $\mu_k^* = \|r\| \cdot \mu_k^*$
- 8: $z_k^* = \langle w_y, z \rangle \cdot w_y + \mu_k^*$
- 9: **end for**
- 10: **return** The optimal expanded vectors $z_i^*, i \in \overline{2, \dots, n_{\text{aug}} + 1}$.

Discussion. Incorporating SEE into a deep metric model yields two pronounced effects. Firstly, the synthetic embeddings foster a more generalized model; trivial samples, augmented with additional information, now facilitate a more comprehensive exploration of under-represented regions. Secondly, the generation of hard negative samples effectively distances the representations from the anchors of other classes. This results in an amplified inter-class differentiation and a more compact intra-class representation.

Consider the normalized softmax loss in Eq. 1 as a simple example. This loss can be expressed in a more generalized form as:

$$\begin{aligned}\ell(z|y, \mathbf{w}) &= -\tau \log \frac{e^{-d_y/\tau}}{\epsilon \cdot e^{-d_y/\tau} + \sum_{j \neq y}^C e^{-d_j/\tau}} = \tau \log [\epsilon + \sum_{j \neq y} e^{(d_y - d_j)/\tau}] \\ &= \tau \text{Softplus} [\text{LSE}_{j \neq y}(d_y - d_j)/\tau],\end{aligned}\quad (5)$$

where $\epsilon \geq 0$, and d_j represents the distance between z and the proxy w_j , such as $d_j = -\langle w_j, z \rangle = -\cos \theta_j$, and $\text{Softplus}(x) = \log(\epsilon + e^x)$. As illustrated in Fig. 3, the synthetic feature points z_i^* maintain consistent distances to their respective proxy anchors; that is, all d_y 's are identical. Furthermore, the Log-Sum-Exp (LSE) function serves as a smooth approximation to the maximum function [23]. Consequently, by probing various directions using MMD centers with optimal dispersion, our SEE is adept at effectively pushing the most challenging negative anchors (represented by the smallest d_j) with every synthetic feature point.

4 Experiments

4.1 Experiment Setting

Datasets. We use the following four popular benchmark datasets for evaluating our method: 1) CUB-200-2011 (CUB) [32], 2) Cars-196 (Cars) [20], 3) Stanford Online Product (SOP) [24], 4) In-shop Clothes Retrieval (In-Shop) [21].

Metrics. We utilize the Recall@k metric which is used to quantify the fraction of query images that have, within their k-nearest neighbors in the embedding space, at least one instance belonging to the same class.

Networks. To ensure fair comparison with prior works, we adopt ResNet50 [15] (R) with an embedding size of $d = 512$ and three versions of vision transformer architecture: DeiT-S [30] (D), DINO [3] (DN), and ViT-S [7] (V), each with embedding sizes $d = 128$ and $d = 384$. All the backbones are pre-trained on the ImageNet dataset. The linear projection layer used for patch embedding in the variants of the vision transformer is kept frozen during training.

Training. The embedding models are optimized using the AdamW optimizer [22], with a learning rate of 10^{-5} for ViT-S and DeiT-S, and 5×10^{-6} for DINO models. For the ResNet50 backbone, we adopt the training settings from [17]. Training images are randomly resized and cropped to 224×224 using bicubic interpolation and are subjected to random horizontal flips. For test images, we resize them to 256×256 and then apply a center crop of size 224×224 . Across our experiments, we use a consistent set of hyperparameters: $\lambda = 0.2$, with $n_{\text{aug}} = 16$ for CNN backbones ($d = 512$), and $n_{\text{aug}} = 8$ and $n_{\text{aug}} = 16$ for ViT backbones ($d = 128$ and $d = 384$, respectively). Our training settings is consistent with prior work without additional training epoch.

Table 1. Performance of metric learning methods on the four datasets. The employed network architectures are represented using abbreviations: R signifies ResNet50 [15], B denotes Inception with BatchNorm, De corresponds to DeiT [30], DN is representative of DINO [3], and V stands for ViT [7]. It should be highlighted that ViT [7] is pre-trained on ImageNet-21k. † indicates models using larger input images.

Methods	Arch.	CUB			Cars			SOP			In-Shop		
		R@1	R@2	R@4	R@1	R@2	R@4	R@1	R@10	R@100	R@1	R@10	R@20
<i>Backbone architecture: CNN</i>													
NSoftmax [39]	R ¹²⁸	56.5	69.6	79.9	81.6	88.7	93.4	75.2	88.7	95.2	86.6	96.8	97.8
MIC [28]	R ¹²⁸	66.1	76.8	85.6	82.6	89.1	93.2	77.2	89.4	94.6	88.2	97.0	-
XBM [37]	R ¹²⁸	-	-	-	-	-	-	80.6	91.6	96.2	91.3	97.8	98.4
XBM [37]	B ⁵¹²	65.8	75.9	84.0	82.0	88.7	93.1	79.5	90.8	96.1	89.9	97.6	98.4
HTL [11]	B ⁵¹²	57.1	68.8	78.7	81.4	88.0	92.7	74.8	88.3	94.8	80.9	94.3	95.8
MS [36]	B ⁵¹²	65.7	77.0	86.3	84.1	90.4	94.0	78.2	90.5	96.0	89.7	97.9	98.5
SoftTriple [26]	B ⁵¹²	65.4	76.4	84.5	84.5	90.7	94.5	78.6	86.6	91.8	-	-	-
PA [17]	B ⁵¹²	68.4	79.2	86.8	86.1	91.7	95.0	79.1	90.8	96.2	91.5	98.1	98.8
NSoftmax [39]	R ⁵¹²	61.3	73.9	83.5	84.2	90.4	94.4	78.2	90.6	96.2	86.6	97.5	98.4
†ProxyNCA++ [29]	R ⁵¹²	69.0	79.8	87.3	86.5	92.5	95.7	80.7	92.0	96.7	90.4	98.1	98.8
Hyp [10]	R ⁵¹²	65.5	76.2	84.9	81.9	88.8	93.1	79.9	91.5	96.5	90.1	98.0	98.7
SEE (ours)	R ⁵¹²	69.3	79.0	87.3	88.5	93.4	95.9	80.3	91.5	96.5	92.8	98.3	98.8
<i>Backbone architecture: ViT</i>													
IRT _R [9]	De ¹²⁸	72.6	81.9	88.7	-	-	-	83.4	93.0	97.0	91.1	98.1	98.6
Hyp [10]	De ¹²⁸	74.7	84.5	90.1	82.1	89.1	93.4	83.0	93.4	97.5	90.9	97.9	98.6
SEE (ours)	De ¹²⁸	75.1	84.1	90.1	85.2	91.5	94.8	83.0	93.1	97.2	91.2	98.0	98.6
Hyp [10]	DN ¹²⁸	78.3	86.0	91.2	86.0	91.9	95.2	84.6	94.1	97.7	92.6	98.4	99.0
SEE (ours)	DN ¹²⁸	78.8	86.5	91.6	89.0	93.6	96.3	84.8	94.1	97.5	92.6	98.6	99.0
Hyp [10]	V ¹²⁸	84.0	90.2	94.2	82.7	89.7	93.9	85.5	94.9	98.1	92.7	98.4	98.9
SEE (ours)	V ¹²⁸	84.1	90.2	93.5	86.8	91.7	95.1	85.9	94.7	97.9	92.8	98.6	99.1
IRT _R [9]	De ³⁸⁴	76.6	85.0	91.1	-	-	-	84.2	93.7	97.3	91.9	98.1	98.9
DeiT-S [30]	De ³⁸⁴	70.6	81.3	88.7	52.8	65.1	76.2	58.3	73.9	85.9	37.9	64.7	72.1
Hyp [10]	De ³⁸⁴	77.8	86.6	91.9	86.4	92.2	95.5	83.3	93.5	97.4	90.5	97.8	98.5
SEE (ours)	De ³⁸⁴	78.3	86.5	91.9	88.8	93.7	96.3	83.6	93.4	97.4	91.7	98.1	98.7
DNO [3]	DN ³⁸⁴	70.8	81.1	88.8	42.9	53.9	64.2	63.4	78.1	88.3	46.1	71.1	77.5
Hyp [10]	DN ³⁸⁴	80.9	87.6	92.4	89.2	94.1	96.7	85.1	94.4	97.8	92.4	98.4	98.9
SEE (ours)	DN ³⁸⁴	81.9	88.8	92.9	91.5	95.2	97.3	85.5	94.6	97.9	93.0	98.5	99.1
ViT-S [20]	V ³⁸⁴	83.1	90.4	94.4	47.8	60.2	72.2	62.1	77.7	89.0	43.2	70.2	76.7
Hyp [10]	V ³⁸⁴	85.6	91.4	94.8	86.5	92.1	95.3	85.9	94.9	98.1	92.5	98.3	98.8
SEE (ours)	V ³⁸⁴	85.8	91.4	94.6	88.8	93.8	96.4	86.3	95.0	98.2	93.2	98.6	99.1

4.2 Quantitative Results

In our evaluation, we benchmark the efficacy of our proposed approach against existing state-of-the-art methods across four canonical datasets. For these experiments, we employ the proxy anchor loss as our metric learning objective \mathcal{L}_{ML} . To ensure an equitable comparison, the results are systematically tabulated in Table 1, segmented based on the backbone architecture, specifically, CNN to ViT, and the embedding dimensions, namely 128, 384, and 512.

Our experimental findings underscore the effectiveness of our proposed methodology. When compared with other CNN-based techniques, our approach, utilizing ResNet50 as the backbone, consistently outperforms competitors across multiple datasets, with the exception of SOP. It is crucial to mention that ProxyNCA++ [29] employs a more expansive input size, and XBM [37] leverages an extensive memory bank to augment their training process. Nevertheless, our method still surpasses them in most datasets. Specifically, we observe improvements of 2% and 2.4% on the Cars and In-shop datasets, respectively, when compared to ProxyNCA++.

In the context of ViT-based experiments, our proposed methodology exhibits a discernible advantage over competing baselines, particularly with respect to R@1 scores, spanning various embedding dimensions. For instance, on the Cars dataset in comparison with the Hyp method [10], our approach registers performance gains of 3.5%, 3%, and 4.1% when employing the $D\ell^{128}$, DN^{128} , and V^{128} backbones, respectively. It is noteworthy that, when utilizing the DN^{128} backbone, our method surpasses all other approaches, even those configured with 512-dimensional CNN settings. Furthermore, even on challenging datasets such as SOP or In-shop, our technique continues to demonstrate marked improvements, even at reduced embedding dimensions like 128 and 384. This enhancement can be attributed to the optimal dispersion of synthetic feature points within the embedding space.

Table 2. Recall@k accuracy for proxy-based losses integrated with our synthesis method on the Cars dataset, using CNN-based networks with 512-dimensional embeddings.

Method	Arch.	R@1	R@2	R@4	R@8	R@16
NSoftmax [39]	R^{512}	84.2	90.4	94.4	96.9	-
NSoftmax+SEE	R^{512}	86.5	92.0	95.4	97.4	98.7
CosFace [34]	R^{512}	86.9	92.3	95.3	97.4	98.6
CosFace+SEE	R^{512}	87.1	92.5	95.4	97.5	98.7
ArcFace [6]	R^{512}	86.8	92.1	95.3	97.3	98.7
ArcFace+SEE	R^{512}	87.6	92.8	95.9	97.6	98.7
[†] ProxyNCA++ [29]	R^{512}	86.5	92.5	95.7	97.7	-
[†] ProxyNCA+++SEE	R^{512}	88.3	93.4	96.4	98.0	99.0
PA [17]	B^{512}	86.1	91.7	95.0	97.0	98.3
PA+SEE	B^{512}	86.2	91.9	95.2	97.2	98.4
PA [17]	R^{512}	87.7	92.7	95.5	97.3	98.4
PA+SEE	R^{512}	88.5	93.4	95.9	97.5	98.8

Table 3. Recall@1 accuracy of Proxy-Anchor (PA) [17] across various augmentation types, including our method, using the DeiT [30] network architecture with 128-dimensional embeddings.

Methods	CUB	Cars	SOP	In-Shop
PA[17]	74.7	84.3	82.3	90.4
PA+Mixup	75.0	86.2	82.2	91.0
PA+ManifoldMixup	74.4	85.4	81.9	90.5
PA+SEE (<i>random</i>)	74.6	85.1	82.4	91.2
PA+SEE	75.1	85.2	83.0	91.2

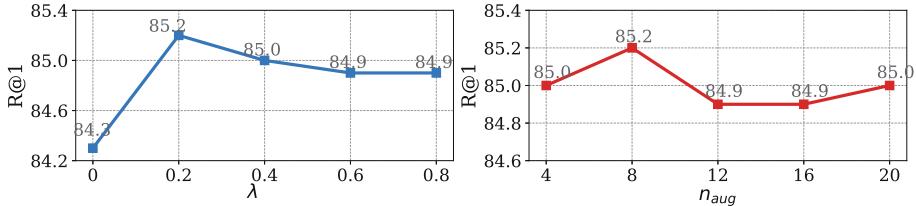


Fig. 4. Ablation studies: augmentation loss weight λ and number of synthesis points n_{aug} .
Note that $\lambda = 0$ denotes not using our SEE.

4.3 Ablation Studies

Proxy-based Losses. In this section, we demonstrate the enhancements achieved by our proposed synthesis method when applied to various proxy-based metric learning losses, specifically when integrated with CNN-based architectures. The detailed results are presented in Table 2. As evident from the table, our synthesis approach consistently enhances performance across different proxy-based losses, achieving up to a 1.8% improvement in Recall@1 accuracy. Furthermore, this enhancement persists even as we increase the number of neighbors, k , in Recall@ k .

Augmentation Types. In Table 3, we juxtapose our SEE with alternative augmentation strategies, notably Mixup [40] and ManifoldMixup [31], and our variation employing a randomized μ , when integrated with the Proxy-anchor [17] loss using De^{128} . Evidently, each methodology exhibits enhancements over the baseline Proxy-anchor. However, our proposal, characterized by the optimal dispersion of μ , consistently yields performance amplifications across the quartet of datasets. It is worth acknowledging that Mixup manifests a pronounced improvement on the Cars dataset. Yet, for other fine-grained datasets, namely CUB, SOP, and In-shop, its efficacy pales in comparison to our approach. This can be attributed to our method’s adeptness at augmenting based on the feature relationship within an image relative to its corresponding anchor.

Impact of Hyperparameters. We explore the influence of two critical hyperparameters on the efficacy of our method: the loss weight of SEE, denoted by λ , and the number of synthetic feature points, n_{aug} . Note that, as mentioned, the radius r depends on the distance between the embedding vector and its proxy, so we do not include it in our studies. For this analysis, we employ the De^{128} backbone and evaluate the performance on the Cars dataset. As depicted in Fig. 4, our method exhibits significant robustness to variations in these hyperparameters, implying consistent performance irrespective of their specific settings.

5 Conclusion

In this work, we have introduced a spherical embedding expansion technique for augmentation within the embedding space, designed to complement existing proxy-based

metric learning losses. Within this space, we augment a sample around its anchor by adhering to the MMD centers situated within the anchor's nullspace, thereby ensuring a thorough exploration. Our proposed method is streamlined and straightforward, obviating the need to modify model architecture or incur computational overhead. Empirical results reveal that our approach considerably enhances the efficacy of established proxy-based losses across a range of model architectures and benchmark datasets. Notably, it yields substantial improvements in challenging learning scenarios characterized by low-dimensional spaces and a vast number of classes.

Acknowledgements. This work was partly supported by Institute for Information & communication Technology Planning & evaluation (IITP) grants funded by the Korean government MSIT: (No. 2022-0-01199, Graduate School of Convergence Security at Sungkyunkwan University), (No. 2022-0-01045, Self-directed Multi-Modal Intelligence for solving unknown, open domain problems), (No. 2022-0-00688, AI Platform to Fully Adapt and Reflect Privacy-Policy Changes), (No. 2021-0-02068, Artificial Intelligence Innovation Hub), (No. 2019-0-00421, AI Graduate School Support Program at Sungkyunkwan University), and (No. RS-2023-00230337, Advanced and Proactive AI Platform Research and Development Against Malicious Deepfakes). Lastly, this work was supported by Korea Internet & Security Agency (KISA) grant funded by the Korea government (PIPC) (No.RS-2023-00231200, Development of personal video information privacy protection technology capable of AI learning in an autonomous driving environment).

References

1. Aziere, N., Todorovic, S.: Ensemble deep manifold similarity learning using hard proxies. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7299–7307 (2019)
2. Bengio, Y., Louradour, J., Collobert, R., Weston, J.: Curriculum learning. In: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 41–48 (2009)
3. Caron, M., et al.: Emerging properties in self-supervised vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9650–9660 (2021)
4. Chen, W., Chen, X., Zhang, J., Huang, K.: Beyond triplet loss: a deep quadruplet network for person re-identification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 403–412 (2017)
5. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), vol. 1, pp. 539–546. IEEE (2005)
6. Deng, J., Guo, J., Xue, N., Zafeiriou, S.: ArcFace: additive angular margin loss for deep face recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4690–4699 (2019)
7. Dosovitskiy, A., et al.: An image is worth 16×16 words: transformers for image recognition at scale. arXiv preprint [arXiv:2010.11929](https://arxiv.org/abs/2010.11929) (2020)
8. Duan, Y., Zheng, W., Lin, X., Lu, J., Zhou, J.: Deep adversarial metric learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2780–2789 (2018)
9. El-Nouby, A., Neverova, N., Laptev, I., Jégou, H.: Training vision transformers for image retrieval. arXiv preprint [arXiv:2102.05644](https://arxiv.org/abs/2102.05644) (2021)
10. Ermolov, A., Mirvakhabova, L., Khrulkov, V., Sebe, N., Oseledets, I.: Hyperbolic vision transformers: combining improvements in metric learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7409–7419 (2022)

11. Ge, W., Huang, W., Dong, D., Scott, M.R.: Deep metric learning with hierarchical triplet loss. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11210, pp. 272–288. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01231-1_17
12. Grill, J.B., et al.: Bootstrap your own latent-a new approach to self-supervised learning. *Adv. Neural. Inf. Process. Syst.* **33**, 21271–21284 (2020)
13. Gu, G., Ko, B.: Symmetrical synthesis for deep metric learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 10853–10860 (2020)
14. Harwood, B., Kumar BG, V., Carneiro, G., Reid, I., Drummond, T.: Smart mining for deep metric learning. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2821–2829 (2017)
15. 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)
16. Huang, Y., et al.: CurricularFace: adaptive curriculum learning loss for deep face recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5901–5910 (2020)
17. Kim, S., Kim, D., Cho, M., Kwak, S.: Proxy anchor loss for deep metric learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3238–3247 (2020)
18. Kim, S., Seo, M., Laptev, I., Cho, M., Kwak, S.: Deep metric learning beyond binary supervision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2288–2297 (2019)
19. Ko, B., Gu, G.: Embedding expansion: augmentation in embedding space for deep metric learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7255–7264 (2020)
20. Krause, J., Stark, M., Deng, J., Fei-Fei, L.: 3D object representations for fine-grained categorization. In: Proceedings of the IEEE International Conference on Computer Vision Workshops, pp. 554–561 (2013)
21. Liu, Z., Luo, P., Qiu, S., Wang, X., Tang, X.: DeepFashion: powering robust clothes recognition and retrieval with rich annotations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1096–1104 (2016)
22. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint [arXiv:1711.05101](https://arxiv.org/abs/1711.05101) (2017)
23. Nielsen, F., Sun, K.: Guaranteed bounds on the Kullback-Leibler divergence of univariate mixtures. *IEEE Signal Process. Lett.* **23**(11), 1543–1546 (2016)
24. Oh Song, H., Xiang, Y., Jegelka, S., Savarese, S.: Deep metric learning via lifted structured feature embedding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4004–4012 (2016)
25. Pang, T., Du, C., Zhu, J.: Max-mahalanobis linear discriminant analysis networks. In: International Conference on Machine Learning, pp. 4016–4025. PMLR (2018)
26. Qian, Q., Shang, L., Sun, B., Hu, J., Li, H., Jin, R.: SoftTriple Loss: deep metric learning without triplet sampling. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6450–6458 (2019)
27. Qiao, L., Shi, Y., Li, J., Wang, Y., Huang, T., Tian, Y.: Transductive episodic-wise adaptive metric for few-shot learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 3603–3612 (2019)
28. Roth, K., Brattoli, B., Ommer, B.: MIC: mining interclass characteristics for improved metric learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 8000–8009 (2019)

29. Teh, E.W., DeVries, T., Taylor, G.W.: ProxyNCA++: revisiting and revitalizing proxy neighborhood component analysis. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12369, pp. 448–464. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58586-0_27
30. 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)
31. Verma, V., Lamb, A., Beckham, C., Najafi, A., Mitliagkas, I., Lopez-Paz, D., Bengio, Y.: Manifold mixup: Better representations by interpolating hidden states. In: International Conference on Machine Learning, pp. 6438–6447. PMLR (2019)
32. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The caltech-UCSD birds-200-2011 dataset (2011)
33. Wang, F., Xiang, X., Cheng, J., Yuille, A.L.: NormFace: L2 hypersphere embedding for face verification. In: Proceedings of the 25th ACM International Conference on Multimedia, pp. 1041–1049 (2017)
34. Wang, H., et al.: CosFace: large margin cosine loss for deep face recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5265–5274 (2018)
35. Wang, J., et al.: Learning fine-grained image similarity with deep ranking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1386–1393 (2014)
36. Wang, X., Han, X., Huang, W., Dong, D., Scott, M.R.: Multi-similarity loss with general pair weighting for deep metric learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5022–5030 (2019)
37. Wang, X., Zhang, H., Huang, W., Scott, M.R.: Cross-batch memory for embedding learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6388–6397 (2020)
38. Wu, C.Y., Manmatha, R., Smola, A.J., Krahenbuhl, P.: Sampling matters in deep embedding learning. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2840–2848 (2017)
39. Zhai, A., Wu, H.Y.: Classification is a strong baseline for deep metric learning. arXiv preprint [arXiv:1811.12649](https://arxiv.org/abs/1811.12649) (2018)
40. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: beyond empirical risk minimization. arXiv preprint [arXiv:1710.09412](https://arxiv.org/abs/1710.09412) (2017)
41. Zhao, Yiru, Jin, Zhongming, Qi, Guo-jun, Lu, Hongtao, Hua, Xian-sheng: An adversarial approach to hard triplet generation. In: Ferrari, Vittorio, Hebert, Martial, Sminchisescu, Christian, Weiss, Yair (eds.) ECCV 2018. LNCS, vol. 11213, pp. 508–524. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01240-3_31
42. Zheng, W., Chen, Z., Lu, J., Zhou, J.: Hardness-aware deep metric learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 72–81 (2019)



Multi-modal Recurrent Graph Neural Networks for Spatiotemporal Forecasting

Nicholas Majeske and Ariful Azad^(✉)

Department of Intelligent Systems Engineering, Indiana University Bloomington, Bloomington,
IN, USA

{nmajeske, azad}@iu.edu

Abstract. The spatial and temporal dynamics of many real-world systems present a significant challenge to multi-variate forecasting where features of both forms, as well as their inter-dependencies, must be modeled correctly. State-of-the-art approaches utilize a limited set of exogenous features (outside the forecast variable) to model temporal dynamics and Graph Neural Networks, with pre-defined or learned networks, to model spatial dynamics. While much work has been done to model dependencies, existing approaches do not adequately capture the explicit and implicit modalities present in real-world systems. To address these limitations we propose MMR-GNN, a spatiotemporal model capable of (a) augmenting pre-defined (or absent) networks into optimal dependency structures (b) fusing multiple explicit modalities and (c) learning multiple implicit modalities. We show improvement over existing methods using several hydrology and traffic datasets. Our code is publicly available at <https://github.com/HipGraph/MMR-GNN>.

Keywords: forecasting · spatiotemporal · graph · clustering · traffic · hydrology

1 Introduction

Spatiotemporal systems such as streamflow [13], traffic flow [1, 11], and energy systems [14, 21] possess complex dynamics that span both space and time. Components of these systems often influence one another giving rise to an explicit or implicit dependency structure captured by a graph or network (e.g., road networks, river networks, etc.). In more complex cases, these networks generate *explicit multi-modal signals* such as static and dynamic node features, dynamic dependency structure, and dynamic graph features. Even further, *implicit multi-modal signals* can be observed within these features and must be captured for accurate forecasting. This paper presents MMR-GNN, a machine learning (ML) model capable of forecasting highly complex spatiotemporal systems containing explicit and implicit multi-modal signals.

As an example of multi-modal spatiotemporal systems, we consider the Wabash River basin which spans Indiana and parts of Ohio and Illinois in the USA (see Fig. 1(a)). In this dataset, various hydrological and meteorological features (streamflow, soil water content, temperature, etc.) were recorded from 1929 to 2013 at 1276 unique regions (sub-basins). This basin can be modeled as a graph where sub-basins

denote nodes and streamflow denotes the connectivity pattern. Each node in this graph has *explicit multi-modal signals* including spatiotemporal features such as streamflow, precipitation, snowfall, etc., static features, such as location in latitude-longitude, and shared dynamic features such as seasonality in time-of-year as shown in Fig. 1(b). Furthermore, sub-basin streamflow is the result of broad and localized properties including season, geology, local weather, and tributary streams from neighboring sub-basins. These properties give rise to *implicit multi-modal signals* where particular sub-basins show high feature correlation and form natural clusters (see Fig. 1(c)). Based on these observations in the Wabash River basin, we define a comprehensive system with three challenging aspects:

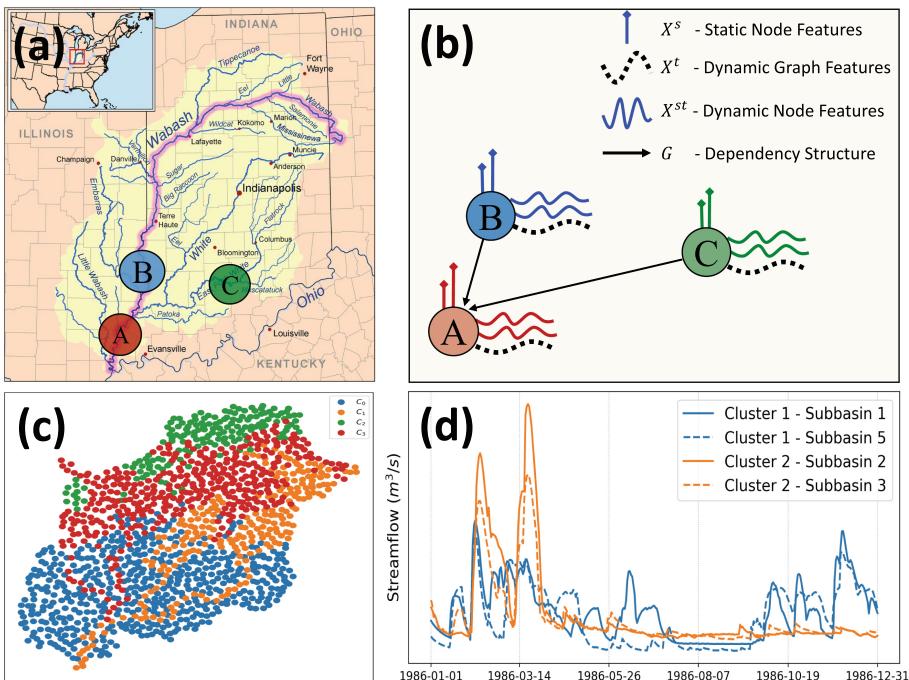


Fig. 1. (a) The Wabash River basin. Three of the 1276 sub-basins are marked with A, B, and C. (b) Three sub-basins (nodes) impact each other through a streamflow network G , generate dynamic node features \mathbf{X}^{st} (e.g., streamflow and precipitation), posses static node features \mathbf{X}^s (e.g., location), and exhibit dynamic graph features \mathbf{X}^t (e.g., seasonality). These features represent *explicit multi-modal signals* of the system. (c) 1276 sub-basins are grouped into four clusters where each cluster includes highly correlated sub-basins (e.g., red and orange clusters capture several major rivers). The strong intra-cluster feature correlation represents *implicit multi-modal signals* of the system. (d) Streamflow of two sub-basins in the orange cluster and two sub-basins in the blue cluster demonstrate this correlation which MMR-GNN can capture. (Color figure online)

- *Explicit multi-modality.* The system has explicit multi-modal spatial, temporal, and spatiotemporal signals as shown in Fig. 1(b).
- *Implicit multi-modality.* The system exhibits implicit modality where node groups exhibit strong feature correlation and form natural clusters as shown in Fig. 1(c).
- *Implicit and explicit dependencies.* Dependency among nodes can be both explicit (e.g., a streamflow network) and implicit (similar streamflow in distant sub-basins).

Despite tremendous progress in spatiotemporal ML, no state-of-the-art (SOTA) method can efficiently capture all explicit and implicit modalities and dependencies in a single model. Most recent methods are based on graph neural networks (GNNs) trained to predict future values of time series at different nodes of a given or learned graph [2, 18]. While GNN-based models perform well for simpler systems such as prediction of traffic speeds on road networks, they may perform sub-optimally on more complex system such as the streamflow network considered above.

Contributions. To capture all aspects of a spatiotemporal system, we developed MMR-GNN which uses recurrent graph neural networks (R-GNN) to capture spatial and temporal aspects. It also uses three new modules to capture explicit and implicit modalities present in the data. An explicit multi-modal module fuses data from multiple explicit modalities (spatial, temporal, and spatiotemporal) into rich embeddings. An implicit multi-modal module partitions nodes of the system into observed implicit modalities to learn their distinct dynamics. Finally, a dependency augmentation module learns an optimal dependency structure given an existing, or absent, graph structure. We combine these three modules in an encoder-decoder framework capable of faithfully capturing all components shown in Fig. 1. Even though MMR-GNN is designed for the most complex systems with multi-modal features, it is general enough to perform well for less-complex systems. We evaluate MMR-GNN on six real-world datasets and show good improvement in most cases.

2 Related Work

Much work has been done in forecasting spatiotemporal systems with methods ranging from pure temporal models [3, 5, 8] to complex architectures featuring temporal, geometrical, and attention-based embeddings [2, 18, 23]. Many methods have focused on traffic forecasting but recent work has shown applications in energy, medicine, and commerce [4, 24]. Table 1 organizes prior work according to their usage of multi-modal variables, graph structures, and graph convolutions.

Multi-modal Variable Correlation. Methods that can capture correlations among multiple time-series include GRU, TCN [3], DCRNN [11], T-GCN [22], A3T-GCN [1], ST-GCN [20], AST-GCN [7], AGCRN [2], and Graph WaveNet [19]. Most methods can model multi-variate relationships but several recent methods are still limited to the uni-variate case [4, 12, 23]. Less common among prior methods is the incorporation of features from spatial and temporal modalities which may also be highly correlated. The methods EA-LSTM [10] and MTGNN [18] integrate spatial features via the LSTM input gate and as priors for a graph learning module, respectively. GMAN [23] combines spatial and temporal features via a spatiotemporal embedding module but temporal features are limited to just two categorical variables. The literature appears to lack any comprehensive module for learning correlations across multiple explicit modalities.

Table 1. Related work grouped into three categories based on their usage of (a) variables from multiple explicit modalities, (b) graph dependency structures, and (c) graph convolution for geometrical embedding. Model components captured by our method are indicated in the last column.

Model components	Examples	MMR-GNN
Use multi-variate features	GRU, EA-LSTM [10], TCN [3], DCRNN [11], T-GCN [22], A3T-GCN [1], ST-GCN [20], AST-GCN [7], AGCRN [2], Graph WaveNet [19], MTGNN [18]	✓
Use spatial features	EA-LSTM, GMAN [23], MTGNN	✓
Use temporal features	GMAN	✓
Graphs not used	RNN, GRU, LSTM, EA-LSTM, TCN	✗
Pre-defined graphs	T/A3T-GCN, ST/AST-GCN, DCRNN, MTGNN	✓
Learned graphs	StemGNN [4], SCINet [12], GMAN, MTGNN, AGCRN	✓
GNN: one encoder	DCRNN, StemGNN, SCINet, GMAN, MTGNN	✗
GNN: shared encoders	AGCRN	✓
GNN: partitioned encoders	MMR-GNN (this paper)	✓

Dependency Structure Modeling. Proper modeling of the inter-node dependency structure is pivotal for accurate forecasting but these structures are often undefined or non-optimal. These scenarios have inspired researchers to construct graphs heuristically [11] or from the minimization of an objective function [18]. We distinguish prior work according to their inter-node dependency modeling including (a) no dependency (b) pre-defined dependency and (c) dependency learned through minimization of loss. Many methods utilize a pre-defined graph [1, 7, 20, 22] but they are ultimately limited by the optimality of that graph in forecasting. Given that there is no guarantee of optimality, it is unsurprising that recent work has moved towards learning the dependency structure entirely [2, 4, 12, 18, 19, 23]. SCINet [12] proposed the interaction module, StemGNN [4] proposed self-attention on GRU embeddings, while Graph WaveNet, MTGNN, and AGCRN proposed approximating the dependency via non-linear activations on learned node embeddings. Our method aims to capture the benefits of both aspects by augmenting pre-defined graphs (if they exist) to improve forecasting.

Graph Convolution Methods. GNN-based methods utilize various forms of graph convolution including classical GCN [9], Chebyshev polynomial expansion [6], and novel forms such as SCINet’s interaction module. However, these approaches mostly use a single parameter set when embedding across all nodes of the system. This presents a limitation for complex spatiotemporal systems where node subsets can exhibit significantly different dynamics in the form of multiple implicit modalities (see Fig. 1). AGCRN [2] addresses this issue by assigning a unique set of parameters (weights

and biases) to each node using a full-rank parameter tensor. To avoid over-fitting, the full-rank parameter tensor is approximated via factorization with factors placed in d -dimensional space representing d modalities. We argue that a multi-modal learning module, constrained to the binary mapping of node subsets to individual modalities instead of unconstrained mixture of modalities, improves performance and interpretability.

3 Methods

3.1 Problem Formulation

As shown in Fig. 1(b), we model a spatiotemporal system as N inter-dependent entities (e.g., traffic sensors, stream gauges, etc.) and their spatial and temporal features. The dependency structure is represented by a graph $G = (V, E)$, where V is the set of nodes (entities) with $|V| = N$, and E is the set of edges (dependencies). Every node has F^s static spatial features (e.g., latitude, longitude, etc.) and F^{st} spatiotemporal features that vary with time (e.g., streamflow, traffic speed, etc.). Additionally, the overall system has F^t dynamic features shared by all nodes (e.g., day of year, season, etc.). We represent all features of a system as $\mathcal{X} = \{\mathbf{X}^s, \mathbf{X}^t, \mathbf{X}^{st}\}$, where $\mathbf{X}^s \in \mathbb{R}^{N \times F^s}$ denotes spatial features, $\mathbf{X}^t = \{\mathbf{X}_0^t, \mathbf{X}_1^t, \dots, \mathbf{X}_\tau^t, \dots\}$ denotes temporal features with $\mathbf{X}_\tau^t \in \mathbb{R}^{F^t}$, and $\mathbf{X}^{st} = \{\mathbf{X}_0^{st}, \mathbf{X}_1^{st}, \dots, \mathbf{X}_\tau^{st}, \dots\}$ denotes spatiotemporal features with $\mathbf{X}_\tau^{st} \in \mathbb{R}^{N \times F^{st}}$. We select one spatiotemporal feature $\mathbf{Y} \in \mathbf{X}^{st}$ as the target variable and solve Eq. 1. That is, given graph G and S historical time-steps of all variables \mathcal{X} , we aim to learn a function \mathcal{F}_Θ that can forecast T future time-steps of \mathbf{Y} :

$$\arg \min_{\Theta} = \mathcal{L}(\mathbf{Y}_{(\tau+1):(\tau+T)}, \mathcal{F}_\Theta(\mathcal{X}_{\tau-S+1:\tau}; G)) \quad (1)$$

where Θ denotes learnable model parameters and \mathcal{L} denotes forecasting loss. This model is flexible enough to capture most spatiotemporal methods proposed in the literature. For example, if we consider one spatiotemporal feature and do not include spatial or temporal features (i.e., $F^{st} = 1, F^s = 0, F^t = 0$), Eq. 1 is equivalent to the multi-step traffic forecasting problem in AGCRN [2].

3.2 Model Design

In this section, we cover the design of MMR-GNN starting with (a) its high-level encoder-decoder architecture (b) its module for fusing explicit modalities (\mathbf{X}^s , \mathbf{X}^t , and \mathbf{X}^{st}) into rich embeddings (c) its module for partitioning node-space and learning implicit modalities and (d) its graph augmentation module for learning an optimal dependency structure G^* . Note that symbol C defines the mapping from nodes to implicit modalities and is discussed later in section *Implicit Multi-Modality Learning Module*.

$$\begin{aligned} stGRU(\mathcal{X}_{\tau-S+1:\tau}, G, C) &\rightarrow \mathcal{E}^{enc} \in \mathbb{R}^{N \times H} \\ stGRU(\mathcal{E}^{enc}, G, C) &\rightarrow \mathcal{E}^{dec} \in \mathbb{R}^{T \times N \times H} \\ mLinear(\mathcal{E}^{dec}, C) &\rightarrow \hat{\mathbf{Y}} \in \mathbb{R}^{T \times N} \end{aligned} \quad (2)$$

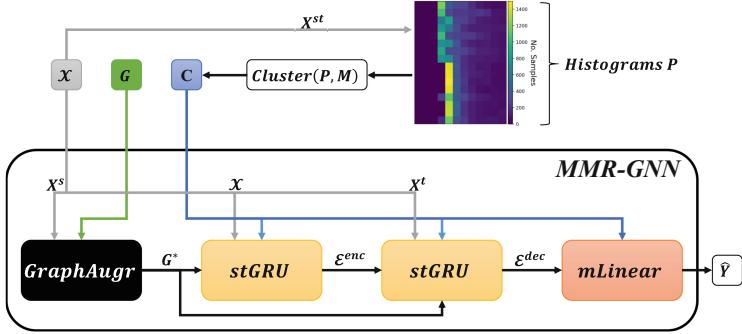


Fig. 2. Overview of MMR-GNN’s architecture. Prior to the forward-pass, nodes are clustered from histograms of the spatiotemporal features. The first step of forward-pass is a graph augmentation module called *GraphAugr* that constructs a graph from node embeddings, prunes it, and then joins it to an existing graph (if present). The augmented graph, all features (spatial, temporal, and spatiotemporal), and clusters are then fed into the multi-modal encoder cell called *stGRU*. Summary embeddings, future temporal features, augmented graph, and clusters are fed into the auto-regressive decoder, another *stGRU* cell. Finally, decoded embeddings and clusters are fed into the final projection layer called *mLinear* to produce forecastings.

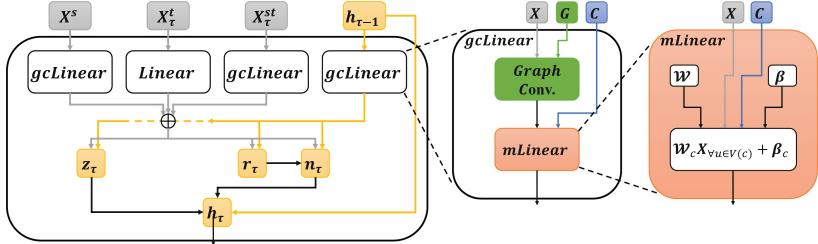


Fig. 3. (left) Overview of the stGRU cell. The forward pass projects and then fuses spatial, temporal, and spatiotemporal features. Fused features and previous hidden state are used to compute reset and update gates. Fused features, previous hidden state, and reset gate are used to compute the new gate. Finally, previous hidden state, update gate, and new gate are used to compute the new hidden state. (middle) Overview of the *gcLinear* layer. The forward-pass first performs graph convolution on features according to the given graph. Convolved features and clusters are then fed into the multi-modal linear projection layer *mLinear* to compute embeddings. (right) The *mLinear* layer projects nodes of each cluster using its assigned weights and biases.

Encoder-Decoder Framework. We utilize an encoder-decoder framework (Fig. 2) to encode S historical time-steps into summary embeddings and causally decode them into T future time-steps via auto-regression. Equation 2 details the forward pass which involves (a) encoding $\mathcal{X}_{\tau-S+1:\tau}$ into embedding \mathcal{E}^{enc} (b) auto-regressively decoding \mathcal{E}^{enc} into embedding \mathcal{E}^{dec} then (c) projecting \mathcal{E}^{dec} from embedding to output dimension \hat{Y} .

$$\begin{aligned} \sigma(gcL(\mathbf{X}^s, G, C) + L(\mathbf{X}_\tau^t) + gcL(\mathbf{X}_\tau^{st}, G, C) + gcL(\mathbf{h}_{\tau-1}, G, C)) &\rightarrow \mathbf{r}_\tau, \mathbf{z}_\tau \\ \tanh(gcL(\mathbf{X}^s, G, C) + L(\mathbf{X}_\tau^t) + gcL(\mathbf{X}_\tau^{st}, G, C) + \mathbf{r}_\tau \odot gcL(\mathbf{h}_{\tau-1}, G, C)) &\rightarrow \mathbf{n}_\tau \\ (1 - \mathbf{z}_\tau) \odot \mathbf{n}_t + \mathbf{z}_\tau \odot \mathbf{h}_{\tau-1} &\rightarrow \mathbf{h}_\tau \end{aligned} \quad (3)$$

Explicit Multi-modality Fusion Module. Spatiotemporal systems generate signals from multiple sources in \mathcal{X}_τ and possess complex dependency structures in G . In order to achieve complete embeddings and accurate forecasts, it is necessary that the model incorporate all components of the system. To this end, we propose *stGRU* which takes the form of a GRU cell (Eq. 3) but computes gates from the combination of all features (see Fig. 3). For spatial and spatiotemporal features (\mathbf{X}^s , \mathbf{X}^{st}), we use our custom graph convolution layer *gcLinear* which combines graph convolution and linear projection to embed dependencies. The standard linear projection layer is used for temporal features \mathbf{X}_τ^t since they do not exist at node-level. Equation 3 uses $gcL(\cdot)$ and $L(\cdot)$ to denote *gcLinear* and *Linear* layers. The forward pass of layers *gcLinear* and *Linear* are defined in Eq. 4 and 5 respectively where \mathbf{X}_τ^{s-st} indicates spatial or spatiotemporal features. The *gcLinear* layer applies Chebyshev graph convolution [6] and a new linear projection layer *mLinear* discussed in the next section.

$$mLinear(GCN(\mathbf{X}_\tau^{s-st}, G), C) \rightarrow \mathcal{E}_\tau^{s-st} \in \mathbb{R}^{N \times H} \quad (4)$$

$$W\mathbf{X}_\tau^t + b \rightarrow \mathcal{E}_\tau^t \in \mathbb{R}^H \quad (5)$$

Implicit Multi-modality Learning Module. We regularly observe multiple implicit modalities across spatiotemporal features \mathbf{X}^{st} representing different node dynamics (e.g., creeks vs. rivers, highway vs. suburban roads, etc.). Traditionally, a single parameter set $\theta = (W \in \mathbb{R}^{H \times F}, b \in \mathbb{R}^H)$ would be used to capture all node dynamics but this is an unnecessary constraint. Instead, we propose to separately learn the dynamics of M implicit modalities by partitioning the nodes into M clusters. We derive clustering C according to similarity of node feature distribution by applying agglomerative clustering on the histograms of \mathbf{X}^{st} . Multiple parameter sets $\Theta = (\mathcal{W} \in \mathbb{R}^{M \times H \times F}, \beta \in \mathbb{R}^{M \times H})$ are then used to learn the unique dynamics of M implicit modalities. We implement implicit modality learning as a new linear projection layer *mLinear* (Eq. 6) and use it for linear projection in *gcLinear* and final projection in the greater encoder-decoder.

$$\mathcal{W}_c \mathbf{X}_{\forall u \in V(c)} + \beta_c \rightarrow \mathcal{E}_c \in \mathbb{R}^{|V(c)| \times H} \quad (6)$$

Dependency Augmentation Module. An integral part of any spatiotemporal system is its dependency structure (e.g., streamflow network, road network, etc.) represented by graph G . However, this structure is often undefined and some methods [11] apply heuristics to infer G as a pre-processing step. Moreover, neither existing nor inferred graphs are guaranteed to be optimal in forecasting. To address these challenges, we propose *GraphAugr* which aims to learn an optimal dependency structure G^* .

Table 2. Details of datasets including sample count, graph size, and feature dimensions.

Dataset	Time-steps	Nodes	F^s	F^t	F^{st}	G	Resolution	Default prediction horizon
Little River	13,515	8	2	1	4	✓	1 day	1 time-step (1 day)
Wabash River	31,046	1,276	11	1	5	✓	1 day	1 time-step (1 day)
METR-LA	34,272	207	2	1	1	✓	5 min	12 time-steps (1 h)
E-METR-LA	34,272	207	12	1	5	✓	5 min	12 time-steps (1 h)
PEMS-BAY	52,116	325	2	1	1	✓	5 min	12 time-steps (1 h)
E-PEMS-BAY	52,116	325	12	1	5	✓	5 min	12 time-steps (1 h)

$$\begin{aligned} \text{Softmax}(\mathbf{E} \cdot \mathbf{E}^T) &\rightarrow \mathcal{S} \in \mathbb{R}^{N \times N} \\ \Phi(\mathbf{W} \odot \mathcal{S} + \mathbf{B}) &\rightarrow \hat{G} \quad G \cup \hat{G} \rightarrow G^* \end{aligned} \tag{7}$$

We define the forward-pass of *GraphAugr* in Eq. 7 where $\mathbf{E} \in \mathbb{R}^{N \times d}$ are learnable node embeddings, $\mathcal{S} \in \mathbb{R}^{N \times N}$ is the computed similarity matrix, matrices $\mathbf{W}, \mathbf{B} \in \mathbb{R}^{N \times N}$ are weights and biases on similarity, and $\Phi(\cdot)$ is a pruning function. The *GraphAugr* layer is generalized to cover a broad span of use-cases including (a) using an existing graph alone (b) augmenting an existing graph with learned edges or (c) constructing a graph entirely. Weights and biases \mathbf{W}, \mathbf{B} allow for fine control over edge construction. For example, \mathbf{W} can be used as a mask to restrict certain edges (e.g., intra-highway edges) and \mathbf{B} may be used to favor certain edges (e.g., existing edges). Finally, by feeding G^* to subsequent layers (*gcLinear* in encoder and decoder) we constrain graph construction to minimize forecasting loss, and hence, be optimal in forecasting.

4 Experiments

For evaluation, we consider six public datasets spanning hydrology, and traffic. We summarize these datasets in Table 2.

Hydrological Data. In hydrology, we consider the datasets Little River and Wabash River [13]. These datasets record multiple hydrological and meteorological features across unique regions of their watersheds. They also include multiple spatial features and pre-defined graphs in the form of streamflow networks. For these datasets, we forecast next-day streamflow across all sub-basins of the watershed given the previous 7 d of meteorological and hydrological records.

Traffic Data. In traffic, we consider METR-LA and PEMS-BAY [11], which are well-studied in the literature. A limitation of METR-LA and PEMS-BAY is that they contain only traffic speed and are restricted to uni-variate modeling. We extend these datasets by adding four spatiotemporal features (Samples, Percent Observed, Total Flow, and Average Occupancy) and ten spatial features acquired from [17]. We also create new pre-defined graphs (detailed in the appendix) using these new spatial features. These extended datasets are called E-METR-LA and E-PEMS-BAY. The task

Table 3. MAE, RMSE, and MAPE scores of MMR-GNN and other baseline models for two datasets. Emboldened numbers indicate the best performance while underlined numbers indicate second-best. An ‘N/A’ means the model was incompatible with a dataset or exceeded GPU runtime/memory.

	Wabash River			E-PEMS-BAY		
Model	MAE	RMSE	MAPE	MAE	RMSE	MAPE
GRU	4.01 ± 0.08	10.23 ± 0.03	27.60 ± 2.16	2.04 ± 0.07	4.03 ± 0.11	4.45 ± 0.15
TCN	4.04 ± 0.24	10.37 ± 0.04	27.21 ± 1.30	2.20 ± 0.02	4.52 ± 0.01	4.84 ± 0.03
FEDformer	4.26 ± 0.01	9.75 ± 0.01	36.15 ± 0.04	2.85 ± 0.03	4.96 ± 0.03	5.81 ± 0.05
LTSFDLinear	4.05 ± 0.00	10.77 ± 0.00	30.13 ± 0.00	2.37 ± 0.00	4.75 ± 0.00	5.08 ± 0.00
T-GCN	7.04 ± 0.13	14.268 ± 0.11	35.27 ± 0.70	2.67 ± 0.01	4.58 ± 0.015	5.67 ± 0.03
A3T-GCN	7.14 ± 0.12	14.43 ± 0.11	35.47 ± 1.32	N/A	N/A	N/A
ST-GCN	4.28 ± 0.15	10.24 ± 0.13	33.15 ± 2.48	N/A	N/A	N/A
AST-GCN	N/A	N/A	N/A	N/A	N/A	N/A
GMAN	N/A	N/A	N/A	N/A	N/A	N/A
StemGNN	9.60 ± 10.21	17.46 ± 12.55	38.30 ± 19.07	2.03 ± 0.06	4.03 ± 0.07	4.40 ± 0.14
MTGNN	N/A	N/A	N/A	2.48 ± 0.08	5.24 ± 0.19	5.73 ± 0.20
AGCRN	$\underline{2.98} \pm 0.01$	$\underline{9.03} \pm 0.10$	$\underline{17.59} \pm 0.68$	$\underline{2.05} \pm 0.04$	4.16 ± 0.09	4.38 ± 0.14
SCINet	6.31 ± 6.29	13.61 ± 8.92	32.96 ± 13.38	2.11 ± 0.02	3.96 ± 0.02	4.47 ± 0.04
MMR-GNN	2.96 ± 0.04	8.85 ± 0.10	17.14 ± 0.25	2.01 ± 0.04	3.85 ± 0.06	4.32 ± 0.09

for these datasets is to forecast the next hour (12 time-steps) of traffic speed given the previous hour.

Data Processing. We apply the same method of imputation and normalization to each dataset. In the event of missing values (found in LittleRiver, E-METR-LA, and E-PEMS-BAY), we impute with the periodic mean computed at each node from available samples. We standardize each node feature according to their mean and standard deviation computed from the training set. Each model is trained on standardized inputs/outputs with final forecasts computed from inverse standardization on model output.

Performance Metrics. To quantify forecast accuracy, we use Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE). We ran all experiments 10 times using 10 random initialization seeds and report their mean and standard deviation.

4.1 Model Baselines

For each dataset, we evaluate MMR-GNN against thirteen models found throughout the literature. The components of these baselines are listed in Table 1 separated according to pure temporal methods, GNN-based methods with pre-defined graphs, and GNN-based methods with learned graphs. Implementation of GRU was acquired from [13], while T-GCN, A3T-GCN, and GMAN were acquired from PyTorch Geometric Temporal [16].

Table 4. (a) RMSE scores of MMR-GNN and other baseline models for remaining datasets. Emboldened numbers indicate the best performance while underlined numbers indicate second-best. An ‘N/A’ means the model was incompatible with a dataset or exceeded GPU runtime/memory. (b) MAE scores for three models at multiple output horizons.

(a) RMSE scores for other datatsets			
Model	Little River	METR-LA	E-METR-LA
GRU	0.67 ± 0.01	<u>11.08 ± 0.081</u>	5.11 ± 0.02
TCN	0.66 ± 0.01	11.49 ± 0.05	5.74 ± 0.02
FEDformer	0.55 ± 0.00	12.53 ± 0.04	6.51 ± 0.05
LTSF-DLinear	0.64 ± 0.00	11.59 ± 0.00	5.73 ± 0.00
T-GCN	0.79 ± 0.02	13.6 ± 0.01	5.66 ± 0.01
A3T-GCN	0.87 ± 0.01	N/A	5.69 ± 0.01
ST-GCN	0.55 ± 0.01	N/A	N/A
AST-GCN	0.54 ± 0.02	11.60 ± 0.14	5.12 ± 0.05
GMAN	0.65 ± 0.03	12.91 ± 0.16	N/A
StemGNN	0.68 ± 0.03	11.31 ± 0.08	5.16 ± 0.07
MTGNN	0.5 ± 0.01	11.34 ± 0.05	<u>4.57 ± 0.03</u>
AGCRN	0.46 ± 0.01	11.78 ± 0.08	4.76 ± 0.02
SCINet	0.6 ± 0.01	12.345 ± 0.72	4.96 ± 0.03
MMR-GNN	0.46 ± 0.01	10.91 ± 0.11	4.56 ± 0.03

(b) Impact of output horizon		
	Wabash River	
Model	Horizon	MAE
FEDformer	1	4.26
LTSF-DLinear		<u>4.04</u>
MMR-GNN		2.96
FEDformer	14	16.90
LTSF-DLinear		<u>14.82</u>
MMR-GNN		13.05
FEDformer	28	20.19
LTSF-DLinear		<u>18.84</u>
MMR-GNN		16.53

All other models were acquired from their published GitHub repositories. We train and evaluate all models on an Nvidia A100 GPU with 40 GB memory.

4.2 Primary Results

We break down the primary results into two tables. Table 3 shows MAE, RMSE, and MAPE of all models on our two largest multi-modal datasets. Due to space limitations, Table 4a shows only RMSE for three other datasets. In all tables, we group models by (a) pure temporal and transformers, (b) GNN-based models with existing graphs, and (c) GNN-based models with learned graphs. A3T-GCN, GMAN, and MTGNN often show prohibitive scaling with either excessive runtime and/or memory requirements.

We observe that GNN models using pre-defined graphs (middle group of Table 3) struggle to out-perform their pure temporal counter-parts. In comparison, most GNN models using learned graphs perform on-par or better than pure temporal models. This suggests pre-defined graphs are not necessarily optimal in forecasting and can even degrade performance. Looking at MMR-GNN in Tables 3 and 4a, we see consistent improvement with lower errors than the best baseline model across all datasets.

Effect of Forecast Horizon. Thus far, we mostly considered short-term predictions as mentioned in Table 2. It is well documented that transformers [25] and some decou-

Table 5. Alternative RNN cells

Cell	MAE	RMSE	MAPE
gcRNN	3.019 ± 0.056	9.106 ± 0.218	18.901 ± 0.471
gcGRU	2.962 ± 0.044	8.812 ± 0.184	17.231 ± 0.287
gcLSTM	<u>2.977 ± 0.060</u>	<u>8.988 ± 0.245</u>	<u>17.269 ± 0.524</u>

Table 6. Alternative fusion method

Fusion	MAE	RMSE	MAPE
Addition	3.00 ± 0.066	9.091 ± 0.268	17.127 ± 0.441
Attention	2.979 ± 0.0545	8.003 ± 0.174	16.977 ± 0.309

Table 7. Alternative clustering algorithms

Method	MAE	RMSE	MAPE
Random	2.984 ± 0.051	8.901 ± 0.189	17.648 ± 0.417
KMeans	<u>2.977 ± 0.058</u>	<u>8.866 ± 0.154</u>	<u>17.608 ± 0.295</u>
Agglomerative	2.911 ± 0.022	8.601 ± 0.046	17.269 ± 0.007

pled linear models such as LTSF-DLinear [21] perform better in long-term predictions. Table 4b shows that MMR-GNN outperforms FEDformer and LTSF-DLinear in mid to long-term forecasting of streamflow.

4.3 Ablation Study

We conduct a variety of experiments to test different components of MMR-GNN. These tests include: (1) substitution of our proposed *stGRU* cell with vanilla GRU, (2) sampling the number of implicit modalities, and (3) sampling the space of graph augmentation. We perform all tests on the Wabash River dataset since it includes a pre-defined graph and has the greatest complexity in terms of available features and scale.

Explicit Multi-modality Fusion. We proposed the *stGRU* cell to learn from the multiple explicit modalities present in spatiotemporal systems and test its contribution here. Table 8a shows performance when using the vanilla GRU cell or our *stGRU* cell. We see improvement across all metrics demonstrating that features from additional modalities are informative to forecasting.

Implicit Modality Learning. We test the effect of increasing the number of implicit modalities learned by our multi-modal projection layer *mLinear*. Table 8b shows MMR-GNN’s performance when increasing implicit modality count M starting from the traditional single modality (single parameter set) case. By increasing M , we partition the system into smaller problems, and thus, expect performance to improve to a point of saturation where over-fitting occurs. This effect is seen in Table 8b where the optimal modality is found at $M = 8$ according to two metrics.

Table 8. Ablation studies

(I) Graph augmentation rate				
G	Density	MAE	RMSE	MAPE
\times	0%	3.392	10.538	2.184
\checkmark	25%	3.111	9.315	2.177
\checkmark	50%	2.953	8.872	2.218
\checkmark	75%	2.969	8.862	2.186
\checkmark	99%	2.897	8.614	2.189

(b) Implicit modality count			
M	MAE	RMSE	MAPE
1	3.066	9.076	2.179
4	3.103	9.336	2.192
8	2.873	8.647	2.217
12	3.038	9.118	2.179
16	3.038	9.123	2.212

(a) stGRU vs. vanilla GRU			
stGRU	MAE	RMSE	MAPE
\times	3.035	9.038	2.247
\checkmark	2.937	8.763	2.180

Graph Augmentation. Here we aim to understand the contribution of learned graph augmentation by testing various scenarios including (a) no pre-defined graph or augmentation (b) pre-defined graph without augmentation and (c) increasing rates of augmentation on the pre-defined graph. Table 8c shows monotonic improvement for RMSE with increased graph augmentation.

Other Ablation Studies. We also test the impact of other RNN cells (Table 5), attention for fusion of explicit modalities (Table 6), and various clustering algorithms when finding implicit modality (Table 7).

5 Conclusions

This paper presents MMR-GNN, a network capable of learning from the multiple explicit and implicit modalities present in many spatiotemporal systems. For multi-modal data, MMR-GNN shows consistent improvement over SOTA methods but we see one limitation that, if addressed, could further improve accuracy. The projection layer mLinear depends on the quality of computed clusters which may be erroneous and inhibit implicit modality learning. A possible solution is to cluster on the learned node embeddings (from graph augmentation) allowing implicit modality assignment to be learned indirectly. While this does not remove potentially erroneous clustering algorithms, it allows the model to adapt towards an optimal clustering. Our future work will aim to address this limitation and expand MMR-GNN into more application domains.

Acknowledgements. This research is supported by the Applied Mathematics Program of the DOE Office of Advanced Scientific Computing Research under contracts numbered DE-SC0022098 and DE-SC0023349 and by the NSF OAC-2339607 grant.

7 Appendix

Road Network Inference. E-METR-LA and E-PEMS-BAY are extensions of the quintessential METR-LA and PEMS-BAY datasets adding new spatial and spatiotem-

poral features. Following [11], we wanted to offer pre-defined graphs for each dataset and chose to infer them using new spatial features. We apply *GraphAugr* to construct a graph heuristically using exponentiated Minkowski ($p = 2$) distance to define node similarity and \mathbf{W}, \mathbf{B} to define rules for edge construction. Each traffic sensor of E-METR-LA and E-PEMS-BAY includes latitude, longitude, freeway name, and freeway bearing. We use latitude and longitude to define node similarity S and freeway/bearing to define a mask \mathbf{W} that restricts to intra-highway edges of the same bearing.

Using OpenStreetMap [15], we found many sensor pairs placed closely together but recording traffic in opposite bearings (e.g., U.S. Route 101 in CA with North/South bearing). Using distance-based similarity alone leads to connections between these and many other unrelated sensors. With the previous similarity and restrictions, the modified similarity becomes $S^* = \mathbf{W} \odot S$ which we prune using K -NN with $k = 2$. The original graph of METR-LA and our new graph for E-METR-LA are shown in Fig. 4.

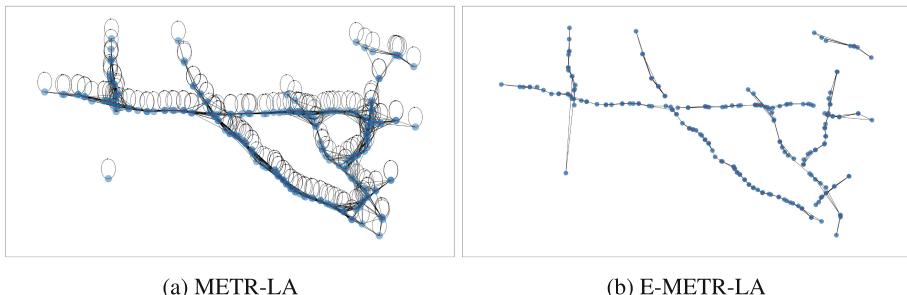


Fig. 4. The pre-defined road network from METR-LA (a) and our new road network for E-METR-LA (b)

References

1. Bai, J., et al.: A3t-GCN: attention temporal graph convolutional network for traffic forecasting. *ISPRS Int. J. Geo Inf.* **10**(7), 485 (2021)
2. Bai, L., Yao, L., Li, C., Wang, X., Wang, C.: Adaptive graph convolutional recurrent network for traffic forecasting. In: *Advances in Neural Information Processing Systems*, vol. 33, pp. 17804–17815 (2020)
3. Bai, S., Kolter, J.Z., Koltun, V.: An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271* (2018)
4. Cao, D., et al.: Spectral temporal graph neural network for multivariate time-series forecasting. In: *Advances in Neural Information Processing Systems*, vol. 33, pp. 17766–17778 (2020)
5. Contreras, J., Espinola, R., Nogales, F.J., Conejo, A.J.: ARIMA models to predict next-day electricity prices. *IEEE Trans. Power Syst.* **18**(3), 1014–1020 (2003)
6. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: *Advances in Neural Information Processing Systems*, vol. 29 (2016)

7. Guo, S., Lin, Y., Feng, N., Song, C., Wan, H.: Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 922–929 (2019)
8. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
9. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint [arXiv:1609.02907](https://arxiv.org/abs/1609.02907) (2016)
10. Kratzert, F., Klotz, D., Shalev, G., Klambauer, G., Hochreiter, S., Nearing, G.: Towards learning universal, regional, and local hydrological behaviors via machine learning applied to large-sample datasets. *Hydrolog. Earth Syst. Sci.* **23**(12), 5089–5110 (2019)
11. Li, Y., Yu, R., Shahabi, C., Liu, Y.: Diffusion convolutional recurrent neural network: data-driven traffic forecasting. arXiv preprint [arXiv:1707.01926](https://arxiv.org/abs/1707.01926) (2017)
12. Liu, M., et al.: Scinet: time series modeling and forecasting with sample convolution and interaction. In: Thirty-Sixth Conference on Neural Information Processing Systems (NeurIPS) (2022)
13. Majeske, N., Zhang, X., Sabaj, M., Gong, L., Zhu, C., Azad, A.: Inductive predictions of hydrologic events using a long short-term memory network and the soil and water assessment tool. *Environ. Model. Softw.* **152**, 105400 (2022)
14. NREL: Solar power data for integration studies (2006). <https://www.nrel.gov/grid/solar-power-data.html>
15. OpenStreetMap contributors: Planet dump retrieved from <https://planet.osm.org> (2017). <https://www.openstreetmap.org>
16. Rozemberczki, B., et al.: PyTorch geometric temporal: spatiotemporal signal processing with neural machine learning models. In: Proceedings of the 30th ACM International Conference on Information and Knowledge Management, pp. 4564–4573 (2021)
17. Varaiya, P.P.: Freeway performance measurement system (pems), pems 7.0. Technical report (2007)
18. Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X., Zhang, C.: Connecting the dots: multivariate time series forecasting with graph neural networks. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 753–763 (2020)
19. Wu, Z., Pan, S., Long, G., Jiang, J., Zhang, C.: Graph wavenet for deep spatial-temporal graph modeling. arXiv preprint [arXiv:1906.00121](https://arxiv.org/abs/1906.00121) (2019)
20. Yu, B., Yin, H., Zhu, Z.: Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. arXiv preprint [arXiv:1709.04875](https://arxiv.org/abs/1709.04875) (2017)
21. Zeng, A., Chen, M., Zhang, L., Xu, Q.: Are transformers effective for time series forecasting? (2023)
22. Zhao, L., et al.: t-GCN: a temporal graph convolutional network for traffic prediction. *IEEE Trans. Intell. Transp. Syst.* **21**(9), 3848–3858 (2019)
23. Zheng, C., Fan, X., Wang, C., Qi, J.: Gman: a graph multi-attention network for traffic prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 1234–1241 (2020)
24. Zhou, H., et al.: Informer: Beyond efficient transformer for long sequence time-series forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 11106–11115 (2021)
25. Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., Jin, R.: FEDformer: frequency enhanced decomposed transformer for long-term series forecasting. In: Proceedings of 39th International Conference on Machine Learning (ICML 2022) (2022)



Layer-Wise Sparse Training of Transformer via Convolutional Flood Filling

Bokyeong Yoon[✉], Yoonsang Han[✉], and Gordon Euhyun Moon^(✉)

Sogang University, Seoul, Republic of Korea
`{bkyoon, han14931, ehmoon}@sogang.ac.kr`

Abstract. Sparsifying the Transformer has garnered considerable interest, as training the Transformer is very computationally demanding. Prior efforts to sparsify the Transformer have either used a fixed pattern or data-driven approach to reduce the number of operations involving the computation of multi-head attention, which is the main bottleneck of the Transformer. However, existing methods suffer from inevitable problems, including potential loss of essential sequence features and an increase in the model size. In this paper, we propose a novel sparsification scheme for the Transformer that integrates convolution filters and the flood filling method to efficiently capture the layer-wise sparse pattern in attention operations. Our sparsification approach significantly reduces the computational complexity and memory footprint of the Transformer during training. Efficient implementations of the layer-wise sparsified attention algorithm on GPUs are developed, demonstrating our SPION that achieves up to $2.78\times$ speedup over existing state-of-the-art sparse Transformer models and maintain high evaluation quality.

Keywords: Deep Learning · Sparse Transformer · Convolutional Flood Filling

1 Introduction

The Transformer is a state-of-the-art deep neural network developed for addressing sequence tasks, originally proposed by Vaswani et al. [18]. One of the main advantages of the Transformer is that, given a sequence of input data points (e.g., a sentence of word tokens), it is able to compute the multi-head attention (MHA) operation in parallel, thereby quickly and accurately capturing long-term dependencies of data points. However, as the sequence length increases, the overall computational cost and memory space required for training the Transformer also increase quadratically [1]. Especially, a MHA sub-layer in the Transformer occupies a substantial portion of the total execution time and becomes the main bottleneck as the sequence length increases. The MHA operation requires a large number of dot-product operations to compute the similarity between all data points in the sequence. However, the dot-product operation inherently has limitations in memory bandwidth since it performs only two floating-point operations

for each pair of data elements read from memory. Hence, in order to mitigate computational complexity and improve data locality of the Transformer, several approaches have addressed the sparsification of computations associated with the MHA operation [2, 16, 19–21]. However, previous approaches suffer from two primary limitations. First, when the Transformer adopts identical fixed patterns of non-zero entries in the attention matrices across all layers during training, it becomes difficult to effectively capture key features within the sequence. Second, when the Transformer employs additional parameters to learn the sparsity pattern in the attention matrices, both the model size and the computational overhead increase. To address the limitations of previous approaches, we focus on developing a specialized sparse attention scheme that significantly reduces memory consumption and efficiently handles variations of sparse patterns across different types of Transformers and datasets.

In this paper, we present a new sparsity-aware layer-wise Transformer (called **SPION**) that dynamically captures the variations in sparse pattern within the MHA operation for each layer. SPION judiciously explores the sparsity pattern in the MHA operation based on a novel convolutional flood filling method. To precisely detect the characteristics of the sparse pattern in the attention matrices, SPION identifies the shape of sparse pattern by utilizing a convolution filter and the degree of sparsity through a flood filling-based scheme. During the generation of the sparse pattern for each layer, we construct a sparsity pattern matrix with a blocked structure to enhance data locality for the MHA operation that involves sparse matrix multiplication. Furthermore, by capturing layer-specific sparse pattern for each layer, SPION performs layer-wise MHA computations iteratively until convergence is achieved. As the sparse MHA operations contribute significantly to the overall training workload, we develop an efficient GPU implementation for sparse MHA to achieve high performance with quality of results.

We conduct an extensive comparative evaluation of SPION across various classification tasks, including image and text classification, as well as document retrieval involving lengthy sequences. Experimental results demonstrate that our SPION model achieves up to a $5.91\times$ reduction in operations for MHA computation and up to a $2.78\times$ training speedup compared to existing state-of-the-art sparse Transformers while maintaining a better quality of solution.

2 Background and Related Work

2.1 Transformer

The encoder-only Transformer, which is one of the variants of the Transformer model, is widely used for various classification tasks using text and image datasets [5, 6, 17]. In the encoder-only Transformer, each encoder layer consists of a MHA sub-layer and a feed-forward sub-layer. For each encoder layer, the query ($Q \in \mathbb{R}^{L \times D}$), key ($K \in \mathbb{R}^{L \times D}$), and value ($V \in \mathbb{R}^{L \times D}$) are obtained by performing linear transformations on the input embedding. Hereafter, we denote L , D and H as the length of input sequence, the size of the embedding

for each data point in the sequence, and the number of heads, respectively. To efficiently perform MHA computation, each Q , K , and V matrix is divided into H sub-matrices (multi-heads) along the D dimension.

$$A = \text{softmax} \left(\frac{Q \times K^T}{\sqrt{(D/H)}} \right) \times V \quad (1)$$

The MHA computation is defined by Eq. 1. After computing the attention for each head, all matrices A_0 through A_{H-1} are concatenated to form the final attention matrix A . Then, the attention matrix A is passed through the feed-forward sub-layer to produce new embedding vectors, which is then fed into the next encoder layer. Therefore, in terms of computational complexity, the main bottleneck of the encoder layer is associated with processing the MHA sub-layer, which involves a large number of matrix-matrix multiplications across multiple heads. More specifically, the number of operations required for computing the attention matrix A is $2L^2(2D + 1) - L(D + 1)$, indicating a quadratic increase in operations as the input sequence length (L) increases.

Sparse MHA. Given the long input sequences, several sparse attention techniques have been proposed to reduce the computational complexity involved in multiplying Q and K^T in the MHA operation. The basic intuition behind sparse attention techniques is that a subset of data points in the long sequence can effectively represent the entire input sequence. In other words, only the highly correlated necessary elements (i.e., data points) in Q and K can be utilized to reduce the computational workload. Therefore, when employing sparse attention, the number of operations required to compute $Q \times K^T$ is $C(2D - 1)$, where $C \ll L^2$ represents the number of critical elements in the resulting matrix. In contrast, without sparsification, the original computation of $Q \times K^T$ requires $L^2(2D - 1)$ operations.

2.2 Related Work on Sparse Attention

Many previous efforts to achieve efficient sparse Transformers have sought to sparsify the MHA operation both before and during model training/fine-tuning.

Fixed Sparse Pattern. One of the strategies for performing sparse MHA is to use a predetermined sparsity patterns, where only specific data points in the input sequence are selected to perform the MHA operation before training the model. Several variants of the Transformer model adapt the sliding windows approach in which the attention operations are performed using only the neighboring data points in the matrices Q and K . The Sparse Transformer [3] originally employs the sliding windows attention to sparsify the MHA operation. The Longformer [2] is an extension to the Sparse Transformer and introduces dilated sliding windows, which extend the receptive field for computing similarity by skipping one data point at a time while performing the sliding windows

attention. Furthermore, ETC [1] and BigBird [20] incorporate global attention that performs similarity calculations between a given data point and all other data points in the input sequence.

The main advantage of utilizing the fixed sparsity pattern is the reduction in computational overhead and memory footprint. However, the primary problem with a fixed pattern is that it may lose the fine-grained important features and dependencies in the input sequence. For example, the attention mechanisms utilized in Longformer and BigBird have limitations when applied to image classification tasks. Since image data is typically shaped in two dimensions (except for RGB channels), neighboring image pixels (or patches) arranged vertically do not appear side by side in sequential data. Consequently, the sliding window attention-based Longformer is unable to recognize connections between vertically adjacent data points, as it attends only to neighbors within the sequence. Moreover, the global attention used in BigBird only focuses on specific tokens. However, it is possible that there can be other data points besides these specific tokens that are important to all the other data points in a sequence. In such cases, models based on fixed sparse patterns may lose critical information from the data points in the sequence.

Data-Driven Sparse Pattern. Sparse patterns in the MHA operation can also be generated by leveraging a data-driven approach, which clusters and sorts the data points of the input sequence during training. Most recently, the LSG Attention [4] extends pretrained RoBERTa [11] for fine-tuning various language related tasks and outperforms Longformer and BigBird models. LSG Attention dynamically captures data patterns by generating sparse masks by utilizing local, sparse, and global attention mechanisms. However, since LSG Attention generates sparse masks at every training step, it results in additional computational overhead during training. Reformer [9] utilizes locality-sensitive hashing to calculate a hash-based similarity and cluster multiple data points into chunks. Similarly, Routing Transformer [16] performs k-means clustering given data points. In the process of training the model, these clustering-based attention approaches also require learning additional functions to identify the relevant dependencies of data points in the input sequence. However, even though utilizing data-driven sparsification techniques during training produces a high quality model, this approach requires additional parameters and operations to learn the sparse patterns, resulting in larger memory space and higher computational cost.

3 Motivation: Analysis of Sparse Patterns in MHA

In order to identify common sparsity patterns in the MHA operation, we conducted experiments on the encoder-only Transformer [6] pretrained with the ImageNet-21k and ImageNet-1k datasets. Hereafter, we denote A^s as the attention score matrix obtained after computing $\text{softmax}(Q \times K^T / \sqrt{D/H})$ in MHA operation. Figure 1 shows A^s from different encoder layers during the inference. Since the sparsity patterns of multiple A^s within the same encoder layer typically

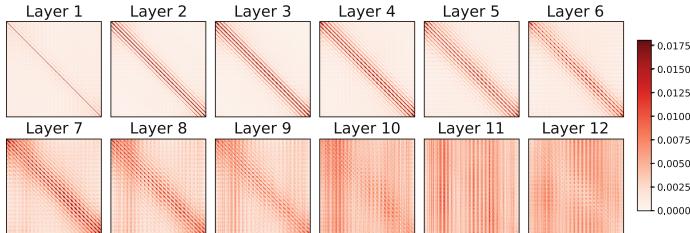


Fig. 1. Sparsity patterns in the attention score matrices across different encoder layers during inference of the encoder-only Transformer for image classification.

show similar patterns, we averaged the A^s across multiple heads in each encoder layer. The results clearly show that most of the elements in A^s are close to zero, indicating that only a few data points in the input sequence are correlated to each other. In practice, a number of studies have shown that considering only the critical data points does not adversely affect the convergence of the model [2, 14, 20]. The characteristics of attention score matrices A^s described below motivate us to develop a new layer-wise sparse attention algorithm.

Shape of Sparse Pattern. As shown in Fig. 1, the attention score matrices produced by the different encoder layers exhibit distinct sparsity patterns. For example, in the first to ninth encoder layers, the diagonal elements have relatively large values, similar to a band matrix that stores nonzeros within the diagonal band of the matrix. It is obvious that the MHA operation relies on the self-attention scheme and therefore, the resulting values of the dot-product between linearly transformed vectors for the same data points tend to be larger compared to the resulting outputs produced with different data points. In addition to the diagonal sparsity pattern, encoder layers 10, 11 and 12 show a vertical sparsity pattern, with nonzeros mostly stored in specific columns. This vertical sparsity pattern emerges when the attention operation focuses on the similarity between all data points in Q and particular data points in K . In light of these observations, applying the same fixed sparse pattern to all layers may lead to the exclusion of unique essential features that need to be captured individually at different layers. Hence, it is crucial to consider layer-wise sparsification of the MHA based on the sparse pattern observed across different layers. Furthermore, it is necessary to generate domain-specific flexible sparse patterns for various tasks and datasets.

Degree of Sparsity. Across different encoder layers, there exists variation not only in the shape of the sparse pattern but also in the number of nonzero elements in attention score matrices. For example, layer 12 has a higher number of nonzero elements compared to layer 1, indicating that layer 12 extensively computes the MHA operation using a larger number of data points in the sequence. Hence, it is crucial to consider varying degrees of sparsity for every encoder layer.

Considering the irregular distribution of non-zero entries in different attention matrices is essential for effectively reducing computational operations while preserving key features across distinct encoder layers.

4 SPION: Layer-Wise Sparse Attention in Transformer

In this section, we provide a high-level overview and details of our new SPION that dynamically sparsifies the MHA operation, incorporating the major considerations described in Sect. 3.

4.1 Overview of SPION

In SPION, the overall training process is decoupled into three phases: dense-attention training, sparsity pattern generation, and sparse-attention training. Our SPION is capable of sparsifying the MHA operation after training the model for a few steps with dense-attention training. The dense-attention training follows the same training procedure as the original Transformer, without sparsifying the MHA operation. The original dense MHA continues until the attention score matrix A^s of each encoder layer exhibits a specific sparsity pattern. In order to determine the end of the dense-attention training or the start of the sparse-attention training, for each step, we first measure the Frobenius distance between the A^s produced in the previous step $i - 1$ and the current step i as defined in Eq. 2.

$$\text{distance}_i = \left| \sqrt{\sum(A_{i-1}^s)^2} - \sqrt{\sum(A_i^s)^2} \right| \quad (2)$$

Then, we compare the previous distance_{i-1} with the current distance_i to ensure whether a common pattern of nonzeros has emerged. Intuitively, when the difference between distance_{i-1} and distance_i is very small, it is possible to assume that A^s ends up with a specialized sparsity pattern. Hence, if the difference in Frobenius distance between the previous step and the current step is less than a threshold value, we cease the dense-attention training phase. Thereafter, we dynamically generates the sparsity pattern matrix P for each encoder layer based on our novel convolutional flood-filling scheme, as described in Sect. 4.2.

After identifying the sparsity pattern, SPION proceeds with the sparse-attention training phase until convergence by adapting the sparsity pattern. Given the matrices Q and K , along with the sparsity pattern matrix P , the SDDMM (Sampled Dense-Dense Matrix Multiplication) operation is utilized to accelerate producing the sparsified attention score matrix. Next, the sparse attention score matrix is used to apply the sparse softmax function. After computing the sparse softmax operation, since the sparsified attention score matrix S^s remains sparse, we utilize SpMM (Sparse-Dense Matrix Multiplication) operation to obtain final attention matrix S by multiplying the sparse matrix S^s with the dense matrix V . To accelerate both SDDMM and SpMM operations

on GPUs, we utilize the `cusparseSDDMM()` and `cusparseSpMM()` functions provided by the NVIDIA cuSPARSE library [13]. Moreover, we implement a custom CUDA kernel for the sparse softmax function by leveraging warp-level reduction to accelerate the sparse-attention training phase.

4.2 Sparsity Pattern Generation with Convolutional Flood Fill Algorithm

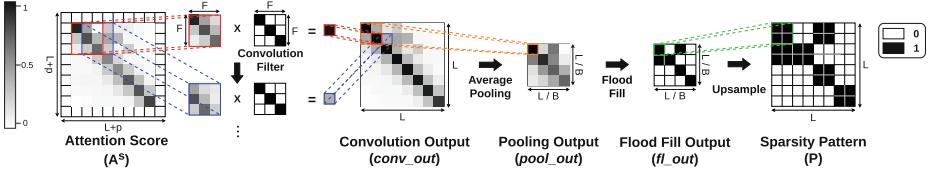


Fig. 2. Overview of our convolutional flood filling method for generating the sparsity pattern in the attention score matrix.

To precisely identify the sparsity patterns in the attention score matrix A^s , we develop a new convolutional flood fill algorithm that extensively explores the shape, degree and locality of sparsity patterns in A^s for each encoder layer. Figure 2 shows the overview for generating the sparsity pattern in A^s . An initial step is to apply a diagonal convolution filter to A^s in order to identify the shape of sparsity pattern. If the diagonal elements in A^s have larger values compared to the others, applying a diagonal convolution filter increases the values of the diagonal elements in the convolution output ($conv_out$). This leads to the emergence of a diagonal sparsity pattern. Otherwise, if the off-diagonal elements, especially the vertical ones, in A^s have larger values compared to the others, applying a diagonal convolution filter results in a vertical sparsity pattern in $conv_out$ matrix. In order to ensure that A^s and $conv_out$ have the same size ($L \times L$), we adopt zero-padding to A^s during computing the convolution operation defined in Eq. 3.

$$conv_out(i, j) = \sum_{f=1}^F A^s(i + f, j + f) \times filter(f, f) \quad (3)$$

After generating the $conv_out$ through a diagonal convolution operation, our algorithm performs average pooling on the $conv_out$ using a kernel/block of size $(B \times B)$ as defined in Eq. 4.

$$pool_out\left(\frac{i}{B}, \frac{j}{B}\right) = \frac{1}{B^2} \sum_{p=1}^B \sum_{q=1}^B conv_out(i + p, j + q) \quad (4)$$

Instead of analyzing the sparsity pattern of A^s element by element, applying average pooling enables capturing block sparsity pattern, which considers both

the critical data points and their surrounding data points. Hence, since the output of average pooling ($pool_out$) has a smaller size ($L/B \times L/B$) compared to the attention score matrix A^s , $pool_out$ can be considered as a block-wise abstract sparsity representation of A^s .

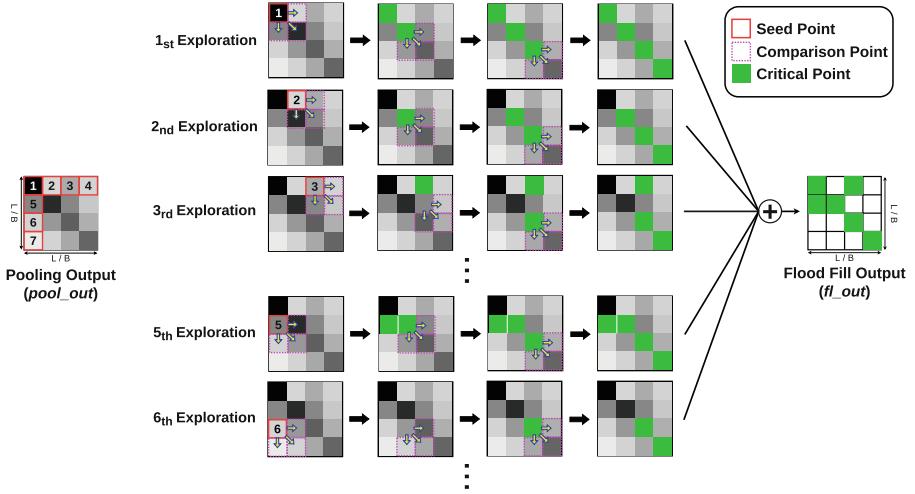


Fig. 3. Walk-through example of the identification of critical elements using the flood fill algorithm. (Colour figure online)

In order to dynamically explore the critical elements in the $pool_out$, we develop a novel method inspired by the flood fill algorithm. The flood fill algorithm is originally developed to determine the area connected to a given cell/pixel in a multi-dimensional array and a bitmap image [7]. Unlike the traditional flood fill algorithm, which compares all neighbors of a current element to find the element with the largest value, our scheme only compares to the neighboring elements on the right, below, and diagonally below, as shown in Fig. 3. Essentially, this is because the $pool_out$ matrix follows the sequential order from left to right and top to bottom. In the process of capturing the pattern, it is necessary to sequentially follow the important features starting from the top-left corner to the bottom of the matrix. All the elements in the first row and all the elements in the first column of the $pool_out$ are used as the seed starting points. From the particular seed point, our algorithm compares the values of elements to its right, below, and diagonally below to extract the element with the largest value in order to check whether the neighbors of the current element are relevant. If the neighboring elements are not relevant to the current element, our algorithm moves to the element diagonally below to continue comparing the neighbors. When the largest value is greater than a predefined threshold, we determine the corresponding element as the critical element (green colored elements in Fig. 3). Here, the threshold is determined by calculating the $\alpha\%$ quantile of $pool_out$. By

utilizing the threshold, our flood fill algorithm ensures that the values of selected critical elements are sufficiently large. After determining the critical elements, our algorithm recursively compares the values of elements to the right, below, and diagonally below the critical element while avoiding duplicate comparisons with elements that have already been selected as critical points. This process is repeated until the selected critical element reaches the end of a row or column in the matrix $pool_out$. After conducting explorations from all seed points, the resulting critical points from each exploration are combined into a single matrix called the flood fill output (fl_out). Therefore, the fl_out can be considered as an explicit sparsity pattern captured from $pool_out$. fl_out also represents the compressed block-level sparsity pattern of A^s since the average pooling operation is applied before processing the flood fill algorithm.

Next, to utilize newly generated sparse pattern of fl_out in the sparse MHA operation, the size of the fl_out needs to be the same as the size of the attention score matrix A^s . Therefore, we upsample the fl_out using nearest-neighbor interpolation, resulting in each nonzero element in fl_out forming a block of nonzero elements in the final sparsity pattern matrix (P), as shown on the right of Fig. 2. Utilizing a block sparse matrix P improves the quality of model since it incorporates not only the critical elements but also their surrounding elements for MHA operation. Moreover, an optimized blocked matrix multiplication further enhances performance through improved data locality. In the blocked sparsity pattern matrix P , critical elements involving MHA calculation are set to 1, and the rest of the non-critical elements are set to 0. Finally, during the sparse MHA, only the elements of A^s that have the same indices with a value of 1 in P will be computed.

5 Experimental Evaluation

This section provides both performance and quality assessments of our SPION implementation. Our SPION is compared with various state-of-the-art sparse Transformer models. All the experiments were run on four NVIDIA RTX A5000 GPUs. Our GPU implementation integrates optimized CUDA kernels and CUDA libraries for sparse-attention training with PyTorch, utilizing the ctypes library (CDLL) provided in Python.

Datasets and Tasks. We evaluated our SPION on four tasks: image classification, text classification, ListOps, and document retrieval. For image classification, we utilized the CIFAR-10 [10] and iNaturalist 2018 [8] datasets, with the former having images of 32×32 pixels and the latter resized to this resolution, resulting in a sequence length of 1,024. In text classification, we used the AG News [22] and Yelp Reviews [22] datasets, both having a maximum sequence length of 1,024. The ListOps task, based on the dataset from Nangia et al. [12], involves classifying answers from 0 to 9 from sequences of numbers and operators, with a sequence length of 2,048. For the document retrieval task, we used the AAN dataset [15], classifying whether two given documents are related or not,

with a sequence length of 4,096. All the text data is evaluated at the character level.

Models Compared. We compared our SPION with four state-of-the-art sparse Transformers: LSG Attention [4], Longformer [2], BigBird [20] and Reformer [9]. In addition, we evaluated variations of our SPION model: SPION-C, SPION-F and SPION-CF. Specifically, the SPION-C model omits the flood filling scheme and selects the top few percent of block elements after convolution filter and average pooling (*pool_out*), facilitating an adjustable sparsity ratio during generating the P matrix. In contrast, the SPION-F model bypasses the convolution filter, applying the flood fill-based algorithm immediately after the average pooling. The SPION-CF integrates both the convolution filter and the flood fill-based approach while generating a sparsity pattern. For all models, the sizes of configuration (window, bucket, block) were determined by the maximum sequence length of dataset. For example, Longformer used 64 or 128, while all other compared models, including our SPION variants, used 32 or 64.

We set the embedding dimension (D) to a size of 64. For image classification and document retrieval tasks, we used two layers, whereas for text classification and ListOps tasks, we employed four layers. The batch size was determined by 512 for image classification and text classification, 256 for ListOps, and 32 for document retrieval. Across all experiments, the size of the convolution filter in SPION was fixed at (31×31) . As for the threshold in the flood fill algorithm, we configured α to 96 for image classification and text classification tasks, 98 for ListOps, and 99.5 for the document retrieval task. Note that all experimental results presented in this section are averaged over three distinct runs.

5.1 Performance Evaluation

Convergence. Table 1 shows the accuracy of six models on four different tasks. Our SPION-CF consistently achieved the highest accuracy in all tasks, surpassing the highest accuracy obtained from other compared models by +0.812%, +0.115%, +0.697%, +2.735%, +0.707%, and +0.205% for the evaluation tasks. It is interesting to see that among the SPION variants, incorporating both the convolution filter and flood-filling scheme led to higher accuracy for all tasks. This indicates that the convolution filter and flood-filling method synergize with each other. Additionally, we observed that the flood filling method shows more significant effect on accuracy compared to the convolution filter. This result demonstrates that considering the connectivity between elements is an important factor when capturing sparse patterns.

Speedup and Memory Reduction. Table 2 shows the time and memory required to train each model across four tasks, using the CIFAR-10 dataset for image classification and Yelp-review dataset for text classification. Our SPION-CF, compared to BigBird, achieved speedups of up to $2.78 \times$ per training step. Particularly in longer sequence tasks like ListOps and document retrieval,

Table 1. Classification accuracy (%) of various tasks

Model	Image Classification		Text Classification		ListOps	Document Retrieval
	CIFAR-10	iNaturalist	Yelp-Review	AG-News		
LSG Attention	43.435	53.758	80.449	79.371	39.137	80.538
Longformer	42.845	54.269	80.858	75.465	39.620	80.595
BigBird	41.978	52.964	76.813	74.177	39.658	80.396
Reformer	40.824	51.298	74.625	67.164	38.058	78.891
SPION-C	41.355	53.759	76.971	72.796	40.290	80.266
SPION-F	43.846	53.553	81.339	81.645	40.160	80.423
SPION-CF	44.247	54.384	81.555	82.106	40.365	80.800

SPION showed greater efficiency. Against LSG-Attention, SPION excelled in tasks with shorter sequences, and also required less training time for longer sequences. Table 2 further shows SPION-CF reducing memory footprint by over $7.24\times$ in comparison to BigBird on ListOps task. Our SPION variants consistently had the smallest memory footprints. Remarkably, despite memory savings, SPION-CF maintained the highest accuracy across all tasks.

Table 2. Comparison of elapsed time (ms) and memory usage (GB) per step for training on various classification tasks

Model	Image Classification		Text Classification		ListOps		Document Retrieval	
	Time	Memory	Time	Memory	Time	Memory	Time	Memory
LSG Attention	97.064	12.390	233.947	31.532	197.602	31.835	104.699	11.652
Longformer	154.511	26.418	385.044	63.022	480.873	91.038	208.290	29.645
BigBird	217.889	35.458	503.207	82.698	539.533	104.134	276.082	34.300
Reformer	138.936	18.545	354.041	49.067	309.772	50.614	144.285	19.590
SPION-C	87.471	6.536	217.360	13.770	187.382	14.259	106.608	5.854
SPION-F	85.298	6.550	212.343	13.745	197.928	15.267	106.111	5.877
SPION-CF	86.257	6.530	206.387	13.769	194.256	14.368	103.395	5.799

5.2 Computational Complexity Analysis

Figure 4 shows the FLOPs (Floating Point Operations) for computing sparse attention matrix S for various sparse Transformers. Note that all sparse Transformers maintain a total of C non-zero elements in the sparse attention matrix S . Therefore, the total operations required for computing S for each head in all compared sparse Transformers is the same as $2C(2D + 1) - L(D + 1)$. As shown in Fig. 4, our SPION-CF achieves up to a $5.91\times$ reduction in FLOPs compared to BigBird on the document retrieval task. This result demonstrates the

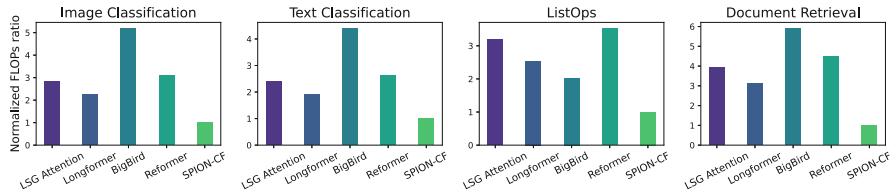


Fig. 4. Comparison of FLOPs required for computing the attention score matrix on various tasks. All the results are normalized to SPION-CF.

effectiveness of our convolutional flood-fill attention scheme, which dynamically captures relevant elements specific to the dataset and task. Furthermore, unlike the Longformer and BigBird models that apply the same fixed sparse pattern for every layer, our layer-wise sparse MHA enables the avoidance of unnecessary computations with non-relevant elements at each layer.

6 Conclusion

Due to the compute-intensive nature of the Transformer, that must perform a large number of MHA operations, we develop a novel sparsification scheme. This scheme leverages convolution filters and the flood fill algorithm to reduce the overall complexity of MHA operations. Our sparsification technique dynamically identifies the critical elements in the attention score matrix on a layer-wise basis. This method is applicable to many other Transformer models, not limited to the encoder-only Transformer. Experimental results on various datasets demonstrate that our sparse MHA approach significantly reduces training time and memory usage compared to state-of-the-art sparse Transformers, while achieving better accuracy on various classification tasks.

Acknowledgments. This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(RS-2024-00259099) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation), and in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. NRF-2021R1G1A1092597).

References

1. Ainslie, J., Ontanon, S., et al.: Etc: encoding long and structured inputs in transformers. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 268–284 (2020)
2. Beltagy, I., Peters, M.E., Cohan, A.: Longformer: the long-document transformer. arXiv preprint [arXiv:2004.05150](https://arxiv.org/abs/2004.05150) (2020)
3. Child, R., Gray, S., Radford, A., Sutskever, I.: Generating long sequences with sparse transformers. arXiv preprint [arXiv:1904.10509](https://arxiv.org/abs/1904.10509) (2019)

4. Condevaux, C., Harispe, S.: Lsg attention: extrapolation of pretrained transformers to long sequences. In: Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 443–454 (2023)
5. Devlin, J., Chang, M.W., et al.: Bert: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (2019)
6. Dosovitskiy, A., Beyer, L., et al.: An image is worth 16×16 words: transformers for image recognition at scale. In: International Conference on Learning Representations (2021)
7. Goldman, R.: Graphics gems, p. 304 (1990)
8. iNaturalist 2018 competition dataset. (2018)
9. Kitaev, N., Kaiser, L., Levskaya, A.: Reformer: the efficient transformer. In: Proceedings of the International Conference on Learning Representations (2020)
10. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
11. Liu, Y., Ott, M., et al.: Roberta: a robustly optimized bert pretraining approach. arXiv preprint [arXiv:1907.11692](https://arxiv.org/abs/1907.11692) (2019)
12. Nangia, N., Bowman, S.R.: Listops: a diagnostic dataset for latent tree learning. arXiv preprint [arXiv:1804.06028](https://arxiv.org/abs/1804.06028) (2018)
13. Nvidia: the api reference guide for cusparse, the cuda sparse matrix library. Technical report (2023). <https://docs.nvidia.com/cuda/cusparse/index.html>
14. Qiu, J., Ma, H., et al.: Blockwise self-attention for long document understanding. In: Findings of the Association for Computational Linguistics: EMNLP 2020, pp. 2555–2565 (2020)
15. Radev, D.R., Muthukrishnan, P., et al.: The ACL anthology network corpus. Lang. Res. Eval. **47**, 919–944 (2013)
16. Roy, A., Saffar, M., et al.: Efficient content-based sparse attention with routing transformers. Trans. Assoc. Comput. Linguist. **9**, 53–68 (2021)
17. Tay, Y., Dehghani, M., et al.: Efficient transformers: a survey. ACM Comput. Surv. **55**(6), 1–28 (2022)
18. Vaswani, A., Shazeer, N., et al.: Attention is all you need. Adv. Neural Inf. Process. Syst. **30** (2017)
19. Wang, S., Li, B.Z., et al.: Linformer: self-attention with linear complexity. arXiv preprint [arXiv:2006.04768](https://arxiv.org/abs/2006.04768) (2020)
20. Zaheer, M., Guruganesh, G., et al.: Big bird: transformers for longer sequences. Adv. Neural. Inf. Process. Syst. **33**, 17283–17297 (2020)
21. Zhang, H., Gong, Y., et al.: Poolingformer: long document modeling with pooling attention. In: International Conference on Machine Learning, pp. 12437–12446. PMLR (2021)
22. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. Adv. Neural Inf. Process. Syst. **28** (2015)



Towards Cost-Efficient Federated Multi-agent RL with Learnable Aggregation

Yi Zhang^{1,2}, Sen Wang¹, Zhi Chen¹, Xuwei Xu^{1,2}, Stano Funiak²,
and Jiajun Liu^{1,2(\square)}

¹ School of Electrical Engineering and Computer Science, The University of Queensland, St Lucia, Brisbane, QLD 4066, Australia
`{uqyzha91, sen.wang, zhi.chen, xuwei.xu}@uq.edu.au`

² DATA61, Commonwealth Scientific and Industrial Research Organisation (CSIRO),
Pullenvale, QLD 4069, Australia
`{stano.funiak, ryan.liu}@data61.csiro.au`

Abstract. Multi-agent reinforcement learning (MARL) often adopts centralized training with a decentralized execution (CTDE) framework to facilitate cooperation among agents. When it comes to deploying MARL algorithms in real-world scenarios, CTDE requires gradient transmission and parameter synchronization for each training step, which can incur disastrous communication overhead. To enhance communication efficiency, federated MARL is proposed to average the gradients periodically during communication. However, such straightforward averaging leads to poor coordination and slow convergence arising from the non-*i.i.d.* problem which is evidenced by our theoretical analysis. To address the two challenges, we propose a federated MARL framework, termed cost-efficient federated multi-agent reinforcement learning with learnable aggregation (FMRL-LA). Specifically, we use asynchronous critics to optimize communication efficiency by filtering out redundant local updates based on the estimation of agent utilities. A centralized aggregator rectifies these estimations conditioned on global information to improve cooperation and reduce non-*i.i.d.* impact by maximizing the composite system objectives. For a comprehensive evaluation, we extend a challenging multi-agent autonomous driving environment to the federated learning paradigm, comparing our method to competitive MARL baselines. Our findings indicate that FMRL-LA can adeptly balance performance and efficiency. Code and appendix can be found in https://github.com/ArronDZhang/FMRL_LA.

Keywords: Multi-agent reinforcement learning · Federated learning

1 Introduction

Federated reinforcement learning (RL) [4, 9, 11] has exhibited immense potential in integrating deep reinforcement learning models into a client-server paradigm. It has been proven effective in balancing communication efficiency and privacy

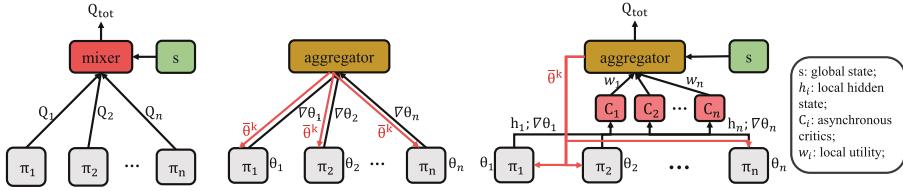


Fig. 1. Framework Comparison. π_i : local policy whose parameters are θ_i ; Q_i : local value function; Q_{tot} : joint value function; $\nabla\theta_i$: local gradients; $\bar{\theta}^k$: global model's parameters at round k ; h_i : local hidden state; w_i : local utility. The left subplot indicates the CTDE framework where the agents share parameters. The middle one represents the current vanilla FMARL where the agents maintain a global model through periodical averaging and the right figure refers to our FMRL-LA framework which selectively chooses and weights the involved agents to maximize the system utility.

preservation. With the burgeoning rise of the Internet of Things [20] that requires agent cooperation, and the prevalent use of multi-agent systems (MAS) [15, 17], it is desirable to develop federated multi-agent reinforcement learning (MARL).

In MARL, centralized training with decentralized execution (CTDE) [12, 21, 24] is a conventional learning regime lying between independent learning [28] and fully centralized learning [27]. This middle-ground strategy can not only mitigate the non-stationarity caused by agents' simultaneous decision-making but also prevent state and action spaces from expanding exponentially with agent number. Nevertheless, the training phase of CTDE requires continual communication between agents and servers. Thus, simply incorporating CTDE into federated learning (FL) [16] will lead to intractable communication overhead and bandwidth burdens. On the other hand, agent interactions with their local environments make their experiences non-independent and identically distributed (non-*i.i.d.*).

While recent efforts like FMARL [29] and Fed-MADRL [23] have marked advances in federated MARL [4], they typically assume an implicit *i.i.d.* in agent interactions and lack server-side coordination. Furthermore, these methods tend to optimize singular, task-oriented objectives, *e.g.*, the average speed in multi-vehicle autonomous systems [29] and the system throughput in wireless communications [23]. Such settings may be impractical for complex real-world settings with composite objectives. For instance, in autonomous driving [6, 13, 19], apart from communication efficiency, we also consider factors like success rate in reaching destinations, overall safety, and average vehicle speed.

In response to these challenges, we introduce Cost-Efficient Federated MARL with Learnable Aggregation (FMRL-LA). It decouples the CTDE by separating the training steps of the server and the client. On the server side, we propose two components for learnable aggregation: 1. Asynchronous critics evaluate the utility of learning agents, guiding selection for optimal system communication. 2. A centralized aggregator integrates global information with agent utilities to periodically update the global model, thus maximizing composite system tar-

gets. This design facilitates FMRL-LA to improve coordination under the federated paradigm. Delving deeper into the non-*i.i.d.* challenge posed by federated MARL, we theoretically delineate its adverse effects, providing a convergence upper bound. We further prove that the proposed learnable aggregation can mitigate the challenge. The comparison of different frameworks is exhibited in Fig. 1.

To conduct experiments with FMRL-LA, we resort to real-world multi-agent environment simulations based on MetaDrive [13], an intricate autonomous driving benchmark out of its flexibility across diverse scenarios. We extend it to support a client-server learning paradigm, incorporating communication efficiency. To further enhance the practicality, in addition to the existing navigation tasks, we design a multi-vehicle cooperative exploration task. Notably, we have integrated baselines from the representative methods of cooperative MARL [28] and communication-inclusive MARL [7], as well as the state-of-the-art method [19] using MetaDrive. Our experimental evaluations in navigation and exploration tasks underscore that FMRL-LA can optimize system performance and efficiency simultaneously, delivering a balanced performance across the metrics corresponding to composite objectives.

2 Preliminary

Cooperative MARL. Cooperative MARL can be formulated as Decentralized Partially Observable Markov Decision Processes (Dec-POMDPs) [12, 31], described by a tuple $G = \langle n, S, O, A, P, r, Z, \gamma \rangle$, where n is the number of agents, and S, O denote the state and observation spaces. A , the joint action space, is the product of all agents' action spaces, *i.e.*, $A = \prod_{i=1}^n A_i$, where i is the agent index. We use lowercase s, o, a to represent an element in the corresponding space. The environments' dynamics are characterized by the transition function $P(s'|s, a) : S \times A \times S \rightarrow [0, 1]$. The system has a shared team reward function $r(s, a) : S \times A \rightarrow \mathbb{R}$. In the aspect of each agent, due to the partially observable setting, at time step t , its observation o_t is drawn by applying the function Z to the current state s_t . Thus, $o_t^i = Z_i(s^t) : S \rightarrow O$. γ is the discount factor. The solution of a Dec-POMDP is a joint policy $\bar{\pi} = (\pi_1, \pi_2, \dots, \pi_n)$, where π_i stands for the policy of agent i and we use $\theta_i, \bar{\theta}$ to represent the parameters of agent i and the joint policy, respectively. Each agent policy is trained with the agent's experience comprised of a collection of agent observation-action history denoted as $\xi_i = \{(o^t, a^t, o^{t+1})\}_{t=0}^T$, where T denotes the time horizon. In addition, we use $\xi = \{(s^t, a^t, s^{t+1}, r^t)\}_{t=0}^T$ to represent one global team episode. The goal of MARL is to learn a joint policy that can maximize the expected cumulative reward, *i.e.*, $\pi^* = \arg \max_{\bar{\pi}} \mathbb{E}_{\tau \sim \bar{\pi}}[R_T(\tau)]$, where $R_T(\tau) = \sum_{t=0}^T \gamma^t r^{(t)}$.

Federated MARL. We use τ to represent the number of local updates. K is the termination condition of the training process, which is usually set as maximum communication rounds [4]. ψ denotes the system communication efficiency. We use the parameter θ to represent policy π . $F(\cdot)$ represents the global objective function, while $F_i(\cdot)$ stands for the local objective function for each agent i .

Their relationship between the global objective and the locals in [4, 23, 29] are the same: $F(x) = \frac{1}{n} \sum_{i=1}^n F_i(x)$. In round k , all agent policies are synchronized as $\bar{\theta}^k$, which is drawn from the server. Then, each agent interacts with the environment concurrently to accumulate local experience for updating the local policy indicated by $\{\theta_i^{k,\tau_i}\}_{i=1}^n$. Next, the parameters $\{\theta_i^{k,\tau_i}\}_{i=1}^n$ or stochastic gradients $\{g(\theta_i^{k,j}; \xi_i^{k,j})\}_{j=1}^{\tau_i}$ for $i \in 1, 2, \dots, n$ will be uploaded to the server. To sum up, the update rules on the server and client side are:

$$\bar{\theta}^{k+1} = \bar{\theta}^k - \eta \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^{\tau_i} g(\theta_i^{k,j}), \quad \theta_i^{k+1,j} = \begin{cases} \bar{\theta}^{k+1}, & j \bmod \tau_i = 0, \\ \theta_i^{k,j} - \eta g(\theta_i^{k,j}), & \text{otherwise.} \end{cases} \quad (1)$$

To indicate the convergence of the algorithm, we use the expected averaged gradient norm to guarantee convergence to a stationary point [2, 25, 26]:

$$\mathbb{E}\left[\frac{1}{K} \sum_{k=0}^{K-1} \|\nabla F(\bar{\theta}^k)\|^2\right] \leq \epsilon, \quad (2)$$

where $\|\cdot\|$ is the ℓ_2 -norm and ϵ is used to describe the sub-optimality. When the above condition holds, we say the algorithm achieves an ϵ -suboptimal solution.

3 Federated MARL with Learnable Aggregation

Server Side. When federated MARL [23, 29] adopts Eq. (1) as the update rule for the server, it implicitly assumes that the agents are homogeneous. However, in real-world environments, the agents are diverse in various aspects such as computing capability, network connection, and local observation distributions, which results in heterogeneous agents with non-*i.i.d.* experience distribution.

To deal with these issues, we introduce **Asynchronous Critics** to dynamically evaluate the agent utilities in each communication round. Each critic corresponds to one learning agent. Its goal is to maximize the return of the current agent. The inputs are hidden information h_i^k , accumulated rewards in recent communication round $r^k := \sum_{j=\tau_i^{k-1}}^{\tau_i^k} r_i^j$ and in agent history $\sum_{j=0}^{\tau_i^k} r_i^j$. The output is a prediction of the agent's local utility:

$$w_i^k = C_i \left(h_i^k, r^k, \sum_k r^k \right), \quad (3)$$

where C_i is the asynchronous critic network of agent i . The output w_i^k can be zero. The corresponding agent does not need to upload its training parameters to the server in the current communication round, achieving client selection.

Next, the agent utilities are passed through a **Centralized Aggregator** to facilitate coordination. It works similarly to the mixing network in value decomposition methods such as [21, 24], which takes the local utility function as input and facilitates agent coordination by maximizing the system utility

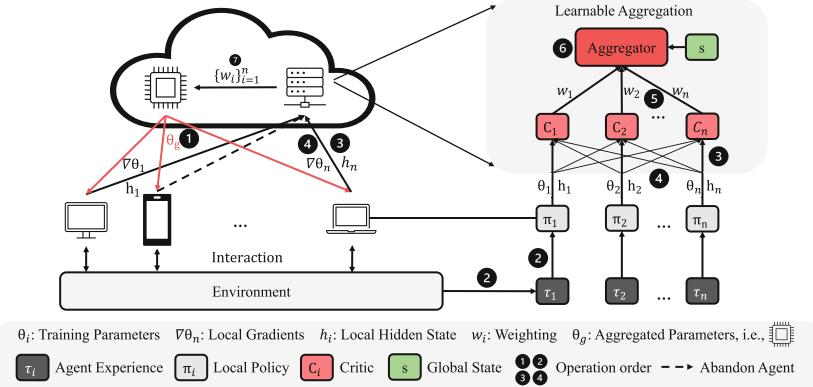


Fig. 2. The workflow of our proposed framework.

condition on the global state. The RL loss is back-propagated to the critics for improving the local utility estimation:

$$Q_{tot} = Mix(w_1^k, w_2^k, \dots, w_n^k | s), \quad (4)$$

where Q_{tot} denotes the system utility, which is reflected by the composite objectives. The server aggregates the gradients based on the local utilities to update the global policy. Thus, the update rule of the server is:

$$\bar{\theta}^{k+1} = \bar{\theta}^k - \eta \sum_{i=1}^n w_i^k \sum_{j=1}^{\tau_i^k} g(\theta_i^{k,j}), \quad (5)$$

while the update rule for the clients remains the same as the client side of Eq. 1.

Client Side. Considering the generalization of our method, we choose an independent reinforcement learning algorithm and take the hidden states as additional outputs. During the communication, the upload process of the clients can be divided into two stages. In the first stage, the agents upload their rewards and hidden information to the asynchronous critics for local utility estimation and agent selection, optimizing communication efficiency. In the second stage, the selected agents upload their gradients to the server for aggregation.

Framework Design. Compared to the update rule for FMARL [23, 29] represented by Eq. (1), we adopt a weighted aggregation for global policy update implemented by the learnable aggregation module. The workflow of FMRL-LA is illustrated by Fig. 2. Specifically, 1. the server broadcasts the global model $\bar{\theta}^k$ to each agent; 2. The agents learn local behavior policies $\{\pi_i\}_{i=1}^n$ by interacting with the environment and maintaining hidden states; 3 and 4. During client-server communication, agents conduct the two-stage upload described in the above subsection; 5. The centralized aggregator maximizes the composite system objectives condition on the global states to facilitate coordination; 6. The

global model is updated based on the local utilities. The corresponding pseudo code is in Appendix A.1.

4 Convergence Analysis

In FL theory, a substantial body of research is devoted to exploring convergence properties under diverse settings. These settings predominantly fall into two categories, *i.i.d.* [4, 16, 23, 29], and non-*i.i.d.* [10, 14, 18, 26]. While *i.i.d.* settings facilitate robust theoretical results, non-*i.i.d.* settings are more realistic about data distribution. Despite the theoretical progresses in FL schemes in supervised learning, the influence of non-*i.i.d.* in federated MARL remains uncharted. To address this issue, we conduct our theoretical analysis in the following paragraphs.

We begin with showing the convergence under the ideal *i.i.d.* setting. To do that, we first list out the following assumptions:

Assumption 1 (*Lipschitz continuity*). *The local loss functions at the client side are Lipschitz continuous, which means $\|\nabla F_i(\theta_1) - \nabla F_i(\theta_2)\| \leq L\|\theta_1 - \theta_2\|$, $\forall i \in \{1, 2, \dots, n\}$.*

Assumption 2 (*Unbiased gradients and bounded variance under *i.i.d.**). *The stochastic gradients at the client side are unbiased estimators of the global gradient, i.e., $\mathbb{E}_\xi[g_i(\theta)] = \nabla F(\theta)$ and $\mathbb{E}_\xi[\|g_i(\theta) - \nabla F(\theta)\|^2] \leq \mu\|\nabla F(\theta)\|^2 + \sigma^2$, $\forall i \in \{1, 2, \dots, n\}$, μ and σ^2 are non-negative.*

Assumption 3 (*Unbiased local gradients and bounded variance under non-*i.i.d.**). *The stochastic gradient at each client is an unbiased estimator of the local gradient, i.e., $\mathbb{E}_\xi[g_i(\theta)] = \nabla F_i(\theta)$ and $\mathbb{E}_\xi[\|g_i(\theta|\xi) - \nabla F_i(\theta)\|^2] \leq \mu\|\nabla F_i(\theta)\|^2 + \sigma^2$, $\forall i \in \{1, 2, \dots, n\}$, μ and σ^2 are non-negative.*

Assumption 4 (*Bounded Dissimilarity*). *For any sets of weights $\{w_i^k \geq 0\}_{i=1}^n$, $\sum_{i=1}^n w_i^k \leq M^k$, M^k is finite, $\forall k \in [0, K]$, there exist constants $\beta^2 \geq 1$, $\kappa^2 \geq 0$ such that $\sum_{i=1}^n w_i^k \|\nabla F_i(\theta)\|^2 \leq \beta^2 \{\|\sum_{i=1}^n w_i^k \nabla F_i(\theta)\|^2, \|\sum_{i=1}^n \frac{1}{n} \nabla F_i(\theta)\|^2\}_{min} + \kappa^2$, $\forall k \in [0, K]$. If local loss functions are identical to each other, then we have $\beta^2 = 1$, $\kappa^2 = 0$.*

Assumption 1 is Lipschitz continuity, a common assumption in the convergence analysis in FL theory. Assumption 2 states that the local stochastic gradient is an unbiased estimation of the local gradient and the variance of the deviation is bounded to support our exploration under a *i.i.d.* setting. Assumption 3, on the other hand, is the gradient bias and variance assumption under non-*i.i.d.* setting. Assumption 4 is inspired by FedNova [26], which bounds the dissimilarities on the weighted norm of local gradients.

We provide the convergence bound under the *i.i.d.* and non-*i.i.d.* settings as Theorem 1 and Theorem 2, respectively. We further provide a special case to show that there is room for tightening the bound with the learnable aggregation mechanism as Theorem 3 in the Appendix A.4. In addition, the proof of these theorems as well as more theoretical details are provided in the Appendix A.4.

Theorem 1. Suppose $\{\theta_i^{k,j}\}$ and $\{\bar{\theta}^k\}$ are parameters' sequences generated by Eq.(1). The federated MARL is conducted under Assumptions 1 and 2. If the total communication rounds K is large enough, which can be divided by τ , and the learning rate η satisfies:

$$\{L\eta < 1, 2L^2\eta^2\tau(2\mu + 1 + \tau) < 1\}, \quad (6)$$

then the expected gradient norm after K iterations is bounded by:

$$\mathbb{E}\left[\frac{1}{K} \sum_{k=1}^K \|\nabla F(\bar{\theta}^k)\|^2\right] \leq \frac{2[F(\bar{\theta}^1) - F(\bar{\theta}^K)]}{\eta K} + \frac{\eta L\sigma^2}{n} + \eta^2 L^2 \sigma^2 (\tau + 1), \quad (7)$$

where $\bar{\theta}^1$ stands for one lower bound of the objective function.

Theorem 2. Suppose $\{\bar{\theta}^k\}$ are parameters' sequences generated by the weighted gradients Eq.(5), while the $\{\theta_i^{k,j}\}$ remains the same. The federated MARL is conducted under Assumptions 1, 3 and 4. If the total communication rounds K is large enough, and the learning rate η satisfies (6), then the expected gradient norm after K iterations is bounded by:

$$\begin{aligned} \mathbb{E}\left[\frac{1}{K} \sum_{k=1}^K \|\nabla F(\bar{\theta}^k)\|^2\right] &\leq \frac{4(E[F(\bar{\theta}^1) - E[F(\bar{\theta}^K)])}{K\eta} \\ &+ 4 \left(C + D + E + F + \mu\eta C \sum_{k=0}^K \frac{1}{K} \sum_{i=1}^n w_i^2 \tau_i^k \right), \end{aligned} \quad (8)$$

where $A = \frac{1}{K} \sum_{i=1}^K A$, $B = 2L^2\eta^2\tau(2\mu + 1 + \tau)$, $C = \frac{\eta^2\sigma^2L^2}{\mu L\eta\tau\beta^2 + 2B\beta^2}$, $D = \frac{(1-2\mu L\eta\tau\beta^2)\kappa^2}{(2\mu L\eta\tau\beta^2 + 4B\beta^2)(1+4\beta^2)}$, $E = \frac{\mu L\eta\tau\kappa^2}{2\mu L\eta\tau\beta^2 + 4B\beta^2}$, and $F = \frac{L\eta\sigma^2}{2K} \sum_{k=1}^K (M^k)^2$.

Discussion. The result of Theorem 1 is an ideal upper bound where the distribution of each client is *i.i.d.* More generally, in Theorem 2, we provide another upper bound to illustrate the impact introduced by the non-*i.i.d.* issue. Comparing the bounds in these two theorems, it is obvious that the convergence bound in the non-*i.i.d.* setting is greater than that in the *i.i.d.* setting.

Special Cases. When $w_i^k \equiv \frac{1}{n}$, the convergence upper bound degenerates to the same as FMARL [29], which derives the same upper bound as in its Theorem



Fig. 3. The six extended scenarios used in our evaluation.

2. When $w_i^k \equiv \frac{1}{n}$ and $\tau_i^k \equiv 1$, the convergence upper bound further degenerates to the same as PASGD, which coincides with the conclusions drawn from [25].

Discussion with Federated Learning in Supervised Learning. We compare our method with FedNova [26] – a general federated method targeting supervised learning. It induces several federated learning methods in a general form and targets the problem of an unbalanced number of local updates by regularizing the weights for one-period local gradients with the number of local updates. However, in MARL, the different number of policy iterations may not be a more significant reason than the diversity of local environments to the non-*i.i.d.* issue. In other words, this issue cannot be rectified by simply regularizing the weights of local gradients by the number of local policy iterations, which necessitates the importance of our learnable aggregation mechanism.

5 Experiments

Baseline Methods. We present a comparative analysis of our proposed method alongside strong baselines covering a wide range of related fields, namely conventional MARL (**IPPO** [28]), communication-based MARL (**RIAL** and **DIAL** [7]), the state-of-the-art method **CoPO** [19] and **FMARL** [29] in a multi-agent autonomous driving simulation environment, MetaDrive [13]. We provide a detailed introduction and adaptation of these methods in the Appendix A.6.

Implementation by Extending MetaDrive. The MetaDrive benchmark represents a flexible and lightweight simulation benchmark for autonomous driving, encompassing a variety of tasks that serve as a reasonable abstraction of real-world environments. In this paper, we focus on its multi-agent tasks. The agents adopt the conventional MARL suite, including parameter sharing and disregarding communication overhead. We add six challenging scenarios whose maps are depicted in Fig. 3. Originally, MetaDrive used parameter sharing for all the methods, so we first expand this benchmark into a client-server learning setting by adopting a non-parameter sharing scheme and simulating a virtual server. This server only periodically collects local gradients and hidden states from the clients, aggregates the gradients for the update of the global model, and then sends it back to the agents. When an existing vehicle terminates and a new vehicle spawns, it accepts the latest global model from the server to prevent the “cold start” problem. To enrich the testing bed and serve as a real-world simulation, in addition to the existing navigation task, we extend a cooperative exploration task where vehicles cooperatively explore the specified destinations.

Evaluation Metrics. MetaDrive provides three evaluation metrics: success rate, efficiency, and safety, which respectively reflect navigation capability (the success ratio of vehicles relative to the total number of vehicles), navigation efficiency (the differences between the successes and failures within a unit of time), and safety driving (the number of crushes within an episode). In our cooperative exploration task, we adapt the navigation success rate to the exploration success rate. For realistic concerns, we also record the communication overhead which



Fig. 4. The system performance and efficiency comparison with baselines in six scenarios of the cooperative navigation tasks.

is reflected by the number of parameters exchanged between the agents and the server. The **system utility** is derived from the weighted average of these metrics, where the weights reflect the specific requirements of particular scenarios. In our experiments, we employ a simple average to gauge overall performance.

In summary, in the **cooperative navigation** task, our evaluation metrics including the navigation success rate (*Success*), safety (*Safety*), overall agents' speed (*Speed*), and communication efficiency (*Comm-efficiency*). As for the **cooperative exploration**, our evaluation metrics including the exploration success rate (*Explore*), safety (*Safety*), overall agents' speed (*Speed*), communication efficiency (*Comm-efficiency*).

Main Results Analysis. The experiment results on cooperative navigation and exploration across six scenarios can be found in Fig. 4 and Fig. 5, respectively. More detailed results related to the performance on two tasks can be found in Table 2 to Table 5 in the Appendix A.7.

Our Performance. In both tasks, FMRL-LA achieves or is comparable to the best success, speed, safety, and system utility. From the perspective of communication efficiency, since RIAL [7] uses a simple actor-critic algorithm with few parameters, its communication efficiency serves as a reference of the upper bound for the methods with a relatively complex algorithm on the client side. Actually, more than half of the baselines adapt PPO [22] as the clients' algorithm. Among them, our method exhibits the capacity to dynamically harmonize the system performance and communication efficiency.

In detail, we focus on the performance of IPPO, FMARL, and our FMRL-LA in both tasks. The three methods have nearly the same client-side algorithms but differ from each other on the server side. IPPO only conducts direct training parameter averaging, while the FMARL adds a weight decay mechanism during the averaging. And FMRL-LA dynamically learns the aggregation weights. From IPPO to FMARL, then to FMRL-LA, the performance of success, safety, speed,

and system utility follow an ascending manner. We believe that it is because the performance of IPPO is bound by the averaging capability of all learning agents while FMARL can enlarge the bound to some extent by weight decay. Nonetheless, the potential performance of FMRL-LA is bound by the best agent, which improves the generalization of our method since we can deploy a suitable behavior model for the clients in advance if we can make use of prior knowledge about the environments.

Task Comparison. Comparing the overall performance of all methods on cooperative navigation and cooperative exploration on six scenarios, we can find navigation is more difficult than exploration, especially in relatively complex scenarios, *e.g.*, scenario 4, 5, and 6. We notice that in the navigation task, each agent has its own destination. Therefore, we believe the different performance on the two tasks may be because the relationship between the local utilities and the system utility is easier to capture in exploration than in navigation.

Scenario Difficulty. In both tasks, if we compare the performance pair by pair such as scenario 1 and scenario 5, we observe that generally, the more building blocks involved in a scenario, the more challenging it is. Then, looking into the performance of scenario 1 and 2, both of them consists of four building blocks while scenario 2 contains one more roundabout than scenario 1. If we compare the safety of CoPO, FMARL, and our FMRL-LA, three robust methods in the two tasks on these six scenarios, we can find that roundabout tends to result in more crushes. Further, if we compare the performance of scenarios 1 and 4 on two tasks, we observe that though the two scenarios both contain one roundabout and the same number of building blocks, the performance of our method and other baselines is generally better in scenario 1. Considering the difference in these two scenarios, we hypothesize that it is due to the influence of wide turn. For intuitive methods like IPPO and RIAL, it is difficult for them to avoid crushing or driving out of the roads during the wide turns. On the other hand, the safety of CoPO in scenarios 3 and 4 is relatively high, it may benefit from its explicit modeling of the surrounding agents.

Ablation Study. To investigate the effectiveness of our design and components in FMRL-LA, we conduct an ablation study about the usage of asynchronous critics and a centralized aggregator as well as an alternative design for the centralized aggregator. We intuitively use the average of multiple metrics as the system utility. From Table 1 we can observe that if we directly use critics without the centralized aggregator, the performance is unstable, resulting in large standard errors. In scenario 4, the performance w.r.t. system utility is worse than federated IPPO. We believe that without the coordination of the centralized aggregator, the server cannot filter out less valuable agents, so their parameters can depreciate the update of the global model in the current round. Meanwhile, the asynchronous critics are useful in our method since the variant that only uses an aggregator performs worse than the full model. We believe that accepting information from all involved agents can stagnate the learning of an aggregator due to redundant information. When we change the centralized

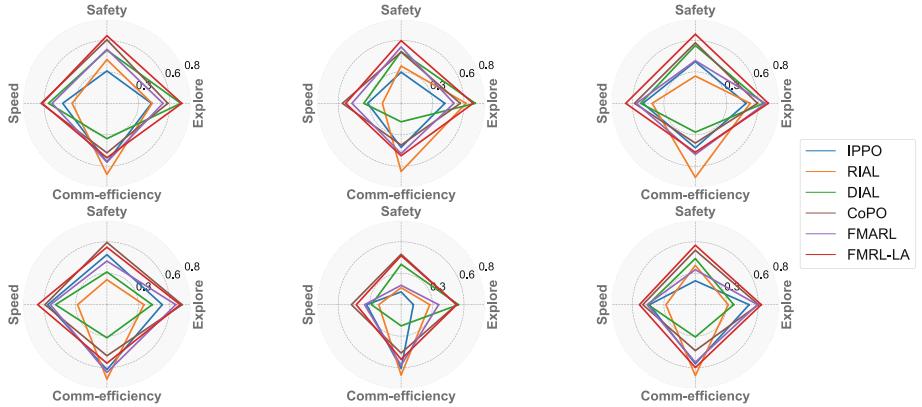


Fig. 5. The system performance and efficiency comparison with baselines on six scenarios of the cooperative exploration tasks.

Table 1. Ablation study on the effectiveness of our critical components on navigation task. The **system utility** is provided.

Experiment	Scenario1	Scenario2	Scenario3	Scenario4	Scenario5	Scenario6
IPPO	49.96±3.06	45.13±3.68	51.49±2.73	33.12±5.29	34.41±6.17	34.33±5.97
FMRL-LA w/o aggregator	54.75±6.00	49.28±5.23	52.06±4.71	40.43±7.02	32.85±8.92	40.22±7.74
FMRL-LA w/o critics	52.69±3.55	48.46±4.65	55.49±3.18	45.35±5.26	38.49±6.19	48.07±5.57
FMRL-LA w/ vdn-aggregator	57.98±2.97	55.84±4.39	57.26±3.69	52.50±5.80	46.98±6.81	50.86±5.09
FMRL-LA	59.84±2.82	62.30±4.29	63.16±3.33	57.27±4.70	50.28±6.67	56.42±5.63

aggregator to a VDN-based [24] one, it yields an inferior performance compared to our QMIX-based [21] design, which suggests the non-linear modeling of the relationship between the agents and the server is more suitable for complex realistic environments than a simple sum as in VDN.

Client Selection Analysis. To verify that FMRL-LA can effectively select involved agents to save communication costs, we also conduct experiments with varying numbers of agents to show the client selection effect. The experiment results and elaboration can be found in the Appendix A.7.

6 Related Work

Cooperative MARL. Cooperative MARL has widespread applications in real-world scenarios [13, 30]. Current methods are mainly developed in game scenarios [3, 15, 21] where the methods can focus on technical design rather than practical details. These environments support parameter sharing (PS) [5] and CTDE regime [8, 15] to enable multiple agents to be trained on one device and facilitate cooperation, respectively. However, when it is the stage to consider practical MARL in realistic environments [1, 19], either PS or CTDE cannot be simply applied due to privacy concerns and communication overhead.

Federated MARL. Federated MARL [23] appears to be a feasible way towards realistic MARL. Most of these methods enable agents to learn individual behavior policies and set a virtual server to maintain a global policy. The agents’ policies are aggregated and updated periodically through communication with the server [4]. In this way, the communication overhead is reduced, and the majority of them aggregate the local gradients by direct averaging [29] or weighted by the relative mini-batch size [23]. Although this oversimplified update may work well under *i.i.d.* setting, the MAs are naturally non-*i.i.d.* due to the interaction among agents. The notorious non-*i.i.d.* issue can stagnate convergence [14, 26]. Besides, without centralized training, it is hard for MARL to learn coordination [12].

7 Conclusion

We aim to adapt MARL for real-world applications by introducing a hybrid distributed, client-server learning framework that takes into account communication and computation overhead. Our framework offers theoretical guarantees even under the influence of non-*i.i.d.* distribution of agents in local environments. To empirically validate the efficacy of our proposed Cost-Efficient Federated Multi-Agent Reinforcement Learning with Learnable Aggregation (FMRL-LA) method, we modify an existing multi-agent autonomous driving simulation environment to conform to a client-server scheme. Experimental results emphasize the superior performance against baseline methods.

Acknowledgements. This work is supported by project DE200101610 funded by Australian Research Council and CSIRO’s Science Leader project R-91559.

References

1. Abegaz, M., Erbad, A., Nahom, H., Albaseer, A., Abdallah, M., Guizani, M.: Multi-agent federated reinforcement learning for resource allocation in UAV-enabled internet of medical things networks. *IoT-J* (2023)
2. Bottou, L., Curtis, F.E., Nocedal, J.: Optimization methods for large-scale machine learning. *SIAM* (2018)
3. Chaudhuri, R., Mukherjee, K., Narayananam, R., Vallam, R.D.: Collaborative reinforcement learning framework to model evolution of cooperation in sequential social dilemmas. In: *PAKDD* (2021)
4. Chen, T., Zhang, K., Giannakis, G.B., Başar, T.: Communication-efficient policy gradient methods for distributed reinforcement learning. *TCNS* (2021)
5. Christianos, F., Papoudakis, G., Rahman, A., Albrecht, S.V.: Scaling multi-agent reinforcement learning with selective parameter sharing. In: *ICML* (2021)
6. Du, X., Wang, J., Chen, S.: Multi-agent meta-reinforcement learning with coordination and reward shaping for traffic signal control. In: *PAKDD* (2023)
7. Foerster, J., Assael, I.A., De Freitas, N., Whiteson, S.: Learning to communicate with deep multi-agent reinforcement learning. In: *NeurIPS* (2016)
8. Hu, S., Zhu, F., Chang, X., Liang, X.: UPDeT: universal multi-agent reinforcement learning via policy decoupling with transformers. In: *ICLR* (2021)

9. Jin, H., Peng, Y., Yang, W., Wang, S., Zhang, Z.: Federated reinforcement learning with environment heterogeneity. In: AISTATS (2022)
10. Karimireddy, S.P., Kale, S., Mohri, M., Reddi, S., Stich, S., Suresh, A.T.: Scaffold: stochastic controlled averaging for federated learning. In: ICML (2020)
11. Khodadadian, S., Sharma, P., Joshi, G., Maguluri, S.T.: Federated reinforcement learning: linear speedup under Markovian sampling. In: ICML (2022)
12. Kuba, J.G., Chen, R., Wen, M., Wen, Y., Sun, F., Wang, J., Yang, Y.: Trust region policy optimisation in multi-agent reinforcement learning. In: ICLR (2022)
13. Li, Q., Peng, Z., Feng, L., Zhang, Q., Xue, Z., Zhou, B.: MetaDrive: composing diverse driving scenarios for generalizable reinforcement learning. TPAMI (2022)
14. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. In: MLSys (2020)
15. Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., Mordatch, I.: Multi-agent actor-critic for mixed cooperative-competitive environments. In: NeurIPS (2017)
16. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: AISTATS (2017)
17. Mo, J., Xie, H.: A multi-player MAB approach for distributed selection problems. In: PAKDD (2023)
18. Pang, Y., Zhang, H., Deng, J.D., Peng, L., Teng, F.: Rule-based collaborative learning with heterogeneous local learning models. In: PAKDD (2022)
19. Peng, Z., Hui, K.M., Liu, C., Zhou, B.: Learning to simulate self-driven particles system with coordinated policy optimization. In: NeurIPS (2021)
20. Pinto Neto, E.C., Sadeghi, S., Zhang, X., Dadkhah, S.: Federated reinforcement learning in IoT: applications, opportunities and open challenges. Appl. Sci. (2023)
21. Rashid, T., Samvelyan, M., De Witt, C.S., Farquhar, G., Foerster, J., Whiteson, S.: Monotonic value function factorisation for deep multi-agent reinforcement learning. JMLR (2020)
22. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. [arXiv:1707.06347](https://arxiv.org/abs/1707.06347) (2017)
23. Song, Y., Chang, H.H., Liu, L.: Federated dynamic spectrum access through multi-agent deep reinforcement learning. In: GLOBECOM (2022)
24. Sunehag, P., et al.: Value-decomposition networks for cooperative multi-agent learning. [arXiv:1706.05296](https://arxiv.org/abs/1706.05296) (2017)
25. Wang, J., Joshi, G.: Cooperative SGD: a unified framework for the design and analysis of local-update SGD algorithms. JMLR (2021)
26. Wang, J., Liu, Q., Liang, H., Joshi, G., Poor, H.V.: Tackling the objective inconsistency problem in heterogeneous federated optimization. In: NeurIPS (2020)
27. Wen, M., et al.: Multi-agent reinforcement learning is a sequence modeling problem. Front. Comput. Sci. (2022)
28. de Witt, C.S., et al.: Is independent learning all you need in the starcraft multi-agent challenge? [arXiv:2011.09533](https://arxiv.org/abs/2011.09533) (2020)
29. Xu, X., Li, R., Zhao, Z., Zhang, H.: The gradient convergence bound of federated multi-agent reinforcement learning with efficient communication. TWC (2023)
30. Yu, C., Velu, A., Vinitsky, E., Gao, J., Wang, Y., Bayen, A., Wu, Y.: The surprising effectiveness of PPO in cooperative multi-agent games. In: NeurIPS (2022)
31. Zhou, X., Matsubara, S., Liu, Y., Liu, Q.: Bribery in rating systems: a game-theoretic perspective. In: PAKDD (2022)



LongStory: Coherent, Complete and Length Controlled Long Story Generation

Kyeongman Park, Nakyeong Yang, and Kyomin Jung^(✉)

Seoul National University, Seoul, Republic of Korea
{zzangmane,kjung,yny0506}@snu.ac.kr

Abstract. A human author can write any length of story without losing coherence. Also, they always bring the story to a proper ending, an ability that current language models lack. In this work, we present the LongStory for coherent, complete, and length-controlled long story generation. LongStory introduces two novel methodologies: (1) the long and short-term contexts weight calibrator (CWC) and (2) long story structural positions (LSP). The CWC adjusts weights for long-term context Memory and short-term context Cheating, acknowledging their distinct roles. The LSP employs discourse tokens to convey the structural positions of a long story. Trained on three datasets with varied average story lengths, LongStory outperforms other baselines, including the strong story generator Plotmachine, in coherence, completeness, relevance, and repetitiveness. We also perform zero-shot tests on each dataset to assess the model’s ability to predict outcomes beyond its training data and validate our methodology by comparing its performance with variants of our model.

Keywords: Long Story generation · Completeness · CWC · LSP

1 Introduction

The story generation task is one of the most challenging problems in natural language processing since it requires writing long lengths with a consistent context. Existing studies have tried to solve this problem but have failed to generate variable lengths of stories since they have only considered fixed lengths when generating stories.

Longformer [1] has tried to solve this problem of handling longer sequences by combining global and local attention in sliding windows. However, they only focused on increasing the input context window size. Their maximum generation length is only 1,024 tokens. Short-length problems in text generation also occur in recent large language models such as GPT-4 [4]. Plug-and-blend [2] has also covered a similar problem, but the length limitation remains since it only aimed to control a few sentences.

To address this challenge, a recursive paragraph generation approach is necessary to compose long stories, given the length limitations imposed by existing

language models. However, this recursive generation of stories may cause undesirable forgetting of the previous context since the information leak may occur in the recursive process of information transfer (*coherence*) [13]. Also, existing studies have mentioned that language models tend to repeat the same story in a recursive generation setting. (*variety* and *repetitiveness*) [6, 33].

Thus, previous works [6, 12, 13] have tried to define coherence and repetitiveness metrics and used them to measure the ability of story generators.

However, their attempts are also limited only to short sentence generation problems and have never considered **completeness**. Completeness is the ability to conclude a story of any length properly. It is a significant metric not only for evaluating story generation but also for a wide range of open-domain generation and dialogue system tasks. Therefore, in our model evaluation, we consider not only coherence and repetitiveness but also prioritize completeness as a critical measure of performance.

In this paper, we tackle this challenging problem by introducing a novel long story generator, *Coherent, Complete and Length Controlled Long story Generator* (LongStory), which covers from a few hundred tokens to 10k tokens length, not limited to only a few sentences. The LongStory utilizes two novel methods: (1) long and short-term contexts weight calibrator (CWC) and (2) long story structural positions (LSP) to tackle coherence and completeness.

Specifically, we implement the CWC using BERT-tiny to calibrate the weight to which long-term and short-term contexts are utilized since both contexts contribute differently in every paragraph when writing a story. For the LSP, we use discourse tokens, which means the order of paragraphs, to give information about the structural positions of paragraphs to the story generator (e.g., *< front >*, *< middle >*, *< ending >*, *< next_is-ending >*). Our model uses more abundant discourse tokens than the previous study [6] since a detailed understanding of story structure is essential to generate much longer posts.

We use three diverse story generation datasets with varying average lengths to train our model on representations of different story lengths. We introduce quantitative metrics for coherence, completeness, and repetitiveness, evaluating the model's performance against other baselines, including the established story generator Plotmachine [6]. The experimental results for three story generation datasets demonstrate that our model outperforms the other baselines in coherence, completeness, and relevance. Surprisingly, our model also shows better results in repetitiveness, suggesting that our methods are effective for the variety. Furthermore, we performed zero-shot tests on each dataset to assess how well our model predicts outcomes in settings beyond its training data. Additionally, our analysis of the augmented CWC version suggests that elevating relevance does not always translate to improvements in coherence and completeness.

The main contributions of this paper are (1). a new open-domain metric called completeness. (2) a new challenge, *Coherent, Complete and Length Controlled Long story Generation* (LongStory), incorporating CWC and LSP methodologies, and (3) the presentation of datasets with varying average document lengths along with zero-shot tests for them.

2 Related Works

Many contemporary automatic story generation models have employed prompt engineering strategies, manipulating input prompts for large pretrained language models like GPT-3 [2, 7, 13]. Despite the effectiveness of these black-box models, they could not directly enhance the internal structure for optimized performance. In contrast, our approach focuses on directly improving the structural aspects of existing language models.

2.1 Neural Story Generation

Story generators using neural networks have been developed in various ways [3, 9, 18, 25, 30]. The main approaches have included outline-based generation [3, 6, 7, 13, 25, 27], event graph-based [8, 15], goal-oriented methodologies [16, 17], and common sense reasoning [18, 19, 32]. Outline-based generation methods often involve interpolating plot keywords [6, 25]. Our model also uses keywords to plan plots and create relevant stories. Event graph-based approaches have proven highly effective, but only if given a detailed plot of the entire article. Our model uses keywords for the entire story but not the plot of the story, thus performing a more challenging task. Goal-oriented methodologies have aimed to make the story’s characters achieve given goals, akin to controlling the ending of a story [5, 12]. Common sense inference methods have focused on generating realistic texts by training on common sense datasets.

Length-Controllable Story Generation. Many models have generated text within a fixed range of length. The length has been mostly determined when selecting the training datasets. For example, Plotmachine [6] has fixed the number of paragraphs it generates, while EtriCA [8] and Fusion [3] also have set their generation lengths by their training dataset. Re3 [7] has generated narratives by applying pre-trained GPT-3 [9], and the length has been adjusted by manipulating the prompts to GPT-3. However, Re3 has not solely focused on controlling length. As such, it has not been specifically designed to ensure that the models generate without losing coherence and completeness with various lengths.

2.2 Recursive Models

Due to finite parameters, there has always been a limit to the length a language model can generate at once. While Longformer [1] and GPT-4 [4] have introduced 5K or even 100K context windows, their output length remains only a few thousand tokens at most. In contrast, human writers can produce thousands of pages without encountering such limitations. This reality implies the necessity of employing a language model recursively [6, 7, 25].

2.3 Autometric Metrics

In NLP generation tasks, traditional evaluation methods commonly involve metrics such as ROUGE [21], BLEU [22]. These metrics predominantly rely on n-gram matching between labels and predictions. However, applying these

evaluation methods to the story generation task may not be sufficient. The reason lies in the fact that a well-crafted story may not necessarily exhibit similarity to the label; in fact, a story resembling the label might score poorly in terms of diversity and creativity [23, 26]. In our case, we additionally employ *coherence*, *completeness*, and *repetitiveness* as the primary evaluation metrics along with the ROUGE scores. While *coherence* and *repetitiveness* is a well-established metric in many natural language generation tasks [5–7, 13, 24, 25], we are the first to use *completeness* as a metric (Fig. 1).

3 Methodology

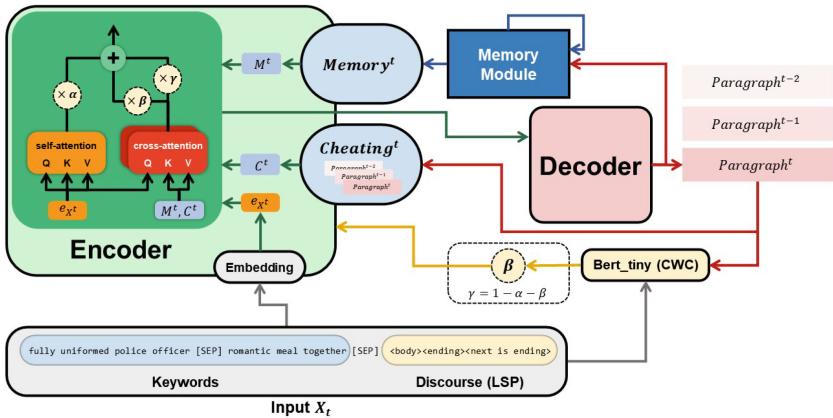


Fig. 1. Model architecture. LongStory takes the keywords of the entire story and the discourse tokens(LSP) representing the order of the target paragraphs as input. The BERT-tiny serves as the long and short-term context weight calibrator (CWC), determining the degree to which long-term and short-term contexts are employed. The CWC takes the discourse tokens and the last generated paragraph as inputs and outputs the optimal β and γ (defined as $1-\alpha-\beta$) for every paragraph. While the α is a hyperparameter applied to input embedding, β is a learnable parameter for the long-term context $Memory(M^t)$ and short-term context $Cheating(C^t)$

3.1 Task Description

Our challenge can be described as follows: Given keywords $K = (k_1, k_2, k_3 \dots)$, discourse tokens $D_t = (d_1^t, d_2^t, d_3^t \dots)$ (each k_i, d_i^t is a token), and a separation token [SEP], an input $X^t = (k_1, [SEP], k_2, [SEP], k_3, \dots, [SEP], d_1^t, d_2^t, d_3^t \dots)$. The model should generate each t-th paragraph recursively: $Y^t = (y_1^t, y_2^t, y_3^t, \dots)$ (each y_i^t is a sentence, a set of tokens) from given contexts of previous paragraphs and the input X^t , which is coherent and compleutive. The model should minimize negative log-likelihood loss L_{LM} of each paragraph t:

$$L_{LM} = - \sum_{i=1}^n \log P(y_j^t | y_{<j}^t, X^t) \quad (1)$$

$$\log P(y_j^t | y_{<j}^t, X^t) = \text{softmax}(H_j^t W + b) \quad (2)$$

$$H_j^t = LM(X^t, M^t, C^t) \quad (3)$$

where LM is an attention-based language model, H_j^t is the j-th position of the language model's last hidden state, M^t is the *Memory* which saves the long-term context by the t-th paragraph, and C^t is *Cheating* which keeps the short term context from the very last paragraphs it generated. The M^t ensures a sustained understanding of the broader context, while C^t contributes to short-term continuity.

3.2 Long and Short Term Contexts Weight Calibrator(CWC)

BERT-tiny is inserted into our model as a calibrator and trained on the same negative log likelihood loss, L_{LM} . The pooler output of BERT-tiny is used for the determination of the context weight, β, γ .

$$B_k^t = Bert(D^t, Y^{t-1}) \quad (4)$$

$$\beta = (1 - \alpha) \cdot \sigma(B_1^t W + b), \quad \gamma = 1 - \alpha - \beta \quad (5)$$

where $\alpha \in [0, 1]$ is a hyperparameter¹, β is a learnable parameter for the long-term context *Memory*(M^t) and short-term context *Cheating*(C^t), σ is the sigmoid function, B_k^t is the k-th position of the last hidden state of BERT-tiny, and B_1^t is the position of $[CLS]$ token of it, often called the ‘pooler output.’ The long and short-term contexts, M^t and C^t , are computed as:

$$g^t = \sigma(W_1 M^{t-1} + W_2 h^{t-1}), \quad \hat{M}^t = \tanh(W_3 M^{t-1} + W_4 h^{t-1}) \quad (6)$$

$$M^t = (1 - g^t) \odot M^{t-1} + g^t \odot \hat{M}^t \quad (7)$$

$$C^t = \tanh(Emb(y_{n-c}^{t-1}, \dots, y_{n-1}^{t-1}, y_n^{t-1})) \quad (8)$$

where Emb denotes a positional embedding layer that involves the element-wise sum of word embeddings and positional encoding vectors, h^{t-1} is the average embedding of Y^{t-1} , and c is a hyperparameter determining the size of the cheating window. We utilized the same computation process of initializing and updating M^t as introduced in Plotmachine. Finally, we define the computation of an attention block within a LM as follows:

$$\begin{aligned} LM(X^t, M^t, C^t) &= \alpha \text{Attention}(Q, K, V) + \beta \text{Attention}(Q, M^t, M^t) \\ &\quad + \gamma \text{Attention}(Q, C^t, C^t) \end{aligned} \quad (9)$$

¹ We covered a test where alpha is also a learnable parameter rather than a constant hyperparameter in Sect. 4.3.2. In this test, the BERT-tiny determines α , β , and γ independently, and finally divides by their sum, so that each variable adds up to 1.

where $Q = \text{Emb}(X^t)W^Q$, $K = \text{Emb}(X^t)W^K$, $V = \text{Emb}(X^t)W^V$, and *Attention* is a typical transformer attention block [20]. By weighting the two contexts, the model can generate cohesive and contextually connected text.

3.3 Long Story Structural Positions (LSP)

We must provide structural positional information for each paragraph [31]. All discourse tokens are of the following types $\langle \text{intro} \rangle$, $\langle \text{body} \rangle$, $\langle \text{tail} \rangle$, $\langle \text{front} \rangle$, $\langle \text{middle} \rangle$, $\langle \text{ending} \rangle$, $\langle \text{next_is_ending} \rangle$. $\langle \text{intro} \rangle$ corresponds to the first paragraph, $\langle \text{tail} \rangle$ to the last paragraph, and $\langle \text{body} \rangle$ to everything in between as introduced in the previous work [6]. However, since our task is to generate much more paragraphs, $\langle \text{body} \rangle$ token is not sufficient to give representation to our model. Therefore, four tokens ($\langle \text{front} \rangle$, $\langle \text{middle} \rangle$, $\langle \text{ending} \rangle$, $\langle \text{next_is_ending} \rangle$) are added: $\langle \text{front} \rangle$ for the first 1/3, $\langle \text{middle} \rangle$ for the next 1/3, and $\langle \text{ending} \rangle$ for the last 1/3. $\langle \text{next_is_ending} \rangle$ is for the paragraph immediately before the end, as its name implies.

3.4 Base Pretrained Model

We finetune a Pretrained Language Model(PLM) based on Transformer [20] for this task. Although our algorithm is model-agnostic so that it can apply any of PLM, we choose BART [10] because (1) it had learned denoising sequences to efficiently extract information from a given input, appropriate for our task as the model should extract representations of the inputs and contexts. (2) it is free to download from online². However, our model holds the potential for even more excellent performance when applied to larger language models in the future.

4 Experiments

4.1 Experiments Set-Up

4.1.1 Datasets

We train our models on Writing Prompts [3], Booksum [14], and Reedsy Prompts³ datasets. For Booksum, we use the ‘Chapter’ column as a dataset. Since our task is to generate various lengths of stories with coherence and completeness, we must prepare datasets of different lengths. We use the same train/valid/test split from the original papers [3, 14], but since Reedsy Prompts is not used in any other works, we split it into train/valid/test sets of 60k/4k/4k. The NLTK library’s sentence tokenizer is utilized for document segmentation into sentences during truncation, with each sentence sequentially compiled into a paragraph, ensuring a maximum length of 200 tokens per paragraph. For each

² <https://huggingface.co/facebook/bart-large>.

³ We crawl this dataset from [https://blog.reedsy.com/short-stories/..](https://blog.reedsy.com/short-stories/)

dataset, we extract keywords using the RAKE algorithm [11] to make keyword-story pairs. These keywords are used as input for the model to write each paragraph of a story (Table 1).

Table 1. The average number of tokens, paragraphs per document, and the number of documents used in experiments.

	Avg # of Tokens	Avg # of Paragraphs	# of documents
WritingPrompts	768	3.4	300k
Booksum	6065	27.7	12k
ReedsyPrompts	2426	12.3	68k

4.1.2 Baselines

We mainly compare our model to Plotmachine, the only model that performs paragraph-level recursive story generation. We also compare the performance of our model against GPT-2 [28] and BART with the recursive generation setting, no applied memory, and cheating contexts; thus, it does not utilize the CWC. Plotmachine receives only the discourse tokens from the original paper, while other baselines are given the same as ours. We also compare our model with the ablated version. The *LongStory with no memory* (LongStory^{-M}), *LongStory with no cheating* (LongStory^{-C}) and *LongStory with no new discourse* (LongStory^{-D}) show the effectiveness of context vectors and newly added discourse tokens.

4.1.3 Implementation Details

Experiments are run on an NVIDIA RTX A5000 GPU server. For training steps, we use the teacher-forcing method so each label is used for not only Y_t but also contexts. We set the hyperparameter $\alpha = \frac{1}{2}$ because providing the model with sufficient inductive bias for input embeddings was beneficial for coherence and completeness. We also configured the size of the cheating window c up to the last three paragraphs it generated. For generation, we use beam sampling with $top_k = 50$, $top_p = 0.95$, $no_repeat_ngram_size = 3$ and $repetition_penalty = 3.5$ [18] for every length of paragraphs. To reproduce Plotmachine, we refer to the author’s github repository.⁴.

4.2 Experimental Results

⁴ <https://github.com/hrashkin/plotmachines>.

Table 2. ROUGE-1,2,L scores, in-self-BLEU n-gram scores, Coherence, Completeness scores, and the average length of a paragraph per 5,10,20,30,50 number of paragraphs. These are results from the combined test datasets of Writing Prompts, Reedsy Prompts, and Booksum. We bold the highest model scores in each column and underline the second highest model scores.

5 Paragraphs										
	R-1	R-2	R-L	B2	B-3	B-4	B-5	Coherence	Completeness	AvgL
Golden Label				0.28	0.15	0.07	0.03	82.62	62.5	161
Plotmachine	29.85	7.15	28.16	0.47	0.33	0.23	0.16	60.19	2.47	192
BART	38.77	13.64	37.35	0.61	0.49	0.41	0.33	54.93	2.39	167
GPT-2	33.41	9.25	31.82	0.55	0.42	0.32	0.24	<u>63.08</u>	1.13	193
LongStory	39.41	13.50	37.84	<u>0.49</u>	<u>0.36</u>	0.27	<u>0.20</u>	65.13	21.15	158
<i>LongStory</i> ^{-M}	<u>39.69</u>	13.66	38.20	0.51	0.38	0.29	0.22	59.38	-3.04	167
<i>LongStory</i> ^{-C}	39.35	14.02	37.90	0.62	0.51	0.42	0.35	62.59	8.53	170
<i>LongStory</i> ^{-D}	39.69	<u>13.67</u>	<u>38.08</u>	0.50	0.37	0.28	0.21	62.27	<u>9.21</u>	166
10 Paragraphs										
	R-1	R-2	R-L	B2	B-3	B-4	B-5	Coherence	Completeness	AvgL
Golden Label				0.37	0.19	0.08	0.03	80.44	55.5	164
Plotmachine	30.34	7.74	28.87	0.62	0.47	0.34	0.25	64.17	3	195
BART	36.28	11.71	34.83	0.67	0.53	0.43	0.34	60.46	2.65	172
GPT-2	32.32	8.84	30.74	0.64	0.49	0.36	0.27	66.06	3.85	196
LongStory	<u>37.46</u>	<u>12.11</u>	<u>35.82</u>	0.59	0.43	0.31	0.23	<u>64.8</u>	21.15	166
<i>LongStory</i> ^{-M}	37.32	12.05	35.72	<u>0.61</u>	<u>0.45</u>	0.34	0.25	58.4	3.59	171
<i>LongStory</i> ^{-C}	37.06	12.11	35.49	0.67	0.53	0.43	0.34	63.24	6.56	170
<i>LongStory</i> ^{-D}	37.54	12.30	35.86	0.60	0.44	0.33	0.25	63.4	<u>11.12</u>	171
20 Paragraphs										
	R-1	R-2	R-L	B2	B-3	B-4	B-5	Coherence	Completeness	AvgL
Golden Label				0.45	0.24	0.12	0.05	80.13	57.66	166
Plotmachine	31.45	9.49	29.96	0.71	0.57	0.44	0.34	64.26	4.75	194
BART	35.43	12.29	34.02	0.74	0.62	0.51	0.42	61.37	-1.53	169
GPT-2	33.28	10.52	31.60	0.72	0.58	0.46	0.35	65.8	0.7	198
LongStory	<u>36.85</u>	12.69	<u>35.22</u>	0.68	0.53	0.40	0.31	<u>64.6</u>	6.73	169
<i>LongStory</i> ^{-M}	36.72	<u>12.78</u>	35.17	0.70	0.55	0.43	0.33	58.3	-1.96	170
<i>LongStory</i> ^{-C}	35.44	12.35	33.89	0.74	0.62	0.51	0.46	64.51	0.23	170
<i>LongStory</i> ^{-D}	37.03	12.81	35.35	0.69	0.54	0.41	<u>0.32</u>	61.14	<u>6.38</u>	170
30 Paragraphs										
	R-1	R-2	R-L	B2	B-3	B-4	B-5	Coherence	Completeness	AvgL
Golden Label				0.48	0.25	0.11	0.04	80.19	58.64	164
Plotmachine	28.85	8.89	27.58	0.75	0.62	0.49	0.38	63.24	6.7	193
BART	30.62	10.53	29.25	0.78	0.67	0.55	0.46	61.65	-0.61	164
GPT-2	29.43	9.78	28.06	0.76	0.62	0.49	0.37	65.13	-0.26	199
LongStory	<u>32.24</u>	10.97	30.90	0.73	0.58	0.45	0.35	65	23.9	163
<i>LongStory</i> ^{-M}	31.81	<u>10.96</u>	30.23	0.74	0.60	0.47	<u>0.37</u>	57.76	<u>19.43</u>	167
<i>LongStory</i> ^{-C}	29.96	10.16	28.76	0.79	0.67	0.56	0.46	64.32	-0.1	165
<i>LongStory</i> ^{-D}	32.42	10.95	31.02	0.73	0.59	0.47	0.48	60.4	11.58	168
50 Paragraph										
	R-1	R-2	R-L	B2	B-3	B-4	B-5	Coherence	Completeness	AvgL
Golden Label				0.53	0.30	0.15	0.06	78.06	52.4	166
Plotmachine	29.01	9.92	28.14	<u>0.81</u>	0.70	0.59	0.49	<u>67.95</u>	-9.08	167
BART	27.83	10.07	27.05	0.84	0.75	0.65	0.56	66.73	-5.66	167
GPT-2	28.03	9.94	27.17	0.84	0.73	0.62	0.52	67.73	3.9	198
LongStory	31.81	11.65	30.87	0.79	0.66	0.54	0.43	63.67	31.76	161
<i>LongStory</i> ^{-M}	30.97	<u>11.51</u>	29.84	0.81	<u>0.69</u>	0.58	<u>0.48</u>	63.06	-10.47	168
<i>LongStory</i> ^{-C}	27.87	9.94	27.12	0.84	0.74	0.64	0.54	68.99	20	170
<i>LongStory</i> ^{-D}	<u>30.99</u>	10.80	<u>30.09</u>	0.81	0.69	<u>0.58</u>	0.48	63.25	21.68	167

4.2.1 Coherence and Completeness

For the Coherence scorer, we divide two consecutive paragraphs into true pairs and two randomly selected paragraphs into fake pairs. We then train GPT-2 to use these constructed datasets on a classification task to distinguish whether the two paragraphs are true or fake folds [12, 18], and convert the result of it into a score $\in [0, 1]$ using sigmoid function, as a Coherence score. We do the same for Completeness scorer, except for defining the true fold with the last paragraphs and the fake fold with the non-last paragraphs. It's important to highlight that the reported Completeness scores are calculated as the *last paragraph's Completeness score* minus the *average Completeness scores of non-last paragraphs*. This approach is chosen because an ideal story should generate a last paragraph appropriately and treat the body distinctly. Regarding Coherence, as shown in Table 2, our model is the best or second best in the 5,10,19,30 paragraph length sample. Plotmachine had a higher Coherence score than BART but lower than our model or GPT-2. In the ablated analysis, LongStory^{-C} almost always outperforms LongStory^{-M} and LongStory^{-D} on Coherence. This suggests that *Memory* is central to capturing the natural flow between paragraphs. In the Completeness, our model is the best for all lengths. LongStory^{-M} and LongStory^{-C} have no absolute advantage in this column, so it is difficult to know exactly how *Memory* and *Cheating* affect Completeness. Still, since our model performs the best, we can say that a balanced mix of the two contexts improves performance. The results for LongStory^{-D} also suggest that the newly added discourse tokens help improve Completeness.

4.2.2 Relevance

We also show the average ROUGE scores between the generated documents and the golden labels to evaluate how well the model reproduces the golden label from the keywords. This serves as a metric to assess the relevance of the keywords to the predictions. As shown in Table 2, our model is best for results with 50 paragraph lengths and second best in many cases. Our model outperforms Plotmachine, GPT-2, and BART for all lengths of results. In the ablated analysis, LongStory^{-D} scores the best, for the most part. This shows that the newly added order tokens have little or even a negative effect on relevance. However, in the repeatability test in the next section, LongStory^{-D} performed worse than our model, which suggests that it reproduced the overall repetitive n-gram. For the most part, LongStory^{-M} performed better than LongStory^{-C}, which means that *Cheating* was more effective than *Memory* in relevance.

4.2.3 Repetitiveness

To see how much repetition occurs within a single output, we calculate the n-gram BLEU score by taking one paragraph as a hypothesis and the rest as a reference and averaging the results. We call the averaged results the in-SELF

BLEU score⁵. The higher in-SELF BLEU score means that there are greater repetitions within the text and less diversity. As shown in Table 2, Our model performs best on the in-self-BLEU score for all but five paragraphs length. For five paragraph lengths, Plotmachine performs best, and ours is second highest. However, for longer results, our model is always better. In the ablated analysis, LongStory^{-M} outperforms LongStory^{-C} and BART. This suggests that *Cheating* prevented the model from over-attending to past context or repeating n-grams irrelevant to the overall context. The better performance of our model over LongStory^{-D} suggests that the order tokens we added positively affected reducing repetition.

4.3 Further Analysis

4.3.1 Zero-Shot Test

To see how well LongStory can have representations of the others, we do zero-shot tests on three models: WP, BK, and RP. As shown in Table 3, Writing Prompt is large enough to have some representation on other types of test datasets, while Booksum and ReedsyPrompts are not. Total outperforms all others, underscoring the effectiveness of our methodology in integrating these diverse datasets.

Table 3. Zero-shot test results averaged across test datasets of 5, 10, 20, 30, and 50 paragraphs. WP, RP, and BK are our models trained on Writing Prompts, Reedsy Prompts, and Booksum only, respectively. Total is the model trained on all three datasets combined. We bold the best results for each dataset.

	Writing Prompts			Booksum			Reedsy Prompts		
	RP	BK	Total	WP	RP	Total	WP	BK	Total
Coherence	38.55	43.75	65.78	56.84	48.62	58.70	57.56	47.79	62.11
Completeness	-0.1	1.51	27.06	1.67	-4.03	12.37	7.75	0.93	21.29

4.3.2 Augmented CWC

We experiment with calibrating not only memory and cheating but also the proportion of input embeddings in the attention block. Table 4 shows that the augmented version is better than ours on ROUGE-1,2, L scores but worse on Coherence and Completeness. This indicates that models excelling in relevance may not necessarily outperform in natural flow.

⁵ Note that in-self-BLEU score is not the same as self-BLEU score [6]. The self-BLEU score has taken one whole generated document as a hypothesis and the others as references, which cannot represent inner repetitiveness.

Table 4. A comparison of the Augmented version of our model with the averaged scores of ROUGE-1, ROUGE-2, ROUGE-L, Coherence, and Completeness across test datasets of 5, 10, 20, 30, and 50 paragraphs. We bold the best results for each dataset.

	R-1	R-2	R-L	Coherence	Completeness
Aug-LongStory	38.25	13.06	36.65	61.01	5.25
LongStory	38.04	12.90	36.46	64.92	18.36

5 Conclusion

We introduce a novel task, Length Controllable Story Generation, aimed at recursively producing paragraph-by-paragraph stories of considerable length while ensuring coherence and completeness. To achieve this, we employ the long- and short-term contexts weight calibrator (CWC) and long story structural positions (LSP). The model is trained on three distinct datasets with varying average lengths, enabling it to learn representations of different lengths. Quantitative analysis demonstrates that our model excels in generating longer stories that exhibit coherence, completeness, relevance, and reduced repetitiveness compared to other baseline models.

Acknowledgements. We thank anonymous reviewers for their constructive and insightful comments. K. Jung is with ASRI, Seoul National University, Korea. This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) [NO.2022-0-00184, Development and Study of AI Technologies to Inexpensively Conform to Evolving Policy on Ethics & NO.2021-0-01343, Artificial Intelligence Graduate School Program (Seoul National University)].

References

1. Beltagy, I., Peters, M.E., Cohan, A.: Longformer: the long-document transformer. arXiv preprint [arXiv:2004.05150](https://arxiv.org/abs/2004.05150) (2020)
2. Lin, Z., Riedl, M.O.: Plug-and-blend: a framework for plug-and-play controllable story generation with sketches. In: Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, vol. 17, No. 1, pp. 58–65 (2021)
3. Fan, A., Lewis, M., Dauphin, Y.: Hierarchical neural story generation. arXiv preprint [arXiv:1805.04833](https://arxiv.org/abs/1805.04833) (2018)
4. OpenAI. GPT-4 Technical Report (2023)
5. Peng, N., Ghazvininejad, M., May, J., Knight, K.: Towards controllable story generation. In: Proceedings of the First Workshop on Storytelling, pp. 43–49 (2018)
6. Rashkin, H., Celikyilmaz, A., Choi, Y., Gao, J.: PlotmaChines: outline-conditioned generation with dynamic plot state tracking. arXiv preprint [arXiv:2004.14967](https://arxiv.org/abs/2004.14967) (2020)
7. Yang, K., Peng, N., Tian, Y., Klein, D.: Re3: generating longer stories with recursive reprompting and revision. arXiv preprint [arXiv:2210.06774](https://arxiv.org/abs/2210.06774) (2022)

8. Tang, C., Lin, C., Huang, H., Guerin, F., Zhang, Z.: EtriCA: event-triggered context-aware story generation augmented by cross attention. arXiv preprint [arXiv:2210.12463](https://arxiv.org/abs/2210.12463) (2022)
9. Brown, T., et al.: Language models are few-shot learners. In: Advances in Neural Information Processing Systems, vol. 33, pp. 1877–1901 (2020)
10. Lewis, M., et al.: Bart: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. arXiv preprint [arXiv:1910.13461](https://arxiv.org/abs/1910.13461) (2019)
11. Rose, S., Engel, D., Cramer, N., Cowley, W.: Automatic keyword extraction from individual documents. Text Mining: Applications and Theory, 1–20 (2010)
12. Wang, S., Durrett, G., Erk, K.: Narrative interpolation for generating and understanding stories. arXiv preprint [arXiv:2008.07466](https://arxiv.org/abs/2008.07466) (2020)
13. Yang, K., Klein, D., Peng, N., Tian, Y.: DOC: improving long story coherence with detailed outline control. arXiv preprint [arXiv:2212.10077](https://arxiv.org/abs/2212.10077) (2022)
14. Kryściński, W., Rajani, N., Agarwal, D., Xiong, C., Radev, D.: Booksum: a collection of datasets for long-form narrative summarization. arXiv preprint [arXiv:2105.08209](https://arxiv.org/abs/2105.08209) (2021)
15. Yao, L., Peng, N., Weischedel, R., Knight, K., Zhao, D., Yan, R.: Plan-and-write: towards better automatic storytelling. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01, pp. 7378–7385 (2019)
16. Alabdulkarim, A., Li, W., Martin, L.J., Riedl, M.O.: Goal-directed story generation: augmenting generative language models with reinforcement learning. arXiv preprint [arXiv:2112.08593](https://arxiv.org/abs/2112.08593) (2021)
17. Pradyumna, T., Murtaza, D., Lara, J. M., Mehta, A., Harrison, B.: Controllable neural story plot generation via reward shaping. In: Proceedings of the International Joint Conference Artificial Intelligence, pp. 5982–5988 (2019)
18. Guan, J., Huang, F., Zhao, Z., Zhu, X., Huang, M.: A knowledge-enhanced pre-training model for commonsense story generation. In: Transactions of the Association for Computational Linguistics, vol. 8, pp. 93–108 (2020)
19. Peng, X., Li, S., Wiegreffe, S., Riedl, M.: Inferring the reader: guiding automated story generation with commonsense reasoning. arXiv preprint [arXiv:2105.01311](https://arxiv.org/abs/2105.01311) (2021)
20. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
21. Lin, C.Y.: Rouge: a package for automatic evaluation of summaries. In: Text summarization branches out, pp. 74–81 (2004)
22. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the 40th annual meeting of the Association for Computational Linguistics, pp. 311–318 (2002)
23. Safovich, Y., Azaria, A.: Fiction sentence expansion and enhancement via focused objective and novelty curve sampling. In: 2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI), pp. 835–843. IEEE (2020)
24. Li, J., Bing, L., Qiu, L., Chen, D., Zhao, D., Yan, R.: Learning to write stories with thematic consistency and wording novelty. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, vol. 01, pp. 1715–1722 (2019)
25. Hu, Z., Chan, H.P., Liu, J., Xiao, X., Wu, H., Huang, L.: Planet: dynamic content planning in autoregressive transformers for long-form text generation. arXiv preprint [arXiv:2203.09100](https://arxiv.org/abs/2203.09100) (2022)
26. Yang, K., Klein, D.: FUDGE: controlled text generation with future discriminators. arXiv preprint [arXiv:2104.05218](https://arxiv.org/abs/2104.05218) (2021)

27. Sakaguchi, K., Bhagavatula, C., Bras, R. L., Tandon, N., Clark, P., Choi, Y.: proscript: partially ordered scripts generation via pre-trained language models. arXiv preprint [arXiv:2104.08251](https://arxiv.org/abs/2104.08251) (2021)
28. Budzianowski, P., Vulić, I.: Hello, it's GPT-2—how can I help you? towards the use of pretrained language models for task-oriented dialogue systems. arXiv preprint [arXiv:1907.05774](https://arxiv.org/abs/1907.05774) (2019)
29. Welleck, S., Kulikov, I., Kim, J., Pang, R.Y., Cho, K.: Consistency of a recurrent language model with respect to incomplete decoding. arXiv preprint [arXiv:2002.02492](https://arxiv.org/abs/2002.02492) (2020)
30. Zellers, R., et al.: Defending against neural fake news. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
31. Guan, J., Mao, X., Fan, C., Liu, Z., Ding, W., Huang, M.: Long text generation by modeling sentence-level and discourse-level coherence. arXiv preprint [arXiv:2105.08963](https://arxiv.org/abs/2105.08963) (2021)
32. Lewis, P., et al.: Retrieval-augmented generation for knowledge-intensive NLP tasks. In: Advances in Neural Information Processing Systems, vol. 33, pp. 9459–9474 (2020)
33. McCoy, R.T., Smolensky, P., Linzen, T., Gao, J., Celikyilmaz, A.: How much do language models copy from their training data? evaluating linguistic novelty in text generation using raven. Trans. Assoc. Comput. Linguist. **11**, 652–670 (2023)



Relation-Aware Label Smoothing for Self-KD

Jeongho Kim¹ and Simon S. Woo²

¹ Korea Advanced Institute of Science and Technology, Daejeon, South Korea
r1awjdhek@kaist.ac.kr

² Sungkyunkwan University, Seoul, South Korea
swoo@g.skku.edu

Abstract. Knowledge distillation (KD) is widely used to improve models' performances by transferring a larger teacher's knowledge to a smaller student model. However, KD has a disadvantage where a pre-trained teacher model is required, which can lead to training inefficiency. Therefore, self-knowledge distillation, enhancing the student by itself, has been proposed. Although self-knowledge distillation shows remarkable performance improvement with fewer resources than conventional teacher-student based KD approaches, existing self-KD methods still require additional time and memory for training. We propose Relation-Aware Label Smoothing for Self-Knowledge Distillation (RAS-KD) that regularizes the student model itself by utilizing the inter-class relationships between class representative vectors with a light-weight auxiliary classifier. Compared to existing self-KD methods that only consider the instance-level knowledge, we show that proposed global-level knowledge is sufficient to achieve competitive performance while being extremely efficient training cost. Also, we achieve extra performance improvement through instance-level supervision. We demonstrate RAS-KD outperforms existing self-KD approaches in various tasks with negligible additional cost.

1 Introduction

Recently, deep neural networks (DNNs) based approaches have achieved remarkable performance across various research areas. However, high-performing DNN models tend to become huge with a large number of parameters and complex internal model architectures. Typically, training such models requires prohibitive computing resources. However, such resources are not always available in practice. To address such issues, several approaches such as weight pruning have been proposed to compress or reduce the model architecture while maintaining its high performance. Also, knowledge distillation (KD) has been proposed by Hinton *et al.* [4]. Although conventional KD [4] achieved remarkable results in various research fields, it also has the unavoidable disadvantage of requiring a

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-981-97-2253-2_16.

high-performing pre-trained teacher network. Further, it can cause the following issues: 1) training the student model can lead to inefficiency because training the teacher model should be performed first. 2) Also, finding the best teacher networks is not always trivial, considering the relative size or model structure of the student networks. Such inefficiency and cost issues from the teacher model have opened up a new research area, self-knowledge distillation (self-KD), a self-improving training strategy without the pre-trained teacher model. Yun *et al.* [14] proposed the class-wise self-KD, which utilizes the output distribution of intra-class data for supervision, and Kim *et al.* [6] used the inference results of the previous epoch as a regularizer of the current epoch to improve performance. Also, Zhang *et al.* [15] introduced the softmax representation from intermediate branch networks, and Ji *et al.* utilized a larger auxiliary self-teacher network based on BIFPN [10] structure. However, those self-KD approaches cause training inefficiency because of their additional forward pass [14] and require large memory to save all the previous results [6]. Also, they use additional networks such as branch networks [15] or an auxiliary self-teacher network [5], which burden the training even more.

In this paper, we propose a novel Relation-aware Label Smoothing for Self-KD (RAS-KD), which shows efficient training cost via global-level supervision, while showing competitive performance and achieve additional performance improvement with instance-level soft targets from a lightweight auxiliary classifier. Compared to other self-KD methods, our soft targets use a negligible number of parameters, meaning that the temporal and spatial cost are nearly the same as the vanilla training method that uses the cross entropy loss only using hard-labels. We demonstrate the effectiveness of RAS-KD through extensive experiments with other baselines, achieving SOTA performance, not only on a large-scale image classification but also on object detection and transfer learning tasks.

2 Related Work

Many approaches have been proposed to prevent overfitting and enhance the generalization ability of DNNs. In particular, label smoothing (LS) [9] is one of the popular methods to mitigate an overconfidence issue by smoothing the one-hot encoded hard-label. Also, based on the powerful teacher network in knowledge distillation, a variety of approaches [8, 12] has been proposed. While the aforementioned distillation methods effectively conjugate and distill the representations of the teacher models, their remarkable results were achieved by exploiting a pre-trained larger network, leading to training inefficiency. Recently, to alleviate the burden of conventional teacher-student distillation frameworks, self-KD has been proposed, aiming to improve performance without a larger teacher model. Zhang *et al.* [15] introduced more powerful soft targets from the ensemble of students and several branch networks. Moreover, Ji *et al.* [5] also utilized a BIFPN-based auxiliary self-teacher network to transfer the refined feature-map and soft target to the student network. However, during training, the entire network to train is larger than the original student network due to the

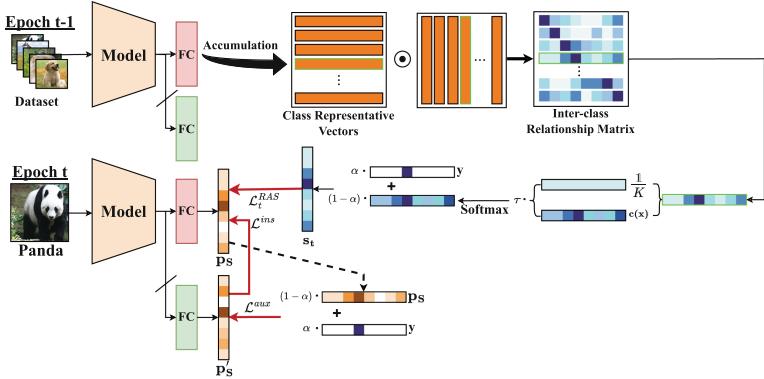


Fig. 1. Overview of RAS-KD. First, class representative vectors are obtained by averaging the entire predictions for each class. Next, to acquire an inter-class relationship prior matrix used for transferring information of class relationship, RAS-KD utilizes the Pearson correlation to map inter-class relationships. Then, we provide more weights to similar classes by multiplying τ to relationship vectors to emphasize the differing relationships between relevant classes and dissimilar classes. Moreover, the auxiliary classifier provides diversity via instance-level logit supervision.

intermediate branch networks or the auxiliary self-teacher network, consuming more VRAM and training time as in conventional KD. Another approach is to match the output of different input images. Class-wise self-KD (CS-KD) was proposed by Yun *et al.* [14], to minimize the KL divergence between two images in the same class. However, this approach requires a forward process to acquire additional supervision, requiring an additional computational cost. Also, Kim *et al.* [6] proposed progressive self-knowledge distillation (PS-KD), which utilizes a predictive probability of the student network at the previous epoch to compose soft targets. While existing self-KD approaches solely rely on instance-level soft targets, leading to expensive overhead, we propose to use the global-level knowledge to obtain better performance with more efficient computational memory and training time.

3 Our Approach

Soft Targets. In KD, instead of directly using the hard-label (i.e., one-hot vector) as target prediction, the student can learn from the soft targets consisting of the hard-label y and the teacher’s predictive probabilities \mathbf{p}_T . Hinton *et al.* [4] proposed a temperature τ , which is used to scale the softmax output \mathbf{p}_T of the teacher and \mathbf{p}_S of the student. In particular, τ has an effect of smoothing the distribution to handle the classes with low probabilities better as follows:

$$\hat{p}_i = \phi(\mathbf{z}; \tau)_i = \frac{\exp(z_i/\tau)}{\sum_j \exp(z_j/\tau)}, \quad (1)$$

where $\mathbf{z} = [z_1, z_2, \dots, z_K]$ denotes the logit vector. And \mathbf{z}_t and \mathbf{z}_s are the logit vector from a teacher and a student, respectively. Using Eq. 1, we can train the student network by minimizing the cross entropy \mathcal{H} not only between \mathbf{y} and \mathbf{p}_s , but also the scaled probabilities $\hat{\mathbf{p}}_T$ and $\hat{\mathbf{p}}_S$ as follows:

$$\mathcal{L} = \alpha \cdot \mathcal{H}(\mathbf{y}, \mathbf{p}_s) + (1 - \alpha) \cdot \tau^2 \cdot \mathcal{H}(\hat{\mathbf{p}}_T, \hat{\mathbf{p}}_S). \quad (2)$$

Inter-Class Relationship Prior. We define the similarity as the inter-class relationship prior, $\mathbf{c}(\mathbf{x})$ from the given input \mathbf{x} . To characterize the inter-class relationships, we use the Pearson correlation coefficient (PCC) [3] to calculate the correlations between the mean logit per class of ResNet18 trained only with hard-label on CIFAR-100, which contains 100 classes over 20 super-classes consisting of 5 sub-classes. To provide better visualization, we group the 5 sub-classes within the same super-class, and plot the heatmap of PCC in Fig. 2. As shown, we can observe that the classes within the same super-classes clearly demonstrate strongly correlated relationships with high intensity forming squares along the diagonal. Therefore, this result confirms that the logits from the trained vector can represent relationships between classes similar to each other. Moreover, this can be regarded as utilizing the attention scores of the self-attention mechanism [11] directly in the learning process. Thus, we use the inter-class relationship prior to better transfer knowledge in our RAS-KD.

Smoothed Distribution. The next important component in our work is generating a smoothed distribution over all classes to construct soft targets. As Yuan *et al.* [13] show that LS regularization is an ad-hoc version of KD, this has a similar effect on the teacher to provide a uniform distribution for distilling knowledge. Surprisingly, they show that even a poorly-trained teacher network with much lower accuracy can enhance the student’s performance with the noisy logits over all classes produced by a teacher, where the noisy logits have a similar effect of having the uniform distribution on distilling knowledge. From K classes, we can create, $\frac{1}{K}$, the uniform distribution as a smoothing method. Therefore, we consider the LS as an effective method to improve performance, while focusing more on classes with high probabilities, which is different from other KDs.

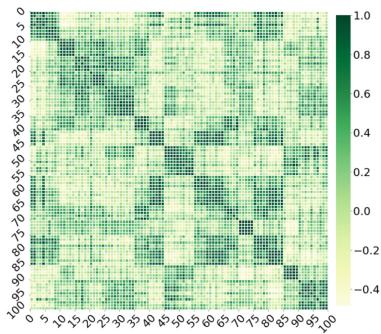


Fig. 2. Heatmap of the Pearson correlation matrix on CIFAR-100, which contains 100 classes over 20 super-classes and 5 sub-classes for each super-class.

Definition of Overall Knowledge. In this work, we define knowledge from the trained network in a more constructive way incorporating 1) inter-class

relationship prior and 2) smoothed distribution over all the classes. From the ground-truth label information \mathbf{y} , we can have the final soft target vector \mathbf{s} for the student network as follows:

$$\mathbf{s} = a_1 \cdot \mathbf{y} + a_2 \cdot \mathbf{c}(\mathbf{x}) + a_3 \cdot \frac{1}{K}, \quad (3)$$

where $\mathbf{c}(\mathbf{x})$ is the inter-class relationship prior from the given input image \mathbf{x} , and $\frac{1}{K}$ is a smoothing vector over K classes, and a_1 , a_2 , and a_3 are the weight coefficients.

3.1 RAS-KD

Inter-Class Relationship Prior Matrix. Let $\mathbf{Z}_t = [\bar{\mathbf{z}}_t^1, \dots, \bar{\mathbf{z}}_t^K]^T \in \mathbb{R}^{K \times K}$ be a matrix of the mean logits from the student network for each class during t -th epoch. From \mathbf{Z}_t , we can obtain the representative inter-class relationships, similarity score from the Pearson correlation, σ_t^{ij} between i -th and j -th class ($1 \leq i, j \leq K$). This is the same as the matrix $\sum_{\mathbf{Z}_t \mathbf{Z}_t} = [\sigma_t^1, \sigma_t^2, \dots, \sigma_t^K]^T \in \mathbb{R}^{K \times K}$, where a vector $\sigma_t^i = [\sigma_t^{i1}, \sigma_t^{i2}, \dots, \sigma_t^{iK}]$ ($-1 \leq \sigma_t^{ij} \leq 1$). For example, classes that are significantly different (i.e., weak or negative correlations) from the i -th class would have correlation values close to -1. Next, we set the lower bound to 0 from -1 so that we can treat such weak or irrelevant classes as noise. Hence, we obtain all the elements in $\sum_{\mathbf{Z}_t \mathbf{Z}_t}$ to be in $[0, 1]$, which is denoted as $\sum'_{\mathbf{Z}_t \mathbf{Z}_t} = [\sigma_t'^1, \sigma_t'^2, \dots, \sigma_t'^K]^T$. We define this matrix $\sum'_{\mathbf{Z}_t \mathbf{Z}_t}$ as an inter-class relationship prior matrix in RAS-KD.

Relation-aware Label Smoothing. The conventional KD utilizes τ to control the probability distribution to smooth the logits. However, we apply τ to emphasize the inter-class relationships more, boosting the probabilities of highly correlated classes. Therefore, unlike dividing τ in Hinton *et al.*'s KD [4], we divide $1/\tau$ to $\sum'_{\mathbf{Z}_t \mathbf{Z}_t}$, to make the range of values to be changed from $[0, 1]$ to $[0, \tau]$ so that the final similarity distribution would have a higher peak, providing more weights to similar classes. Next, we apply the softmax function to obtain the probability distribution. Therefore, our final soft target vector \mathbf{s}_t for i -th class and the relation-aware label smoothing objective function (\mathcal{L}^{RAS}) at t -th epoch are defined as follows:

$$\mathbf{s}_t = \alpha \cdot \mathbf{y} + (1 - \alpha) \cdot \phi(\sigma_{t-1}^{ii}; 1/\tau), \quad (4)$$

$$\mathcal{L}_t^{RAS} = \mathcal{H}(\mathbf{s}_t, \mathbf{p}_S) = \alpha \cdot \mathcal{H}(\mathbf{y}, \mathbf{p}_S) + (1 - \alpha) \cdot \mathcal{H}(\phi(\sigma_{t-1}^{ii}; 1/\tau), \mathbf{p}_S), \quad (5)$$

where α is a weighting hyper-parameter. For training, we initialize \mathbf{Z}_0 to the zero-matrix at the first epoch. Also, by taking the softmax to σ_{t-1}^{ii} , the soft target vector \mathbf{s}_t in Eq. 4 provides a label smoothing effect. Indeed, we demonstrate Eq. 4 has an equivalent form to Eq. 3.

Table 1. Avg. top-1 and top-5 accuracies (%) on fine-grained image classification datasets. The best top-1 and top-5 performances are highlighted in bold, and the second-best performances are underlined for each model, respectively.

ResNet18	CUB-200	Stanford Cars	FGVC-Aircraft
CE	53.17 (77.16)	72.20 (90.14)	66.46 (89.58)
LS	57.14 (78.06)	<u>76.08</u> (91.98)	74.91 (91.90)
BYOT	62.03 (83.57)	82.67 (95.85)	76.48 (92.83)
CS-KD	64.24 (85.34)	86.97 (97.34)	73.21 (92.56)
FRSKD	65.4 (85.00)	84.22 (96.38)	79.51 (94.93)
PS-KD	50.34 (75.04)	73.81 (91.46)	68.10 (89.89)
RAS-KD w/o $\mathcal{L}^{ins,aux}$	<u>65.67</u> (85.55)	85.13 (96.40)	77.34 (94.37)
RAS-KD	66.91 (86.31)	<u>86.03</u> (97.00)	<u>78.88</u> (93.85)
ResNext50	CUB-200	Stanford Cars	FGVC-Aircraft
CE	52.78 (78.84)	79.34 (94.83)	71.26 (91.15)
LS	59.89 (80.83)	84.35 (95.86)	72.82 (90.70)
BYOT	63.05 (84.12)	86.27 (96.41)	78.25 (94.08)
CS-KD	64.55 (86.14)	85.59 (93.89)	76.63 (93.87)
FRSKD	63.36 (85.05)	85.06 (96.21)	<u>80.17</u> (94.12)
PS-KD	52.62 (78.71)	79.85 (94.12)	76.87 (93.22)
RAS-KD w/o $\mathcal{L}^{ins,aux}$	<u>66.36</u> (86.43)	86.46 (72.22)	79.65 (94.57)
RAS-KD	67.07 (87.16)	86.95 (97.00)	80.71 (94.33)
ResNext101	CUB-200	Stanford Cars	FGVC-Aircraft
CE	59.01 (82.57)	83.12 (95.89)	78.34 (93.67)
LS	62.85 (82.60)	86.60 (96.49)	71.95 (90.94)
BYOT	66.10 (88.75)	87.26 (97.11)	80.32 (94.92)
CS-KD	<u>67.86</u> (87.36)	87.46 (97.86)	<u>80.94</u> (95.11)
FRSKD	—	—	—
PS-KD	56.01 (79.67)	84.12 (95.91)	77.22 (93.64)
RAS-KD w/o $\mathcal{L}^{ins,aux}$	67.38 (86.74)	87.39 (97.30)	80.01 (94.42)
RAS-KD	69.31 (88.04)	89.62 (98.23)	82.00 (94.81)
ViT-Base	CUB-200	Stanford Cars	FGVC-Aircraft
CE	80.03 (95.58)	75.09 (93.35)	70.56 (92.13)
LS	79.56 (94.96)	80.07 (95.56)	77.07 (94.33)
CS-KD	83.21 (96.57)	86.42 (97.51)	81.49 (95.77)
PS-KD	80.34 (95.63)	74.44 (93.01)	69.72 (91.62)
RAS-KD w/o $\mathcal{L}^{ins,aux}$	<u>83.75</u> (96.63)	87.58 (97.89)	81.90 (95.61)
RAS-KD	84.77 (96.78)	87.77 (97.93)	82.98 (95.49)

Instance-level Soft Target. Proposed soft target in Eq. 4 successfully regularizes the network itself, while it considers the representative similarity between the classes. In other words, data samples in the same class are supervised with the same global target during training. Meanwhile, existing works such as PS-KD [6] or CS-KD [14] show the promised performance via instance-level logit supervision. To this end, we introduce an auxiliary classifier with negligible cost, to provide the instance-level soft targets. As shown in Fig. 1, we attach a fully connected layer as an auxiliary classifier trained with the auxiliary loss (\mathcal{L}^{aux}) through distilling the knowledge of the student network:

$$\mathcal{L}^{aux} = \alpha \cdot \mathcal{H}(\mathbf{y}, \mathbf{p}'_{\mathbf{S}}) + (1 - \alpha) \cdot \mathcal{H}(\mathbf{p}_{\mathbf{S}}, \mathbf{p}'_{\mathbf{S}}), \quad (6)$$

where α is a weighting hyper-parameter with the same value as in Eq. 4 and \mathbf{p}'_S is a predictive probability of the auxiliary classifier. Moreover, we constrain the auxiliary classifier from directly imitating the original forward path by detaching the gradient from the auxiliary classifier to the feature extractor. Finally, \mathbf{p}'_S provides the instance-level loss (\mathcal{L}^{ins}) to the student network via KL divergence:

$$\mathcal{L}^{ins} = \mathcal{H}(\mathbf{p}'_S, \mathbf{p}_S). \quad (7)$$

Therefore, our framework is trained with the following objective function at t -th epoch:

$$\mathcal{L}^{RAS-KD} = \beta \cdot \mathcal{L}_t^{RAS} + (1 - \beta) \cdot \mathcal{L}^{ins} + \mathcal{L}^{aux}, \quad (8)$$

where β is a weighting hyper-parameter. While the proposed \mathcal{L}^{ins} appears to be similar to Eq. 6, \mathcal{L}^{ins} functions differently from Eq. 6. Inspired by PS-KD, which exploits the logits from the previous epoch, we introduce a lightweight auxiliary classifier. In particular, our proposed auxiliary classifier shares the feature extractor but applies a gradient stop to the feature extractor.

4 Experimental Results

We compare our method with four popular self-KD approaches: BYOT, CS-KD, TF-KD, FRSKD, PS-KD, and LS.

Fine-grained Classification. As shown in Table 1, RAS-KD achieves the best top-1 accuracy in all datasets, except for the ResNet18 on Stanford Cars dataset. Note that in the case of ViT-Base [1], due to architectural differences, we only measured the accuracy of the response-based KD methods, excluding cases that involve the utilization of intermediate feature maps such as BYOT and FRSKD. Specifically, in the FGVC-Aircraft dataset, we achieve performance improvement across all the existing self-KD approaches up to 2.46%. Note that RAS-KD requires only a few parameters and computational cost, while other methods

Table 2. Avg. top-1 accuracies (%) of ours, label smoothing (LS) and existing self-KD methods on CIFAR-100. We rerun all the results three times, and report the average of the results. We highlight the best scores in bold, and the second-best performance in underlining for each model, respectively.

Model	ResNet18	ResNet50	DenseNet121	ResNeXt29
CE	75.55	78.07	79.58	80.59
LS	78.21	79.31	79.86	81.44
BYOT	78.51	80.56	80.27	81.51
CS-KD	78.19	78.69	79.76	82.16
TF-KD	76.59	78.45	80.24	82.23
FRSKD	77.71	74.72	—	—
PS-KD	78.89	79.33	81.29	82.27
RAS-KD w/o $\mathcal{L}^{ins, aux}$	<u>79.17</u>	79.56	80.31	<u>82.58</u>
RAS-KD	79.28	<u>79.86</u>	<u>80.89</u>	82.8

need additional memory to save all predictions or store the network to re-forward images to the model. Furthermore, we visualize the t-SNE results based on the outputs of ResNet18 trained with cross entropy (CE), LS, and RAS-KD in Fig. 3. Clearly, CE shows the scattered and overlapped distributions compared to other approaches for both datasets, where such distributions make it difficult for the model to find decision boundaries that separate classes. While LS has better boundaries than CE, t-SNE results from RAS-KD show more compact clusters. This clearly shows that our RAS-KD effectively reduces the intra-class variation, while maximizing the inter-class boundaries, demonstrating that RAS-KD successfully learns more useful knowledge from soft targets from our approach.

CIFAR-100 Classification. We compare our method with three popular self-KD approaches: CS-KD, TF-KD, and PS-KD, and additional regularization method, LS. For self-KD methods, we use the same hyper-parameter settings reported in their respective research, and we use the smoothing parameter ϵ of 0.1 for LS. As shown in Table 2, our RAS-KD outperforms other approaches in most of the experimental settings. Compared to LS, RAS-KD achieves higher performance, demonstrating the effectiveness of using inter-class relationships. Even though BYOT shows the highest performance in ResNet50, it requires more resources to build a powerful ensemble model, which works as a teacher network.

Data Augmentation. Also, we exploit data augmentation methods for evaluation. As shown in Table 3, overall, RAS-KD achieves the highest accuracies in all of the results, except for the DenseNet121 with CutMix.

Effectiveness of Inter-Class Relationship Prior. To illustrate that our RAS-KD effectively learns the inter-class relationship during training, we present PCC heatmaps in Sect. 3 to visualize the correlations of ResNet18 trained with LS and RAS-KD in Fig. 4. Compared to CE, LS (Fig. 4 (b)) smooths most

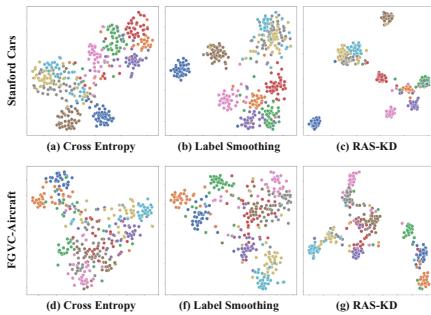


Fig. 3. t-SNE visualization results of two baselines and our RAS-KD. For clear visualization, we use the first 10 classes of Stanford Cars and FGVC-Aircraft and use the outputs from ResNet18.

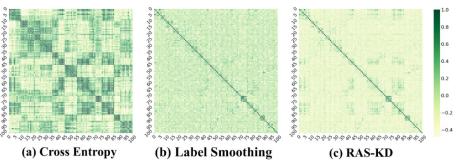


Fig. 4. Heatmaps of the Pearson correlation matrix on CIFAR-100. RAS-KD shows much lower correlation between classes in different super-classes.

Table 3. Top-1 accuracies (%). of four baseline models with CutOut and CutMix augmentation-based regularization on CIFAR-100 dataset. Bold values illustrate the best result and the highest increase.

Method	ResNet18	ResNet50	DenseNet121	ResNeXt29
CE	75.55	78.07	79.58	80.59
LS	78.21	79.31	79.86	81.44
RAS-KD	79.28	79.86	80.89	82.8
CE + CutOut	76.92 ($\uparrow 1.37$)	79.15 ($\uparrow 1.08$)	80.06 ($\uparrow 0.48$)	81.76 ($\uparrow 1.17$)
LS + CutOut	78.72 ($\uparrow 0.51$)	79.90 ($\uparrow 0.59$)	81.18 ($\uparrow 0.55$)	81.62 ($\uparrow 0.18$)
RAS-KD + CutOut	79.51 ($\uparrow 0.23$)	80.49 ($\uparrow 0.63$)	81.83 ($\uparrow 0.94$)	83.22 ($\uparrow 0.42$)
CE + CutMix	78.68 ($\uparrow 3.13$)	80.54 ($\uparrow 2.47$)	81.68 ($\uparrow 2.1$)	82.94 ($\uparrow 2.35$)
LS + CutMix	79.70 ($\uparrow 1.49$)	81.04 ($\uparrow 1.73$)	82.49 ($\uparrow 2.63$)	83.67 ($\uparrow 2.23$)
RAS-KD + CutMix	80.51 ($\uparrow 1.23$)	81.34 ($\uparrow 1.48$)	82.32 ($\uparrow 1.43$)	84.19 ($\uparrow 1.39$)

Table 4. Top-1 accuracies (%) of label smoothing (LS) and self-KD methods on ImageNet. We highlight the best results in bold and underline the second-best results.

ImageNet	CE	LS	BYOT	TF-KD	CS-KD	FRSKD	PS-KD	RAS-KD
ResNet50	75.51	<u>75.88</u>	75.24	75.12	75.62	–	75.73	76.16
ResNet101	77.66	77.7	–	<u>77.98</u>	77.35	–	77.78	78.29
ResNeXt101	79.17	79.78	–	80.26	79.36	–	79.76	80.02

Table 5. Performance of mAP (%) scores with 0.5 of IoU threshold. The best result is highlighted in bold.

Backbone	CE	LS	CS-KD	PS-KD	RAS-KD
ResNet50	78.5	78.5	78.3	78.5	79.4
ResNet101	79.8	80.0	80.2	80.2	81.1

of the correlation values. On the other hand, we can observe that our RAS-KD (Fig. 4 (c)) not only significantly decreases the similarity score between classes in different-super classes (non-diagonal region), but also increases the correlations within the same super class from the distinct squares along the diagonal.

ImageNet Classification. As shown in Table 4, RAS-KD achieves the SOTA performance on ImageNet [7] dataset, compared to other self-KD methods. In ResNet101, RAS-KD further improves accuracy up to 0.63% compared to CS-KD and PS-KD. In addition, memory-based methods such as TF-KD and PS-KD, which have to store predictive outputs of the teacher network or previous epoch, need more GPU memory to load the fixed models and require re-forwarding data to the model. On the other hand, the training algorithm with RAS-KD requires the same cost as CE, as shown in the ablation section.

Object Detection (Transfer Learning Task). Transfer learning provides a practical and efficient approach to improving performance by leveraging a pre-trained model. Our proposed RAS-KD can serve as a great backbone network for various computer vision tasks. To evaluate the performance on transfer learning, we replace the feature extractor pre-trained with ImageNet using vanilla CE to other training methods and then fine-tune the entire model with PascalVOC [2] dataset. In Table 5, we present the mean Average Precision (mAP) for five pre-trained backbone networks trained by existing KD methods. Compared to existing self-KD methods, our RAS-KD improves the mAP score by 0.9%.

Table 6. Ablation results on τ and α . We gradually increase τ from Config. A to D, while adjusting α from E to G with no instance-level supervision (i.e., $\beta = 1.0$). We color the best results from the two settings in red and blue, respectively.

Configuration	ResNet18	ResNet50	ResNeXt29	DenseNet121
A. $\alpha = 0.8, \tau = 1$	78.82	78.3	81.87	80.30
B. $\alpha = 0.8, \tau = 1.5$	79.17	79.56	82.58	80.31
C. $\alpha = 0.8, \tau = 3$	78.75	79.50	82.03	80.21
D. $\alpha = 0.8, \tau = 5$	78.38	78.31	82.31	79.76
E. $\alpha = 0.7, \tau = 1$	78.48	78.39	81.82	79.86
F. $\alpha = 0.8, \tau = 1$	78.82	78.30	81.87	80.30
G. $\alpha = 0.9, \tau = 1$	78.53	78.78	81.84	80.00

5 Ablation Study

Table 7. Ablation results on β . We gradually increase β values from 0.6 to 0.9 with fixed α and τ to 0.8 and 1.5, respectively. We can observe that beyond a certain threshold value of β (i.e., 0.7), no significant difference in performance is discernible.

$\alpha = 0.8, \tau = 1.5$	Model	$\beta = 0.6$	$\beta = 0.7$	$\beta = 0.8$	$\beta = 0.9$
CIFAR-100	ResNet18	78.90	79.17	79.28	79.27
	ResNet50	79.41	79.45	79.86	78.63
	ResNext29	82.47	82.75	82.80	82.70
	DenseNet121	80.51	80.79	80.89	81.00
CUB-200	ResNet18	65.53	66.67	66.91	66.15
	ResNext50	63.34	67.04	67.07	67.12
Stanford Cars	ResNet18	85.73	85.33	86.03	85.67
	ResNext50	85.44	86.98	86.95	86.96
FGVC-Aircraft	ResNet18	78.12	79.20	78.88	78.61
	ResNext50	78.73	79.98	80.71	78.49

Our RAS-KD employs three main components in producing the soft targets: 1) inter-class relationships prior, 2) label smoothing, as shown in Eq. 3, and 3) logits from the auxiliary classifier. Since the auxiliary classifier is supervised by the student network from the first two terms, we only conduct ablation

experiments to explore how the first and second component affects performance improvement, while setting $\beta = 1$ instead of 0.8. We first adjust τ to 1, 1.5, 3, and 5. If τ increases, the soft targets provide more weight to the classes with closer similarity. We report the results according to different τ from Config. A to D in Table 6. All baseline models consistently show the best performance with $\tau = 1.5$, but we observe an accuracy drop as τ increases from 1.5. On the other hand, when $\alpha = 0.7$, we can observe the performance degradation compared to $\alpha = 0.8$ or 0.9. This indicates that too less weighting on hard-labels (more weighting on the soft targets) can hinder the optimization. With the best configuration for \mathcal{L}^{RAS} (Config. B), we extend \mathcal{L}^{RAS} to \mathcal{L}^{RAS-KD} ($\beta = 0.8$) by including the additional instance-level supervision. As shown in Table 2, the proposed auxiliary classifier achieves performance improvement, which demonstrates that global and instance-level supervision regularizes the student independently and effectively. Moreover, we have observed a significant improvement up to 2.0% in top-1 accuracy in fine-grained datasets through the instance-level supervision, as shown in Table 1. Lastly, we conduct additional ablation experiments for the sensitivity of the β . As reported in Table 7, we can observe that across the four datasets, there is a noticeable performance degradation at a common value of $\beta = 0.6$, while $\beta > 0.7$, does not result in significant changes. If β is too small, the weight of hard-label decreases, leading to unstable training of the main classifier.

Training Efficiency of RAS-KD.

We compare our RAS-KD with the existing methods to time and space efficiency. For both experiments, we used ResNet18 architecture trained on the CUB-200 dataset. As shown in Fig. 5(a), existing self-KD approaches spend more time on training the model. In particular, CS-KD requires more time than other approaches due to the additional forward process. PS-KD and BYOT also require extra time, because they utilize additional computation or use additional branch networks. On the other hand, our RAS-KD has similar training time performance to CE and LS because it only requires little computation for calculating the class representative vectors and gradient of one fully connected layer for the auxiliary classifier. We also show the space efficiency of RAS-KD by utilizing the VRAM occupancy, as shown in Fig. 5(b). Especially, PS-KD consumes significant time and space due to the previous model being stored separately. However, RAS-KD has an advantage in terms of memory footprints, because of requiring minimal memory to compute the class representative vectors and one linear layer for the auxiliary classifier.

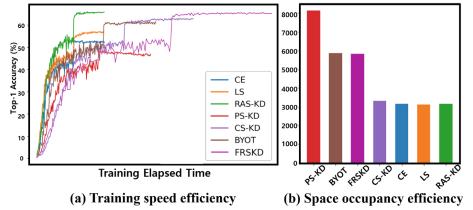


Fig. 5. Time and space efficiency of RAS-KD vs. other methods. RAS-KD costs much less training time and VRAM compared to other methods.

6 Conclusion

We proposed the novel Relation-aware Label Smoothing for Self-Knowledge distillation (RAS-KD). RAS-KD leverages the inter-class relationship matrix obtained by class representative vectors from the previous epoch of the model. For global-level supervision, our soft targets can be categorized into: 1) one-hot encoded hard-label, 2) inter-class relationship prior, and 3) smoothed distribution over all classes. With the global-level approach, we can significantly reduce the temporal and spatial training cost. Moreover, we provide the diverse logits via instance-level supervision from an auxiliary classifier with negligible parameters. Through extensive experiments, we show that RAS-KD outperforms existing self-KD methods on several vision tasks.

Acknowledgements. This work was partly supported by Institute for Information & communication Technology Planning & evaluation (IITP) grants funded by the Korean government MSIT: (No. 2022-0-01199, Graduate School of Convergence Security at Sungkyunkwan University), (No. 2022-0-01045, Self-directed Multi-Modal Intelligence for solving unknown, open domain problems), (No. 2022-0-00688, AI Platform to Fully Adapt and Reflect Privacy-Policy Changes), (No. 2021-0-02068, Artificial Intelligence Innovation Hub), (No. 2019-0-00421, AI Graduate School Support Program at Sungkyunkwan University), and (No. RS-2023-00230337, Advanced and Proactive AI Platform Research and Development Against Malicious Deepfakes). Lastly, this work was supported by Korea Internet & Security Agency (KISA) grant funded by the Korea government (PIPC) (No.RS-2023-00231200, Development of personal video information privacy protection technology capable of AI learning in an autonomous driving environment).

References

1. Dosovitskiy, A., et al.: An image is worth 16×16 words: transformers for image recognition at scale. arXiv preprint [arXiv:2010.11929](https://arxiv.org/abs/2010.11929) (2020)
2. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. Int. J. Comput. Vision **88**(2), 303–338 (2010)
3. Freedman, D., Pisani, R., Purves, R.: Statistics (international student edition). Pisani, R. Purves, 4th edn. WW Norton & Company, New York (2007)
4. 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)
5. Ji, M., Shin, S., Hwang, S., Park, G., Moon, I.C.: Refine myself by teaching myself: feature refinement via self-knowledge distillation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10664–10673 (2021)
6. Kim, K., Ji, B., Yoon, D., Hwang, S.: Self-knowledge distillation with progressive refinement of targets. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6567–6576 (2021)
7. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Adv. Neural. Inf. Process. Syst. **25**, 1097–1105 (2012)
8. Park, W., Kim, D., Lu, Y., Cho, M.: Relational knowledge distillation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3967–3976 (2019)

9. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)
10. Tan, M., Pang, R., Le, Q.V.: Efficientdet: scalable and efficient object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10781–10790 (2020)
11. Vaswani, A., et al.: Attention is all you need. *Adv. Neural Inf. Process. Syst.* **30** (2017)
12. Yang, J., Martinez, B., Bulat, A., Tzimiropoulos, G., et al.: Knowledge distillation via softmax regression representation learning. In: International Conference on Learning Representations (ICLR) (2021)
13. Yuan, L., Tay, F.E., Li, G., Wang, T., Feng, J.: Revisiting knowledge distillation via label smoothing regularization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3903–3911 (2020)
14. Yun, S., Park, J., Lee, K., Shin, J.: Regularizing class-wise predictions via self-knowledge distillation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13876–13885 (2020)
15. Zhang, L., Song, J., Gao, A., Chen, J., Bao, C., Ma, K.: Be your own teacher: improve the performance of convolutional neural networks via self distillation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 3713–3722 (2019)



Bi-CryptoNets: Leveraging Different-Level Privacy for Encrypted Inference

Man-Jie Yuan^{1,2}, Zheng Zou^{1,2}, and Wei Gao^{1,2(✉)}

¹ National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
`{yuanmj, zouz}@lamda.nju.edu.cn`

² School of Artificial Intelligence, Nanjing University, Nanjing, China
`gaow@lamda.nju.edu.cn`

Abstract. Privacy-preserving neural networks have attracted increasing attention in recent years, and various algorithms have been developed to keep the balance between accuracy, computational complexity and information security from the cryptographic view. This work takes a different view from the input data and structure of neural networks. We decompose the input data (e.g., some images) into sensitive and insensitive segments according to importance and privacy. The sensitive segment includes some important and private information such as human faces and we take strong homomorphic encryption to keep security, whereas the insensitive one contains some background and we add perturbations. We propose the bi-CryptoNets, i.e., plaintext and ciphertext branches, to deal with two segments, respectively, and ciphertext branch could utilize the information from plaintext branch by unidirectional connections. We adopt knowledge distillation for our bi-CryptoNets by transferring representations from a well-trained teacher neural network. Empirical studies show the effectiveness and decrease of inference latency for our bi-CryptoNets.

Keywords: Neural network · Encryption · Privacy-preserving inference

1 Introduction

Recent years have witnessed increasing attention on privacy-preserving neural networks [4, 14, 25], which can be viewed as a promising security solution to the emerging Machine Learning as a Service (MLaaS) [13, 15, 30]. Specifically, some clients can upload their encrypted data to the powerful cloud infrastructures, and then obtain machine learning inference services; the cloud server performs inference without seeing clients' sensitive raw data by cryptographic primitives, and preserve data privacy.

Privacy-preserving neural networks are accompanied with heavy computational costs because of homomorphic encryption [2, 12], and various algorithms and techniques have been developed to keep the balance between accuracy, information security and computation complexity. For example, Brutzkus et al. proposed the LoLa network for fast inference over a single image based on well-designed packing methods [4], and Lou and Jiang introduced the circuit-based network SHE with better accuracies [27], implemented with the TFHE scheme [7]. Dathathri et al. presented the compiler CHET to optimize data-flow graph of HE computations for privacy-preserving neural networks [10]. Yin et al. presented a comprehensive survey on privacy-preserving networks [38].

Previous studies mostly encrypted the entire input data and treated indiscriminately. In some applications, however, input data may consist of sensitive and insensitive segments according to different importance and privacy. As shown in Fig. 1, a fighter is more sensitive than background such as sky and mountains in a military picture, and a human face is more private than landscape in a photo.

This work presents new privacy-preserving neural network from the view of input data and network structure, and the main contributions can be summarized as follows:

- We decompose input data (e.g., images) into sensitive and insensitive segments according to importance and privacy. We adopt strong homomorphic encryption to keep the security of sensitive segment, yet mingle some perturbations [11,37] to insensitive segment. This could reduce computational overhead and perform private inference of low latency, without unnecessary encryption on insensitive segment.
- We propose the bi-CryptoNets, i.e., ciphertext and plaintext branches, to deal with sensitive and insensitive segments, respectively. The ciphertext branch could use information from plaintext branch by unidirectional connections, but the converse direction does not hold because of the spread of ciphertexts. We integrate features for the final predictions, from the outputs of both branches.
- We present the feature-based knowledge distillation to improve the performance of our bi-CryptoNets, from a teacher of convolutional neural network trained on the entire data without decomposition.
- We present empirical studies to validate the effectiveness and decrease of inference latency for our bi-CryptoNets. We could improve inference accuracy by 0.2%–2.1%, and reduce inference latency by $1.15\times$ – $3.43\times$ on a single image, and decrease the amortized latency by $4.59\times$ – $13.7\times$ on a batch of images.

The rest of this work is organized as follows: Sect. 2 introduces relevant work. Section 3 presents our bi-CryptoNets. Section 4 proposes the feature-based knowledge distillation. Section 5 conducts experiments, and Sect. 6 concludes with future work.

2 Relevant Work

Homomorphic Encryption. Homomorphic Encryption (HE) is a cryptosystem that allows operations on encrypted data without requiring access to a secret key [12]. In addition to encryption function E and decryption function D , HE scheme provides two operators \oplus and \otimes such that, for every pair of plaintexts x_1 and x_2 ,

$$D(E(x_1) \oplus E(x_2)) = x_1 + x_2, \quad D(E(x_1) \otimes E(x_2)) = x_1 \times x_2,$$

where $+$ and \times are the standard addition and multiplication, respectively. Hence, we could directly perform private addition, multiplication and polynomial functions over encrypted data, by using \oplus and \otimes operators without knowing true values in plaintexts.



Fig. 1. An illustration for input data (e.g., some images (Download from ILSVRC dataset [37].)), which consists of two segments with different importance and privacy.

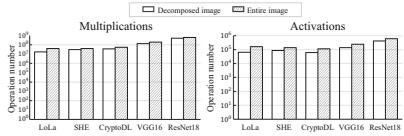


Fig. 2. Counts of HE multiplications and activations for decomposed and entire image.

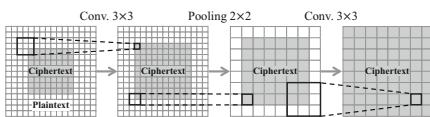


Fig. 3. The fast spread of ciphertext via CNN layers.

The commonly-used HE cryptosystems include CKKS [6], BGV [3] and TFHE [7] for privacy-preserving machine learning. The BGV and TFHE schemes support integer and bits computations, while the CKKS scheme supports fixed-point computations. For CKKS, a ciphertext is an element in the ring \mathcal{R}_p^2 , where $\mathcal{R}_p = \mathbb{Z}_p[x]/(x^N + 1)$ is the residue ring of polynomial ring, with polynomial degree N . In addition to HE, a recent study introduced a new data encryption scheme by incorporating crucial ingredients of learning algorithm, specifically the Gini impurity in random forests [36].

Privacy-Preserving Neural Networks. Much attention has been paid on the privacy-preserving neural networks in recent years. For example, Gilad-Bachrach et al. proposed the first CryptoNets to show the feasibility of private CNN inference by HE scheme [14]. Brutzkus et al. proposed the LoLa to optimize the implementation of matrix-vector multiplication by different packing methods, and achieve fast inference for single image [4]. In those studies, the ReLU activation has been replaced by squared activation because the used HE schemes merely support polynomial functions. Lou and Jiang presented a circuit-based quantized neural network SHE implemented by TFHE scheme, which could perform ReLU and max pooling by comparison circuits to obtain good accuracies [27]. Recent studies utilized the CKKS scheme to avoid quantization of network parameters for better accuracies [9, 25].

Knowledge Distillation. Knowledge distillation focuses on transferring knowledge from a pretrained teacher model to a student model [22, 26]. Traditional methods aimed to match the output distributions of two models by minimizing the Kullback-Leibler divergence loss [22]. Subsequently, a variety of innovative distillation techniques have been developed. Romero et al. proposed Fitnets [31], which matches the feature activations, while Zagoruyko and Komodakis suggested matching attention maps between two models [40]. Gou et al. gave an extensive review on knowledge distillation [17].

Threat Model. Our threat model follows previous studies on private inference [1, 8, 28]. A honest but curious cloud-based machine learning service hosts a network, which is trained on plaintext data (such as public datasets) and hence the network weights are not encrypted [14]. To ensure the confidentiality of client’s data, the client could encrypt the sensitive segment of data by using HE scheme and send data to the cloud server for performing private inference service without decrypting data or accessing the client’s private key. Only the client could decrypt the inference result by using the private key.

3 Our Bi-CryptoNets

In this section, we will present new privacy-preserving neural network according to different-level privacy of input data. Our motivation is to decompose input data into *sensitive* and *insensitive* segments according to their privacy. The sensitive segment includes some important and private information such as human faces in an image, whereas the insensitive one contains some background information, which is not so private yet beneficial to learning algorithm.

Based on such decomposition, we could take the strongest homomorphic encryption to keep data security for sensitive segment, while mingle with some perturbations [11, 37] to the data of insensitive segment. This is quite different from previous private inference [14, 25], where homomorphic encryption is applied to the entire input data.

Notice that we can not directly use some previous neural networks to tackle such decomposition with much smaller computational cost, as shown in Fig. 2. We compare with three private networks LoLa [4], SHE [27], CryptoDL [20], and two conventional networks VGG16 and ResNet18. Prior networks can not reduce HE operations because ciphertexts could quickly spread over the entire image via several convolution and pooling operations, as shown in Fig. 3.

Our idea is quite simple and intuitive for the decomposition of input data. We construct a bi-branch neural network to deal with the sensitive and insensitive segments, which are called *ciphertext* and *plaintext* branch, respectively. The ciphertext branch can make use of features from plaintext branch by unidirectional connections, while the converse direction does not hold because of the quick spread of ciphertexts. We take the feature integration for final predictions from the outputs of two branches of network.

Figure 4 presents an overview for our new privacy-preserving neural network, and it is short for *bi-CryptoNets*. We will go to the details of bi-CryptoNets in the following.

3.1 The Bi-branch of Neural Network

We construct ciphertext and plaintext branches to deal with the sensitive and insensitive segments of an input instance, respectively. The plaintext branch deals with the entire input instance, where the insensitive segment is mingled with some perturbations [11, 37], while the sensitive segment is simply filled with zero. The ciphertext branch tackles the sensitive segment with the homomorphic encryption due to its privacy.

The HE operations are restricted in the ciphertext branch to avoid the ciphertext spread on plaintext branch. This could keep the proportion of HE operations in a stable level, rather than previous close approximation to 1 as depth increases [14]. Hence, our bi-branch structure could decease HE operations. Moreover, plaintext branch can be computed with full precision for performing inference, rather than quantized homomorphic ciphertext in prior privacy-preserving neural networks [20].

3.2 The Unidirectional Connections

It is well-known that one great success of deep learning lies in the strong features or representations by plentiful co-adaptations of neural network [16, 21, 34]. Our bi-branch neural network reduces some co-adaptations of features obviously, to restrict

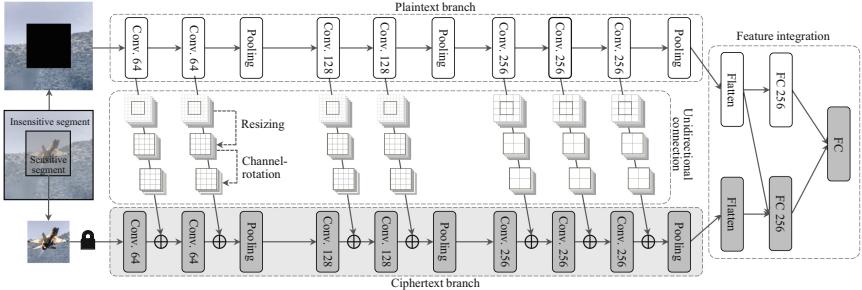


Fig. 4. The overview of our proposed bi-CryptoNets, where grey layers are computed with encrypted inputs, and other layers are computed with plaintext inputs.

the spread of ciphertexts. Hence, it is necessary to strengthen features' representations by exploiting some correlations in bi-branches.

Notice that the ciphertext branch can make use of features from plaintext branch, while the converse direction does not hold because of the spread of ciphertexts. In this work, we consider the simple addition to utilize features from plaintext branch as convolutional neural network [19, 29, 35]. Concatenation is another effective way to utilize features [23, 24], while it yields more HE operations and computational costs.

One reason for addition is of small computational costs, since it yields relatively few homomorphic additions between plaintext and ciphertext, as shown in Table 1. We compare with addition (Add_{PC}) and multiplication (Mul_{PC}) between a plaintext and a ciphertext, as well as addition (Add_{CC}) and multiplication (Mul_{CC}) between two ciphertexts. We compare with four operations under three popular HE schemes: BGV [3], CKKS [6] and TFHE [7], and it is evident that the Add_{PC} has the smallest latency.

For unidirectional connections, we first resize the feature map of plaintext branch to fit the addition with ciphertext branch, and then introduce a channel-rotation for ciphertext channel to extract some information from multiple plaintext channels. Specifically, suppose there are L layers in ciphertext and plaintext branches, and denote by $\mathcal{F}_{c,l}$ and $\mathcal{F}_{p,l}$ their respective l -th layer. For every $l \in [L]$, we consider the following two steps:

i) Resizing feature map of plaintext branch

We begin with the resizing function

$$\mathbf{y}_{p,l} = \text{Resize}_l(\mathbf{x}_{p,l}) : \mathbb{R}^{h_{p,l} \times w_{p,l} \times \text{ch}_{p,l}} \rightarrow \mathbb{R}^{h_{c,l} \times w_{c,l} \times \text{ch}_{p,l}}$$

where $\mathbf{x}_{p,l} \in \mathbb{R}^{h_{p,l} \times w_{p,l} \times \text{ch}_{p,l}}$ denotes the output feature maps from the l -th plaintext layer $\mathcal{F}_{p,l}$ with $h_{p,l}$ rows, $w_{p,l}$ columns and $\text{ch}_{p,l}$ channels, and $h_{c,l}$ and $w_{c,l}$ denote the number of row and column in ciphertext branch.

Table 1. The comparisons of latency (ms) for three schemes and four operations.

HE Scheme	Add_{PC}	Add_{CC}	Mul_{PC}	Mul_{CC}
BGV	0.049	0.077	2.055	3.379
CKKS	0.039	0.077	0.173	0.390
TFHE	56.03	256.8	1018	1585

For resizing function, a simple yet effective choice is the cropping [32], which maintains the center features of size $h_{c,l} \times w_{c,l} \times ch_{p,l}$ because those features can capture more important information around the sensitive segment. We can also select pooling or convolution operations for resizing function [19].

ii) Channel-rotation for ciphertext channel

We now introduce the channel-rotation function as follows:

$$\mathbf{z}_{p,l} = CRot_l(\mathbf{y}_{p,l}) = [CRot_l^1(\mathbf{y}_{p,l}), \dots, CRot_l^i(\mathbf{y}_{p,l}), \dots, CRot_l^{ch_{c,l}}(\mathbf{y}_{p,l})]$$

where $ch_{c,l}$ is number of channels in ciphertext branch, and $CRot_l^i(\mathbf{y}_{p,l})$ denotes the i -th channel of $CRot_l(\mathbf{x})$, that is,

$$CRot_l^i(\mathbf{y}_{p,l}) = \sum_{j=1}^{ch_{p,l}} \mathbf{W}_{CRot,l}^{i,j} \cdot \mathbf{y}_{p,l}^j \quad \text{for } i \in [ch_{c,l}] .$$

Here, $\mathbf{y}_{p,l}^j \in \mathbb{R}^{h_{c,l} \times w_{c,l}}$ denotes the j -th channel of $\mathbf{y}_{p,l}$ for $j \in [ch_{p,l}]$, and $\mathbf{W}_{CRot,l} = (\mathbf{W}_{CRot,l}^{i,j})_{ch_{c,l} \times ch_{p,l}}$ is the weight matrix for channel-rotation in the l -th layer.

The channel-rotation is helpful for ciphertext channels to extract useful information automatically from plaintext channels. This is because channel-rotation could compress, rotate and scale different channels of plaintext feature maps, without the requirements of channel-wise alignments and identical numbers and magnitudes of channels. Moreover, each channel, in connection with ciphertext branch, can be viewed as a linear combination of multiple plaintext channels, rather than one plaintext channel. Therefore, the ciphertext branch can get better information from plaintext branch and achieve better performance with the help of channel-rotation.

After resizing and channel-rotation, the ciphertext branch can make use of features from plaintext branch by addition. The unidirectional connections can be written as

$$\mathbf{x}_{c,l+1} = \mathcal{F}_{c,l+1}(\mathbf{x}_{c,l} + \mathbf{z}_{p,l}) = \mathcal{F}_{c,l+1}(\mathbf{x}_{c,l} + CRot_l(Resize_l(\mathbf{x}_{p,l}))) ,$$

where $\mathbf{x}_{c,l} \in \mathbb{R}^{h_{c,l} \times w_{c,l} \times ch_{c,l}}$ denotes the output feature maps of $\mathcal{F}_{c,l}$ with $h_{c,l}$ rows, $w_{c,l}$ columns and $ch_{c,l}$ channels.

Finally, it is feasible to improve computational efficiency by implementing plaintext and ciphertext branches in parallel, because computing the plaintext branch and unidirectional connections is much faster than that of ciphertext branch.

3.3 The Feature Integration

We now design a two-layer neural network to integrate the outputs from ciphertext and plaintext branches. In the first layer, we split all neurons into two halves, one half for ciphertext yet the other for plaintext. The plaintext neurons are only connected with the outputs from plaintext branch, whereas the ciphertext neurons have full connections. We take full connections in the second layer.

Specifically, there are n_1 (even) and n_2 neurons in the first and second layer, respectively. Let $\mathbf{x}_c \in \mathbb{R}^{n_c}$ and $\mathbf{x}_p \in \mathbb{R}^{n_p}$ denote the flattened outputs from ciphertext

and plaintext branches with n_c and n_p features, respectively. Then, the outputs of ciphertext and plaintext neurons in the first layer can be given by, respectively,

$$\mathbf{x}_p^{(1)} = \sigma(\mathbf{W}'_{p,1} \mathbf{x}_p + \mathbf{b}'_1), \quad \mathbf{x}_c^{(1)} = \sigma(\mathbf{W}_{c,1} \mathbf{x}_c + \mathbf{W}_{p,1} \mathbf{x}_p + \mathbf{b}_1),$$

where $\sigma(\cdot)$ is an activation function, $\mathbf{b}_1, \mathbf{b}'_1 \in \mathbb{R}^{n_1/2}$ are bias vectors, $\mathbf{W}_{c,1} \in \mathbb{R}^{n_c \times n_1/2}$ and $\mathbf{W}_{p,1} \in \mathbb{R}^{n_p \times n_1/2}$ are the weight for ciphertext neurons, yet $\mathbf{W}'_{p,1} \in \mathbb{R}^{n_p \times n_1/2}$ is the weight for plaintext neurons. The final output in the second layer can be given by

$$\mathbf{x}_{\text{out}} = \sigma(\mathbf{W}_{c,2} \mathbf{x}_c^{(1)} + \mathbf{W}_{p,2} \mathbf{x}_p^{(1)} + \mathbf{b}_2),$$

where $\mathbf{b}_2 \in \mathbb{R}^{n_2}$ is a bias vector, and $\mathbf{W}_{c,2} \in \mathbb{R}^{n_1/2 \times n_2}$ and $\mathbf{W}_{p,2} \in \mathbb{R}^{n_1/2 \times n_2}$ are the weight matrices for ciphertext and plaintext neurons, respectively. Here, we consider two-layer neural network for simplicity, and similar constructions could be made for deeper neural networks based on the splitting of plaintext and ciphertext neurons.

4 Knowledge Distillation for Bi-CryptoNets

Knowledge distillation has been an effective way to improve learning performance by distilling knowledge from a teacher network [17, 22]. This section develops a feature-based knowledge distillation for bi-CryptoNets, as shown in Fig. 5. The basic idea is to supplement the representations of two branches of bi-CryptoNets by imitating a teacher network.

For the teacher network, we learn a classical convolutional neural network from training data over the entire images without decomposition. Hence, the teacher network has plentiful connections between different neurons, and strengthens the intrinsic correlations for better performance.

We learn the representations of our two branches from the corresponding intermediate representations of teacher network, and also learn the final outputs of our bi-CryptoNets from that of teacher network. This is partially motivated from previous knowledge distillations on internal representations [31].

Specifically, we first pad the output of ciphertext branch with 0 to match the size of plaintext output and add them together, as in convolutional neural network literature [16, 18]. We then train two branches of bi-CryptoNets to learn teacher's intermediate representations by minimizing the following loss function:

$$\mathcal{L}_{\text{IR}}(\mathbf{W}_T^h, \mathbf{W}_c, \mathbf{W}_p) = \|\mathcal{F}_T^h(\mathbf{x}, \mathbf{W}_T^h) - (\text{Pad}(\mathcal{F}_c(\mathbf{x}, \mathbf{W}_c)) + \mathcal{F}_p(\mathbf{x}, \mathbf{W}_p))\|_2^2, \quad (1)$$

where $\mathcal{F}_T^h(\cdot; \mathbf{W}_T^h)$ is the first h layers of teacher network of parameter \mathbf{W}_T^h , and $\mathcal{F}_c(\cdot; \mathbf{W}_c)$ and $\mathcal{F}_p(\cdot; \mathbf{W}_p)$ denote the ciphertext and plaintext branches of parameter \mathbf{W}_c and \mathbf{W}_p , respectively, and $\text{Pad}(\cdot)$ is the zero-padding function.

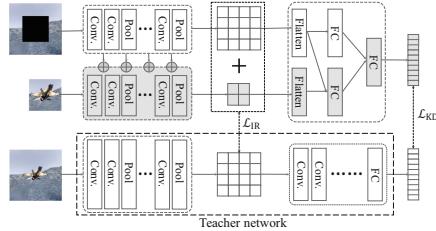


Fig. 5. The overview of our feature-based knowledge distillation.

Here, we denote by h the corresponding h -th layer in the teacher network that has the same output size as plaintext branch's output. Such loss can help the sum of outputs from two branches to approximate the teacher's intermediate output. In this manner, two branches could obtain stronger representations, and this makes it much easier for further learning in the following layers of our bi-CryptoNets.

After training two branches, we then perform knowledge distillation to the whole network. The student network is trained such that its output is similar to that of the teacher and to the true labels. Given bi-CryptoNets' output \mathbf{a}_S and teacher's output \mathbf{a}_T , we first get softened output as

$$\mathbf{p}_T^\tau = \text{softmax}(\mathbf{a}_T/\tau) \quad \text{and} \quad \mathbf{p}_S^\tau = \text{softmax}(\mathbf{a}_S/\tau),$$

where $\tau > 1$ is a relaxation parameter. We try to optimize the following loss function:

$$\mathcal{L}_{\text{KD}}(\mathbf{W}_T, \mathbf{W}_{\text{BCN}}) = \mathcal{H}(\mathbf{y}_{\text{True}}, \mathbf{p}_s) + \lambda \mathcal{H}(\mathbf{p}_T^\tau, \mathbf{p}_s^\tau), \quad (2)$$

where \mathbf{y}_{True} is the true label, $\mathbf{p}_s = \text{softmax}(\mathbf{a}_S)$, \mathcal{H} refers to the cross-entropy, \mathbf{W}_T and \mathbf{W}_{BCN} are the parameters of teacher network and our bi-CryptoNets, respectively, and $\lambda \in [0, 1]$ is a hyperparameter balancing two cross-entropies. With our proposed method, two branches in our bi-CryptoNets can learn better representations, enhancing overall performance. More details can be found in [39].

5 Experiments

This section presents empirical studies on datasets¹ MNIST, CIFAR-10 and CIFAR-100, which have been well-studied in previous private inference studies [4, 25, 27]. We develop different backbones for our bi-CryptoNets according to different datasets. We adopt the 3-layer CNN [4, 14] as backbone for MNIST, and we take 11-layer CNN [5], VGG-16 [33] and ResNet-18 [19] for CIFAR-10 and CIFAR-100. Figure 4 presents our bi-CryptoNets with VGG-16 backbone, and more details are shown in [39].

We take the CKKS scheme for sensitive segment with polynomial degree $N = 2^{14}$, and mingle with Gaussian noise for insensitive segment. We further improve packing method for our bi-CryptoNets to perform inference for multiple images simultaneously.

For simplicity, we focus on the regular images, where the area of sensitive segment is restricted to a quarter in the center of every image, and we could take some techniques of zooming, stretching and rotating to adjust those irregular images.

Ablation Studies

We conduct ablation studies to verify the effectiveness of feature-based knowledge distillation and the structures of our bi-CryptoNets. Table 2 presents the details of experimental results of inference accuracies on three datasets.

¹ Downloaded from yann.lecun.com/exdb/mnist and www.cs.toronto.edu/~kriz/cifar.

Table 2. The accuracies (%) on MNIST, CIFAR-10 and CIFAR-100 datasets (KD refers to the conventional knowledge distillation; FKD refers to our feature-based knowledge distillation).

Models	Knowledge Distillation	MNIST	CIFAR-10			CIFAR-100	
		CNN-3	CNN-11	VGG-16	ResNet-18	VGG-16	ResNet-18
Backbone network (w/o decomposition)	w/o KD	99.21	90.99	93.42	94.30	72.23	74.00
bi-CryptoNets	w/o KD	98.60	81.91	90.36	92.04	64.85	69.46
bi-CryptoNets	KD	98.61	80.03	88.91	92.43	64.96	65.67
bi-CryptoNets (w/o uni. connections)	FKD	98.89	84.69	91.78	91.73	70.61	70.29
bi-CryptoNets (w/o channel-rotations)	FKD	98.90	90.09	92.14	91.86	71.43	71.40
bi-CryptoNets	FKD	99.15	90.30	93.27	93.91	72.35	73.33

We first exploit the influence of feature-based knowledge distillation in our bi-CryptoNets. We consider two different methods: bi-CryptoNets without knowledge distillation and bi-CryptoNets with conventional knowledge distillation. As can be seen from Table 2, it is observable that our feature-based knowledge distillation could effectively improve the inference accuracy by 0.54%–8.39%, in comparison to bi-CryptoNets without knowledge distillation; it also achieves better inference accuracy by 1.45%–10.27% than that of conventional knowledge distillation, which fails to stably improve accuracy with our bi-branch structure.

We then study the influence of unidirectional connections in bi-CryptoNets, and consider two variants: our bi-CryptoNets without unidirectional connections and bi-CryptoNets with resizing yet without channel-rotations. As can be seen from Table 2, the unidirectional connections are helpful for ciphertext branch to extract useful information from plaintext branch. This is because the unidirectional connections with only resizing can enhance inference accuracy by 0.13%–5.40%, and it could further improve accuracy by 0.26%–5.61% with both resizing and channel-rotations.

We finally compare the inference accuracies of our bi-CryptoNets with the backbone network, which is trained and tested on the entire input images without decomposition. From Table 2, it is observable that our bi-CryptoNets with feature-based knowledge distillation achieves comparable or even better inference accuracies than the corresponding backbone networks without data decomposition; therefore, our proposed bi-CryptoNets could effectively compensate for the information loss of plaintext and ciphertext branches under the help of feature-based knowledge distillation.

Experimental Comparisons

We compare our bi-CryptoNets with the state-of-the-art schemes on privacy-preserving neural networks [4, 8, 9, 20, 25, 27]. We also implement the structure of backbone network with square activation for fair comparisons. We take the batch size 20 and 32 for MNIST and CIFAR-10, respectively. We employ commonly-used criteria in experiments, i.e., inference accuracy, inference latency and the number of homomorphic

Table 3. The experimental comparisons on MNIST.

Scheme	HEOPs	Addcc	Mulpc	Actc	Latency (s)	Amortized Latency (s)	Acc (%)
FCryptoNets [8]	63K	38K	24K	945	39.1	2.0	98.71
CryptoDL [20]	4.7M	2.3M	2.3M	1.6K	320	16.0	99.52
LoLa [4]	573	393	178	2	2.2	2.2	98.95
SHE [27]	23K	19K	945	3K	9.3	9.3	99.54
EVA [9]	8K	4K	4K	3	121.5	121.5	99.05
VDSCNN [25]	4K	2K	2K	48	105	105	99.19
Backbone CNN-3	1962	973	984	5	7.2	1.4	98.95
bi-CryptoNets (CNN-3)	830	406	418	6	2.1	0.1	99.15

operations as in [8, 28]. We also consider the important amortized inference latency in a batch of images [25], and concern the number of activations for ciphertexts [4, 9, 27]. Tables 3 and 4 summarize experimental comparisons on MNIST and CIFAR-10, respectively. Similar results could be made for CIFAR-100, presented in our full work [39].

From Table 3, our bi-CryptoNets can reduce HE operations by $2.35\times$ and inference latency by $3.43\times$ in contrast to backbone network. Our bi-CryptoNets could decrease the amortized latency by $13.7\times$, and achieve better accuracy (about 0.2%) than backbone because of our knowledge distillation and precise inference in the plaintext branch.

From Table 4, our bi-CryptoNets reduces HE operations by $1.43\times$, and inference latency by $1.15\times$ in contrast to backbone network. Our bi-CryptoNets could decrease the amortized latency by $4.59\times$ and improve accuracy by 2.1%. We also implement our bi-CryptoNets with VGG-16 and ResNet-18 as backbones, and deeper networks yield better inference accuracies, i.e., 93.27% for VGG-16 and 93.91% for ResNet-18.

From Tables 3 and 4, our bi-CryptoNets takes a good balance between inference accuracies and computation cost in comparison with other neural networks. Our bi-CryptoNets achieves lower inference latency and amortized latency than other methods except for LoLa, where light-weight neural network is implemented with complicated packing method and smaller HE operations. Our bi-CryptoNets takes relatively-good inference accuracies except for SHE, CryptoDL and VDSCNN, where deeper neural networks are adopted with larger computational overhead. Based on deeper backbones such as VGG-16 and ResNet-18, our bi-CryptoNets could achieve better inference accuracy, comparable inference latency and smaller amortized latency.

Table 4. The experimental comparisons on CIFAR-10.

Scheme	HEOPs	AddCC	MulPC	ActC	Latency (s)	Amortized Latency (s)	Acc (%)
FCryptoNets [8]	701M	350M	350M	64K	22372	1398	76.72
CryptoDL [20]	2.4G	1.2G	1.2G	212K	11686	731	91.50
LoLa [4]	70K	61K	9K	2	730	730	76.50
SHE [27]	4.4M	4.4M	13K	16K	2258	2258	92.54
EVA [9]	135K	67K	67K	9	3062	3062	81.50
VDSCNN [25]	18K	8K	9K	752	2271	2271	91.31
Backbone CNN-11	1.0M	493K	521K	246	1823	228	88.21
bi-CryptoNets (CNN-11)	709K	341K	368K	246	1587	49	90.30
bi-CryptoNets (VGG-16)	3.4M	1.5M	1.9M	1.1K	2962	92	93.27
bi-CryptoNets (ResNet-18)	15.5M	6.9M	8.6M	2.8K	6760	211	93.91

6 Conclusion

Numerous privacy-preserving neural networks have been developed to keep the balance between accuracy, efficiency and security. We take a different view from the input data and network structure. We decompose the input data into sensitive and insensitive segments, and propose the bi-CryptoNets, i.e., plaintext and ciphertext branches, to deal with two segments, respectively. We also introduce feature-based knowledge distillation to strengthen the representations of our network. Empirical studies verify the effectiveness of our bi-CryptoNets. An interesting future work is to exploit multiple levels of privacy of input data and multiple branches of neural network, and it is also interesting to generalize our idea to other settings such as multi-party computation.

Acknowledgements. The authors want to thank the reviewers for their helpful comments and suggestions. This research was supported by National Key R&D Program of China (2021ZD0112802), NSFC (62376119) and CAAI-Huawei MindSpore Open Fund.

References

- Boemer, F., Costache, A., Cammarota, R., Wierzyński, C.: nGraph-HE2: a high-throughput framework for neural network inference on encrypted data. In: WAHC@CCS, pp. 45–56 (2019)
- Boulemtafes, A., Derhab, A., Challal, Y.: A review of privacy-preserving techniques for deep learning. Neurocomputing **384**, 21–45 (2020)
- Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (leveled) fully homomorphic encryption without bootstrapping. ACM Trans. Comput. Theory **6**(3), 1–36 (2014)
- Brutzkus, A., Gilad-Bachrach, R., Elisha, O.: Low latency privacy preserving inference. In: ICML, pp. 812–821 (2019)
- Chabanne, H., Wargny, A., Milgram, J., Morel, C., Prouff, E.: Privacy-preserving classification on deep neural network. Cryptol. ePrint Arch. (2017)

6. Cheon, J.H., Kim, A., Kim, M., Song, Y.: Homomorphic encryption for arithmetic of approximate numbers. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017. LNCS, vol. 10624, pp. 409–437. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70694-8_15
7. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: TFHE: fast fully homomorphic encryption over the torus. *J. Cryptol.* **33**(1), 34–91 (2020)
8. Chou, E., Beal, J., Levy, D., Yeung, S., Haque, A., Fei-Fei, L.: Faster CryptoNets: leveraging sparsity for real-world encrypted inference. CoRR/abstract **1811.09953** (2018)
9. Dathathri, R., Kostova, B., Saarikivi, O., Dai, W., Laine, K., Musuvathi, M.: EVA: an encrypted vector arithmetic language and compiler for efficient homomorphic computation. In: PLDI, pp. 546–561 (2020)
10. Dathathri, R., et al.: CHET: an optimizing compiler for fully-homomorphic neural-network inferencing. In: PLDI, pp. 142–156 (2019)
11. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. *J. Priv. Confidentiality* **7**(3), 17–51 (2016)
12. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC, pp. 169–178 (2009)
13. Ghodsi, Z., Jha, N., Reagen, B., Garg, S.: Circa: stochastic ReLUs for private deep learning. In: NeurIPS, pp. 2241–2252 (2021)
14. Gilad-Bachrach, R., Dowlin, N., Laine, K., Lauter, K., Naehrig, M., Wernsing, J.: CryptoNets: applying neural networks to encrypted data with high throughput and accuracy. In: ICML, pp. 201–210 (2016)
15. Gong, X., Chen, Y., Yang, W., Mei, G., Wang, Q.: InverseNet: augmenting model extraction attacks with training data inversion. In: IJCAI, pp. 2439–2447 (2021)
16. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)
17. Gou, J., Yu, B., Maybank, S., Tao, D.: Knowledge distillation: a survey. *Int. J. Comput. Vis.* **129**(6), 1789–1819 (2021)
18. Hashemi, M.: Enlarging smaller images before inputting into convolutional neural network: zero-padding vs. interpolation. *J. Big Data* **6**(1), 1–13 (2019)
19. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR, pp. 770–778 (2016)
20. Hesamifard, E., Takabi, H., Ghasemi, M.: Deep neural networks classification over encrypted data. In: CODASPY, pp. 97–108 (2019)
21. Hinton, G., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Improving neural networks by preventing co-adaptation of feature detectors. CoRR/abstract **1207.0580** (2012)
22. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. CoRR/abstract **1503.02531** (2015)
23. Huang, G., Liu, Z., Maaten, L., Weinberger, K.: Densely connected convolutional networks. In: CVPR, pp. 2261–2269 (2017)
24. Iandola, F., Moskewicz, M., Ashraf, K., Han, S., Dally, W., Keutzer, K.: SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size. CoRR/abstract **1602.07360** (2016)
25. Lee, E., et al.: Low-complexity deep convolutional neural networks on fully homomorphic encryption using multiplexed parallel convolutions. In: ICML, pp. 12403–12422 (2022)
26. Li, Z., et al.: Curriculum temperature for knowledge distillation. In: AAAI, pp. 1504–1512 (2023)
27. Lou, Q., Jiang, L.: SHE: a fast and accurate deep neural network for encrypted data. In: NeurIPS, pp. 10035–10043 (2019)
28. Lou, Q., Lu, W., Hong, C., Jiang, L.: Falcon: fast spectral inference on encrypted data. In: NeurIPS, pp. 2364–2374 (2020)
29. Radosavovic, I., Kosaraju, R., Girshick, R., He, K., Dollár, P.: Designing network design spaces. In: CVPR, pp. 10425–10433 (2020)

30. Ribeiro, M., Grolinger, K., Capretz, M.: MLaaS: machine learning as a service. In: ICMLA, pp. 896–902 (2015)
31. Romero, A., Ballas, N., Kahou, S., Chassang, A., Gatta, C., Bengio, Y.: FitNets: hints for thin deep nets. In: ICLR (2015)
32. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
33. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR/abstract **1409.1556** (2014)
34. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. JMLR **15**(1), 1929–1958 (2014)
35. Tan, M., Le, Q.: EfficientNet: rethinking model scaling for convolutional neural networks. In: ICML, pp. 6105–6114 (2019)
36. Xie, X.R., Yuan, M.J., Bai, X.T., Gao, W., Zhou, Z.H.: On the Gini-impurity preservation for privacy random forests. In: NeurIPS (2023)
37. Yang, K., Yau, J., Fei-Fei, L., Deng, J., Russakovsky, O.: A study of face obfuscation in imagenet. In: ICML, pp. 25313–25330 (2022)
38. Yin, X., Zhu, Y., Hu, J.: A comprehensive survey of privacy-preserving federated learning: a taxonomy, review, and future directions. ACM Comput. Surv. **54**(6), 131:1–131:36 (2021)
39. Yuan, M.J., Zou, Z., Gao, W.: Bi-cryptonets: leveraging different-level privacy for encrypted inference. CoRR/abstract **2402.01296** (2024)
40. Zagoruyko, S., Komodakis, N.: Paying more attention to attention: improving the performance of convolutional neural networks via attention transfer. In: ICLR (2017)



Enhancing YOLOv7 for Plant Organs Detection Using Attention-Gate Mechanism

Hanane Ariouat^{1(✉)}, Youcef Sklab¹, Marc Pignal², Florian Jabbour², Régine Vignes Lebbe², Edi Prifti^{1,3}, Jean-Daniel Zucker^{1,3}, and Eric Chenin¹

¹ IRD, Sorbonne Université, UMMISCO, 93143 Bondy, France
hanane.ariouat@ird.fr

² Institut de Systématique, Evolution, Biodiversité (ISYEB), Muséum national d'Histoire naturelle, CNRS, Sorbonne Université, Université des Antilles, EPHE, 57, rue Cuvier, CP39, 75005 Paris, France

³ Sorbonne Université, INSERM, Nutrition et Obésités; Systemic Approaches, NutriOmique, AP-HP, Paris, France

Abstract. Herbarium scans are valuable raw data for studying how plants adapt to climate change and respond to various factors. Characterization of plant traits from these images is important for investigating such questions, thereby supporting plant taxonomy and biodiversity description. However, processing these images for meaningful data extraction is challenging due to scale variance, complex backgrounds that contain annotations, and the variability in specimen color, shape, and orientation of specimens. In addition, the plant organs often appear compressed, deformed, or damaged, with overlapping occurrences that are common in scans. Traditional organ recognition techniques, while adept at segmenting discernible plant characteristics, are limited in herbarium scanning applications. Two automated methods for plant organ identification have been previously reported. However, they show limited effectiveness, especially for small organs. In this study we introduce YOLOv7-ag model, which is a novel model based on the YOLOv7 that incorporates an attention-gate mechanism, which enhances the detection of plant organs, including stems, leaves, flowers, fruits, and seeds. YOLOv7-ag significantly outperforms previous state of the art as well as the original YOLOv7 and YOLOv8 models with a precision and recall rate of 99.2% and 98.0%, respectively, particularly in identifying small plant organs.

Keywords: Object Detection · Attention-gate · YOLOv7 · Herbarium specimen

1 Introduction

Plants are a crucial component of our planet's biodiversity, by shaping landscapes and being the foundation of trophic networks, and by playing a major role in

Our code is available at <https://github.com/IA-E-Col/YOLOv7-ag>

H. Ariouat and Y. Sklab—Both authors contributed equally to this work.

© The Author(s), under exclusive license to Springer Nature Singapore Pte Ltd. 2024
D.-N. Yang et al. (Eds.): PAKDD 2024, LNAI 14646, pp. 223–234, 2024.

https://doi.org/10.1007/978-981-97-2253-2_18

the balance between carbon dioxyde and oxygen [4]. Natural history collections are house to several hundreds years of data on the evolution of the environment and biodiversity [3]. Herbarium collections are gaining interest in the scientific community since they can lead to our improved understanding and mitigate serious threats to biodiversity thanks to the information they carry [1,2,5–8]. The collections of several herbaria have recently been extensively digitized [9], creating several virtual collections (online biodiversity data portals) and making possible the use of computer-based processing techniques, such as Deep Learning (DL) to describe plant morphology, such as leaf size, perimeter and width, and counting organs [10], for assessing crop quality after harvest [11] and/or for disease classification [12]. More specifically, with object detection techniques, organs or prominent areas can be automatically detected and measured [15].

The objective of this work is to improve existing computer vision techniques in the task of predicting bounding boxes and classify them in a set of categories, such as plant organs. Here we focus on a subset of digitized herbarium specimens from the Paris Museum collection. More specifically, we focus on the images of the scanned 2D herbarium sheets and aim at identifying plants' organs (leaf, flower, fruit, stem, root, seed) for further analyses. Recently, there has been important progress in the field of object detection, and most of the existing AI-based models produce good results for medium and big objects [19]. However, since there are possibly a large number and overlapping plant organs of different sizes in the images, using existing models to detect objects in herbarium scans may not be appropriate, particularly, for small organs detection.

In this article, we propose an extension of the YOLOv7 model [20], which improves plant organ identification, particularly the smaller organs. This new model called YOLOv7-ag integrates an attention mechanism: the attention-gate [21]. YOLOv7-ag better exploits spatial information like the arrangement of organs and the presence of other organs in the spatial perimeter of a given organ. Using an attention mechanism essentially improves identifying regions of interest in plants, which is critical in the detection of the targeted organs.

2 Related Work

A herbarium is a collection of dried plant specimens that are arranged on herbarium sheets [8]. A typical herbarium sheet scan might contain non-vegetal elements such as stamps, scale bars, color palettes, annotation labels, barcodes, and envelopes (*cf.* Fig. 1). A first analysis of the herbarium scans allowed us to notice that they include several elements that bring significant difficulties for object detection techniques based on conventional networks. For instance, they have important scale variance and complex backgrounds with several non-vegetable elements. Moreover, we noticed that the color, shape, and orientation of specimen images vary greatly and in several images the plant's organs are condensed and deformed/damaged. Most often, organs such as leaves, flowers and fruits are overlapped on a reduced space of the herbarium scan.



Fig. 1. Typical herbarium sheet scans, including plant material and non-biological material, with sometimes overlapping, incomplete, compressed and deformed (or even missing) objects.

YOLO-based algorithms have been used in numerous works on object detection. Zhang et al. [10] focused on identifying and detecting fruits with two different algorithms: OrangeYolo to detect the orange fruits and OrangeSort to track them. Both algorithms were based on YOLOv3 algorithm. Zhao et al. [23] used YOLOv7 and enhanced it for object detection for maritime UAV images. The authors added one head at the detection phase to allow the network to identify small people or objects and implemented a SimAM attention module to allow the network to find the scene's attention region. Almost 59% of precision was reached. James et al. [25] provided a whole pipeline for head density estimation for sorghum from RGB images, collected via unmanned aerial vehicle (UAV). The authors used YOLO-v5 and Faster RCNN algorithms. Based on YOLOv4, Zhang et al. [24] proposed the ViT-YOLO (Vision Transformer) object detection approach for drone captured images (UAV images, VisDrone2019-Det dataset). The authors enhanced YOLOv4 by adding a multi-head attention block using the MHSA-Darknet, which is an upgraded Darknet backbone. Xu et al. [26] proposed TrichomeYolo to identify the density and length of maize leaf trichomes from scanning electron microscopy images. They used YOLOv5 as the base model and integrated a Transfomer into it, which reached 92.5 % accuracy. In the same line of YOLO-based approaches, Zhou et al. [19] enhanced YOLOv5 to better detect small objects in infrared imaging. The authors proposed an image Super-Resolution reconstruction based on Generative Adversarial Network (SRGAN), Self-Adaption Squeeze-and-Excitation (SASE) module and a YOLO based network, called a multi-receptive field adaptive channel attention network.

Despite the existence of a large number of object detection approaches in the literature, there is a lack of work that addresses plant organ's detection on herbarium scans. Identifying these organs can considerably help provide valuable information to investigate several research questions such as long-term pheno-

logical investigations about the consequences of climatic change [22]. In this regard, Triki et al. [8] focused on the identification of four plant organs (leaves, flowers, fruits, and buds) and proposed a YOLOv3 based algorithm. To enhance feature extraction and better detect small plant organs, the authors integrated two architectures, namely, ResNet and DenseNet, and included a new feature scale. By doing so, they reached 94.2% precision rate. Although this method appears to provide interesting results, we could not assess its applicability on our dataset as the authors did not publish their models, nor their dataset. In the same context, a Faster RCNN (Region Convolutional Neural Network) based approach for detecting plant organs (leaf, stem, seed, fruit, flower, and root) on herbarium specimens scans was proposed by Souhaib et al. [5]. Only 22.8% of precision (AP) was reached with IoU (Intersection over Unions) equal to 0.5 and only the leaf and the stem organs were well detected with this approach.

2.1 Attention-Gate Mechanism

Attention mechanisms are used to focus on specific regions of interest (spatial information) in images that are deemed more relevant or important for the prediction task. We believe that using these mechanisms in object detection would improve the ability of the neural network to recognize plant organs, particularly in the regions of the images where there is a high concentration of organs. In this paper, we focus on the attention-gate mechanism (*cf.* Fig. 2), which is frequently used in natural image analysis, knowledge graph, and classification tasks [27]. Attention-gate directs the model's focus to crucial locations while suppressing the activation of features in irrelevant regions in an input image. It is able to focus on targets of varied sizes and shapes through an automated learning process.

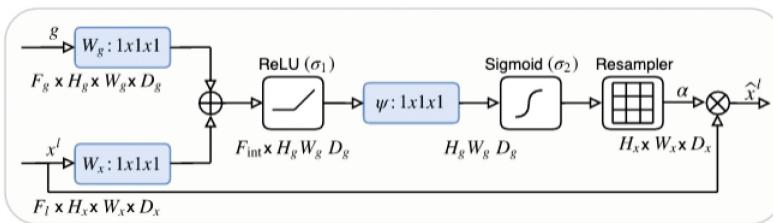


Fig. 2. Attention-gate block structure diagram as proposed by [21]. \oplus and \otimes represent element wise sum and matrix multiplication respectively, while 1×1 represents a pointwise convolution.

In the Fig. 2, attention coefficients (α) are computed in order to scale the input features (x^l). Spatial regions are selected by analysing both the activation and contextual information provided by the gating signal (g), which is collected

from a coarser scale. Grid re-sampling of attention coefficients is done using trilinear interpolation. The activations and contextual information provided by the gating signal (g), which is acquired from a coarser scale, are analyzed in order to select the spatial regions. Then, a trilinear interpolation is utilized in the process of resampling the attention coefficients. The features that are propagated through the skip connections (x^l) are filtered by attention gates and in order to achieve feature selectivity, contextual information (gating) gathered at coarser scales is used (g). More precisely, models trained with Attention-gate intuitively learn to highlight prominent features that are helpful for a particular task while suppressing irrelevant regions in an input image.

3 YOLOv7 with Attention-Gate Mechanism

We added two instances of the attention-gate module into two parts of the YOLOv7 model, as illustrated in Fig. 3. To the best of our knowledge, the attention-gate mechanism had not previously been incorporated into an object detection algorithm. YOLOv7-ag is structured in three main parts i) Backbone ii) Head and iii) Detection. The first one consists of the backbone and is the same as the one proposed in YOLOv7 [20]. It aims to extract the feature maps and consists of a succession of several blocks, namely, CBS, ELAN and MPCConv. The Fig. 4 illustrates the layers that constitute these blocks. The second part of the model architecture as described in [20] is called the Head. It mainly consists of a succession of several blocks (CBS, ELAN-H, SPPCSP, MPCConv), concatenation (Concat) and up-sampling (UPSample) blocks. With stride 1 and kernel sizes 3 or 1, ELAN and ELAN-H blocks in YOLOv7 [20] (see Fig. 4) have 8 stages (or block groups) of CBS (Convolution, Batch normalization, and activation function SILU). The MPCConv block contains five layers, one of which is max-pooling and three are CBS. The SPP block, which is a mixture of three max-pooling, is one of the nine blocks that forms the SPPCSP blocks.

YOLOv7 uses a combination of low-level and high-level characteristics through a summation or concatenation operation. However, doing so without making any distinctions may sometimes result in feature mismatches and performance deterioration. Our aim is to integrate learnable weights to determine the significance/importance of various input features. In our approach, we modified the HEAD block by adding the attention-gate mechanism, followed by a CBS block. More specifically, we integrated two attention-gate blocks into the YOLOv7 architecture¹, which allow exploiting the spatial information coming from the upper layers (which is collected from a coarser scale) to compute a coefficient matrix, called attention coefficient. It is a condensed representation of the contextual information derived from the spatial information coming from the upper layers. Then, applying the computed attention coefficient on the deep layers, (which provide feature maps) allows to determine which features coming from the upper layers are determining in the feature maps. This aims to consider more the relevant features of the target contained in the shallow network and

¹ After the backbone network and before the detection layer.

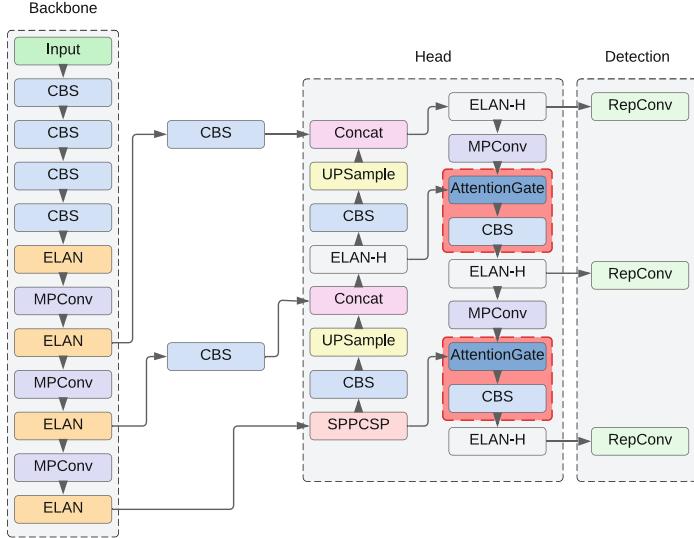


Fig. 3. YOLOv7-ag: YOLOv7 with attention-gate.

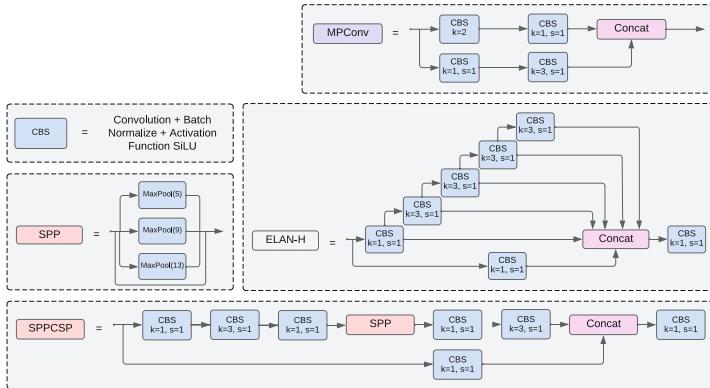


Fig. 4. YOLOv7 component blocks

give less importance to the irrelevant information, improving the detection performance of the algorithm. Because self-attention across n entities needs $O(n^2d)$ memory and computation [13], we believe that the simplest configuration that complies with the aforementioned considerations would be to include the attention mechanism in the two head's lowest resolution feature maps. For the third part of the model architecture, which aims at predicting layers at 3 different scales², we kept the detection part as it was originally introduced in [20].

² The three scales are designed to detect small, medium, and large objects, respectively. They are represented by feature maps that are extracted at different depths of the

4 Experiments

We conducted experiments using a dataset of several herbarium specimen scans that contains plants with organs of varying sizes and diverse shapes. We compared the results of YOLOv7-ag with four models: Faster R-CNN [5], YOLOv7 [20], YOLOv8 [28] and YOLOv8-ag. YOLOv8-ag is the version of YOLOv8 with the attention-gate mechanism, that we integrated similarly to our approach in YOLOv7-ag. We trained the YOLO models on two NVIDIA A100 GPUs with 80 GB memory for 800 epochs. The optimizer used was Adam, with a default weight decay of 0.0005 and a momentum of 0.937. The batch size was set to 16, and the initial learning rate was 0.001. The input image size was 640×640 . We also compared the obtained results from YOLOv7-ag with the plant organ detection model of Younis et al. [5].

4.1 Experiment Materials

For the training data, we used the publicly available dataset, published by Younis et al. [5], which contains 635 images. The height of the images approximately ranges between 3500 and 1.600 pixels and the width ranges between 300 and 500 pixels. Taking into account their diversity observed in our dataset of herbarium scanned images (over 10 million images), the Younis et al. dataset is relatively small, which needs to be increased to cover a large spectrum of possible cases and improve the performance of our model. We therefore used a rotational data augmentation technique, with various angles. For each image we performed 5 rotations at 5 angles [45, 180, 225, 270, 315] and adjusted the bonding boxes accordingly. Note that we deactivated YOLOv7's (and YOLOv8's) default augmentation pipeline as it performs such transformations that produce images that aren't present in our herbarium scans (e.g., mosaic augmentation, cutout augmentation, hsv color augmentation, etc.). At the end we increased the initial dataset by a factor of 5, resulting in 3810 images, from which 3000 were allocated to the training set, and the remaining 784 to the validation set. For the test dataset, we used 26 images, representing various species with distinct sizes and shapes.

4.2 Evaluation Metrics

The prediction task in object detection is a bounding box along with the object at hand and its corresponding label (class). To evaluate the performance of YOLOv7-ag, we used a set of metrics widely used in object detection tasks: i) Precision, ii) Recall and iii) mean Average Precision (mAP). The Intersection over Union (*IoU*) metric is used to quantify how the predicted box is aligned compared to the actual ground truth of the bounding box³.

neural network, thus allowing for precise detection across a varied range of object sizes.

³ Ground truth represents the desired bounding box as output of the object detection algorithm.

$$IoU = \frac{\text{Area of overlap}}{\text{Area of union}} \quad (1)$$

We consider the values bigger than 0.5 for the prediction frame and the target frame's intersection over union (IoU), so $IoU > 0.5$, as the criterion for evaluating target detection.

4.3 Experimental Results

The results (*cf.* Table 1) demonstrate that our model YOLOv7-ag, has superior performance compared to YOLOv7, as well as YOLOv8, YOLOv8-ag, and FasterR-CNN. This superiority is visible across all the used metrics, including precision, recall, and mean average precision (mAP).

Table 1. Precision and Recall Metrics for All Classes on the Validation Dataset.

Method	Precision	Recall	mAP0.5
FasterR-CNN [5]	9.7%	NM	22.5%
YOLOv7 [28]	97.3%	96.2%	97%
YOLOv7-ag	99.19%	98%	99.1%
YOLOv8 [20]	94.5%	92.8%	96.1%
YOLOv8-ag	93%	93.6%	94.7%

Table 2. Precision Metrics by Class on the Validation Dataset

Method	leaf	seed	root	flower	fruit	stem
FasterR-CNN [5]	26.5%	0%	9.4%	4.7%	7.8%	9.9%
YOLOv7 [20]	99.2%	97.4%	97%	95%	97.5%	97%
YOLOv7-ag	99.8%	98.1%	98.9%	98.5%	99.3%	99.5%
YOLOv8 [28]	95.3%	97.4%	95%	90.1%	92.6%	97%
YOLOv8-ag	95.9%	97.4%	96.1%	92.7%	94.8%	81.5%

Table 2 further details the precision values for different organ classes, underscoring the particularly enhanced performance of YOLOv8-ag. It is also important to note that the integration of the attention-gate mechanism into YOLOv8 contributes to an improvement in its performance metrics as well (*cf.* Table 2). However, it still inferior to that of YOLOv7-ag. A key strength of YOLOv7-ag is its ability to detect smaller objects, even with varying degrees of occlusion, demonstrating its advanced capabilities in handling complex visual scenarios. When analyzing the precision for different plant organ classes individually,

YOLOv7-ag showed higher values, surpassing other models for every class as detailed in Table 2. The recall rates are also higher for leaves, seeds, roots, flowers, fruits, and stems, recorded at 99.6%, 100%, 100%, 93.9%, 95.9%, and 98.8% respectively.

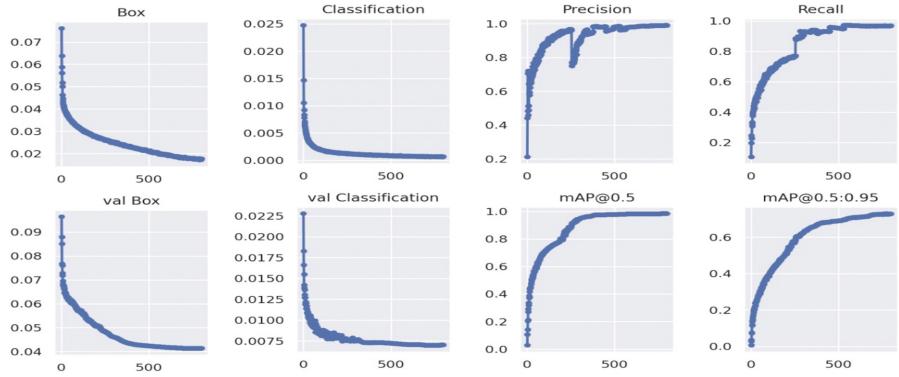


Fig. 5. Plots of box loss, classification loss, precision, recall and mean average precision (mAP) over the training epochs for the training and validation set for YOLOv7-ag.

The Fig. 5 shows various performance measures for both the training and validation set for YOLOv7-ag. The box loss represents how well the model can locate the center of an object and how well the predicted bounding box covers an object. Classification loss gives an idea of how well the model can predict the correct class of a given object. As illustrated in Fig. 5, before plateauing after roughly 500 epochs, the model quickly increased in terms of precision, recall, and mean average precision. The validation data's box, objectness, and classification losses all displayed a sharp decline.

The Fig. 6 shows a comparison of the prediction of the two models YOLOv7 and YOLOv7-ag, as they are the two models with the best performance, on the same test image. YOLOv7-ag can identify more organs, such as leaves, flowers, stems, etc., with greater accuracy than YOLOv7. It is particularly better at detecting small organs. However, both models struggle to identify leaves when there are many overlapping leaves. Additionally, it proves challenging for models to accurately identify stems obscured by leaves in an image. We observed that YOLOv7 faced difficulties in accurately distinguishing between fruits and flowers. The use of attention-gate mechanisms enhances the detection of plant organs in herbarium images by focusing the model's attention on relevant features. It helps in distinguishing between overlapping or closely situated organs, by emphasizing unique characteristics and reducing background noise. It also aid in handling the inherent variability in plant images and provide contextual understanding, crucial for accurately identifying and classifying the plant organs based on their surrounding features. Interestingly, both models have very

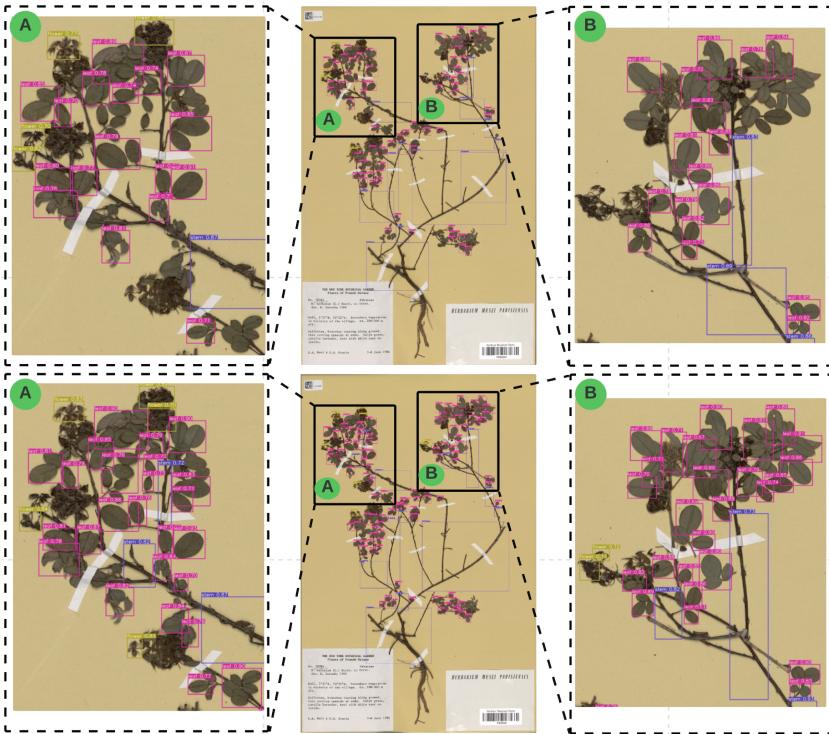


Fig. 6. Prediction example (YOLOv7 on the top and YOLOv7-ag on the bottom). 25 leaves (YOLOv7-ag) vs. 18 leaves (YOLOv7) in box A, 26 leaves leaves (YOLOv7-ag) vs. 17 leaves (YOLOv7) in box B, and 5 flowers (YOLOv7-ag) vs. 4 flowers (YOLOv7) in box A, 2 flowers (YOLOv7-ag) vs. 0 flowers (YOLOv7) in box B and 3 stems (YOLOv7-ag) vs. 1 stem (YOLOv7) in box A.

close inference times, with YOLOv7 at approximately 85 ms and YOLOv7-ag at 90 ms, while YOLOv8 and YOLOv8-ag have approximately 22 ms and 26 ms respectively.

5 Conclusion

Plant organ identification poses significant problems for general object detectors due to the tiny target size and the variety of forms and shapes. In response to these issues, we proposed the YOLOv7-ag model which outperformed existing plant organs detectors and achieved 99% precision in identifying plant organs. The accuracy and recall rate of YOLOv7-ag exceeded those of YOLOv7 by 2% and 3%, respectively, in the experiment, and significantly improved the stability of the results. Nevertheless, the detection of plant organs still requires further improvements. The most promising avenues for boosting performance involve

refining the data annotation process and modifying the attention block within the original architecture to operate across multiple levels.

Recognizing the bounding boxes assigned to each organ type, in the herbarium images, could allow for targeted datasets for subsequent recognition of organ properties. It is thus a crucial step towards the automatic morphological annotation of millions of herbarium specimens, and future extended research interfaces on collection databases.

References

1. Ariouat, H., et al.: Extracting masks from herbarium specimen images based on object detection and image segmentation techniques. *Biodiv. Inf. Sci. Stand.* **7**, e112161 (2023)
2. Sahraoui, M., Sklab, Y., Pignal, M., Lebbe, R.V., Guigue, V.: Leveraging Multi-modality for Biodiversity Data: exploring joint representations of species descriptions and specimen images using CLIP. *Biodivers. Inf. Sci. Stand.* **7**(2023), e112666 (2023)
3. Meredith, L.: Roles of natural history collections. *Ann. Mo. Bot. Gard.* **4**(83), 536–545 (1996)
4. Raven, P.H.: Saving plants, saving ourselves. *Plants People Planet* **1**, 8–13 (2019)
5. Younis, S., Schmidt, M., Weiland, C., Dressler, S., Seeger, B., Hickler, T.: Detection and annotation of plant organs from digitised herbarium scans using deep learning. *Biodiv. Data J.* **8**, e57090 (2020)
6. Besnard, G., et al.: Herbarium-based science in the twenty-first century. *Botany Lett.* **165**, 323–327 (2018)
7. Soltis, P.: Digitization of herbaria enables novel research. *Am. J. Bot.* **104**, 1281–1284 (2017)
8. Abdelaziz, T., Bassem, B., Walid, M.: A deep learning-based approach for detecting plant organs from digitized herbarium specimen images. *Eco. Inform.* **69**, 101590 (2022)
9. Patrick, W., et al.: Large-scale digitization of herbarium specimens: development and usage of an automated, high-throughput conveyor system. *Taxon* **67**, 165–178 (2018)
10. Wenli, Z., et al.: Deep-learning-based in-field citrus fruit detection and tracking. *Horticult. Res.* **9**, uhac003 (2022)
11. Jiang, Y., Li, C.: Convolutional neural networks for image-based high-throughput plant phenotyping: a review. *Plant Phenom.* **9** (2020)
12. Borhani, Y., Khoramdel, J., Najafi, E.: A deep learning based approach for automated plant disease classification using vision transformer. *Sci. Rep.* **12**, 11554 (2022)
13. Ashish, V., et al.: Attention is all you need. *CoRR* abs/1706.03762 (2017)
14. Sue Han, L., Chee Seng, C., Simon, J., Paolo, R.: How deep learning extracts and learns leaf features for plant classification. *Pattern Recogn.* **71**, 1–13 (2017)
15. Mochida, K., et al.: Computer vision-based phenotyping for improvement of plant productivity: a machine learning perspective. *GigaScience* **8**, giy153 (2018)
16. Shaoqing, R., Kaiming, H., Ross, B.G., Jian, S.: Faster R-CNN: towards real-time object detection with region proposal networks. *Computer Vision and Pattern Recognition* abs/1506.01497 (2015)

17. Kaiming, H., Georgia, G., Piotr, D., Ross, B.G.: Mask R-CNN. Computer Vision and Pattern Recognition, vol. abs/1703.06870 (2017)
18. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016)
19. Xiao, Z., Lang, J., Shuai, L., Tingting, Z., Xingang, M.: YOLO-SASE: an improved YOLO algorithm for the small targets detection in complex backgrounds. Computer Vision and Pattern Recognition, vol. abs/2207.02696 (2022)
20. Chien-Yao, W., Alexey, B., Hong-Yuan Mark, L.: YOLOv7: trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. Computer Vision and Pattern Recognition, vol. abs/2207.02696 (2022)
21. Ozan, O., et al.: Attention U-Net: learning where to look for the pancreas. Computer Vision and Pattern Recognition, vol. abs/1804.03999 (2018)
22. Lang, P.M., Willems, F., Scheepens, J.F., Burbano, H., Bossdorf, O.: Using herbaria to study global environmental change. *New Phytol.* **2021**, 110–122 (2019)
23. Zhao, H., Zhang, H., Zhao, Y.: YOLOv7-Sea: object detection of maritime UAV images based on improved YOLOv7. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops (2023)
24. Zixiao, Z., et al.: ViT-YOLO: transformer-based YOLO for object detection. In: IEEE/CVF International Conference on Computer Vision Workshops, ICCVW 2021, Montreal, BC, Canada, 11–17 October 2021
25. James, C., et al.: From prototype to inference: a pipeline to apply deep learning in sorghum panicle detection. *Plant Phenomics* **5**, 0017 (2023)
26. Jie, X., et al.: TrichomeYOLO: a neural network for automatic maize trichome counting. *Plant Phenom.* **5**, 0024 (2023)
27. Zhaoyang, N., Guoqiang, Z., Hui, Y.: A review on the attention mechanism of deep learning. *Plant Neurocomput.* **452**, 48–62 (2021)
28. Dillon, R., Jordan, K., Jacqueline, H., Ahmad, D.: Real-time flying object detection with YOLOv8. Computer Vision and Pattern Recognition (2023)



On Dark Knowledge for Distilling Generators

Chi Hong¹, Robert Birke³, Pin-Yu Chen⁴, and Lydia Y. Chen^{1,2(✉)}

¹ Delft University of Technology, Delft, Netherlands

c.hong@tudelft.nl, lydiaychen@ieee.org

² University of Neuchatel, Neuchatel, Switzerland

³ University of Torino, Turin, Italy

robert.birke@unito.it

⁴ IBM Research, New York, USA

pin-yu.chen@ibm.com

Abstract. Knowledge distillation has been applied on generative models, such as Variational Autoencoder (**VAE**) and Generative Adversarial Networks (**GANs**). To distill the knowledge, the synthetic outputs of a teacher generator are used to train a student model. While the dark knowledge, i.e., the probabilistic output, is well explored in distilling classifiers, little is known about the existence of an equivalent dark knowledge for generative models and its extractability. In this paper, we derive the first kind of empirical risk bound for distilling generative models from a Bayesian perspective. Through our analysis, we show the existence of the dark knowledge for generative models, i.e., Bayes probability distribution of a synthetic output from a given input, which achieves lower empirical risk bound than merely using the synthetic output of the generators. Furthermore, we propose a **Dark Knowledge based Distillation**, **DKtill**, which trains the student generator based on the (approximate) dark knowledge. Our extensive evaluation on distilling VAE, conditional **GANs**, and translation **GANs** on Facades and **CelebA** datasets show that the FID of student generators trained by **DKtill** combining dark knowledge are lower than student generators trained only by the synthetic outputs by up to 42.66%, and 78.99%, respectively.

Keywords: Knowledge distillation · Generators · Risk bounds

1 Introduction

Generative models, such as Variational autoencoders (**VAE**) and generative adversarial networks (**GANs**), are increasingly applied to synthesize images. To practically deploy those models on edge devices, namely the generators, it is critical to distill/compress the models first due to the memory and computation constraints. Knowledge distillation via teacher and student models can effectively compress the machine learning models, especially for classification models [2, 4, 6]. The student model tries to imitate the (non-compressed) teacher model through the input and output pairs from the teacher model such that the same learning

efficacy can be achieved via a smaller model. Dark knowledge [21] is shown particularly critical for the distillation quality of classifiers. Recently, some related studies [1, 3, 19, 19] focus on distilling the generator of GANs. This prior art empirically distills the teacher generator by directly leveraging the synthetic outputs. While the studies in [13, 16, 20] provide theoretical analysis for distilling classifiers, e.g., distillation bound, little is on the theoretical understanding of distilling generative models, for instance, the empirical risk bound and the existence of dark knowledge of the generators.

In this paper, we rethink the generator distillation from a theoretical Bayesian perspective. We hypothesize the existence the “dark knowledge”, which is embedded in the synthetic output, e.g., image, but not directly observable. To such an end, we model the synthetic outputs of a generator as a conditional Bayes probability distribution, $\mathbb{P}(y|x)$, where y is the synthetic output and x is the input of the generator. We derive two types of empirical risks for distillation: regular empirical risk using only synthetic inputs, and Bayes empirical risk using the proposed conditional probability distribution. We show that the variance of the Bayes empirical risk is lower than the variance of regular empirical risk, demonstrating better generalization capability. We thus refer to the knowledge of $\mathbb{P}(y|x)$ as dark knowledge. Our analysis can be applied on both probabilistic generative models, e.g., VAE, and non-probabilistic models, e.g., GANs. To derive such Bayes empirical risk bound for GANs, we approximate the conditional probability distribution and quantify its impact in the distillation bound.

Motivated by the effectiveness of such dark knowledge, we further propose the **Dark Knowledge Distillation**, **DKtill**, algorithm to train a student generator model through the (approximate) dark knowledge. Specifically, we try to minimizing the difference between teacher and student outputs by controlling the tensor of the second last layer of their networks, which is an approximated conditional Bayes probability. We extensively evaluate **DKtill** on distilling VAE, conditional GANs and translational GANs for Facades, and CelebA datasets. Our results show that the student generators from **DKtill** achieve lower FID than the ones trained on only synthetic images by up to 78.99% and a slightly higher FID than the ones from the teacher model by 17.77%.

Our key contributions for this work can be summarized as follows. *i)* Having Theorem 1 and Proposition 1 as the distillation empirical risk analysis to show the effectiveness of dark knowledge in distilling generative models. *ii)* Deriving Proposition 2 as the approximate distillation empirical risk analysis to capture the impact of approximating dark knowledge in distilling the GANs. *iii)* Proposing **DKtill** which is a dark knowledge based distillation algorithm for training student generative models. *iv)* Achieving higher distillation quality of **DKtill** on three different generative models and 3 different datasets, compared to distillation algorithms which do not use the proposed (approximated) dark knowledge.

2 Preliminary

We consider general generative models for synthesizing images, including non-probabilistic generators and probabilistic generators. In existing generator dis-

tillation works [1, 3, 19] the basic idea is as follows. First, they provide the same inputs x into a trained teacher model \mathcal{T} and a student model f where x is the generator input. x has different formats in different generative models. For vanilla GANs or VAE, x is a noise vector. For conditional GANs, x is a noise and a conditional label. x can also be a picture in image translation GANs. Second, the distance between the two model outputs $\mathcal{T}(x)$ and $f(x)$ is minimized. In this way, the student f can mimic the input-output mapping of \mathcal{T} and extract the knowledge from the teacher's synthetic image outputs $\{\mathcal{T}(x)\}$.

In classifier distillation [4], the final predicted labels of a teacher are not sufficient to train a student. The logits or the softmax outputs of the teacher model are needed to train the student so as to capture the underlying “dark knowledge”, which is the probability the teacher assigns to “wrong” labels. Some basic settings and terms of generator distillation used in this paper are introduced in the following.

Synthetic Images. In generator distillation, we input x into a trained teacher¹ \mathcal{T} and obtain the corresponding synthetic output $y = \mathcal{T}(x)$. After N inferences, a training dataset $D = \{(x_n, y_n)\}_{n=1}^N \sim \mathbb{P}^N$ is obtained to train the student model f , where x_n is an input sample and $y_n = \mathcal{T}(x_n)$ is the corresponding synthetic image. To simplify the notation, without loss of generality, y_n is a single channel image and $y_n^{ij} \in \{0, \dots, C\}$ is a pixel of the image, where C indicates the number of possible colors². Let $i \in [1, \dots, I]$ and $j \in [1, \dots, J]$ be the pixel indexes of an image of width I and height J .

Distillation Optimization. To distill the knowledge from the teacher model \mathcal{T} , a student generator which can be defined as $f : \mathcal{X} \rightarrow \mathbb{R}^{I \times J \times C}$ is trained to minimize the following risk:

$$R(f) = \mathbb{E}_{(x,y) \sim \mathbb{P}} [\ell(y, f(x))]. \quad (1)$$

According to the definition of f , the mapping result $f(x)$ is an $I \times J \times C$ tensor, and $f^{ij}(x) \in \mathbb{R}^C$ corresponds to a pixel. The vector $f^{ij}(x)$ represents the weights of taking different colors in the pixel. For each training pair (x, y) collected from \mathcal{T} , the loss of $f(x)$ takes the form $\ell(y, f(x)) = \sum_i \sum_j \ell^{ij}(y^{ij}, f^{ij}(x))$. The term $\ell^{ij}(y^{ij}, f^{ij}(x))$ is the loss value of predicting $f^{ij}(x)$ when the true pixel color is y^{ij} . To do distillation, the student needs to mimic the teacher's outputs. Therefore $\ell^{ij}(y^{ij}, f^{ij}(x))$ should be a loss that minimizes the distance between $y = \mathcal{T}(x)$ and $f(x)$, e.g., the softmax cross-entropy loss.

Distillation Using Only Synthetic Images. Having different definitions of $\ell(y, f(x))$ in $R(f)$ leads to different distillation methods. A common way in most existing works is to define $\ell(y, f(x))$ directly using the synthetic output images $\{y\}$ from the teacher. Given $y = \mathcal{T}(x)$, the basic idea for training the student f is to maximize the element $f^{ij, y^{ij}}(x)$ in the output probabilities $f^{ij}(x)$ of the student for each pixel i, j .

¹ We interchangeably use teacher or target model/generator.

² The analysis of this paper can be straightforwardly extended to three channel images.

3 Theoretical Analysis of Dark Knowledge in Distilling the Generator

This section introduces the novel concept of dark knowledge in distilling generators and demonstrates that harvesting this dark knowledge can train a student generator f which generalizes better. In generator distillation, the goal is to transfer the knowledge of the teacher \mathcal{T} as much as possible to a student f . We use the risk (generalization error) of the student f shown in Eq. 1 to evaluate the quality of the distilled student model. The risk describes the generalization ability of f on unseen new data sampled from \mathbb{P} other than the observed training dataset $D = \{(x_n, y_n)\}_{n=1}^N \sim \mathbb{P}^N$. A lower risk value means having a better distilled f . In the following, we first introduce the concept of dark knowledge in generative models and then derive the distillation empirical risks.

3.1 Dark Knowledge of Generators

From a Bayesian viewpoint, each synthetic image y from the teacher is generated by a conditional distribution $\mathbb{P}(y|x)$ where the distribution \mathbb{P} is potentially determined by the trained teacher generator \mathcal{T} . Obviously, the knowledge of $\mathbb{P}(y|x)$ cannot be fully extracted from the teacher’s synthetic image output. A synthetic image y is only a sample drawn from the distribution $\mathbb{P}(y|x)$. Thus, in generator distillation we call the underlying distribution $\mathbb{P}(y|x)$ the “dark knowledge” of the teacher \mathcal{T} . The next section demonstrates that using the dark knowledge $\mathbb{P}(y|x)$ to define $\ell(y, f(x))$ so as to have a precise empirical estimation of $R(f)$ can facilitate the training of a student in generator distillation.

We consider two ways of distilling teacher generator: i) by solely the synthetic image outputs of \mathcal{T} , and ii) by using the underlying dark knowledge $\mathbb{P}(y|x)$ of \mathcal{T} . We analyze the generalization error of the student f under both distillation ways by comparing their (empirical) risks. According to the introduction in Sect. 2, in generator distillation, $R(f)$ should be minimized by the algorithm where Eq. 1 shows the general definition of $R(f)$. However, calculating the exact value of $R(f)$ is intractable because \mathbb{P} is unknown or it has no explicit expression. Hence empirical risk definitions are required to approximate $R(f)$. The aforementioned two distillation ways leverage two different corresponding empirical risk definitions. These will be illustrated in detail in the following.

3.2 Distillation Empirical Risk

Distillation with Sole Synthetic Images. First, we introduce the distillation empirical risk of using \mathcal{T} ’s synthetic image outputs. In this way of distillation, the dataset $D = \{(x_n, y_n)\}_{n=1}^N$ collected from the teacher \mathcal{T} is used to train the student f . We have the following distillation empirical risk definition to approximate $R(f)$:

$$\hat{R}(f; D) = \frac{1}{N} \sum_{n \in [N]} \sum_i \sum_j \ell^{ij}(y_n^{ij}, f^{ij}(x_n)) = \frac{1}{N} \sum_{n \in [N]} \sum_i \sum_j e_{y_n^{ij}}^\top \ell^{ij}(f^{ij}(x_n)), \quad (2)$$

where $e_{y_n^{ij}}$ is the one-hot vector encoding of the color value of y_n^{ij} , and $\ell^{ij}(f^{ij}(x)) = [\ell^{ij}(1, f^{ij}(x)), \dots, \ell^{ij}(C, f^{ij}(x))] \in \mathbb{R}^C$ is a vector of loss values for each possible color of the pixel with index i, j . As shown in Eq. 2, this empirical risk definition only needs the synthetic image y_n from \mathcal{T} to decide the value of $e_{y_n^{ij}}$. Then we can calculate $\hat{R}(f; D)$ with no additional information from \mathcal{T} .

Distillation with Dark Knowledge. Here we assume that besides the final synthetic image output y we also get the conditional probability $\mathbb{P}(y|x)$ (the dark knowledge) of a given input x from \mathcal{T} . Consequently, we can define another distillation empirical risk $\hat{R}_\alpha(f, D)$ to approximate $R(f)$:

$$\hat{R}_\alpha(f, D) = \frac{1}{N} \sum_{n \in [N]} \sum_i \sum_j p^{ij}(x_n)^\top \ell^{ij}((f^{ij}(x_n))) \quad (3)$$

where the conditional probability of pixel y^{ij} color is denoted by

$$p^{ij}(x) = [\mathbb{P}(y^{ij}|x)]_{y_{ij} \in [C]} \quad (4)$$

Connection to the Empirical Risk. $\hat{R}(f; D)$ (Eq. 2) can be seen as an approximation of $\hat{R}_\alpha(f, D)$ (Eq. 3). Considering that: $R(f) = \mathbb{E}_{(x, y) \sim \mathbb{P}}[\ell(y, f(x))] = \mathbb{E}_{x \sim \mathbb{P}}[\mathbb{E}_{y|x}[\ell(y, f(x))]]$, we know that $\hat{R}_\alpha(f, D)$ is an empirical estimate of $\mathbb{E}_{x \sim \mathbb{P}}[\mathbb{E}_{y|x}[\ell(y, f(x))]]$ over the random variable x . If we further use the one-hot encoding $e_{y_n^{ij}}^\top$ to approximate $p^{ij}(x_n)$ in Eq. 3, we have Eq. 2. To distinguish these two empirical risks, we call $\hat{R}(f; D)$ (Eq. 2) as the *distillation empirical risk with synthetic images* and $\hat{R}_\alpha(f, D)$ (Eq. 3) as the *distillation empirical risk with dark knowledge*.

3.3 Generalization of the Student Generator

Given a teacher \mathcal{T} , a student model f can be trained by optimizing the distillation empirical risk with synthetic images (Eq. 2) or the distillation empirical risk with dark knowledge (Eq. 3). In this section, we analyze the generalization ability of the student f under the two different distillation empirical risk bounds. In the following, we show that a student generator trained with Eq. 3 is expected to generalise better than trained under Eq. 2, which means using the dark knowledge can facilitate generator distillation. Although both Eq. 2 and Eq. 3 are unbiased estimates of $R(f)$, the variances of Eq. 2 and Eq. 3 over the observed training dataset D are different. This is formally demonstrated by Theorem 1 (proof in the supplementary).

Theorem 1. Let D be a training dataset sampled from \mathbb{P}^N . \mathbb{V} represents the variance of a random variable. For any fixed hypothesis $f : \mathcal{X} \rightarrow \mathbb{R}^{I \times J \times C}$, $\mathbb{V}_{D \sim \mathbb{P}^N}[\hat{R}_\alpha(f; D)] \leq \mathbb{V}_{D \sim \mathbb{P}^N}[\hat{R}(f; D)]$

The two variances in Theorem 1 equal to each other when $p^{ij}(x)$ is concentrated on one single color and the probability on all other colors is zero. However, this case rarely happens. The benefit of having lower variance is that the student f generalises better as shown in the following. Applying Theorem 6 of [14], we have the following bound for the distillation empirical risk with dark knowledge.

Proposition 1. *Let $D \sim \mathbb{P}^N$ and \mathcal{F} be a class of hypotheses $f : \mathcal{X} \rightarrow \mathbb{R}^{I \times J \times C}$ with induced class $\mathcal{H} \subset [0, 1]^{\mathcal{X}}$ of $h(x) = \sum_i \sum_j p^{ij}(x)^{\top} \ell^{ij}((f^{ij}(x))$. Suppose \mathcal{H} has uniform covering number \mathcal{N}_{∞} . For any $\delta \in (0, 1)$ and set $\mathcal{M}_N = \mathcal{N}_{\infty}(1/N, \mathcal{H}, 2N)$. Then with probability at least $1 - \delta$ over D we have:*

$$R(f) \leq \hat{R}_{\alpha}(f; D) + \mathcal{O}\left(\sqrt{\mathbb{V}_N(f)/N} \cdot \sqrt{\log(\mathcal{M}_N/\delta)} + \log(\mathcal{M}_N/\delta)/N\right)$$

where $\mathbb{V}_N(f)$ is the empirical variance of $\{h(x_n)\}_{n=1}^N$.

Note that using the same procedure we can derive a similar bound for the distillation empirical risk with synthetic images (Eq. 2). Considering Theorem 1 and the two bounds, we know that the bound for Eq. 3 has lower variance penalty. Increasing the training dataset size N (number of queries on \mathcal{T}), Eq. 3 has a risk bound with a better rate of convergence. Thus, a student model trained by minimizing the distillation empirical risk with dark knowledge generalises better compared to one using the distillation empirical risk with synthetic images. We conclude that a student generator f trained by the dark knowledge $\mathbb{P}(y|x)$ is better than using sole synthetic image outputs $\{y\}$ from \mathcal{T} .

3.4 Impact of Probability Approximation

In the aforementioned analysis, we showed that having the dark knowledge $\mathbb{P}(y|x)$ from the teacher \mathcal{T} allows for a more precise empirical approximation of $R(f)$. Thus it benefits the training of the student f in generator distillation. However, as mentioned before, doing distillation with dark knowledge requires the conditional probability $\mathbb{P}(y|x)$ for any given input x . In ideal cases, the intermediate layer output of \mathcal{T} , e.g., the second last layer output in VAE, can provide $\mathbb{P}(y|x)$. Then, it can be used to train the student model. Unfortunately, for some teacher models \mathcal{T} , e.g., GANs, the intermediate layer output cannot directly show $\mathbb{P}(y|x)$. We refer to the generators that can provide $\mathbb{P}(y|x)$ in the intermediate layer output as *probabilistic generators* and the generators that cannot show the probability as *non-probabilistic generators*. To distill the dark knowledge from non-probabilistic generators, it is required to approximate $\mathbb{P}(y|x)$. Let $\tilde{\mathbb{P}}(y|x)$ be an approximated probability of $\mathbb{P}(y|x)$. In this section, we study the impact on the student generalization error using such an approximated distribution to do distillation. Using the approximated probability to train the student f , referring to Eq. 3 we have the following distillation empirical risk:

$$\tilde{R}_{\alpha}(f, D) = \frac{1}{N} \sum_{n \in [N]} \sum_i \sum_j \tilde{p}^{ij}(x_n)^{\top} \ell^{ij}((f^{ij}(x_n))), \quad (5)$$

where $\tilde{p}^{ij}(x_n)$ is the approximation of $p^{ij}(x_n)$. According to Eq. 1 and Eq. 4 we can rewrite the population risk $R(f)$ as:

$$R(f) = \mathbb{E}_x[\mathbb{E}_{y|x}[\ell(y, f(x))]] = \mathbb{E}_x[\sum_i \sum_j p^{ij}(x)^\top \ell^{ij}(f^{ij}(x))]. \quad (6)$$

The following Proposition 2 (proof in the supplementary) reveals the connection between the approximation of $\mathbb{P}(y|x)$ and the generalization error of f .

Proposition 2. *If the loss ℓ is bounded, we train the student model by minimizing the empirical risk shown in Eq. 5. For any hypothesis $f : \mathcal{X} \rightarrow \mathbb{R}^{I \times J \times C}$,*

$$\begin{aligned} \mathbb{E}\left[(\tilde{R}_\alpha(f; D) - R(f))^2\right] &\leq \frac{1}{N} \cdot \mathbb{V}\left[\sum_i \sum_j \tilde{p}^{ij}(x)^\top \ell^{ij}((f^{ij}(x))]\right. \\ &\quad \left. + \mathcal{O}(\mathbb{E}[\sum_i \sum_j \|\tilde{p}^{ij}(x) - p^{ij}(x)\|_2])^2.\right] \end{aligned} \quad (7)$$

On the right side of Eq. (7), when N is big, the second term dominates the upper bound of the gap $\tilde{R}(f; D) - R(f)$. That means minimizing the distance between the approximated probability $\tilde{\mathbb{P}}(y|x)$ and the ground truth probability $\mathbb{P}(y|x)$ yields a tighter upper bound of the risk gap $\tilde{R}(f; D) - R(f)$. Hence, a tighter approximation leads to a better student model.

4 DKtill: Extracting Dark Knowledge for Training Student Generator

In the previous section, we theoretically demonstrate that using the underlying dark knowledge of a teacher \mathcal{T} can improve the generator distillation and train a student f that generalizes better. However, to make use of the underlying dark knowledge in distillation, we first need to know how to extract and use the dark knowledge from a teacher generator \mathcal{T} . In this section, we propose two methods to extract the dark knowledge based on the class of the generator: one for probabilistic generators and one for non-probabilistic generators. For verifying the effectiveness of the extracted dark knowledge, we propose a generator distillation algorithm **DKtill**. When distilling \mathcal{T} , **DKtill** makes the student f to mimic the extracted (approximated) $\mathbb{P}(y|x)$ besides the synthetic images $y = \mathcal{T}(x)$ (see the supplementary for more implementation details).

4.1 Extracting from Probabilistic Generators

Probabilistic generators, e.g., variational auto-encoder (VAE) [9], commonly assume the existence of latent variables. More in detail, they assume that a synthetic image y is generated by some random process involving some latent random variables. In such generators, to do distillation with dark knowledge, the conditional probability $\mathbb{P}(y|x)$ can be calculated by some middle layer outputs of the teacher generator. In the following, we take VAE as an example for dark knowledge extraction in probabilistic generators. The training method of VAE is

a kind of variational inference that uses $q(x|y)$ to approximate the intractable true posterior $p(x|y)$ and maximize the evidence lower bound. For VAE distillation, we focus on its trained decoder. In the decoder, the synthetic image is produced by the conditional distribution $\mathbb{P}(y|x) = \mathcal{N}(y; \mu, \sigma^2 I)$, where the distribution parameters μ and σ are the middle layer outputs of the decoder neural network. Thus, given any input x , we can obtain $\mathbb{P}(y|x)$ by getting the output value of μ and σ from the middle layers.

4.2 Extracting from Non-probabilistic Generators

Non-probabilistic generators, e.g., GANs, do not assume any probability relationship between output image y and latent random variables. y is directly produced by a neural network mapping $\mathcal{T}(x)$, where \mathcal{T} is the teacher generator. Different from probabilistic generators, in non-probabilistic generators we cannot derive the knowledge of $\mathbb{P}(y|x)$ by some middle layer outputs of \mathcal{T} . In the following, we take GANs as an example to show extracting dark knowledge $\mathbb{P}(y|x)$ from non-probabilistic generators. Given a GANs teacher generator \mathcal{T} , we let g_α be the second last layer and g_β be the last layer of \mathcal{T} . Given an input x into the teacher generator, the intermediate output of g_α is represented by α . Then the corresponding synthetic image can be represented as $y = g_\beta(\alpha)$. To get the dark knowledge, we can apply a differentiable probabilistic network (e.g., MLP) g_γ to replace the original last layer g_β . The input of g_γ is g_α 's output, α and the output of g_γ is a tensor that takes the shape corresponding to the underlying Bayes probabilities. The shape of $g_\gamma(\alpha)$ is decided by I , J and C where I and J are decided by the synthetic image size and C depends on the color range. The final synthetic image is now sampled from the tensor $g_\gamma(\alpha)$. If \mathcal{T} is untrained, we can train it from scratch using this new architecture. Thus, after the training, we can easily get $\mathbb{P}(y|x)$ from $g_\gamma(\alpha)$. If the given \mathcal{T} is pre-trained, we can just fine-tune the parameters of g_γ using the optimization objective, $\text{argmin}_{g_\gamma} \|g_\beta(\alpha) - \hat{y}\|_1$, where $\hat{y} \sim \text{Discrete}(g_\gamma(\alpha))$ is the synthetic image sampled from $g_\gamma(\alpha)$.

5 Empirical Illustration

5.1 Setting

Datasets: We consider the datasets, Facades [5] and CelebA [12]. In the case of conditional generation, we focus on the gender class, i.e., male and female, in the CelebA dataset. We downsampled the CelebA images from originally 178×218 to 64×64 pixels. The input images size of Facades is 256×256 .

Networks: The network structures of all generators are based on convolutional neural networks [9, 17]. For VAE, we compress a 27.2 MB teacher into a 2.4 MB student. As for conditional GANs, a 2.8 MB student is distilled from a 14.4 MB teacher. In the image translation experiment, Pix2pix is compressed from 217.8 MB into 14.0 MB.

Distillation Process and Baseline: For both DKtill and the baseline (abbreviated as “image” [3]), we train the student generators, by minimizing the distance of

the generated images to the teacher from the same random inputs (details in Appendix). However, instead of only using generated images (baseline), **DKtill** also uses in parallel the information of dark knowledge (underlying distribution) for loss minimization. Note that although the baseline distillation is originally designed for Image Translation, we adopt it for VAE and conditional GANs.

Evaluation Metrics: we use Fréchet Inception Distance (FID) to evaluate the quality of the images produced by the distilled student model. Lower values of FID indicate higher quality of generated data.

5.2 Distilling Probabilistic Generators

Here, we distill VAE (the teacher generator) using the baseline and **DKtill**.

Figure 1 shows the comparison results based on FID for VAE and conditional GANs (Fig. 1a). We also illustrate the FID of images generated by the teacher VAE, i.e., the horizontal line in Fig. 1a. The FID value decreases during the training, from 30.61 to 28.76, and 22.61 to 16.49, for

baseline and **DKtill** respectively, demonstrating lower distance to ground truth images. When looking at the final FID (after 128K examples), **DKtill** is 42.66% lower than the baseline image and 62.14% higher than the teacher generator. Here FID for the teacher VAE is 10.17, which is much better than both students. This is within our expectation as the teacher model is 10X the size of the students. In general, we can see that **DKtill**, using the dark knowledge provided by the intermediate output of teacher VAE, can significantly improve the quality of distillation. As the number of examples increases, the student learns the teacher’s mapping and distribution better. Using dark knowledge can achieve lower FID and thus it can distill better.

5.3 Distilling Non-probabilistic Generators

Here we implement two tasks for non-probabilistic generators: the conditional GANs and image translation as the baseline [3]. Figure 1b evaluates the distillation on conditional GANs. We can see that FID shares the same trend as distilling the probabilistic generator VAE. Thus, our proposed method

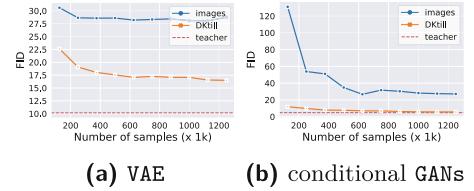


Fig. 1. FID of distilling VAE and conditional GANs on CelebA.

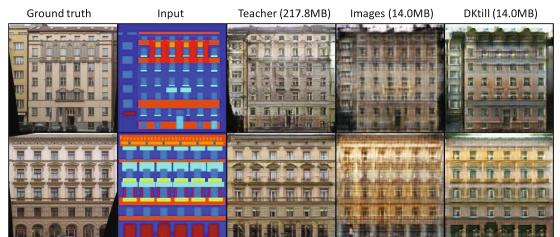


Fig. 2. Pix2pix image translation GANs on Facades.

DKtill is able to effectively extract knowledge from non-probabilistic teacher generators too, given the approximate nature of dark knowledge. The reason why FID of **DKtill** gets much closer to the teacher than Fig. 1a is that the teacher model is only 5X bigger in parameter size than the students. When looking at the final FID (after 128K examples), **DKtill** is 78.99% lower than the baseline image and only 17.77% higher than the teacher generator.

Let us zoom into the distillation results of the image translation task in Fig. 2. The goal is to train the translation GANs so that given input (as the “input” in visualization), the Pix2pix network approximately maps it into the ground truth image. From the results, we observe that even with 6% size of the teacher model (FID: 35), **DKtill** is able to distill the image translation generator at high quality (FID: 35) with dark knowledge, especially compared with the baseline (FID: 37.88). By distillation, the floating point operations per second (FLOPs) on one input image is reduced from 6.06G (Teacher) to 0.83G (Student).

5.4 Small Generators Through **DKtill**

In addition to achieving FID similar to the teacher model, we show another advantage of exploring dark knowledge, i.e., smaller student generator. We first solicit the synthetic images generated by the following model in Fig. 3: the ground truth, teacher VAE, 27.2 MB student VAE trained by baseline, 27.2 MB student VAE from **DKtill**, and 2.4 MB student VAE from **DKtill**. Without any surprise, **DKtill** achieves better distillation quality when using bigger networks, comparing Fig. 3d and Fig. 3e. Another observation is that, smaller student (2.4 MB) VAE trained by **DKtill**, achieve better image quality than baseline with 27.2 MB, comparing Fig. 3c and Fig. 3e.

The conditional GANs result is also presented in Fig. 4 (on gender class, i.e., male and female for the left 3 and right 3 pictures). Note that here we do not show the corresponding real images of the synthetic ones since conditional GANs do not have a latent code encoder as VAE. Thus, having the latent code of a real image for reproducing some corresponding synthetic ones is difficult. Given the same small network with 2.8 MB for **DKtill**

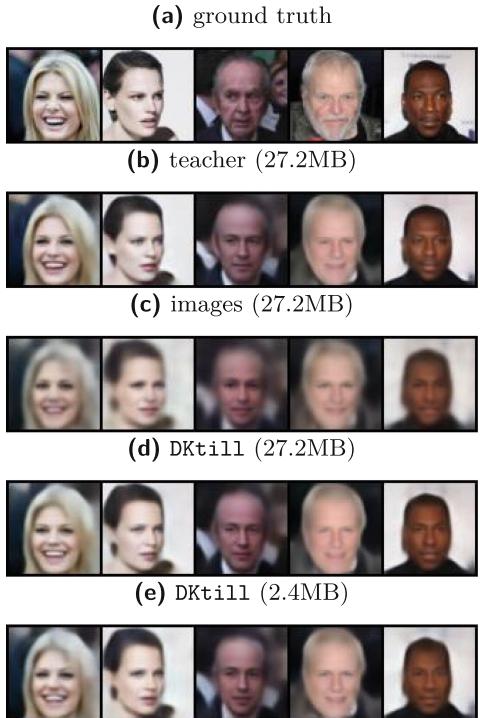


Fig. 3. VAE on CelebA.

and the baseline, **DKtill** shows better generated image quality than baseline. These results again prove the existence of dark knowledge for generators and its benefit for distilling generators.

6 Related Work

Knowledge distillation [2, 4, 6, 8, 11], which transfers the knowledge from a big network to a small one, enables the light-weight deep learning. Generally, knowledge distillation is studied on classifiers for model compression, e.g., [4] compresses the knowledge of an ensemble into a single model which is much easier to deploy. Such techniques are used to train surrogate models [10, 18, 22], even without the necessity of knowing the target model parameters.

Besides, knowledge distillation has also been applied on **GANs** [1, 3, 19, 19]. KDGAN [19] simultaneously optimizes the distillation and adversarial losses between a classifier, a teacher, and a discriminator, to learn the true data distribution at the equilibrium. To further improve the performance, [3] include a student discriminator to measure the distances between real data, and the synthetic data generated by student and teacher generators. Recently, there are theoretical analyses on knowledge distillation for classifiers [7, 13, 15, 16, 20, 21]. On the one hand, [15] first provides the theoretical analysis of self-distillation, fitting a nonlinear function on Hilbert space and L2 regularization. This analysis sheds light on the relation between self-distillation rounds and (under-)over-fitting. On the other hand, based on neural tangent kernel, [7] provides a transfer risk bound for the linearized model of the wide neural networks, revealing the impact of soft (hard) labels for (im)perfect teachers according to the designed data inefficiency metric. Furthermore, [21] explores the bias-variance trade-off brought by distillation with soft labels. According to their analysis, novel weighted soft labels are inspired to help the network adaptively handle the trade-off. However, none of the existing work provides a generalization analysis on generators.

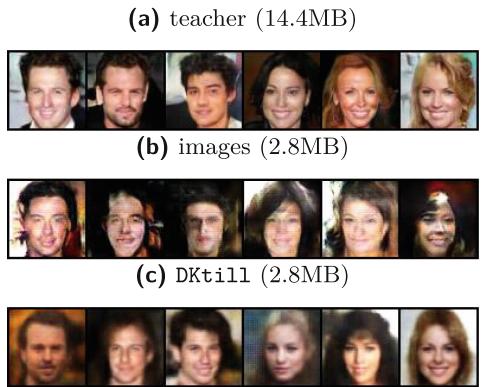


Fig. 4. Conditional **GANs** on **CelebA**.

7 Conclusion

In this paper, we model the knowledge distillation for generative models from a Bayesian perspective, identifying dark knowledge and its influence on the generalization ability of student models, i.e., lower empirical risk. Furthermore, we

propose a dark knowledge based distillation optimization, **DKtill**, to train student generators on both non-probability-based and probability-based generative models. Evaluation results on three datasets across different scenarios show that synthetic images from **DKtill** achieve lower FID by up to 78.99%, in contrast to using images only. **DKtill** also generates images of similar quality as the teacher model, using smaller and more compact generator networks.

Acknowledgements. This work has been supported by the Spoke “FutureHPC & BigData” of the ICSC - Centro Nazionale di Ricerca in “High Performance Computing, Big Data and Quantum Computing”, funded by EU - NextGenerationEU and the EPI project funded by EuroHPC JU under G.A. 101036168.

References

1. Aguinaldo, A., Chiang, P., Gain, A., Patil, A., Pearson, K., Feizi, S.: Compressing GANs using knowledge distillation. CoRR **abs/1902.00159** (2019)
2. Chandrasekaran, V., Chaudhuri, K., Giacomelli, I., Jha, S., Yan, S.: Exploring connections between active learning and model extraction. In: USENIX Security (2020)
3. Chen, H., et al.: Distilling portable generative adversarial networks for image translation. In: AAAI (2020)
4. Hinton, G.E., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. CoRR **abs/1503.02531** (2015)
5. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: CVPR, pp. 1125–1134 (2017)
6. Jagielski, M., Carlini, N., Berthelot, D., Kurakin, A., Papernot, N.: High accuracy and high fidelity extraction of neural networks. In: USENIX Security (2020)
7. Ji, G., Zhu, Z.: Knowledge distillation in wide neural networks: risk bound, data efficiency and imperfect teacher. In: NeurIPS 2020 (2020)
8. Kanwal, N., Eftestøl, T., Khoramnia, F., Zuiverloon, T.C., Engan, K.: Vision transformers for small histological datasets learned through knowledge distillation. In: PAKDD (2023)
9. Kingma, D.P., Welling, M.: Auto-encoding variational Bayes. In: Bengio, Y., LeCun, Y. (eds.) ICLR (2014)
10. Krishna, K., Tomar, G.S., Parikh, A.P., Papernot, N., Iyyer, M.: Thieves on sesame street! Model extraction of BERT-based APIs. In: ICLR (2020)
11. Liu, Z., Zhu, Y., Gao, Z., Sheng, X., Xu, L.: Itreievalkd: an iterative retrieval framework assisted with knowledge distillation for noisy text-to-image retrieval. In: PAKDD (2023)
12. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: (ICCV) (2015)
13. Lopez-Paz, D., Bottou, L., Schölkopf, B., Vapnik, V.: Unifying distillation and privileged information. In: Bengio, Y., LeCun, Y. (eds.) ICLR (2016)
14. Maurer, A., Pontil, M.: Empirical Bernstein bounds and sample variance penalization. In: COLT 2009 - The 22nd Conference on Learning Theory (2009)
15. Mobahi, H., Farajtabar, M., Bartlett, P.L.: Self-distillation amplifies regularization in Hilbert space. In: NeurIPS (2020)
16. Phuong, M., Lampert, C.H.: Towards understanding knowledge distillation. CoRR **abs/2105.13093** (2021)

17. Pu, Y., Gan, Z., Henao, R., Yuan, X., Li, C., Stevens, A., Carin, L.: Variational autoencoder for deep learning of images, labels and captions. In: NIPS **29** (2016)
18. Truong, J., Maini, P., Walls, R.J., Papernot, N.: Data-free model extraction. In: CVPR (2021)
19. Wang, X., Zhang, R., Sun, Y., Qi, J.: KDGAN: knowledge distillation with generative adversarial networks. In: NeurIPS, pp. 783–794 (2018)
20. Zhang, Z., Sabuncu, M.R.: Self-distillation as instance-specific label smoothing. In: NeurIPS (2020)
21. Zhou, H., et al.: Rethinking soft labels for knowledge distillation: a bias-variance tradeoff perspective. In: 9th International Conference on Learning Representations, ICLR 2021 (2021)
22. Zhou, M., Wu, J., Liu, Y., Liu, S., Zhu, C.: Dast: data-free substitute training for adversarial attacks. In: CVPR, pp. 231–240. IEEE (2020)



RPH-PGD: Randomly Projected Hessian for Perturbed Gradient Descent

Chi-Chang Li, Jay Huang, Wing-Kai Hon and Che-Rung Lee

National Tsing Hua University, Hsinchu, Taiwan
`{wkhon,cherung}@cs.nthu.edu.tw`

Abstract. The perturbed gradient descent (PGD) method, which adds random noises in the search directions, has been widely used in solving large-scale optimization problems, owing to its capability to escape from saddle points. However, it is inefficient sometimes for two reasons. First, the random noises may not point to a descent direction, so PGD may still stagnate around saddle points. Second, the size of random noises, which is controlled by the radius of the perturbation ball, may not be properly configured, so the convergence is slow. In this paper, we proposed a method, called RPH-PGD (Randomly Projected Hessian for Perturbed Gradient Descent), to improve the performance of PGD. The randomly projected Hessian (RPH) is created by projecting the Hessian matrix into a relatively small subspace which contains rich information about the eigenvectors of the original Hessian matrix. RPH-PGD utilizes the eigenvalues and eigenvectors of the randomly projected Hessian to identify the negative curvatures and uses the matrix itself to estimate the changes of Hessian matrices, which is necessary information for dynamically adjusting the radius during the computation. In addition, RPH-PGD employs the finite difference method to approximate the product of the Hessian and vectors, instead of constructing the Hessian explicitly. The amortized analysis shows the time complexity of RPH-PGD is only slightly higher than that of PGD. The experimental results show RPH-PGD does not only converge faster than PGD, but also converges in cases that PGD cannot.

Keywords: Optimization · Gradient Descent · Projected Hessian

1 Introduction

The gradient descent is an iterative method that models the objective function as a hyperplane and searches along the steepest descent direction from one point to another. Although it is a kind of greedy strategy, and can only converge to a local minimum for non-convex problems, empirically it performs well and has been widely used in solving large-scale optimization problems. Because of its simplicity and effectiveness, many commonly used methods for machine learning or deep learning, such as stochastic gradient descent (SGD) [14], AdaGrad [6], and Adam [11], are based on its principle and extend with different variations.

However, gradient descent based methods can mis-converge to some non-optimal points, called saddle points, which appear often in high-dimensional and non-convex optimization problems. A saddle point is a first-order stationary point, whose gradient is equal to zero, but the eigenvalues of its Hessian are mixed of different signs, which means it is not an extremum. Theoretically, if the gradient descent method converges to a saddle point, it will get stuck and cannot escape from the saddle point.

Several methods [1, 3, 4] have been proposed to solve the saddle point problem for gradient descent. The major challenge is that the proposed method needs to keep the simplicity and effectiveness of gradient descent. Perturbed Gradient Descent (PGD) method [7] that adds noises in the search directions is one of the promising solutions to escape from the saddle points. Theoretically, Chi Jin et al. [10] have shown that the perturbed gradient descent can converge to local minima in a number of iterations that is almost dimension-free.

Nevertheless, PGD is inefficient sometimes for two reasons. First, the added random noises may not point to a descent direction, so it cannot escape from the saddle points efficiently. Second, both the analysis and algorithms of PGD rely on the understanding of the local geometry of underlying functions, which can be characterized by the eigenvalues of the Hessian matrix. One of the critical hyper-parameters is the radius r of perturbation ball [10], from which the perturbations are sampled. If r is too small, it cannot escape from saddle points; if r is too large, it may not converge. Although the analysis suggests some proper values of the radius r [10], the computation of r is too expensive to perform in practice. Moreover, the value of radius r is changing during the computation as the gradient descent moves to different regions of the function. Finding one constant value of r for the entire optimization process is mostly impossible to satisfy different situations.

The major challenge is how to obtain a good estimation of the Hessian without paying a high computation cost. The exact computation of Hessian matrix takes $O(Tn^2)$ time and $O(n^2)$ space, where n is the number of variables and T is the time for function evaluation. The cost of eigenvalue decomposition is $O(n^3)$. For large-scale problems, such time/space complexity is too high to use for real-world applications. A linear time algorithm is usually required for solving large-scale problems.

In this paper, we present an algorithm, called RPH-PGD (Randomly Projected Hessian for PGD), to improve the convergence of PGD. The projected Hessian is a much smaller matrix compared to the original Hessian. It is created by projecting the Hessian into a small subspace, which may contain rich information about the eigenvectors of the matrix. From the RPH, one can compute the approximate eigenvalues/eigenvectors which can be used to explore the negative curvatures, the descent directions at the saddle points. The RPH can also be used to adjust the radius r of the perturbation ball, which is a hyper-parameter related to the changes of Hessian matrices. In addition, RPH-PGD employs the finite difference method to approximate the product of the Hessian and vectors, instead of constructing the Hessian explicitly. The experimental results show

RPH-PGD does not only converge faster than PGD, but also converges in cases that PGD cannot. The amortized analysis shows the time complexity of RPH-PGD is only slightly higher than that of PGD.

2 Preliminary

2.1 Notation

The notation, definitions, and assumptions of the paper are summarized here. Most of them are the same as those in [8, 10]. We use uppercase letters for matrices and lowercase letters for vectors and scalars. We use $\|\cdot\|$ to denote the Euclidean norm for vectors and matrices, and $\|\cdot\|_F$ for Frobenius norm for matrices. For a matrix A , we use A^* to represent A 's conjugate transpose.

Let the objective function be f . The gradient is ∇f and the Hessian matrix is $\nabla^2 f$. We will use g as an approximation to ∇f and use H for $\nabla^2 f$. We use I for identity matrix and o for zero vector. For example, $\mathcal{N}(o, I)$ means the normal distribution with zero mean and identity covariance.

In this paper, we assume the optimization problem to solve is a minimization problem. We assume the Hessian matrix in problems is symmetric/Hermitian, which is true for most of the cases. Moreover, we assume the function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is twice-differentiable. The function f is called ℓ -gradient Lipschitz smooth :

$$\forall x_1, x_2 \in \mathbb{R}^n, \|\nabla f(x_1) - \nabla f(x_2)\| \leq \ell \|x_1 - x_2\| \quad (1)$$

Moreover, f is ρ -Hessian Lipschitz smooth if

$$\forall x_1, x_2 \in \mathbb{R}^n, \|\nabla^2 f(x_1) - \nabla^2 f(x_2)\| \leq \rho \|x_1 - x_2\| \quad (2)$$

For the domain, we also define some commonly used terms. First, $x \in \mathbb{R}^n$ is called an ϵ -first-order stationary point if $\|\nabla f(x)\| \leq \epsilon$. Second, for a stationary point x , we called x is a δ -approximate local minimum if

$$\exists \delta > 0 : f(x) \leq f(y), \forall y : \|y - x\| \leq \delta; \quad (3)$$

x is a δ -approximate local maximum:

$$\exists \delta > 0 : f(x) \geq f(y), \forall y : \|y - x\| \leq \delta; \quad (4)$$

Otherwise, x is a saddle point.

Last are the definitions of the search directions. At a function point x , a vector d points to a *descent direction* if

$$d^T \nabla f(x) < 0. \quad (5)$$

Furthermore, d is in the direction of *negative curvature* if

$$d^T \nabla^2 f(x) d < 0. \quad (6)$$

For a saddle point, because $\nabla f(x) = 0$, there is no descent direction. However, because $\nabla^2 f(x)$ is indefinite, which means its eigenvalues are mixed of different signs, the eigenvectors corresponding to the negative eigenvalues, as well as their linear combinations, are in the directions of negative curvature.

2.2 Methods to Escape from Saddle Points

There are many studies proposing methods to solve the saddle point problems. In [7], authors showed that stochastic gradient descent (SGD) can converge to a local minimum even with exponentially many local minima and saddle points. In [12], authors indicate the normalized gradient descent with noises can even converge faster. Chi Jin [10] et al. gave a comprehensive study on the convergence of PGD.

Many algorithms also leverage the second-order derivative to escape the saddle points. In [1, 4], the authors used the method of approximating the Hessian-vector product. The approximation is based on the first-order Taylor expanding for multi-variable functions,

$$\nabla f(x + \delta d) \approx \nabla f(x) + \delta \nabla^2 f(x)d,$$

where ∇f is the gradient, $\nabla^2 f$ is the Hessian, d is a vector, and δ is a small non-zero scalar. Therefore, the Hessian vector product can be approximated by

$$\nabla^2 f(x)d \approx \frac{\nabla f(x + \delta d) - \nabla f(x)}{\delta}. \quad (7)$$

The same result can also be obtained from the finite difference method. Equation (7) shows that the computational cost of estimating the Hessian vector product is merely two gradient evaluations, which is as cheap as computing stochastic gradients. [2]

2.3 Perturbed Gradient Descent

In [10], Chi Jin et al. provide an analysis of the perturbed gradient descent method. The key idea is to regard all possible positions of x after perturbation as a ball. If the radius r is chosen correctly, it can be proven that the area where x gets stuck is very small compared to the whole sphere. Thus, it is of high probability to escape from saddle points.

Algorithm 1 sketches the perturbed gradient method (PGD), in which d is the dimension of the objective function. The perturbation ξ is sampled from a zero-mean Gaussian with covariance matrix $(r^2/n)I$ so that the expectation of the perturbation equals to r , $\mathbb{E}\|\xi\|^2 = r^2$. The hyper-parameter r is called the radius of the perturbation ball. In [10], authors show with a properly configured radius, the convergence of PGD can be as fast as gradient descent. The suggested formula is

$$r \leftarrow \epsilon \sqrt{R} \quad (8)$$

Input: initial guess x_0 , step size η , dimension n , perturbation radius r .

Output: a local minimum x^* ;

for $t = 0, 1, \dots, \tau - 1$ **do**

if x_t is trapped into a saddle point **then**

sample ξ_t from $\mathcal{N}(0, (r^2/n)I)$

$x_{t+1} \leftarrow x_t - \eta \xi_t$

end

$x_{t+1} \leftarrow x_t - \eta \nabla f(x_t)$.

end

return x_τ ;

Algorithm 1: Perturbed Gradient Descent (PGD)

where

$$R = 1 + \min\left(1/\epsilon^2 + \ell/\sqrt{\rho\epsilon}, n/\epsilon^2\right). \quad (9)$$

The value ϵ is the tolerance; The variable ℓ is used to define the ℓ -gradient Lipschitz smooth, as show in (1); ρ is for the ρ -Hessian Lipschitz smooth, defined in (2); and n is the dimension.

The perturbation only needs to be added to the search direction when x is trapped to a saddle point. In [10], authors evaluate this condition using a heuristic, which is if $\|\nabla f\| \leq \epsilon$ for some number of iterations t_{inter} , then x might get stuck into a saddle point. However, such strategy may fail sometimes because the added random noise may not point to a descent direction. PGD may wander around the saddle point and cause stagnation.

The advantage of this method is that it is simple and cheap, so it can be combined with any other algorithm. The disadvantage is that the radius r is an additional hyper-parameter, whose theoretical value is defined in (8) and (9). In practice, it is not easy to choose a proper r . Moreover, the radius is changing as the method moves to different regions of the function. So giving a constant number of r cannot fit all the needs during the computation.

3 Algorithms

In this section, we first introduce the algorithm for computing the randomly projected Hessian (RPH), followed by the main algorithm of RPH-PGD.

3.1 Randomly Projected Hessian

Let Q be an orthogonal basis for subspace, and A be a Hessian matrix. The projected Hessian is in the form Q^*AQ , which is also called the Rayleigh quotient. The projected Hessian has been widely used in the constrained optimization problem [13] to avoid the steps moving into the tangent cone of constraints. The idea is to project the Hessian matrix into the null space spanning by the gradients of active constraints. If the subspace is of full column rank, the projected Hessian can still keep its definiteness. Moreover, the dimension of the projected Hessian will be reduced to the dimension of the subspace.

Input: A function g to compute ∇f ;
 the dimension of subspace k .
Output: H : projected Hessian;
 Q : orthogonal basis for the generated subspace.

Generate an $n \times k$ Gaussian random matrix Ω ;

Approximate $W \leftarrow \nabla^2 f \Omega$ using g and (7);

Compute the QR decomposition of W , $QR \leftarrow W$;

Compute $H \leftarrow R(Q^* \Omega)^{-1}$;

return H, Q

Algorithm 2: Randomly Projected Hessian (RPH)

Here we want to generate a projected Hessian that has similar spectrum as the original Hessian. Therefore, the subspace must contain good approximations to the eigenvectors of the original Hessian. Therefore, we will leverage the randomized eigen-decomposition (RED) [8] to construct the subspace and the projected Hessian.

Algorithm 2 shows the procedure of computing the randomly projected Hessian. As can be seen, it is similar to the RED, except in two places. First, for matrix-matrix multiplication, RPH uses the finite difference method, as shown in (7), to approximate the matrix-vector multiplication. Hessian matrix is the second-order derivative of the objective function. Therefore, without forming the Hessian matrix explicitly, one can use the directional derivative of the gradient to approximate the Hessian-vector multiplication.

Second, the procedure does not compute the eigenvalue and eigenvector approximations. It stops at the generation of the projected Hessian. In fact, the small matrix H is not exactly a projected Hessian. Let $A = \nabla^2 f$ be the Hessian matrix. From Step 4 in Algorithm 2, if R and A are non-singular, and $W = A\Omega$, one has

$$H = R(Q^* \Omega)^{-1} = R(Q^* A^{-1} W)^{-1} = R(Q^* A^{-1} Q R)^{-1} = (Q^* A^{-1} Q)^{-1}. \quad (10)$$

Thus, H is the inverse of the Rayleigh quotient of A^{-1} , not A . Theoretically, such Rayleigh quotient produces better approximations to the eigenvalues of small magnitude [9], because the small eigenvalues in A are usually clustered and they will be well separated in A^{-1} . The well separation of eigenvalues implies better conditions for computing eigenvectors. In addition, the matrix Q , whose column vectors form an orthogonal basis for the product $A\Omega$, should contain good approximations to the eigenvectors of A . Combining those two factors, H is a good surrogate of the Hessian in terms of eigenvalues/eigenvectors.

From (7), one can see the computation cost equals one more gradient evaluation, $\nabla f(x + \delta d)$, because $\nabla f(x)$ is already available. Since there are k Hessian vector multiplications, the total computational cost of Step 2 is $O(kT)$, where T is the time for evaluating gradients. The time complexity of Algorithm 2 is

$$O(kT + nk^2 + k^3). \quad (11)$$

If $n \gg k$ and $T = O(n)$, the time complexity of Algorithm 2 is $O(n)$.

Input: A function g to compute ∇f ;
 the dimension of subspace k .
 maximum eigenvalue in the previous iteration λ' .
Output: H : projected Hessian of $(A - \lambda' I)$;
 Q : orthogonal basis for the generated subspace.
 Generate an $n \times k$ Gaussian random matrix Ω ;
 Approximate $W \leftarrow (\nabla^2 f - \lambda' I)\Omega$ using g and (7);
 Compute the QR decomposition of W , $QR \leftarrow W$;
 Compute $H \leftarrow R(Q^* \Omega)^{-1}$;
return H, Q

Algorithm 3: Shifted Randomly Projected Hessian (SRPH)

3.2 Shifted Randomly Projected Hessian

A small subspace will make computation much faster, but the descent direction may not be sampled every time the projected Hessian is calculated. Algorithm 3 shows a solution that shifts the diagonal elements of ∇f by a constant λ' . It can be seen that Algorithm 2 is a special case of Algorithm 3, in which $\lambda' = 0$. By using Algorithm 3, we can ensure that the vectors in W contains better approximations to the eigenvectors of A 's smallest eigenvalues, since now their magnitude becomes larger. Note H in Algorithm 3 is the inverse of the projected Hessian to $(A - \lambda' I)^{-1}$, so when computing the eigenvalues of A , we need to add λ' back.

3.3 RPH-PGD

Algorithm 4 gives a high-level description of the RPH-PGD, which is based on the PGD method (Algorithm 1). The major difference between them is when the original PGD sticks in the saddle points for a long time, a new procedure (Step 6 to Step 20) will be triggered. It first computes the shifted randomly projected Hessian (Algorithm 3), and uses it to escape the saddle points more efficiently.

There are two major mechanisms to assist PGD to escape the saddle points. First, RPH-PGD computes its eigenvalue decomposition $H = V\Lambda V^*$, and find the largest eigenvalue λ_{\max} , the smallest eigenvalue λ_{\min} , and the eigenvector v_{\min} corresponding λ_{\min} .

$$Hv_{\min} = \lambda_{\min} v_{\min}.$$

Since we only need the smallest and the largest eigenvalue approximation, the dimension of the subspace k can be small, such as 5 or 10. As a result, the exact eigenvalue decomposition of H is cheap to compute.

To obtain the direction of negative curvature, we need to project v_{\min} back to the space of n . One common way is using the Ritz vector,

$$u_{\min} = Qv_{\min},$$

where Q is the orthogonal basis of generated by RPH. Based on [9], if Q is close to the eigen-space of $\nabla^2 f$, the Ritz vector will give a good eigenvector

```

Input:  $x_0$ : initial guess;  $\tau$ : number of steps;
         $r$ : initial radius;  $\eta$ : step size;  $\epsilon$ : tolerance;
         $t_{\text{inter}}$ : time interval;
         $t_{\text{rp}}$ : random project interval.
Output: minimum point  $x^*$ 
for  $t = 0, 1, \dots, \tau - 1$  do
    if  $\|\nabla f(x_t)\| \leq \epsilon$  more than  $t_{\text{inter}}$  iterations then
        sample  $\xi_t$  from  $\mathcal{N}(0, (r^2/n)I)$ 
         $x_{t+1} \leftarrow x_t - \eta \xi_t$ 
    end
     $\lambda' = 0$ 
    if  $\|\nabla f(x_t)\| \leq \epsilon$  more than  $t_{\text{rp}}$  iterations then
        Compute the shifted randomly projected Hessian  $H$ 
        Compute  $\lambda_{\max}$ ,  $\lambda_{\min}$ , and  $u_{\min}$  of  $H$ 
        Update  $\lambda_{\max} \leftarrow \lambda_{\max} + \lambda'$ ,  $\lambda_{\min} \leftarrow \lambda_{\min} + \lambda'$ ,  $\lambda' \leftarrow \lambda_{\max}$ 
        if  $\lambda_{\min} > \beta$  then
            return  $x_t$ 
        end
        if  $\lambda_{\min} < 0$  then
             $x_{t+1} \leftarrow x_t + \lambda_{\max}^{-1} u_{\min}$ 
        end
        Update the radius  $r$  using  $H$  and  $\nabla f$ 
    end
     $x_{t+1} \leftarrow x_t - \eta \nabla f(x_t)$ .
end
return  $x_\tau$ 

```

Algorithm 4: Randomly Projected Hessian for PGD (RPH-PGD)

approximation to the Hessian. Then, the largest and the smallest eigenvalues need to add back the value of the last largest eigenvalue since H is shifted (Algorithm 3).

Step 13 and Step 14 show that if λ_{\min} is large enough ($\lambda_{\min} > \beta > 0$), which means the Hessian is positive definite, the algorithm reaches an optimal solution, so RPH-PGD stops earlier.

If λ_{\min} is negative (Step 16 and Step 17), it means H is indefinite, or the Hessian is indefinite. We can move along the direction of negative curvature, along which the function value can be decreased. Suppose x_0 is a saddle point. The Taylor expansion of f around x_0 is

$$f(x_0 + d) \approx f(x_0) + d^* \nabla f(x_0) + \frac{1}{2} d^* \nabla^2 f(x_0) d.$$

Since x_0 is a saddle point, $\nabla f(x_0) = 0$. Therefore,

$$f(x_0 + d) \approx f(x_0) + \frac{1}{2} d^* \nabla^2 f(x_0) d. \quad (12)$$

If d is a direction in negative curvature, which means $d^* \nabla^2 f(x_0) d < 0$, the function value at $x_0 + d$ is less than that at x_0 .

The computed u_{\min} is a unit vector by default, so we need to decide a step length. The rule of thumb is that if the function changes rapidly, we should not move too far. The norm of Hessian matrix, which is λ_{\max} here, can help to determine the sensitivity of the functions. If λ_{\max} is large, the step length should be small. Therefore, we use λ_{\max}^{-1} as the step length.

The second method to help PGD is the dynamic update of the radius of the perturbation ball r . The analysis of PGD [10] shows when the choice of r satisfies (8), the efficiency of PGD can be guaranteed. However, the values of ℓ and ρ are intrinsic for the given function f , and are hardly to know in advance. Here we use the changes of gradient and the projected Hessian to estimate the values of ℓ and ρ . Suppose that $x_i \neq x_j$.

$$\ell \leftarrow \frac{\|\nabla f(x_i) - \nabla f(x_j)\|}{\|x_i - x_j\|}, \text{ and } \rho \leftarrow \frac{\|H_i - H_j\|}{\|x_i - x_j\|} \quad (13)$$

where i, j are indices of iterations. Since H is small, the computation of (2) is simple and fast. They are local measurements of the parameter ℓ and ρ . Based on the same proof, if r can be set based on which, the efficiency of PGD can be guaranteed locally.

The remaining problem is how good the estimation is. Let A_i and A_j be the Hessian matrices in the i th iteration and the j th iteration respectively, and H_i and H_j are their randomly projected Hessian matrices. Their relation can be characterized as

$$A_i - Q_i H_i Q_i^* = E_i, \quad A_j - Q_j H_j Q_j^* = E_j$$

where Q_i and Q_j are the orthogonal bases for the randomly generated subspaces, and E_i and E_j represent their differences. Here we only show a special case that $Q_i = Q_j$.

$$\|H_i - H_j\|_F = \|Q_i H_i Q_i^* - Q_j H_j Q_j^*\|_F = \leq \|A_i - A_j\|_F + \|E_i\|_F + \|E_j\|_F$$

The last line is by triangle inequality for norm. Therefore,

$$\|H_i - H_j\|_2 \leq \|H_i - H_j\|_F \leq \|A_i - A_j\|_F + 2\|E\|_F,$$

where $\|E\|_F = \max\{\|E_i\|_F, \|E_j\|_F\}$. The expectation of $\|E\|_F$ is

$$\mathbb{E}\|E\|_F \leq \mathbb{E}(\sqrt{n}\|E\|_2) < \left[\sqrt{n} + \frac{4n\sqrt{k}}{k-2} \right] \sigma_2.$$

The difference between $\|H_i - H_j\|$ and $\|A_i - A_j\|$ may not be too small, because the additional term $\|E\|$ can be large. However, empirically, this method works well.

The time complexity of the additional steps is bounded by the computation of randomly projected Hessian,

$$O(kT + nk^2 + k^3),$$

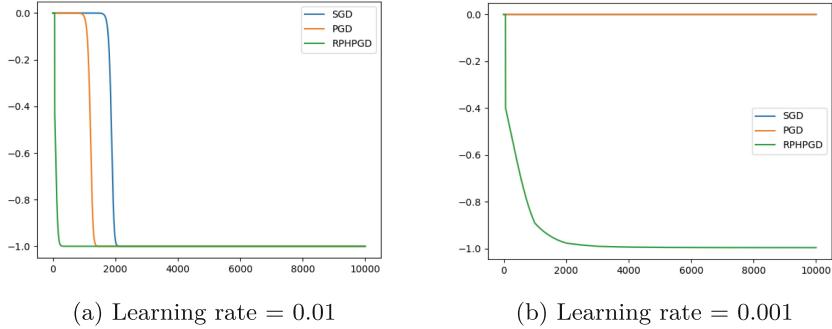


Fig. 1. Convergence of SGD, PGD, and RPH-PGD for function f defined in (14). The x-axis is the number of iterations, and the y-axis is the function value. (a) shows the convergence of three methods for learning rate = 0.01; (b) shows the convergence of three methods for learning rate = 0.001.

seeing the analysis of (11). However, they do not perform often, because t_{rp} is larger than t_{inter} . Suppose that it is performed every k iteration. The amortized time complexity is

$$O(T + nk + k^2).$$

Since the time of computing gradient T dominates the others, the time complexity of RPH-PGD is the same as that of PGD.

4 Experiments

In this section, we evaluate the effectiveness of RPH-PGD, and compare it with other PGD and SGD. We run our experiments on a machine with i5-6300HQ CPU, 8GB 2133MHz RAM, and Windows 10. We used Python 3.9.13 for coding.

The test problem in experiments is

$$f(x_1, x_2) = \frac{1}{16}x_1^4 - \frac{1}{2}x_1^2 + \frac{9}{8}x_2^2 \quad (14)$$

which has a saddle point at $(0, 0)$. The minimum value of this problem is at $(2, 0)$ and $(-2, 0)$, whose value is -1. This is a good benchmark problem for evaluating the ability of optimization methods to escape saddle points [15]. The initial point is at $(0, 0)$.

The convergence of three methods is shown in Fig. 1. We used two learning rates 0.01 and 0.001 to evaluate their convergence, as shown in subfigure (a) and subfigure (b) respectively. As can be seen, when the learning is large, such as 0.01, all three methods can still converge. Although the convergence rates (their slope) of three methods are similar, the convergence of RPH-PGD is much faster than that of SGD and PGD. The major reason is RPH-PGD starts to converge in the beginning, and the convergence of PGD starts much slower than that of

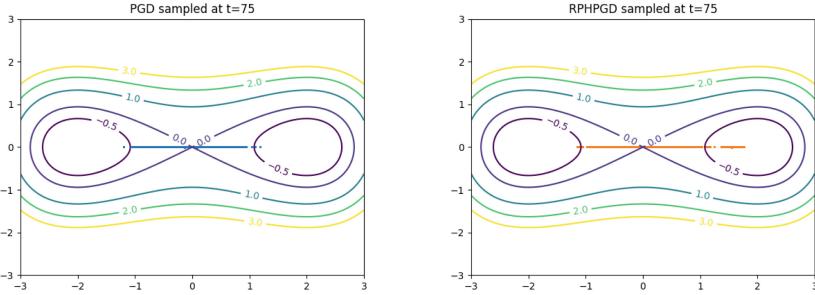


Fig. 2. The traces of PGD (left) and RPH-PGD (right) for function f defined in (14).

RPH-PGD, and the convergence of SDG starts even slower than that of PGD. For SGD, it does not have the ability to detect and avoid the saddle points, and the gradient near $(0, 0)$ is close to 0, so SGD can only move slowly from the saddle point. For PGD, although it can detect the saddle points, and uses perturbation to escape them, its strategy is not strong enough owing to the lack of accurate information about the Hessian matrix.

We can further examine the convergence of PGD and RPH-PGD by comparing their traces in the function contour, as shown in Fig. 2. They show the traces at iteration 75. As can be seen, the traces of PGD show PGD moves slower than RPH-PGD toward to the minimization points.

When the learning rate is small, such as 0.001, SGD and PGD stagnate at the original point. Although the convergence of RPH-PGD also becomes slower, it reaches the minimum points eventually.

5 Conclusion and Future Work

In this paper, we present a new algorithm, called RPH-PGD to improve the convergence of PGD. It uses randomly projected Hessian to provide information about the second-order derivative, which is critical to reveal the properties of local geometry. When PGD fails, RPH-PGD employs the eigenvalues/eigenvectors to explore the negative curvature and utilizes the projected Hessian to adjust the radius of perturbation ball. Experiments show that RPH-PGD can significantly improve the convergence of PGD.

There are several future directions to explore. First, the current implementation of RPH-PGD can work only on some toy problems, as the ones shown in Sect. 4. To run it for some large-scale problems, we need to have a better implementation. Also, we will try it on more real-world applications to demonstrate its power. Second, the randomized projected Hessian algorithm is easier to be parallelized than subspace methods, because the subspace generation and orthogonalization can be done in parallel, such as using TSQR [5]. We want to explore those directions to further sharpen the implementation. Last, although

RPH-PGD performs well empirically, we want to investigate its convergence and stability analytically, which would build a more solid foundation for future directions.

References

1. Agarwal, N., Allen-Zhu, Z., Bullins, B., Hazan, E., Ma, T.: Finding approximate local minima faster than gradient descent. In: Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, pp. 1195–1199. STOC 2017, Association for Computing Machinery, New York, NY, USA (2017)
2. Allen-Zhu, Z.: Natasha 2: faster non-convex optimization than SGD. 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)
3. Allen-Zhu, Z., Li, Y.: NEON2: finding local minima via first-order Oracles. 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)
4. Carmon, Y., Duchi, J.C., Hinder, O., Sidford, A.: Accelerated methods for non-convex optimization. SIAM J. Optim. **28**(2), 1751–1772 (2018)
5. Demmel, J., Grigori, L., Hoemmen, M., Langou, J.: Communication-optimal parallel and sequential QR and LU factorizations. SIAM J. Sci. Comput. **34**(1), A206–A239 (2012)
6. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res. **12**(61), 2121–2159 (2011)
7. Ge, R., Huang, F., Jin, C., Yuan, Y.: Escaping from saddle points: online stochastic gradient for tensor decomposition. J. Mach. Learn. Res. **40**(2015)
8. Halko, N., Martinsson, P.G., Tropp, J.A.: Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. SIAM Rev. **53**(2), 217–288 (2011)
9. Jia, Z., Stewart, G.W.: An analysis of the Rayleigh-Ritz method for approximating Eigenspaces. Math. Comput. **70**, 637–647 (2001)
10. Jin, C., Netrapalli, P., Ge, R., Kakade, S.M., Jordan, M.I.: On nonconvex optimization for machine learning: gradients, stochasticity, and saddle points. J. ACM **68**(2), 1–29 (2021). <https://doi.org/10.1145/3418526>
11. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2014). <https://doi.org/10.48550/ARXIV.1412.6980>, <https://arxiv.org/abs/1412.6980>
12. Levy, K.Y.: The power of normalization: faster evasion of saddle points. CoRR **abs/1611.04831** (2016). <http://arxiv.org/abs/1611.04831>
13. Nocedal, J., Wright, S.J.: Numerical Optimization, 2e edn. Springer, New York (2006). <https://doi.org/10.1007/978-0-387-40065-5>
14. Robbins, H., Monro, S.: A stochastic approximation method. Ann. Math. Stat. **22**(3), 400–407 (1951). <https://doi.org/10.1214/aoms/1177729586>
15. Zhang, C., Li, T.: Escape saddle points by a simple gradient-descent based algorithm. In: Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems, vol. 34, pp. 8545–8556. Curran Associates, Inc. (2021)



Transformer based Multitask Learning for Image Captioning and Object Detection

Debolena Basak^(✉), P. K. Srijith, and Maunendra Sankar Desarkar

Indian Institute of Technology Hyderabad, Sangareddy, India
ai20resch11003@iith.ac.in, {srijith,maunendra}@cse.iith.ac.in

Abstract. In several real-world scenarios like autonomous navigation and mobility, to obtain a better visual understanding of the surroundings, image captioning and object detection play a crucial role. This work introduces a novel multitask learning framework that combines image captioning and object detection into a joint model. We propose **TICOD**, Transformer-based Image Captioning and Object Detection model for jointly training both tasks by combining the losses obtained from image captioning and object detection networks. By leveraging joint training, the model benefits from the complementary information shared between the two tasks, leading to improved performance for image captioning. Our approach utilizes a transformer-based architecture that enables end-to-end network integration for image captioning and object detection and performs both tasks jointly. We evaluate the effectiveness of our approach through comprehensive experiments on the MS-COCO dataset. Our model outperforms the baselines from image captioning literature by achieving a 3.65% improvement in BERTScore.

Keywords: Transformer · Multitask Learning · Image Captioning · Object Detection

1 Introduction

In autonomous navigation and human mobility assistance systems, image captioning and object detection play a vital role in obtaining a better visual understanding of the surroundings. Applications such as human mobility assistance systems would require the detection of objects including their positions and description of the environment in natural language to help in the mobility of visually impaired people.

Most image captioning models follow the encoder-decoder framework along with the visual attention mechanism. The encoder processes input images and encodes them into fixed-length feature vectors, and the decoder utilizes these encoded image features to generate word-by-word descriptions [1, 23, 29, 34]. Most mainstream image captioning models [1, 5] use a two-step training method, which firstly extracts image regional features using a pre-trained object detector like Faster R-CNN [26], then feeds these feature vectors into an encoder-decoder framework for image captioning. However, this approach has a few inherent shortcomings: 1) the object detection model in the first step is trained on specific visual datasets like the Visual Genome, and the visual

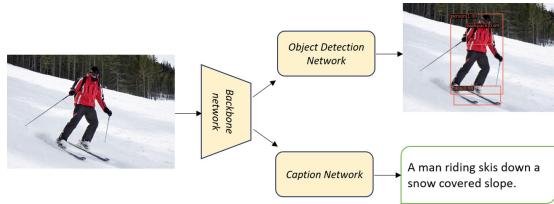


Fig. 1. A high-level framework of our proposed method. Our model TICOD has three major components, which we call as—(a) the *backbone network*, (b) the *object detection network*, and (c) the *caption network*.

representation is not optimized towards the image captioning dataset used (commonly MS-COCO). This may lead to an error propagation problem if the object detector fails to recognize certain important visual information [21], 2) the time-intensive nature of extracting region features causes state-of-the-art models to rely on cached visual features (usually pre-computed) for training and evaluation, imposing constraints on model designs and resulting in run-time inference inefficiencies during prediction [13, 31, 33]. This introduces a two-stage process where a pre-trained object detection model and an image captioning model are used in sequence to extract features and predict the captions. The learning typically updates the image captioning model. This impedes the end-to-end training from image pixels to descriptions in image captioning models, limiting their applicability in real-world scenarios [13, 31].

Inspired by the NLP domain, the transformer architecture has shown its potential in computer vision (CV) tasks [7, 20] and multimodal tasks [25]. Considering the drawbacks of pre-trained CNN and object detector in the encoder and advantages of vision transformers, we integrate the task of image captioning as a single-stage approach, which can also perform object detection parallelly, as shown in Fig. 1. The key idea is *multitask learning across object detection and image captioning* that enables the model to develop a better representation learning capability. This shared representation learning enables the model to leverage the knowledge gained from each task to effectively align the backbone representations, enhancing the overall learning capacity. The model parameters are learned by optimizing a joint loss that combines the losses from both tasks. A key advantage of this approach is that if we want to generate a caption, we can simply enable the captioning network, turning off the object detection network, which helps us to get output without introducing any additional latency. On the other hand, if we need more detailed information, we can enable the detection network to provide us with objects' details along with the captions simultaneously. Our model's generated captions and detected objects can be utilized to generate synthetic data using LLMs like GPT-4. For instance, a recent work [18] has used COCO captions and detection annotations to feed into *text-only* GPT-4 to generate *multimodal instruction-following data*, which includes *conversation*, *detailed description* and *complex reasoning* data. This synthetic data has been used for training the multimodal chatbot LLaVA [18].

We use Swin Transformer [20] as the backbone network for extracting image features. We use GPT2 [24] to decode the image features extracted from the Swin transformer and generate the captions of the corresponding image. For the object detection

part, Cascade R-CNN [2] framework is used with the same Swin backbone. We use the MS-COCO dataset [16, 32] for bounding box information and category labels to train object detection, and the caption annotations corresponding to the same image are used for image captioning together as joint training.

To summarize, our key contributions are – (a) We developed a Transformer-based **Image Captioning and Object Detection (TICOD)** model capable of simultaneously performing image captioning and object detection, (b) TICOD uses a joint loss function that combines the losses from object detection and caption networks while maintaining a trade-off between them, (c) We demonstrate that our multitask method outperforms the task-specific models on image captioning by 3.65% in BERTScore [35] and produces comparable performance in object detection.

2 Related Work

Image Captioning: Most of the existing image captioning works can be broadly categorized into CNN-RNN based models [1, 29, 34] and CNN-Transformer based models [5, 21, 23]. \mathcal{M}^2 Transformer [5] used a mesh-like connection between each of the encoder and the decoder blocks to extract features from all levels of the encoder. It used Faster R-CNN [26] pre-trained on the Visual Genome dataset. The disadvantages of such an approach have been already discussed. After Vision Transformer (ViT) [7] and its variants [20, 25, 27] became popular in CV tasks, people began to explore it for image captioning as well. ClipCap [22] performs image captioning using a CLIP [25] encoder and GPT2 [24] decoder. Oscar [14] and VinVL [36] use BERT [6] but provide additional object tags for supervision, which limits their practical applicability in real-world scenarios. These approaches [14, 36] are constrained to datasets that provide access to object detectors or annotations. *PureT* [31] used Swin Transformer [20] as a backbone network to extract image features for image captioning. They keep the Swin backbone pre-trained weights frozen in their experiments. However, we train end-to-end to leverage the information obtained from the captions to influence the Swin backbone weights.

Object Detection and Transformer-Based Vision Backbones: Object detection research has seen a breakthrough with the introduction of CNN-based models like R-CNN [10], Fast R-CNN [9], and Faster R-CNN [26]. Later, the remarkable success of the Transformer architecture [28] in the NLP domain has motivated researchers to explore its application in computer vision. Transformers were first introduced for vision problems in Vision Transformers (ViT) [7]. Several works on ViT and its variants followed up [11, 27, 30]. However, ViT required large-scale training datasets like JFT-300M to perform optimally. DeiT [27] addresses this limitation by introducing training strategies that enable ViT to work effectively with smaller datasets like ImageNet-1K. While ViT shows promising results in image classification, it is not suitable as a general-purpose backbone network for dense vision tasks or high-resolution input images [20]. In parallel to Swin Transformer [20], other researchers have also modified the ViT architecture to improve image classification [11]. But Swin Transformer [20] demonstrates that they achieve the best speed-accuracy trade-off among the above-mentioned methods on image classification, though it focuses on a general-purpose backbone. Also,

Swin Transformer has linear complexity to image size unlike [30] with quadratic complexity.

Multitask Learning: Early works on multitask image captioning such as [8] incorporate a simple multi-label classification as an auxiliary task for image captioning. [37] expands on this by introducing two auxiliary tasks: multi-label classification and syntax generation. Both papers highlight the benefits of auxiliary tasks in enhancing the performance of image captioning and use CNN-LSTM as an encoder-decoder. In the proposed approach, we use a more complex object detection as the auxiliary task and use Swin Transformer in our architecture. Recently, there has been a growing interest in developing models which can solve multiple tasks together. For instance, Pix2seq-v2 [4] proposes a prompt-based approach in an encoder-decoder framework based on Transformers to solve four tasks, namely, object detection, instance segmentation, key-point detection, and image captioning.

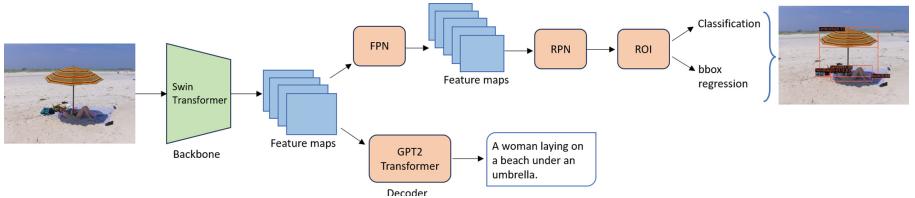


Fig. 2. Architectural overview of the proposed Transformer-based Image Captioning and Object Detection (TICOD) model.

3 Proposed Method

This work aims to leverage multitask learning to train objection detection and image captioning tasks jointly. The overall architecture of our model is shown in Fig. 2. The complete architecture has three major components: (a) an initial image feature extractor, (b) an object identification network, and (c) a caption generation network. As the name suggests, the image feature extractor extracts features from the input image. These features are then passed to the two networks for object detection and caption generation. Swin Transformer [20] is used for the image feature extractor owing to its superior performance in various image understanding tasks [15, 20, 31].

The object detection network involves passing the extracted image features through a sequence of Feature Pyramid Network (FPN) [17], Region Proposal Network (RPN) [26], and Region of Interest (RoI) pooling layer [9] to get the objects and their bounding boxes detected. It can be seen that this flow of encoding the image representation to finally detecting the objects follows the overall framework of Faster R-CNN [26]. This specific instantiation of Faster R-CNN uses Swin Transformer as the backbone to extract the image feature maps. On the other hand, the caption generation network passes the extracted image features through a GPT-2 [24] architecture to get the final captions. These two networks operate in parallel to solve the tasks of object detection

and caption generation and the losses for the tasks are combined into a multitask loss. The use of a common feature extractor for both the tasks, and a combined loss function to guide the training process enables the two tasks to influence the learning of each other and improve their individual performance.

As shown in Fig. 2, our model consists of a common encoder (i.e., image feature extractor) and two parallel decoders (i.e., object identification network and caption generation network), trained as a single model with a joint loss function. Given an input image, it is first divided into image patches with patch size = 4×4 . It is then passed through a Swin backbone, producing four feature maps of the following dimensions:

$$\left\{ \left(bs \times C \times \frac{H}{4} \times \frac{W}{4} \right), \left(bs \times 2C \times \frac{H}{8} \times \frac{W}{8} \right), \left(bs \times 4C \times \frac{H}{16} \times \frac{W}{16} \right), \left(bs \times 8C \times \frac{H}{32} \times \frac{W}{32} \right) \right\}$$

where, bs denotes the batch size, C denotes the channel dimensions, i.e., the patch embedding dimension in each Swin block, and H, W denote the height and width of the image, respectively. We call this as the *backbone network*.

Each of these feature maps is passed through the FPN regarded as the *neck* in object detection literature. It produces five feature maps, each of embedding dimension = 256 [17]. FPN is used at multiple levels to increase the detection of small objects. It provides semantically rich information at multiple scales, which reduces the error rate in detection [17]. Feature maps from the FPN are passed into the RPN [26] and RoI pooling layer [9,26] to finally produce the object classes and bounding box coordinates. We refer to this as the *object detection network*.

The other parallel branch of our multitask model, which we call the *caption network*, consists of a GPT2 [24] Transformer as the decoder. It takes the last feature map of the Swin backbone as input along with a `<start>` token and generates captions word by word in an auto-regressive manner.

Attention: We use Multi-head Self Attention (MSA) [28] in the decoder to calculate the relationship between tokens in a sequence and cross-attention for the relationship between the tokens and image grid features. We also adopt Window MSA/Shifted Window MSA (W-MSA/SW-MSA) proposed in the Swin Transformer work [20]. They are used in the encoder to model the relationship between the image patches.

3.1 Objective Function

For training the caption network, we use the standard language modeling loss, which is the Cross-Entropy loss ($L^C(\theta, \phi)$) computed over all the samples. For a given sample, the loss function aims to predict the next token given the context and can be mathematically formulated as follows:

$$L^C(\theta, \phi) = - \sum_{t=1}^T \log(p(y_t^* | y_{1:t-1}^*, x)) \quad (1)$$

where θ and ϕ represent the parameters of Swin Transformer backbone and GPT2 respectively, $y_{1:T}^*$ is the target ground-truth sequence, and x is the input image.

For objection detection, we consider a loss ($L^O(\theta, \psi)$) which consists of the Cross-Entropy Loss for object classification and smooth L1 Loss [9] for bounding box regression computed over all the samples. Here, ψ represents the parameters of the Faster

R-CNN/Cascade R-CNN network used for object detection. Since the *object detection network* consists of the Region Proposal Network (RPN) and Region of Interest (RoI) pooling layer, L^O can be further subdivided as -

$$L^O(\theta, \psi) = L_{cls}^{RPN} + L_{reg}^{RPN} + L_{cls}^{RoI} + L_{reg}^{RoI} \quad (2)$$

where L_{cls}^{RPN} and L_{reg}^{RPN} denote the losses from RPN and RoI layers respectively. L_{cls}^{RPN} and L_{reg}^{RPN} denote classification and bounding box (bbox) regression losses from the RPN network. L_{cls}^{RoI} and L_{reg}^{RoI} refer to the classification loss and bbox regression loss from the RoI network. L_{cls}^{RPN} is a classification log loss over two classes—object or non-object, i.e., *background* class [26]. L_{reg}^{RPN} is a Smooth L_1 Loss defined in [9] arising from the difference between the ground-truth bbox coordinates and the predicted bbox coordinates [26]. L_{cls}^{RoI} is the log loss over $(C + 1)$ categories— C object classes and a *background* class, and L_{reg}^{RoI} is again a similar Smooth L_1 Loss over the bbox coordinates [9].

For training our model. We use the joint multitask learning objective function, which combine the image captioning and object detection losses as

$$L^T(\theta, \phi, \psi) = L^O(\theta, \psi) + \lambda \cdot L^C(\theta, \phi) \quad (3)$$

where λ is the weightage to be given to the captioning loss. The λ value is chosen to maximize the evaluation scores of both image captioning and object identification. It is determined empirically using a validation data as demonstrated in Table 4. We found the most suitable values of λ to be 0.1 and 0.2 for TICOD-small and TICOD-large, respectively.

4 Experimental Setup

Dataset: We use the MS-COCO 2017 dataset [16,32] containing 118K training and 5K validation images. COCO has five captions per image and separate annotations for object categories and bounding boxes. For comparison with image captioning works, we follow the standard “Karpathy” split.

Evaluation Metrics: *Image Captioning*—We evaluate our captioning performance using the standard metrics used in Natural Language Processing, viz, BLEU, CIDEr, METEOR, ROUGE-L, and SPICE. The metrics above only evaluate the generated captions by matching them with the reference captions at the lexical level. Since these metrics try to find exact matches, the scores might not be a true measure of the quality of the captions, as the presence of synonymous words may lower the scores. So, we also measure the scores based on BERTScore [35], which uses contextual embeddings to find a semantic similarity measure between the candidate sentence and the reference sentence. BERTScore has been shown to exhibit a superior correlation with human judgments [38], and provide strong model selection performance [35].

Object Detection—We use the standard evaluation metrics—mAP as the mean of APs@[.5 : .05 : .95], AP@ $IoU = 0.50$, and AP@ $IoU = 0.75$. We also report the APs across scales, i.e., APs@small, medium, and large objects.

Implementation Details: We keep the same settings as in Swin Transformer [20] work. AdamW optimizer is used with an initial learning rate of 10^{-4} , weight decay of 0.05, and batch size of 2 per GPU. For training, we use 4 Nvidia V100 GPUs and for inference, we use a single V100 GPU. During inference, captions are generated using beam search with beam size = 5. For the image captioning part, we take the encoding from the last feature map of the Swin backbone. The last feature map of Swin-T and Swin-B models have embedding dimensions of $8 \times 96 = 768$ and $8 \times 128 = 1024$, respectively [20], which matches the embedding dimensions of GPT2-small ($dim = 768$) and GPT2-medium ($dim = 1024$). The combined loss from the object detection and the caption networks is backpropagated to update the model weights of the backbone network.

Architecture Details: The authors of Swin Transformer [20] have proposed several variants of the Swin backbone: Swin - tiny, small, base, and large. For our experiments, we have used –

- Swin-tiny [20] with GPT2-small [24] for image captioning and Swin-tiny backbone in Faster R-CNN framework [26] for object detection. We call this *TICOD-small*.
- Swin-base [20] with GPT2-medium [24] for image captioning and Swin-base backbone in Cascade R-CNN [2] for object detection. We call this as *TICOD-large*.

Our model TICOD-large performs better than the other variant. The Swin Transformer acts as a common backbone for image captioning and object detection in our multitask model.

5 Results

5.1 Comparison and Analysis

We compare our multitask model with task-specific baselines from the literature on image captioning and object detection. For image captioning, we compare our work with some recent models like – ClipCap [22], Meshed-memory (\mathcal{M}^2) Transformer [5], and PureT [31] on the MS-COCO [32] offline test split. We also compare our work with Pix2seq-V2 [4], a recently proposed multitask system that jointly learns object detection and image captioning along with two other vision tasks. Table 1 reports the performances of these models and our proposed model.

Table 1. Comparison on MS-COCO [16,32] dataset.

Methods	B1	B2	B3	B4	Meteor	RougeL	CIDEr	Spice	BERTScore	mAP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L
ClipCap (MLP+GPT2 fine-tuning) [22]	70.9	54.4	41.2	31.5	27.7	54.8	106.7	20.5	68.001	–	–	–	–	–	–
ClipCap (Transformer) [22]	74.6	58.5	44.5	33.5	27.6	55.9	112.8	21.0	68.326	–	–	–	–	–	–
\mathcal{M}^2 - Transformer [5]	80.8	–	–	39.1	29.2	58.6	131.2	22.6	64.556	–	–	–	–	–	–
PureT [31]	82.1	67.3	53.0	40.9	30.2	60.1	138.1	24.2	68.303	–	–	–	–	–	–
BUTD [1]	77.2	–	–	36.2	27.0	56.4	113.5	20.3	–	–	–	–	–	–	–
Pix2Seq-V2 [4]	–	–	–	34.9	–	–	–	–	46.5	–	–	–	–	–	–
Swin-B (Cascade R-CNN) [20]	–	–	–	–	–	–	–	–	51.9	70.9	56.5	35.4	55.2	67.3	–
DETR-R101 [3]	–	–	–	–	–	–	–	–	43.5	63.8	46.4	21.9	48.0	61.8	–
Faster R-CNN R101-FPN [26]	–	–	–	–	–	–	–	–	42.0	62.5	45.9	25.2	45.6	54.6	–
TICOD-large (Ours)	75.6	59.0	45.5	35.3	28.3	56.7	115.3	21.1	70.794	52.1	70.6	56.7	34.8	55.3	67.2

From Table 1, it can be seen that the PureT [31] model outperforms all other models in terms of the BLEU (B1 to B4), Meteor, RougeL, CIDEr, and Spice metrics. However, these are all lexical similarity based metrics, and try to match the exact words present in the generated and ground-truth captions. Although a good score in terms of these metrics indicate good match between the generated and the reference captions, a lower score does not necessarily indicate a poor match. This is because any concept or thought can be expressed in multiple ways with very less overlap in the words used in these *parallel expressions*. This drawback is addressed in the BERTScore metric [35], where the semantic embeddings of the texts are compared to decide the performance score. Our model achieves comparable performances with PureT and \mathcal{M}^2 transformer in terms of the lexical overlap based scores. At the same time, it achieves **3.65%** and **9.66%** improvements in BERTScore over these two models respectively, indicating that the proposed model can generate better quality captions at a semantic level. The proposed TICOD-large model also outperforms both model variants of ClipCap [22] in terms of all the metrics as well as BERTScore. The mentioned BERTScores in this table are calculated using Deberta-xlarge-mnli model [12] as it best correlates with human evaluation [35]. Our model also outperforms the popular BUTD model [1] in terms of CIDEr, Meteor, RougeL, Spice.

In addition, we also calculate BERTScore with other models like – Roberta-large [19] and Deberta-xlarge-mnli with idf (Inverse Document Frequency). BERTScore calculated using these three models for the baselines and the proposed multitask model are illustrated in Table 2. Clearly, our proposed multitask model outperforms all the baselines in terms of BERTScore calculated using all three mentioned methods. Comparable scores based on lexical overlap-based metrics and superior scores based on embedding-based methods indicate that the proposed model can generate good-quality captions for the input images.

Table 2. Comparison of captioning performance of our proposed multitask model with some image captioning baselines from literature, in terms of BERTScore [35].

Methods	Deberta-xlarge-mnli [12]	Roberta-large [19]	Deberta-xlarge-mnli with idf
\mathcal{M}^2 - Transformer [5]	64.56	63.11	59.40
ClipCap (MLP+GPT2 fine-tuning) [22]	68.00	65.27	59.13
ClipCap (Transformer) [22]	68.33	66.25	59.37
PureT [31]	68.30	67.69	63.22
TICOD-large ($\lambda = 0.2$) (proposed model)	70.79	68.06	63.23
TICOD-large ($\lambda = 0.5$)	71.69	68.98	63.40

We report the object detection evaluation scores also in Table 1 for convenience of comparison. Our objective of this work is to perform image captioning and object detection simultaneously by improving the performance of image captioning due to joint training with a carefully constructed joint loss function. We demonstrate that we achieve superior image captioning performance in terms of BERTScore while maintaining a comparable performance in object detection. Since our model is developed upon

Swin Transformer architecture [20], we compare our model’s performance on object detection with Swin Transformer [20] in the Cascade R-CNN [2] framework. The comparisons presented in Table 1, show that TICOD has better performance in terms of mAP, AP@0.75, AP@small, and AP@medium objects. This, in turn, shows that the caption generation task has positively influenced the object detection task through the joint training, and has resulted in improved performance for object detection. We also compare our model with other popular baselines like DETR [3] and Faster R-CNN [26]. We compare with Pix2seq-v2 [4] on their reported object detection and image captioning scores, and we can see a clear improvement in performance using our approach for both tasks. This is because Pix2seq-v2 doesn’t use detection-specific architecture but instead uses language modeling to solve “core” vision tasks. Due to the significant departure from conventional architectures, the model needs further improvement to challenge the current SOTA of task-specific models [4]. It also has a slower inference speed (particularly for longer sequences) than the specialized systems, as the approach is based on autoregressive modeling [4].

Qualitative Comparison: While discussing the caption generation performance of the different models, based on the values of the evaluation metrics, we argued that TICOD generates good-quality captions. However, it may use words that are semantically similar but lexically different from the tokens in the corresponding ground-truth captions. Figure 3 presents some example cases to elaborate on this point. It shows a few images with detected objects by our proposed model and their corresponding captions, along with the captions generated by PureT model [31], \mathcal{M}^2 Transformer [5] and the ground-truths. In Fig. 3a, our model generates the correct caption whereas \mathcal{M}^2 incorrectly generates *green tennis racket*. The generation also doesn’t end properly as it produces *on a* at the end. PureT generated *on a table*, which is incorrect. In Fig. 3b, our model generates *red stop sign* which is more detailed than \mathcal{M}^2 and PureT model’s captions. In Fig. 3c, the captions are all similar. However, in Fig. 3d, we notice that our model has slightly under-performed as it generates *A couple of small birds* instead of *Two birds* produced by PureT. \mathcal{M}^2 also under-performed by producing *A small bird*. By qualitatively analyzing the captions produced by our model, we have observed that, while in most cases our model produces better or equivalent captions, there are some cases where our model has slightly regressed performance.

5.2 Ablation Studies

To evaluate the effectiveness of our proposed approach, we perform ablation studies on the COCO dataset [16, 32] using two object detection frameworks – Faster R-CNN [26] and Cascade R-CNN [2]. We also check the model’s performance by keeping different parts of the network frozen.

Effect of Backbone Size and Object Detection Framework: Table 3 shows the performance of our model with different backbone sizes and object detection frameworks. As observed, our model performs better when Cascade R-CNN is used with Swin-base and GPT2-medium. There is a performance gain of $\sim 3\%$ Bleu-1 score, **21.67%** Bleu-4 and **21.09% CIDEr (+18.5 CIDEr)** over Faster-RCNN with Swin-tiny and GPT2-small with frozen backbone and object detection network (fine-tuning only the decoder network). Also, there is an improvement of **6.63%** Bleu-1, **27%** Bleu-4, and **28%** CIDEr

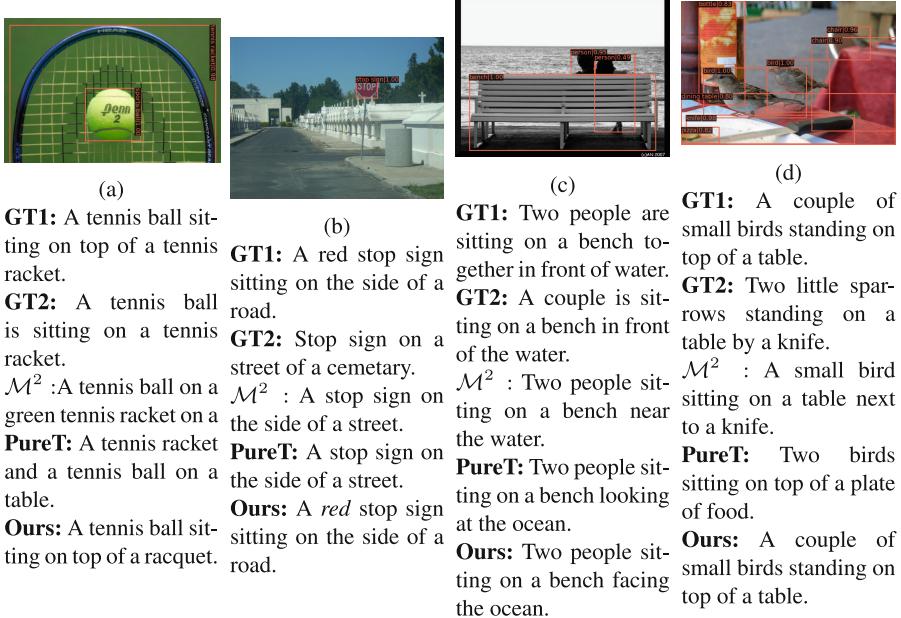


Fig. 3. Examples of captions generated by \mathcal{M}^2 -Transformer [5], PureT [31], our model, and their corresponding ground-truths(GT) [32]. The images also display the detected object categories and their scores as predicted by our proposed model.

(+ 25.2 CIDEr) when the backbone and decoder networks are finetuned. Clearly, with larger backbone and decoder sizes, there is performance improvement.

Effect of Frozen Layers: We perform experiments by keeping different components of the network frozen. The trainable components of the network are mentioned in the fourth column of Table 3. The remaining components of the model were frozen. We tried three combinations of `<frozen-finetuned>` components: (i) Swin backbone, FPN, RPN, ROI layers frozen while finetuning only the GPT2 decoder, (ii) finetuning Swin backbone and GPT2 while keeping the remaining layers required for object detection as frozen, and (iii) finetuning all components by keeping the caption network turned off. It is observed from Table 3 that case (ii) has a performance improvement of **2.74%** CIDEr for Faster R-CNN method and **8.57%** CIDEr for Cascade R-CNN. BERTScore has also improved by **3.65%** for Cascade R-CNN. We also observe that for object detection, the mAP improves from 51.9 to **52.1** when the backbone network is trainable, which demonstrates the positive impact of joint training.

For both the upper and lower halves of Table 3, the last rows represent the setup where the GPT-2 network is frozen, and the object detection network is finetuned. In the proposed TICOD model, the caption network is passed an image embedding. However, the GPT-2 in the caption network is initialized with pre-trained GPT-2 parameters that do not recognize that image embedding as input. Hence, without finetuning the

Table 3. Ablation study with different methods, backbones and decoder sizes on COCO [16, 32] dataset.

Methods	Backbone	Decoder	Finetuned networks	B1	B2	B3	B4	Meteor	RougeL	CIDEr	Spice	BERTScore	mAP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L
Faster R-CNN [26]	Swin-T	GPT2 (small)	GPT2	70.4	51.5	36.8	26.3	23.7	50.9	87.7	16.7	66.473	46.0	68.1	50.3	31.2	49.2	60.1
		GPT2 (small)	X	70.9	52.5	38.2	27.8	24.1	51.3	90.1	17.1	66.934	45.3	67.7	49.9	29.5	48.7	59.0
		Swin.GPT2	-	-	-	-	-	-	-	-	-	-	46.0	68.1	50.3	31.2	49.2	60.1
Cascade R-CNN [2]	Swin-B	GPT2 (med)	GPT2	72.5	55.3	41.8	32.0	27.2	54.6	106.2	19.7	68.301	51.9	70.9	56.5	35.4	55.2	67.3
		GPT2 (med)	X	75.6	59.0	45.5	35.3	28.3	56.7	115.3	21.1	70.794	52.1	70.6	56.7	34.8	55.3	67.2
		Swin.GPT2	-	-	-	-	-	-	-	-	-	-	51.9	70.9	56.5	35.4	55.2	67.3
Swin.FPN, RPN, ROI	Swin-B	Swin.GPT2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
		Swin.FPN, RPN, ROI	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
		Swin.FPN, RPN, ROI	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	

caption-generation network, the generated captions for this setup would be meaningless. Accordingly, the values of the evaluation metrics for image captioning are filled by “–”s for these rows.

The object detection scores in the first and third rows are identical in both halves of the table. The first row corresponds to a setup where the object detection network is frozen, while the third row involves finetuning the parameters of the object detection network. The similarity in scores arises from the fact that the object detection network’s parameters are initialized with a pre-trained checkpoint optimized for this dataset. Further finetuning leads to a drop in performance on the validation set. As a result, the initial parameters (epoch 0) are retained, resulting in similar metric values for these rows for the object detection task.

Task-specific Performance: We examine the impact of multitasking on performance compared to task-specific models through experiments with two baselines: (i) object detection alone without the caption network, and (ii) captioning alone without the object detection network. Results in Table 3 indicate improved image captioning performance without compromising object detection.

Choosing Lambda: We conduct experiments to finetune the hyperparameter λ in Eq. 3. Table 4 demonstrates that 0.1 and 0.2 are the most suitable values for TICOD-small and TICOD-large, respectively. For TICOD-small, the difference in image captioning scores between $\lambda = 0.1$ and 10 is negligible, and as object detection performance degrades with increasing lambda, scores for $\lambda = 0.2$ and 0.5 are not reported.

Table 4. Hyperparameter tuning: Illustration of model performance with different λ values.

Methods	lambda (λ)	B1	B2	B3	B4	Meteor	RougeL	CIDEr	Spice	BERTScore	mAP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L
TICOD-small	0.01	69.7	51.3	36.6	25.8	22.8	50.1	82.1	16.2	65.728	45.4	67.3	50.2	31.1	48.8	59.1
	0.1	70.9	52.5	38.2	27.8	24.1	51.3	90.1	17.1	66.934	45.3	67.7	49.9	29.5	48.7	59.0
	10	70.8	52.8	38.1	27.4	23.8	51.3	88.6	17.2	66.698	37.7	62.3	40.6	25.8	42.0	46.1
TICOD-large	0.01	72.8	56.9	41.8	32.1	26.7	54.2	100.7	18.6	67.801	51.9	70.5	56.6	36.1	55.2	67.4
	0.1	74.3	57.3	43.6	33.5	27.0	55.1	107.3	19.4	69.821	51.9	70.6	56.5	36.1	55.3	67.2
	0.2	75.6	59.0	45.5	35.3	28.3	56.7	115.3	21.1	70.794	52.1	70.6	56.7	34.8	55.3	67.2
	0.5	76.5	60.4	46.9	36.6	28.9	57.6	119.6	21.6	71.686	51.6	70.3	56.3	33.6	55.0	67.4
	10	76.2	60.5	45.7	36.1	28.5	57.1	119.4	21.5	71.221	47.2	67.8	50.4	30.2	51.5	61.1

6 Conclusion

In this work, we presented TICOD, a multitask framework for object detection and image captioning. Empirically, we show that joint learning helps improve image captioning by improving the image representations in the backbone. Swin Transformer is not pre-trained on a vision-language objective, yet we demonstrate that we can use it directly with GPT2 and show superior image captioning performance in BERTScore while maintaining a comparable performance in object detection. Our proposed framework is customizable as Swin Transformer and GPT2 can be replaced with newer specialized SOTA detection and large language models, which will further improve performance over general-purpose multitask models like Pix2seq-V2 [4].

Acknowledgements. This work was supported by DST National Mission on Interdisciplinary Cyber-Physical Systems (NM-ICPS), Technology Innovation Hub on Autonomous Navigation and Data Acquisition Systems: TiHAN Foundations at Indian Institute of Technology (IIT) Hyderabad, India. We also acknowledge the support from Japan International Cooperation Agency (JICA). We express gratitude to Suvodip Dey for his valuable insights and reviews on this work.

References

1. Anderson, P., et al.: Bottom-up and top-down attention for image captioning and visual question answering. In: Proceedings of the IEEE Conference on CVPR (2018)
2. Cai, Z., Vasconcelos, N.: Cascade r-cnn: delving into high quality object detection. In: Proceedings of the IEEE Conference on CVPR (2018)
3. 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.) Computer Vision. ECCV 2020. LNCS, vol. 12346, pp. 213–229. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58452-8_13
4. Chen, T., et al.: A unified sequence interface for vision tasks. In: NeurIPS (2022)
5. Cornia, M., et al.: Meshed-memory transformer for image captioning. In: CVPR (2020)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. arXiv preprint [arXiv:1810.04805](https://arxiv.org/abs/1810.04805) (2018)
7. 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)
8. Fariha, A.: Automatic image captioning using multitask learning. In: NeurIPS, vol. 20 (2016)
9. Girshick, R.: Fast r-cnn. In: International Conference on Computer Vision (ICCV) (2015)
10. Girshick, R., et al.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on CVPR (2014)
11. Han, K., et al.: Transformer in transformer. In: Advances in NeurIPS, pp. 15908–15919 (2021)
12. He, P., et al.: Deberta: decoding-enhanced bert with disentangled attention. In: ICLR (2021)
13. Jiang, H., et al.: In defense of grid features for visual question answering. In: CVPR (2020)
14. Li, X., et al.: Oscar: Object-semantics aligned pre-training for vision-language tasks. In: Computer Vision. ECCV 2020. Springer (2020)
15. Liang, J., et al.: Swinir: image restoration using Swin transformer. In: ICCV (2021)
16. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) Computer Vision. ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48

17. Lin, T.Y., et al.: Feature pyramid networks for object detection. In: CVPR (2017)
18. Liu, H., et al.: Visual instruction tuning. arXiv preprint [arXiv:2304.08485](https://arxiv.org/abs/2304.08485) (2023)
19. Liu, Y., et al.: Roberta: a robustly optimized bert pretraining approach. arXiv preprint [arXiv:1907.11692](https://arxiv.org/abs/1907.11692) (2019)
20. Liu, Z., et al.: Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF ICCV, pp. 10012–10022 (2021)
21. Luo, Y., et al.: Dual-level collaborative transformer for image captioning. Proc. AAAI Conf. Artif. Intell. **35**(3), 2286–2293 (2021). <https://doi.org/10.1609/aaai.v35i3.16328>
22. Mokady, R., Hertz, A., Bermano, A.H.: Clipcap: clip prefix for image captioning. arXiv preprint [arXiv:2111.09734](https://arxiv.org/abs/2111.09734) (2021)
23. Pan, Y., et al.: X-linear attention networks for image captioning. In: CVPR (2020)
24. Radford, A., et al.: Language models are unsupervised multitask learners. OpenAI blog (2019)
25. Radford, A., et al.: Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning, pp. 8748–8763. PMLR (2021)
26. Ren, S., et al.: Faster r-cnn: towards real-time object detection with region proposal networks. In: NeurIPS, vol. 28 (2015)
27. Touvron, H., et al.: Training data-efficient image transformers and distillation through attention. In: International Conference on Machine Learning, pp. 10347–10357. PMLR (2021)
28. Vaswani, A., et al.: Attention is all you need. In: NeurIPS, vol. 30 (2017)
29. Vinyals, O., et al.: Show and tell: a neural image caption generator. In: CVPR (2015)
30. Wang, W., et al.: Pyramid vision transformer: a versatile backbone for dense prediction without convolutions. In: Proceedings of the IEEE/CVF ICCV, pp. 568–578 (2021)
31. Wang, Y., Xu, J., Sun, Y.: End-to-end transformer based model for image captioning. Proc. AAAI Conf. Artif. Intell. **36**(3), 2585–2594 (2022). <https://doi.org/10.1609/aaai.v36i3.20160>
32. Xinlei Chen, H.F., et al.: Microsoft coco captions: data collection and evaluation server. arXiv preprint [arXiv:1504.00325](https://arxiv.org/abs/1504.00325) (2015)
33. Xu, H., et al.: E2e-vlp: end-to-end vision-language pre-training enhanced by visual learning. In: Proceedings of the 59th Annual Meeting of the ACL and the 11th IJCNLP (2021)
34. Xu, K., et al.: Show, attend and tell: neural image caption generation with visual attention. In: Proceedings of the 32nd ICML, vol. 37, pp. 2048–2057. PMLR (2015)
35. Zhang, K., et al.: Bertscore: evaluating text generation with bert. In: ICLR (2020)
36. Zhang, P., et al.: Vinvl: revisiting visual representations in vision-language models. In: Proceedings of the IEEE/CVF Conference on CVPR (2021)
37. Zhao, W., et al.: A multi-task learning approach for image captioning. In: IJCAI 2018
38. Zhuo, T.Y., et al.: Rethinking round-trip translation for machine translation evaluation. In: Findings of the Association for Computational Linguistics: ACL (2023)



Communicative and Cooperative Learning for Multi-agent Indoor Navigation

Fengda Zhu, Vincent CS Lee^(✉), and Rui Liu

Monash University, Clayton, Australia
vincent.cs.lee@monash.edu

Abstract. The ability to cooperate and work as a team is one of the “holy grail” goals of intelligent robots. To address the importance of communication in multi-agent reinforcement learning (MARL), we propose a Cooperative Indoor Navigation (CIN) task, where agents cooperatively navigate to reach a goal in a 3D indoor room with realistic observation inputs. This navigation task is more challenging and closer to real-world robotic applications than previous multi-agent tasks since each agent can observe only part of the environment from its first-person view. Therefore, this task requires the communication and cooperation of agents to accomplish. To research the CIN task, we collect a large-scale dataset with challenging demonstration trajectories. The code and data of the CIN task have been released. The prior methods of MARL primarily emphasized the learning of policies for multiple agents but paid little attention to the communication model, resulting in their inability to perform optimally in the CIN task. In this paper, we propose a MARL model with a communication mechanism to address the CIN task. In our experiments, we discover that our proposed model outperforms previous MARL methods and communication is the key to addressing the CIN task. Our quantitative results shows that our proposed MARL method outperforms the baseline by 6% on SPL. And our qualitative results demonstrates that the agent with the communication mechanism is able to explore the whole environment sufficiently so that navigate efficiently.

Keywords: Multi-agent Reinforcement Learning · Cooperative Navigation

1 Introduction

Cooperative multi-agent problems are ubiquitous in real-world applications, such as multiplayer games [14, 24], multi-robot control [39], language communication [16]. These applications focus on solving the decision-making problem of multiple autonomous agents within a common environment, which are modeled as the multi-agent reinforcement learning (MARL) [17]. Many game-based environments have been proposed to research the cooperative multi-agent problems, such as grounded communication environment [21], StarCraft II [25], multi-agent emergence environments [4], soccer shooting [18], etc.

Many game-based Multi-Agent Reinforcement Learning environments diverge from real-world robotic scenarios. Agents in these environments often receive noise-free, pre-processed data instead of experiencing authentic, first-person observations.

Additionally, the transition function relies on straightforward rules without simulating real-world physics or accounting for actual-world interferences. This significant domain gap hinders the application of current MARL models [24, 37] in real-world scenarios.

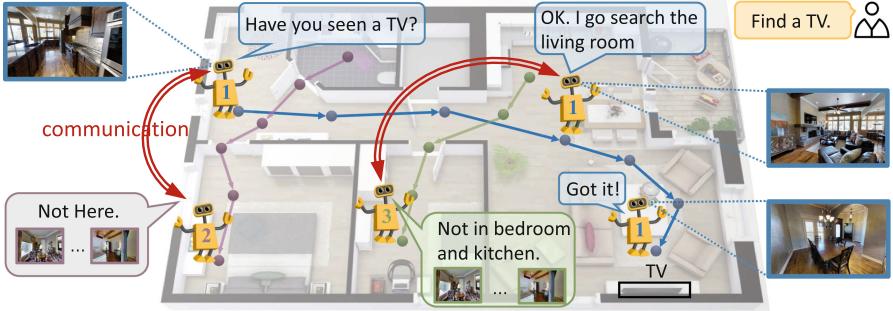


Fig. 1. A demonstration of our Cooperative Indoor Navigation (CIN) task. During the process of navigation, agents exchange the observed semantic information and explore the room toward different directions to find the target efficiently.

To overcome the limitations of previous multi-agent environments, we propose a novel task, Cooperative Indoor Navigation (CIN), where multiple agents are required to navigate to reach targets in a 3D indoor room. To the best of our knowledge, our CIN is the first multi-agent real-world navigation environment. Each agent in CIN observes an RGB-D image from its own first-person perspective and make actions to navigate. Thus, our CIN environment is closer to real-world robotic scenarios, whose realistic observation facilitates the learned agents to be deployed to real-world robotic applications. An overview of the CIN task with the communication mechanism is shown in Fig. 1. Three agents start from different positions and are asked to find a TV in this room. Each agent receives a first-person photo-realistic observation from their perspective respectively. To reach the target, they explore the entire room and communicate with each other in exchanging their discoveries. Through active exploration and cooperative communication, the No. 1 agent finally finds the target TV. To fully investigate the CIN task, we construct a large-scale dataset consisting of 24 million episodes for training, validation, and testing splits. Compared with other embodied navigation datasets [27, 32], our dataset is more challenging because we provide lengthier trajectory data.

The environments of the CIN task bring new challenges. For example, perceiving embodied vision information and Previous MARL methods [1, 12, 24, 31] adopt the Centralized Training Decentralized Execution (CTDE) framework, which ignores the communication among agents. To tackle this communication difficulty, we propose a new cooperative multi-agent communication model to enable the agents to exchange the latent embedding step-by-step during the whole navigation process. Our model first embeds the visual image by a Convolutional Neural Network as a visual embedding. The visual embedding and the word embedding of the global target are concatenated as latent embedding. Then our model adopts a Long short-term memory layer to embed

the latent embeddings in multiple steps into a historical latent embedding. For each step, agents exchange their historical embedding with each other and make the action prediction via a policy network.

Our environment validates that our model outperforms previous MARL methods quantitatively and qualitatively. We discover that without sufficient communication, the navigation performance will not increase by increasing the number of agents. And the communication module we propose could improve the overall success rate in reaching the target. We compare the navigation trajectories between the agents with inter-agent communication and without inter-agent communication and find that our method enables the agents to sufficiently explore the semantic knowledge of the entire room and achieve high performance. We experiment on four kinds variants of our model and find that communicating through historical latent embeddings and using them as input to generate the historical latent embeddings of the next step achieves the best performance (Table 1).

Table 1. Compared with existing MARL environments (N is the number of agents). The ego-centric view means that the agent has a local observation of the environment from its perspective rather than receiving global information. The embodied view represents whether an agent observes the 3D environment from the first-person perspective (we only compare the navigable environments).

Input	Environment	Multi-agent	Embodied	Observation Overlap	Physics Engine
Array	Hanabi [5]	✓	—	$\frac{N-1}{N}$	—
	Diplomacy [22]	✓	—	$\frac{N-1}{N}$	—
	EGCL [21]	✓	✗	100%	—
	DOTA2 [7]	✓	✗	100%	Rubikon
	SMAC [25]	✓	✗	81.0%–89.8%	Havok
	Hide-and-seek [4]	✓	✗	$\frac{1}{N}$ –100%	MuJoCo
	Soccer [18]	✓	✗	100%	MuJoCo
Syn image	MARLÖ [23]	✓	✓	$\frac{1}{N}$	hybrid voxel
	AI2-THOR [15]	✗	✓	$\frac{1}{N}$	Unity
Real image	Habitat [27]	✗	✓	$\frac{1}{N}$	Bullet
	CIN (Ours)	✓	✓	$\frac{1}{N}$	Bullet

2 Related Work

Multi-agent Environments have been proposed to research multi-agent problems. In Table 1, we compare the differences between our CIN environment with previous multi-agent environments. To the best of our knowledge, we claim that our CIN environment is the first multi-agent environment that offers realistic image input. Previous works [4, 7, 25] get clean and formatted array data via programming interfaces.

A particular challenge in the MARL problem is partial observability, where each agent could only observe a part of the global state. However, our investigation reveals that some of the previous MARL environments [4, 13, 25], even though claimed to be partially observable, have large observation overlaps. Little observations overlap makes our benchmark more challenging than the previous environments because little information sharing makes cooperation difficult. Our environment adopts the Bullet engine to simulate physics activities such as acceleration and collision.

Multi-agent Reinforcement Learning extends the problem of reinforcement learning [20, 30] into multi-agent scenario and brings new challenges. Most MARL problems [19] fall into the centralized training with decentralized execution (CTDE) architecture [36]. Some works are aiming at improving the network structure and developing an individual function with better representation and transfer capability [11, 12]. Besides, the utilization of state information varies. IPPO [33] incorporates the global state information barely by sharing network parameters among critics of individual agents. While MAPPO [38] constructs a centralized value function upon agents which takes the aggregated global state information as inputs. We discovered in this paper that exchanging historical information between agent facilitate the accuracy and efficiency of multi-agent navigation. And we have done extensive ablation experiments to validate this conclusion.

Embodied Navigation. Simulations such as Matterport3D simulator [3], Gibson simulator [35] and Habitat [27] propose high-resolution photo-realistic panoramic view to simulate the more realistic environment. Rendering frame rate is also important to embodied simulators since it is critical to training efficiency. MINOS [26] runs more than 100 frames per second (FPS), which is 10 times faster than its previous works. Habitat [27] runs more than 1000 FPS on 512×512 RGB+depth image, making it become the fastest simulator among existing simulators. Some complex tasks may require a robot to interact with objects, such as picking up a cup, moving a chair or opening a door. AI2-THOR [15], iGibson [34] and RoboTHOR [10] provide interactive environments to train such a skill. Multi-agent reinforcement learning [17] is a rising problem of cooperation and competition among agents. Based on the Habitat simulator, we construct a multi-agent environment to research realistic MARL problems.

3 Cooperative Indoor Navigation Task

3.1 Task Definition

Our Cooperative Indoor Navigation (CIN) task requires multiple agents $E = \{e_1, \dots, e_n\}$ to navigate to the target g_i within a common embodied environment starting from different locations. For each step, the agent observes an observation and makes an action decision. An action could be “turn left”, “turn right”, “step forward” to navigate, or “found” to declare the agent has found the target object and stops. The episode of this agent is considered successful if the “found” action is selected while the agent is located with the threshold toward the target object.

We propose two sub-tasks, shared-target navigation, and individual-target navigation. *Shared-target navigation* is a task where all agents are asked to find a shared target g . The task is considered successful if any agent reaches g and considered a failure if any agent fails within its episode. *Individual-target navigation* is a task where each agent i has its own target g_i , and the task is considered successful only when all agents successfully find their own target.

3.2 Multi-agent Indoor Navigation Environment

The CIN environment is built based on the Habitat [27] simulator. Our environment renders the 3D assets of a house and provides a common photo-realistic embodied environment for multiple navigation agents. Our implementation of the CIN environment is shown in Fig. 2. The CIN environment creates B sub-environments for decentralized execution to sample data for training, where B is the size of the minibatch. Each sub-environment creates N processes, where the N is the number of agents. Each process has a copy of a Habitat simulator, and each simulator individually simulates the state of an agent and renders the RGB-D image observation for an agent. The CIN sub-environment synchronizes the processes and interacts with a copy of a multi-agent navigation model. In the decentralized execution, the parameters of the multi-agent navigation model are shared across all sub-environments. The multi-agent navigation model predicts actions for each agent for each step. The predicted actions are sent to the CIN environment and then distributed to each process to execute. The CIN sub-environment calculates the global reward based on the global state and sends the global reward and observations to the model. For each step, the global reward, observations for all agents, and actions that agents predict are stored in the episode memory. We sample the episodes from the episode memory to optimize a centralized model by stochastic gradient descent (SGD). The model after a step of SGD optimization is copied to each CIN sub-environment to update the execution model.

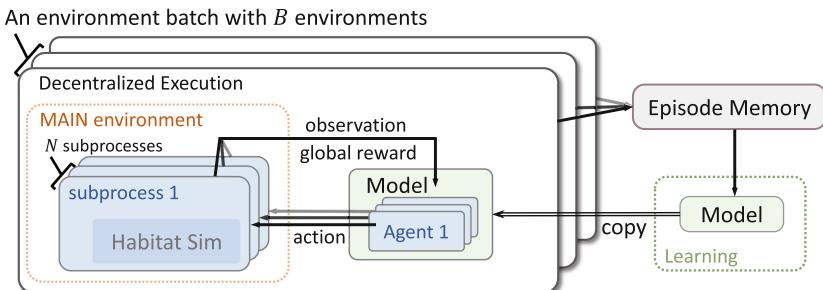


Fig. 2. Training overview of our CIN environment.

3.3 Data Collection

We use the room textures and other 3D assets provided by Matterport3D [8] to build the CIN environment. Matterport3D consists of 10,800 panoramic views constructed from 194,400 RGB-D images of 90 building-scale scenes, where 61 scenes for training, 11 for validation, and 18 for testing following the standard split [8]. Providing episode data for learning and testing, where an episode is defined by a randomly sampled starting position and target position within a navigable environment. And we constrain the length of an episode to be between 2m and 20m, which ensures that each episode is neither too trivial nor too hard.

We compare the distribution of the average trajectory length within a room with MultiON [32] and the object navigation data in the Habitat Challenge [6] in Fig. 3(a). Due to the different structures of the house, the episode data from each house have different average length. We find that with the same room setting, our average trajectory length is longer than both MultiON and Habitat, proving that our data is more challenging. Our dataset provide 24M episodes, 10 times more than the data scale of the Habitat dataset. The Fig. 3(b) shows the average distance that the agents need to navigate to successfully accomplish task. It reveals the gap of the difficulty among the two sub-tasks and the single-agent navigation task accompany with the agent amount using our dataset. With the increase of the agent amounts, the difficulty of individual-target task is significantly increasing while the shared-target task is reducing.

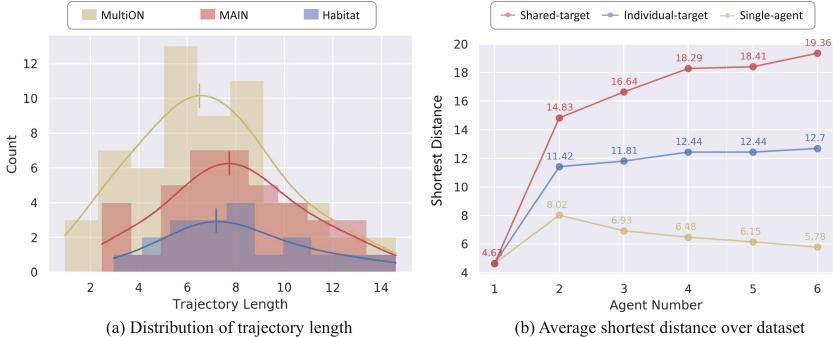


Fig.3. An analysis of our CIN dataset. (a) analysis of the data scale of trajectories and the distribution of the length of trajectories. The curve in (a) stands for the Gaussian smoothing, and the vertical lines are the mean values of the smoothed Gaussian distributions. (b) compares the navigation performance of the IPPO baseline in three-different environment settings.

4 Cooperative Indoor Navigation Models

4.1 Preliminaries

We model our CIN problem as a multi-agent reinforcement learning problem in a partially-observed Markov decision process (POMDP). $P(s'|s, a)$ is the transition probability that transforms the current state space S to the next state space S' conditions on

the a global action $a \in A$. We follow the centralized-training decentralized-execution framework that parameterize the shared policy of each agent as π_θ . For each step t , the agent i receive its partial observation $o_{t,i}$ and choose its action by $a_{t,i} = \pi_{\theta_i}(o_{t,i})$. The global action $a_t = a_{1,t}, \dots, a_{n,t}$ All agents share the same global reward function $r(s, a) : S \times A \rightarrow \mathbb{R}$. And the $\gamma \in [0, 1)$ is a discount factor that defines the length of the horizon. We optimize the parameter θ by minimizing the optimization objective $J(\theta) = \mathbb{E} [\sum_t \gamma^t r(s_t, a_t)]$ by PPO algorithm [29].

4.2 Framework

In this section, we are going to introduce our cooperative communicative navigation model, as shown in Fig. 4. The framework first embeds the target as an embedding feature and extracts visual features using a CNN network:

$$f_{i,t} = \text{CNN}(o_{i,t}), \quad (1)$$

where the f_i denotes visual embedding feature of the i th agent and the t denotes the t th timestep. The parameters of the embedding layer and the CNN layer are shared between agents to ensure the ability of generalization. Then the target feature and visual feature are concatenated to feed the Recurrent Neural Network (RNN) module:

$$g_{i,t} = \text{RNN}_i(f_{i,t}). \quad (2)$$

The historical feature from the RNN is sent to two fully connected layers.

$$\begin{aligned} a_{i,t} &= W_i^\pi(g_{i,t}), \\ v_{i,t} &= W_i^v(g_{i,t}). \end{aligned} \quad (3)$$

The W_π outputs the decision and the W_i^v predicts a value to estimate the effectiveness of the current situation. Note that the parameters of the RNN layer and the fully connected layers are not shared between agents. The action prediction is supervised by the clipped policy gradient loss [29], and the value prediction is supervised by the temporal difference of the bellman equation.

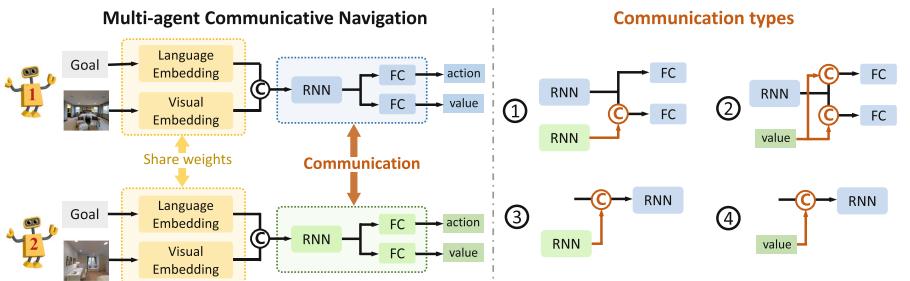


Fig. 4. A demonstration of our multi-agent communicative navigation framework using a two-agent model. The dual-directed orange arrow indicates weight sharing between orange boxes, while the orange arrow represents communication. On the right, we show four communication variants. (Color figure online)

Cooperative Communication. As shown in Fig. 4, the agents cooperatively communicate by exchanging the observed semantic information between the blue block and the green block. The information exchange is implemented by exchanging a embedding feature vector. The feature vector that an agent sent is named a “message”. The agent that receives the message is named the “receiver” and the agent that sends the message is named the “sender”. The gradient is not back-propagated from the ‘receiver’ to the ‘sender’ since it causes severe instability in training, which makes the performance of the learned navigation model to be almost zero. Different exchanging structure led to different experimental results. We design four types of communicative variants and compare their performance in our experiments, as demonstrated in Fig. 4 (right): 1) the sequential communication model (Comm-S), 2) value communication model (Comm-V), 3) recurrent message communication model (Comm-RM), and 4) recurrent value communication model (Comm-RV).

5 Experiment

5.1 Benchmarking CIN with MARL Models

We implement several multi-agent models to investigate the performance of multi-agent models on the CIN task.

Random Navigator with Oracle Founder. We implement a random baseline that randomly samples the action of “turn left”, “turn right” and “go forward”. This baseline model is used to validate if our dataset is too easy or has severe bias.

Multi-single Agent. This PPO-based [29] model, initially trained in a single-agent paradigm, is tested in a multi-agent setup. Our study investigates how altering the number of agents impacts navigation performance in this context.

IPPO. The IPPO [1] model learns the global reward and share network parameters with each agent. PPO gets a reward for a single agent, where as IPPO receives a global reward affected by other agents.

MAPPO. Based on IPPO, MAPPO [38] introduces a centralized value function upon agents with global state inputs without historical encoding and communication.

5.2 Implementation Details

Our communicative model is built based on our implementation of [1]. We train all of our models for 15M iterations. We adopt the Adam optimizer whose learning rate is 2.5×10^{-4} . The discount factor $\gamma = 0.99$ and the TD(λ) factor in GAE [28] is 0.95. The loss weight of the value function is 0.5. The hyperparameter for PPO clip is set to 0.2. Our model is trained on 8 GPUs (7 GPUs for rendering image inputs and 1 GPU for optimization) for 36 h.

5.3 Evaluation Metrics

Distance indicates the average distance forward the target when the agent stops.

Success Rate measures if the agent successfully finds the target when it yields ‘found’. The agent is regarded as ‘success’ only if it is located within 1m towards the target.

SPL, short for Success weighted by Path Length [2], evaluates the accuracy and efficiency simultaneously. The SPL is calculated by $\frac{1}{N} \sum_{i=1}^N S_i \frac{p_i}{l_i}$, where the N is all testing samples, S_i is the success indicator, p_i is the shortest path length, and the l_i is the actual path length in testing.

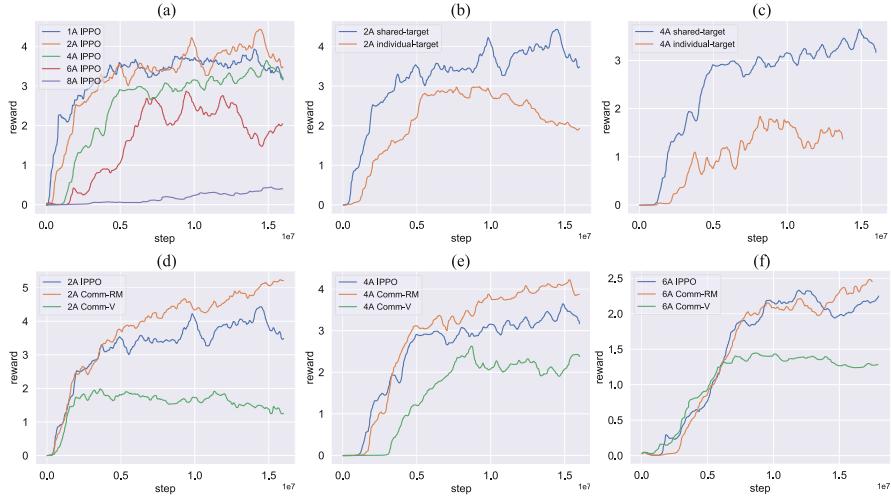


Fig. 5. The reward curves during the training process in the CIN environment: (a) ablates the agent number based on the IPPO baseline. (b) and (c) compare the shared-target task with the individual-target task. (d), (e) and (f) plot the reward curves of IPPO, Comm-RM, and Comm-V during the training process.

5.4 Quantitative and Qualitative Results

Ablation Study of Agent Amount. Here, we investigate if the amount of agents in cooperative navigation affects the navigation performance. Figure 5(a) shows the ablation results of the number of agents. The figure shows that the navigation setting with 2 agents achieves the best performance. The 2-agent model outperforms the 1-agent model indicating that multiple agents are able to find the target more accurately and efficiently than using a single agent. If the number of agents continues to increase, the navigation performance will progressively deteriorate, and the 8-agent model yields the poorest results. Increasing the number of agents can narrow down the search area for finding a target; however, optimizing 8 agents with a global reward is challenging. The actions of other agents can easily influence the gradients of each agent, making it difficult for the global reward to provide clear guidance to individual agents.

The Difficulty of Two Sub-tasks. Here, we compare the shared-target task and the individual-target task and investigate the differences between these two tasks. Figure 5(b), (c) ablate the difficulty of two sub-tasks with different settings. We train the IPPO baselines on individual-target tasks and shared-target tasks respectively. The blue curve represents the average reward from the model training for the shared-target navigation task, while the yellow curve represents the average reward for the individual-target navigation task. The model in the shared-target task significantly outperforms the model in the individual-target task by reward, which reveals that the individual-target task is harder than the shared-target task. Moreover, the reward curves in Fig. 5(b) are significantly higher than those in Fig. 5(c), indicating that the task difficulty is increasing with more agent amounts (Table 2).

Table 2. The comparison of our method with previous multi-agent methods.

Models	2 Agents				3 Agents			
	Length	Distance ↓	Success rate ↑	SPL ↑	Length	Distance ↓	Success rate ↑	SPL ↑
Random	3.39	12.75	0.00	0.00	3.30	16.67	0.00	0.00
Multi-PPO [29]	232.89	10.66	0.21	0.17	47.11	16.10	0.01	0.00
IPPO [1]	256.67	15.55	0.08	0.06	137.24	15.94	0.02	0.01
Comm-S	351.03	10.02	0.12	0.06	75.92	16.16	0.05	0.05
Comm-V	68.14	12.02	0.03	0.03	80.23	10.36	0.05	0.04
Comm-RM	309.7	12.78	0.3	0.23	312.86	14.59	0.13	0.06
Comm-RV	298.1	11.56	0.24	0.17	301.2	12.32	0.08	0.05

Effectiveness of Communication. We train the model with a historical communication mechanism (Comm-RM), the model with a value communication mechanism (Comm-V), and the IPPO baseline (IPPO) in different settings, including 2 agents, 4 agents, and 6 agents. The results are shown in Fig. 5(d), (e), (f). We discover that the reward curves of Comm-V and Comm-RM outperform the reward curves of other methods in all three figures. Moreover, there is a clear performance gap between the Comm-RM and Comm-V in all three figures, indicating that the historical communication mechanism is able to improve the performance of multi-agent navigation.

We detailed analysis of three baseline models (Random, Multi-PPO, and IPPO) and four variants of our proposed methods (Comm-S, Comm-V, Comm-RM, Comm-RV) in the shared-target task, as shown in Table 2. First of all, the Multi-PPO setting outperforms the IPPO method in the 2-agent setting while failing in the 3-agent setting. Secondly, we discover that our proposed method with communication mechanisms significantly outperforms the cooperative and non-cooperative baselines, which reveals that communication is a key to solving the multi-agent navigation problem. We find that the model with adopts historical information exchange (Comm-RM and Comm-RV), as shown in Fig. 4, outperforms the communication methods without considering the historical information such as Comm-S and Comm-V by the success rate and SPL.

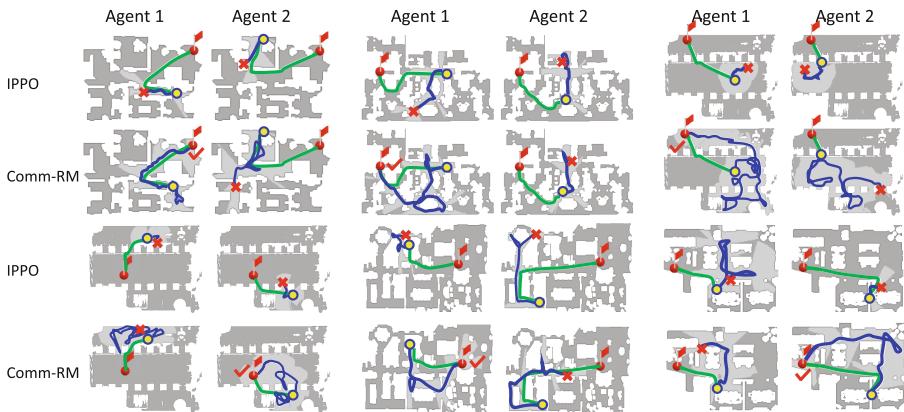


Fig. 6. The trajectory visualization results of the IPPO agent and the communicative agent in the testing environment. The red circle with a red flag shows where the target is located. The yellow circle is the starting position of an agent. The green line indicates the shortest path, and the blue line is the actual navigation path. (Color figure online)

Visualization for Navigation Process. We visualize the navigation trajectories of the IPPO model and our Comm-RM model in the shared-target task. The visualization results are shown in Fig. 6. Green lines represent the shortest path from start to end, while the blue line traces the actual navigation path of the agents. Notably, the blue lines in the second and fourth rows are notably longer than those in the first and third rows. This suggests that cooperative communication enhances the ability of the agents to explore larger areas and cover greater distances. Additionally, in the Comm-RM model, the navigation trajectories of different agents show minimal overlap, indicating effective communication that helps them avoid traversing through already-explored areas.

6 Conclusion

In this paper, we introduce the Cooperative Indoor Navigation (CIN) task for studying the multi-agent problem in realistic environments. We create a large-scale dataset for CIN research and highlight its advantages. Our analysis reveals that traditional MARL methods struggle with CIN challenges like limited observation overlap and high variance in embodied image views. We find that incorporating historical communication messages notably improves multi-agent navigation in CIN. Future work will focus on cooperative navigation using CIN and ongoing dataset updates.

References

1. Aiello, M., et al.: IPPO: A privacy-aware architecture for decentralized data-sharing (2020). [arXiv:2001.06420](https://arxiv.org/abs/2001.06420)
2. Anderson, P., et al.: On evaluation of embodied navigation agents (2018). [arXiv:1807.06757](https://arxiv.org/abs/1807.06757)

3. Anderson, P., et al.: Vision-and-language navigation: interpreting visually-grounded navigation instructions in real environments. In: CVPR (2018)
4. Baker, B., et al.: Emergent tool use from multi-agent autocurricula. In: ICLR (2020)
5. Bard, N., et al.: The Hanabi challenge: a new frontier for AI research. *Artif. Intell.* **280**, 103216 (2020)
6. Batra, D., et al.: ObjectNav revisited: On evaluation of embodied agents navigating to objects (2020). [arXiv:2006.13171](https://arxiv.org/abs/2006.13171)
7. Berner, C., et al.: Dota 2 with large scale deep reinforcement learning (2019). [arXiv:1912.06680](https://arxiv.org/abs/1912.06680)
8. Chang, A., et al.: Matterport3D: Learning from RGB-D data in indoor environments (2017). [arXiv:1709.06158](https://arxiv.org/abs/1709.06158)
9. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation (2014). [arXiv:1406.1078](https://arxiv.org/abs/1406.1078)
10. Deitke, M., et al.: RoboTHOR: an open simulation-to-real embodied AI platform. In: CVPR (2020)
11. Foerster, J., Farquhar, G., Afouras, T., Nardelli, N., Whiteson, S.: Counterfactual multi-agent policy gradients. In: AAAI (2018)
12. Hu, S., Zhu, F., Chang, X., Liang, X.: UPDeT: universal multi-agent RL via policy decoupling with transformers. In: ICLR (2021)
13. Ikram, K., Mondragón, E., Alonso, E., Garcia-Ortiz, M.: HexaJungle: a marl simulator to study the emergence of language (2021)
14. Khan, M.J., Ahmed, S.H., Sukthankar, G.: Transformer-based value function decomposition for cooperative multi-agent reinforcement learning in starCraft. In: Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment. vol. 18, pp. 113–119 (2022)
15. Kolve, E., Mottaghi, R., Gordon, D., Zhu, Y., Gupta, A., Farhadi, A.: AI2-THOR: An interactive 3D environment for visual AI (2017). [arXiv:1712.05474](https://arxiv.org/abs/1712.05474)
16. Lin, T., Huh, J., Stauffer, C., Lim, S.N., Isola, P.: Learning to ground multi-agent communication with autoencoders. *NeurIPS* **34**, 15230–15242 (2021)
17. Littman, M.L.: Markov games as a framework for multi-agent reinforcement learning. In: ICML (1994)
18. Liu, S., Lever, G., Merel, J., Tunyasuvunakool, S., Heess, N., Graepel, T.: Emergent coordination through competition. In: ICLR (2019)
19. Mahajan, A., Rashid, T., Samvelyan, M., Whiteson, S.: MAVEN: Multi-agent variational exploration (2019). [arXiv:1910.07483](https://arxiv.org/abs/1910.07483)
20. Mnih, V., et al.: Playing Atari with deep reinforcement learning (2013). [arXiv:1312.5602](https://arxiv.org/abs/1312.5602)
21. Mordatch, I., Abbeel, P.: Emergence of grounded compositional language in multi-agent populations. In: AAAI (2017)
22. Paquette, P., et al.: No press diplomacy: Modeling multi-agent gameplay (2019). [arXiv:1909.02128](https://arxiv.org/abs/1909.02128)
23. Pérez-Liébana, D., et al.: The multi-agent reinforcement learning in malmÖ (marlÖ) competition (2019). [arXiv:1901.08129](https://arxiv.org/abs/1901.08129)
24. Rashid, T., Samvelyan, M., Schroeder, C., Farquhar, G., Foerster, J., Whiteson, S.: QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In: ICML (2018)
25. Samvelyan, M., et al.: The StarCraft multi-agent challenge (2019). [arXiv:1902.04043](https://arxiv.org/abs/1902.04043)
26. Savva, M., Chang, A.X., Dosovitskiy, A., Funkhouser, T.A., Koltun, V.: MINOS: Multimodal indoor simulator for navigation in complex environments (2017). [arXiv:1712.03931](https://arxiv.org/abs/1712.03931)
27. Savva, M., et al.: Habitat: a platform for embodied AI research. In: ICCV (2019)
28. Schulman, J., Moritz, P., Levine, S., Jordan, M., Abbeel, P.: High-dimensional continuous control using generalized advantage estimation. In: ICLR 2016 (2016)

29. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms (2017). [arXiv:1707.06347](https://arxiv.org/abs/1707.06347)
30. Sutton, R.S., McAllester, D.A., Singh, S.P., Mansour, Y., et al.: Policy gradient methods for reinforcement learning with function approximation. In: NeurIPS (1999)
31. Wang, T., Gupta, T., Mahajan, A., Peng, B., Whiteson, S., Zhang, C.: RODE: Learning roles to decompose multi-agent tasks (2020). [arXiv:2010.01523](https://arxiv.org/abs/2010.01523)
32. Wani, S., Patel, S., Jain, U., Chang, A.X., Savva, M.: MultiON: Benchmarking semantic map memory using multi-object navigation (2020). [arXiv:2012.03912](https://arxiv.org/abs/2012.03912)
33. de Witt, C.S., et al.: Is independent learning all you need in the StarCraft multi-agent challenge? CoRR (2020)
34. Xia, F., et al.: Interactive Gibson benchmark: a benchmark for interactive navigation in cluttered environments. IEEE Robot. Autom. Lett. **5**(2), 713–720 (2020)
35. Xia, F., Zamir, A.R., He, Z., Sax, A., Malik, J., Savarese, S.: Gibson Env: real-world perception for embodied agents. In: CVPR (2018)
36. Yang, Y., et al.: Multi-agent determinantal Q-learning. In: ICML (2020)
37. Yu, C., Velu, A., Vinitsky, E., Wang, Y., Bayen, A.M., Wu, Y.: The surprising effectiveness of MAPPO in cooperative, multi-agent games (2021). [arXiv:2103.01955](https://arxiv.org/abs/2103.01955)
38. Yu, C., Velu, A., Vinitsky, E., Wang, Y., Bayen, A.M., Wu, Y.: The surprising effectiveness of MAPPO in cooperative, multi-agent games. CoRR (2021)
39. Zabounidis, R., Campbell, J., Stepputis, S., Hughes, D., Sycara, K.P.: Concept learning for interpretable multi-agent reinforcement learning. In: Conference on Robot Learning, pp. 1828–1837. PMLR (2023)



Enhancing Continuous Domain Adaptation with Multi-path Transfer Curriculum

Hanbing Liu[✉], Jingge Wang[✉], Xuan Zhang[✉], Ye Guo[✉], and Yang Li^(✉)

Tsinghua Shenzhen International Graduate School (SIGS), Tsinghua University,
Beijing, China

{liuhb21,wjg22}@mails.tsinghua.edu.cn,
{xuanzhang,guo-ye,yangli}@sz.tsinghua.edu.cn

Abstract. Addressing the large distribution gap between training and testing data has long been a challenge in machine learning, giving rise to fields such as transfer learning and domain adaptation. Recently, Continuous Domain Adaptation (CDA) has emerged as an effective technique, closing this gap by utilizing a series of intermediate domains. This paper contributes a novel CDA method, W-MPOT, which rigorously addresses the domain ordering and error accumulation problems overlooked by previous studies. Specifically, we construct a transfer curriculum over the source and intermediate domains based on Wasserstein distance, motivated by theoretical analysis of CDA. Then we transfer the source model to the target domain through multiple valid paths in the curriculum using a modified version of continuous optimal transport. A bidirectional path consistency constraint is introduced to mitigate the impact of accumulated mapping errors during continuous transfer. We extensively evaluate W-MPOT on multiple datasets, achieving up to 54.1% accuracy improvement on multi-session Alzheimer MR image classification and 94.7% MSE reduction on battery capacity estimation.

Keywords: Continuous Domain Adaptation · Wasserstein distance · Transfer curriculum · Optimal Transport · Path Consistency regularization

1 Introduction

Domain shift is a common challenge in many real life applications [15]. For example, in medical imaging, models trained on the data from one institution may not generalize well to another institution with different imaging hardware. Similarly, in battery capacity monitoring, models trained on lab-collected data may perform poorly under diverse operation environments. Obtaining the annotation for

Hanbing Liu, Jingge Wang, Xuan Zhang, and Yang Li are from the Shenzhen Key Laboratory of Ubiquitous Data Enabling, SIGS, Tsinghua University.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-981-97-2253-2_23.

the new domains, however, is often very costly or infeasible. To address this challenge, Unsupervised Domain Adaptation (UDA) has been proposed, leveraging the labeled data from the source domain to improve the performance of learning models on the unlabeled target domain [5]. In particular, UDA aims to align the distributions of the source and target domains using labeled source data and unlabeled target data, typically by learning domain-invariant representations [13] or adversarial learning schemes. Nevertheless, one challenge of UDA is its limited effectiveness when confronted with significant domain shift. Studies conducted by Zhao et al. [17] have shed light on the relationship between the domain shift and generalization error in UDA. They have shown that the effectiveness of UDA is bounded by the distributional divergence between the source and target domain, so the performance of the adapted model on the target domain may not be satisfactory with a substantial domain shift. In addressing this challenge, many works studied the problem of Continuous Domain Adaptation (CDA) [16].

Instead of directly adapting the model from the source to the target domain, CDA captures the underlying domain continuity leveraging a stream of observed intermediate domains, and gradually bridges the substantial domain gap by adapting the model progressively. There are various applications of CDA requiring continuous domains with indexed metadata [6,9]. For example, in medical data analysis, age acts as a continuous metadata for disease diagnosis across patients of different age groups. In the online battery capacity estimation problem, the state of health (SoH) of the battery acts as a continuous metadata that differs across batteries. CDA has attracted a great deal of attention and gained a rapid performance boost by self-training [7,18], pseudo-labeling [8], adversarial algorithms, optimal transport (OT) [10] and so on. In particular, Ortiz et al. [10] designed an efficient forward-backward splitting optimization algorithm for continuous optimal transport (COT) and demonstrated the efficacy of OT in reducing the domain shift and improving the performance of adaptation models.

While there have been significant advances in CDA, it still faces two critical challenges, determining the transfer order of intermediate domains in the absence of continuous metadata and mitigating cumulative errors throughout the continuous adaptation process. For the first issue, metadata could be missing or incorrect, and sometimes metadata alone can not fully explain the difference between data distributions. The proper ordering of intermediate domains is significant for CDA in transferring knowledge all the way to the target domain. Indeed, it is necessary to order the intermediate domains to facilitate continuous transfer without relying on explicit metadata. As a divergence measurement that takes into account the geometry of the data distributions, Wasserstein distance (w-distance) [14] plays an important role in deriving the generalization bound in domain adaptation, which implies the effectiveness of reducing domain divergence in the Wasserstein space. In this work, we propose a transfer curriculum in the Wasserstein space, aimed at determining the optimal sequence of intermediate domains for better knowledge transfer. The incorporation of Wasserstein-based transfer curriculum provides a principled and effective way to order the intermediate domains, enabling more precise and controlled knowledge

transfer. A more comprehensive discussion regarding the process of selecting the appropriate transfer sequence, which ultimately leads to a tighter generalization bound, will be provided in the method section.

For the second issue, as the model progressively adapts to new domains, errors can accumulate and degrade the overall performance. The accumulation of errors can occur during CDA due to the successive estimation of pseudo-labels or intermediate adaptation results, e.g., the error accumulates during each projection of source domain data based on the estimated optimal transport map in [10]. Fourier domain filtering possesses the capability to mitigate cumulative errors [4], but its efficacy is limited to fixed frequencies of errors, thereby exhibiting inadequacies in terms of flexibility and robustness. To tackle this challenge, we introduce a path consistency regularization scheme for OT-based CDA inspired by [10]. Our multi-path regularization scheme enforces consistency among multiple transfer paths, effectively reducing the impact of accumulated errors and improving the robustness and stability of the transferred model.

In summary, the main contribution of this work lies in four aspects:

- 1) **W-MPOT**: The paper proposes a novel CDA framework, named W-MPOT, which incorporates a Wasserstein-based transfer curriculum and multi-path consistency regularization, providing a principled and effective solution for CDA in scenarios where explicit metadata is unavailable.
- 2) **Wasserstein-based Transfer Curriculum**: The method employs w-distance to devise the transfer curriculum, providing theoretical proofs and generalization upper bounds on the error incurred by improper sorting based on w-distance.
- 3) **Multi-Path Optimal Transport**: The paper introduces a multi-path domain adaptation method based on Optimal Transport, namely MPOT, to enforce consistency among multiple adaptation paths. By mitigating the impact of accumulated errors during continuous transfer, MPOT significantly enhances the overall performance and stability of the adaptation process.
- 4) **Comprehensive Empirical Validation**: We conduct a thorough set of experiments to validate the motivation and effectiveness of our proposed methods. These experiments cover various domains, including *ADNI*, *Battery Charging-discharging Capacity* and *Rotated MNIST* datasets, and demonstrate the superiority of our approach compared to alternative methods.

2 Methodology

2.1 Preliminary

We employ optimal transport (OT) to map the source domain into the target domain. OT provides a measurement of the divergence between two probability distributions by finding an optimal transportation plan that minimizes the total cost of mapping one distribution to the other [2]. Detailed explanations regarding OT can be found in Supplementary A.

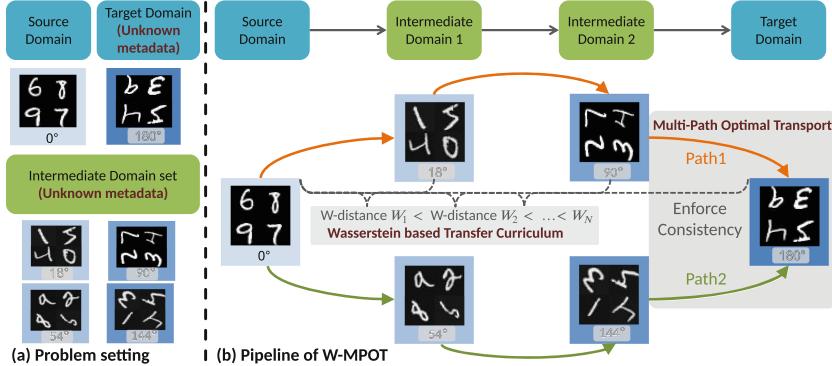


Fig. 1. Illustration of our proposed W-MPOT. (a) In the *Rotated MNIST* example, we are given a source domain of 0° and a target/intermediate domain set with unknown degrees. (b) In our proposed W-MPOT, we address the challenge of unknown domain metadata (angles) and perform CDA. We utilize Wasserstein based transfer curriculum to sort intermediate domains and employ MPOT to enforce consistency, thereby enhancing the transfer effectiveness.

Consider the scenario where a labeled source domain and a collection of unlabeled auxiliary domains are available. Let $\mathcal{X} \subset \mathbb{R}^d$ be the feature space and $\mathcal{Y} \subset \mathbb{R}$ be the label space. Denote the source domain and target domain as $D_S = \{(x_j, y_j)\}_{j=1}^{N_S}$, $D_T = \{x_j\}_{j=1}^{N_T}$, where N_S and N_T are the number of samples in the source domain and target domain, respectively. The candidate set of intermediate domains is denoted as $\mathcal{D}_I = \{D_{I_1}, D_{I_2}, \dots, D_{I_K}\}$, where each domain is denoted as $D_{I_k} = \{x_j\}_{j=1}^{N_{I_k}}$, $k = 1, \dots, K$ and N_{I_k} denotes the number of unlabeled data. Let $\mu_S, \mu_{I_k}, \mu_T \in \mathcal{P}(\mathcal{X})$ be the probability measures on \mathbb{R}^d for the source, intermediate, and target domains, respectively. The objective is to make predictions $\{\hat{y}_j\}_{j=1}^{N_T}$ for samples in the target domain.

While the order of intermediate domains in CDA can be determined using available metadata, the challenge arises when the metadata is absent, requiring further efforts to determine the proper order for CDA. Given the K candidate intermediate domains, a transfer curriculum defines an ordered sequence of N domains from \mathcal{D}_I , denoted as $\widehat{\mathcal{D}}_I$, to be used as intermediate domains for the CDA.

2.2 Method Framework

The framework of our proposed W-MPOT, comprising the Wasserstein-based transfer curriculum module and the Multi-Path Optimal Transport (MPOT) module, is depicted in Fig. 1. The Wasserstein-based transfer curriculum module selects an optimal intermediate domain sequence $\widehat{\mathcal{D}}_I$ for a given target domain D_T . Specifically, after calculating the w-distance [14] between each intermediate domain and the source domain, we sort intermediate domains closer to the source than the target domain by their w-distances. In the MPOT module, we

adopt COT [10] to transfer the source domain knowledge through the sorted intermediate domain sequence, and finally adapt to the target domain in an unsupervised manner. A multi-path consistency term is proposed to regularize the continuous adaptation. Since the divergence between the source and target domains is minimized by the OT-based adaptation process, the final prediction for the target domain can be derived by a regressor or classifier trained on the transported source domain.

2.3 Wasserstein-Based Transfer Curriculum

Properly ordering the intermediate domains ensures a more effective transfer of knowledge. Here we present the motivation for arranging intermediate domains in the Wasserstein space with a simple example.

Given the labeled source domain D_S , the target domain D_T and two candidate intermediate domains D_{I_1} and D_{I_2} . We assume that the optimal transfer order is $D_S \rightarrow D_{I_1} \rightarrow D_{I_2} \rightarrow D_T$, i.e., the intermediate domain D_{I_1} is closer to D_S than to D_{I_2} . Apparently, another possible transfer order would be $D_S \rightarrow D_{I_2} \rightarrow D_{I_1} \rightarrow D_T$. We first present the generalization bound proposed by [12] which relates the source and target errors using w-distance. Let $\mu_{I_1}, \mu_{I_2} \in \mathcal{P}(\mathcal{X})$ be the probability measures for domains D_{I_1}, D_{I_2} , respectively. h and f denote the predicted hypothesis and the true labeling function, respectively. $\epsilon_{\mu}(h, f) = \mathbb{E}_{x \sim \mu} [|h(x) - f(x)|]$ and ϵ is the combined error of the ideal hypothesis h^* . Assume that all hypotheses in the hypothesis set H satisfies the A -Lipschitz continuous condition for some A .

By simply applying the lemma proposed in [12] to each source-target pairs (D_S, D_{I_1}) , (D_{I_1}, D_{I_2}) and (D_{I_2}, D_T) , the generalization bound of the transfer path $D_S \rightarrow D_{I_1} \rightarrow D_{I_2} \rightarrow D_T$ could be derived as the following equation. For a detailed explanation of the lemma, please refer to Supplementary B.

$$\begin{aligned} \epsilon_{\mu_T}(h, f) &\leq \epsilon_{\mu_S}(h, f) + 2A \cdot \mathcal{W}_1(\mu_S, \mu_{I_1}) \\ &\quad + 2A \cdot \mathcal{W}_1(\mu_{I_1}, \mu_{I_2}) + 2A \cdot \mathcal{W}_1(\mu_{I_2}, \mu_T) + \epsilon, \end{aligned} \quad (1)$$

Similarly, for another transfer path $D_S \rightarrow D_{I_2} \rightarrow D_{I_1} \rightarrow D_T$, the generalization bound is

$$\begin{aligned} \epsilon_{\mu_T}(h, f) &\leq \epsilon_{\mu_S}(h, f) + 2A \cdot \mathcal{W}_1(\mu_S, \mu_{I_2}) \\ &\quad + 2A \cdot \mathcal{W}_1(\mu_{I_2}, \mu_{I_1}) + 2A \cdot \mathcal{W}_1(\mu_{I_1}, \mu_T) + \epsilon, \end{aligned} \quad (2)$$

where ϵ is the same. According to the optimal transfer path assumption, it is straightforward to get $\mathcal{W}_1(\mu_S, \mu_{I_2}) > \mathcal{W}_1(\mu_S, \mu_{I_1})$, and $\mathcal{W}_1(\mu_{I_1}, \mu_T) > \mathcal{W}_1(\mu_{I_2}, \mu_T)$. Therefore, better domain transfer order $D_S \rightarrow D_{I_1} \rightarrow D_{I_2} \rightarrow D_T$ will lead to tighter generalization bound. The optimal transfer path order will achieve better performance on the target domain than the other, which justifies the use of the w-distance in our transfer curriculum.

This result leads to our Wasserstein-based transfer curriculum to select and sort for intermediate domains. We use the following form of w-distance to measure the closeness between each intermediate domain D_{I_k} and the source domain D_S ,

$$\begin{aligned} W_k &= \min_{\gamma} \langle \gamma_k, \mathbf{M}_k \rangle_F + \lambda \cdot \Omega(\gamma_k) \\ \text{s.t. } \gamma_k \mathbf{1} &= \mu_S \quad \gamma_k^T \mathbf{1} = \mu_{I_k}, k = 1, \dots, K, \end{aligned} \quad (3)$$

where $\gamma_k \geq 0$ is the transport matrix and \mathbf{M}_k is the cost matrix between the distribution μ_S and μ_{I_k} . W_k measures the similarity between each intermediate domain and the source domain, i.e., the greater the W_k is, the farther the intermediate domain is from the source domain. Note that any intermediate domain that is further from the source domain than the target domain is discarded. The remaining N domains in the intermediate domain set are then sorted in order of W_k and we could obtain a domain sequence $\widehat{\mathcal{D}}_I = \widehat{D}_{I_1} \rightarrow \widehat{D}_{I_2} \rightarrow \dots \rightarrow \widehat{D}_{I_N}$, where $W_1 \leq W_2 \dots \leq W_N$. As a result, the Wasserstein-based transfer curriculum generates a sorted transfer sequence $\widehat{\mathcal{D}}_I$ that represents the desired order of the domains, arranged from those closer to the source domain to those farther away. By utilizing the w-distance to sort multiple intermediate domains, we eliminate the need for meta-information.

2.4 Multi-path Optimal Transport

We apply OT-based domain adaptation iteratively across each intermediate domain indexed by the proposed curriculum. In order to make predictions for the target domain D_T , MPOT constructs sequential transport plans, denoted as γ_m , through a series of successive steps from the source to the target domain. The steps are as follows. Given the sorted sequence of intermediate domain $\widehat{\mathcal{D}}_I$, the source domain is initially mapped to the first intermediate domain \widehat{D}_{I_1} using direct OT [2] and we can derive the first transport plan γ_0 . Following this, the barycentric mapping of the source domain to the first intermediate domain \widehat{D}_{I_1} can be defined by a weighted barycenter of its neighbors, $\mathcal{B}_{I_1}(D_S) = N_S \cdot \gamma_0 \cdot \widehat{D}_{I_1}$. The distribution of the mapped source domain on \widehat{D}_{I_1} is denoted as μ_{SI_1} , which is consistent with the distribution of the target intermediate domain μ_{I_1} .

Then for the following intermediate domains, $\widehat{D}_{I_n}, n \in 2, \dots, N$, the probabilistic coupling γ_{n-1} between the domain $\widehat{D}_{I_{n-1}}$ and the subsequent domain \widehat{D}_{I_n} is calculated using COT [10],

$$\begin{aligned} \gamma_{n-1} &= \underset{\gamma \in \mathbb{R}^{N_S \times N_T}}{\operatorname{argmin}} \left\langle \gamma, \mathbf{M}^{[n-1, n]} \right\rangle + \lambda \Omega(\gamma) + \eta_t R_t(\gamma) \\ \text{s.t. } \gamma \mathbf{1} &= \mu_{SI_{n-1}} \quad \gamma^T \mathbf{1} = \mu_{I_n} \quad \gamma \geq 0, \end{aligned} \quad (4)$$

where $\mathbf{M}^{[n-1, n]}$ is the cost matrix defining the cost to move mass from the distribution of $\mu_{SI_{n-1}}$ to μ_{I_n} . $\Omega(\cdot)$ denotes the entropic regularization term $\Omega(\gamma) = \sum_{i,j} \gamma_{i,j} \log(\gamma_{i,j})$, and $\lambda > 0$ is the weight of entropic regularization. $R_t(\cdot)$ is a time regularizer with coefficients $\eta_t > 0$, which aims to enforce smoothness and coherence across consecutive time steps [10].

Upon the completion of the sequential transfer from the source domain to all intermediate domains, the transport plan γ_N from \widehat{D}_{I_N} to the target domain D_T will be computed using MPOT with the equation as follows,

$$\begin{aligned} \gamma_N &= \underset{\gamma \in \mathbb{R}^{N_S \times N_T}}{\operatorname{argmin}} \left\langle \gamma, \mathbf{M}^{[N, N+1]} \right\rangle + \lambda \Omega(\gamma) + \eta_t R_t(\gamma) + \eta_p R_p(\gamma, \gamma_{p_2}) \\ \text{s.t. } \gamma \mathbf{1} &= \mu_{SI_N} \quad \gamma^T \mathbf{1} = \mu_T \quad \gamma \geq 0, \end{aligned} \quad (5)$$

where $\mathbf{M}^{[N, N+1]}$ is the cost matrix between the distribution μ_{SI_N} and μ_T . $\eta_t > 0$ and $\eta_p > 0$ are the coefficients to adjust the weights. We further introduce a path consistency regularizer $R_p(\cdot)$ by comparing it with another transfer path,

$$R_p(\gamma, \gamma_{p_2}) = \left\| N_S \cdot \gamma \cdot \widehat{D}_{I_n} - N_S \cdot \gamma_{p_2} \cdot \widehat{D}_{I_n} \right\|_F^2, \quad (6)$$

where γ_{p_2} is the transport plan of the second possible path which is utilized to refine the γ of Path 1. During each adaptation step across intermediate domains in the curriculum, the incremental steps introduce minor inaccuracies, potentially impacting overall transfer performance over time. By employing $R_p(\cdot)$, we exploit the complementary information present in diverse paths to alleviate the accumulation of errors. The continuous adaptation results of the original COT and our proposed MPOT on the simulated half-moon dataset are shown in Fig. 2.

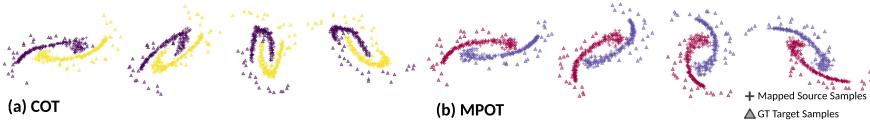


Fig. 2. The Effect of Multi-Path regularization on the Optimal Transport of source domain sample. Visual experiments are conducted on simulated half-moon data to compare the migration effects of (a) COT with our proposed (b) MPOT. The source domain has an angle of 0° , and the angles on the graph increase by 36° from left to right. (a) and (b) depict the mapping results obtained by applying the COT and MPOT methods respectively to sequentially map the source domain to the target domains. The triangles represent the ground truth target domain data, while the plus signs represent the mapped source domain data after the adaptation.

During continuous adaptation, the COT method experiences a substantial increase in cumulative error, leading to deteriorating adaptation results as the rotation angle increases. In contrast, our proposed method, MPOT, effectively addresses the issue of cumulative errors by leveraging multiple transfer paths.

Notably, the paths interact in a bidirectional manner within $R_p(\cdot)$: not only does Path 2 impose constraints on Path 1, but Path 1 likewise restricts Path 2. This creates a reciprocal dynamic and both paths can be jointly optimized simultaneously. Moreover, the above OT problem (5) fits into the forward-backward splitting algorithm [1] as in [10], whose solution could be efficiently computed with the Sinkhorn algorithm [3]. More details about the solution for the Sinkhorn algorithm are given in Supplementary C. Let $\mathcal{J}(\cdot)$ be the combination of

Algorithm 1: Bidirectional Optimization algorithm in MPOT

Input: Transport matrix of Path 1 γ_{p_1} , Transport matrix of Path 2 γ_{p_2} , step size α , cost matrix of Path 1 $\mathbf{M}^{[1]}$, cost matrix of Path 2 $\mathbf{M}^{[2]}$, weight of Path 1 λ_1 and Path 2 λ_2 , iteration times c

Output: Refined transport matrix from N -th intermediate domain to target domain γ'_N

```

1 Initialize:  $\gamma_0 \in (0, +\infty)^{N_S \times N_T}$ 
2 for  $c \leftarrow 0, 1, \dots$  do
3    $\mathbf{M}_c^{[1]} = \alpha \mathbf{M}^{[N, N+1]} + \alpha \nabla \mathcal{J}(\gamma_c, \gamma_{p_2})$ 
4    $\mathbf{M}_c^{[2]} = \alpha \mathbf{M}^{[N, N+1]} + \alpha \nabla \mathcal{J}(\gamma_c, \gamma_{p_1})$ 
5    $\mathbf{M}_c = \lambda_1 \mathbf{M}_c^{[1]} + \lambda_2 \mathbf{M}_c^{[2]}$ 
6    $\gamma_{c+1} = \text{Sinkhorn}(\mathbf{M}_c, 1 + \alpha \lambda, \mu_{SI_N}, \mu_T)$ 
7  $\gamma'_N = \gamma_\infty$ 

```

regularization terms. The details of the bidirectional optimization procedure are depicted in the subsequent Algorithm 1.

Once the refined transport matrix, denoted as γ'_N , has been computed, the barycentric mapping from the mapped source domain to the target domain can be obtained by using $\mathcal{B}_T(D_S^{I_N}) = N_S \cdot \gamma'_N \cdot D_T$, where $D_S^{I_j}$ denotes the domain mapped from the source domain D_S to \widehat{D}_{I_j} . Consequently, a classifier or regressor can be trained on $\mathcal{B}_T(D_S^{I_N})$ using the available source labels and can be directly deployed on the target domain for various applications.

3 Experimental Results

3.1 Datasets and Experimental Configurations

Experiments are conducted on three datasets, each offering unique characteristics and challenges. Details regarding the implementation are presented in Supplementary D.

ADNI. The Alzheimer’s Disease Neuroimaging Initiative (ADNI) dataset is a crucial resource for Alzheimer’s disease research, housing a repository of MRI images that furnish in-depth structural insights into the brain [11]. The 3D MRI data is sliced along the direction of the vertical spinal column, resulting in 2D MRI images. The 2D MRI images are categorized into a source domain (ages 50–70, 190 samples) and intermediate domains (ages 70–72, 72–74, 74–76, 76–78, 78–82, 82–92, 50 samples each). The primary task is classifying MRI images into five disease categories. In our ADNI MRI image experiments (128×128 dimensions), we performed image normalization to a 0–1 range. Subsequently, we utilized a pre-trained VGG16 model, originally trained on the ImageNet dataset, to extract image features. The VGG16 model’s output consists of a

$512 \times 4 \times 4$ feature map, which is condensed into a 16-dimensional feature vector by averaging across the channel dimension.

Battery Charging-Discharging Capacity. The capacity of lithium-ion batteries holds paramount significance in the context of power systems and electric vehicles. Laboratory experiments are conducted via a pulse test [19] to collect voltage-capacity data pairs on batteries during charging and discharging processes at different state of charge (SoC) levels. The dataset includes a source domain (5% SoC) and nine intermediate domains (10%–50% SoC), each with 67 samples. It is a regression problem evaluated with Mean Squared Error (MSE).

Rotated MNIST. The Rotated MNIST dataset is a variation of the MNIST dataset, featuring rotated digit images. It is divided into five domains with different rotation angles. The source domain (0°C) and intermediate domains (18° , 36° , 54° , 72° , and 90°C) each contain 1000 samples.

3.2 Analysis of Wasserstein-Based Transfer Curriculum

This section assesses the effectiveness of our proposed Wasserstein-based transfer curriculum, comparing it with two alternative domain adaptation methods. We examine Directly Optimal Transport (DOT), which transfers data directly from the source to the target domain, and Continuously Optimal Transport (COT), a progressive adaptation approach. COT is explored in two configurations: COT+Metadata, utilizing genuine domain metadata for sorting intermediate domains, and COT+W-dis, employing our Wasserstein-based transfer curriculum in the absence of metadata. In our experimental design, we maintain consistency between the source and target domains across all methods.

As shown in Fig. 3, the superior performance of COT over DOT, coupled with the comparable performance between COT utilizing w-distance and COT with true metadata, shows the effect of intermediate domain sorting. The impact of varying the number of intermediate domains differs across datasets, which is determined by the inherent characteristics of the data itself. These findings highlight the benefits of incorporating COT and the Wasserstein distributional geometric relationships.

3.3 Adaptation Comparison Results

A comprehensive comparison of our proposed method, W-MPOT, is conducted with several classic approaches in the realm of CDA. We present the comparison results in Table 1.

In our experimental setup, we maintain a fixed number of two intermediate domains, and each experiment is repeated 100 times to ensure robustness, with the average results reported. In the bottom section of Table 1, we showcase the outcomes of our proposed algorithm, W-MPOT, applied in three distinct scenarios: $p_2 \rightarrow p_1$, $p_1 \rightarrow p_2$, and $p_1 + p_2$. These scenarios involve using one path

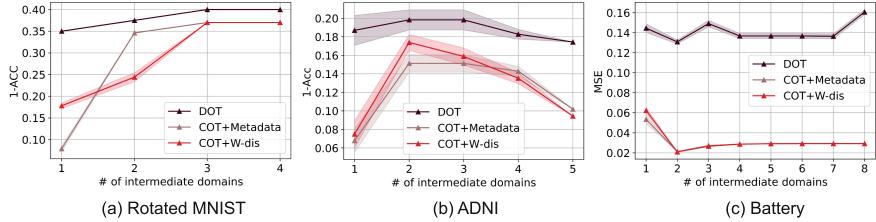


Fig. 3. Domain ordering Results. The results for the (a)Rotated MNIST, (b)ADNI, and (c)Battery Charging-discharging Capacity datasets are shown in this figure. The evaluation metric for Rotated MNIST and ADNI is accuracy, while for the Battery dataset is MSE. The methods compared are DOT (Direct Optimal Transport) and COT (Continuous Optimal Transport). Two sorting approaches are utilized: metadata-based sorting and Wasserstein based transfer curriculum. For ease of comparison, the y-coordinate of the first two datasets represents $1 - ACC$, where lower values indicate superior performance. Each configuration is evaluated 100 times, and the shaded areas represent the variance.

to refine the other and utilizing both paths for a new refined path. Bilateral experiments are conducted by applying a mutual constraint mechanism using regularizers from both paths.

The results indicate the superior performance of W-MPOT over other methods across all three datasets, underscoring its efficacy in mitigating continuous domain shifts. Notably, W-MPOT using both path 1 and path2 shows the optimal performance, providing strong evidence for the effectiveness of the added regularization term $R_p(\cdot)$. The results of W-MPOT in other two scenarios: p2→p1 and p1→p2, showing similar performance, suggest that the regularization approach is robust and does not heavily rely on the specific choice of the second path. By leveraging both paths with mutual constraints, W-MPOT successfully improves the overall robustness of the model.

Table 1. MSE or Accuracy for three datasets of different algorithms

Method	ADNI (\uparrow)	Battery (\downarrow)	ROT MNIST (\uparrow)
Source Model	41.2	0.3731	48.1
CMA [6]	55.4	0.3842	65.3
EAML [9]	68.3	0.2045	70.4
AGST [18]	57.3	0.3534	76.2
Gradual ST [7]	64.5	0.1068	87.9
CDOT [10]	82.6	0.0209	75.6
W-MPOT(p2→p1)	<u>86.7</u>	0.0199	<u>88.3</u>
W-MPOT(p1→p2)	86.5	<u>0.0197</u>	87.2
W-MPOT(p1 + p2)	88.3	0.0185	89.1

3.4 Ablation Study

We conducted ablation studies on the *Battery Charging-discharging Capacity* dataset to investigate the effect of domain partitioning, Wasserstein-based sorting strategy, and path consistency regularization separately. Delimited domains consistently exhibit lower mean and variance of MSE compared to random batch sampling in Fig. 4(a), indicating increased stability. This highlights the necessity of considering distinct domains for improved predictive accuracy. Comparing Unordered COT to Ordered COT reveals consistently lower MSE values in Fig. 4(b), emphasizing the value of the Wasserstein-based transfer curriculum for superior performance in battery capacity prediction tasks. The vital role of the path consistency regularization term in achieving accurate and robust domain mappings is demonstrated by the comparison of MPOT (with $R_p(\cdot)$) and COT (without $R_p(\cdot)$) in Fig. 4(c).

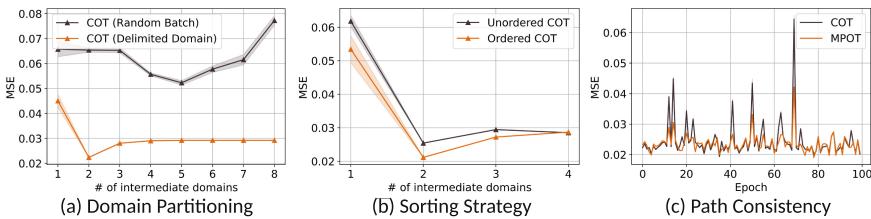


Fig. 4. Ablation study Results. The experiments from a to b involved domain partitioning, sorting strategy, and path consistency. For each number of intermediate domains in a and b, the experiment was randomly repeated 100 times. The solid line represents the mean MSE, while the shaded area represents the variance. The experiment in c is conducted by fixing the number of intermediate domains to 2 and randomly sampling different intermediate domains 100 times.

4 Conclusion

This research introduces the W-MPOT framework for CDA, effectively addressing substantial domain shifts and missing metadata. It comprises the Wasserstein-based Transfer Curriculum for domain ordering and MPOT for cumulative errors during adaptation. Experimental results across diverse datasets demonstrate its superior performance, highlighting the practicality and potential of these methods in handling substantial domain shift challenges. This work advances the field of CDA and offers insights into addressing domain shift effectively, especially in domains like healthcare and energy storage. Future research could focus on establishing generalization bounds and employing reinforcement learning to optimize the selection of intermediate domains, making domain adaptation a sequential decision-making process.

Acknowledgement. This work was supported in part by the Natural Science Foundation of China (Grant 62371270), Shenzhen Key Laboratory of Ubiquitous Data Enabling (No.ZDSYS20220527171406015), and Tsinghua Shenzhen International Graduate School Interdisciplinary Innovative Fund (JC2021006).

References

1. Büi, M.N., Combettes, P.L.: Bregman forward-backward operator splitting. *Set-Valued Variat. Anal.* **29**, 583–603 (2021)
2. Courty, N., Flamary, R., Tuia, D., Rakotomamonjy, A.: Optimal transport for domain adaptation (2016)
3. Cuturi, M.: Sinkhorn distances: lightspeed computation of optimal transport. *Adv. Neural Inf. Process. Syst.* **26** (2013)
4. Durak, L., Aldirmaz, S.: Adaptive fractional fourier domain filtering. *Signal Process.* **90**(4), 1188–1196 (2010)
5. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. In: International Conference on Machine Learning, pp. 1180–1189. PMLR (2015)
6. Hoffman, J., Darrell, T., Saenko, K.: Continuous manifold based adaptation for evolving visual domains. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 867–874 (2014)
7. Kumar, A., Ma, T., Liang, P.: Understanding self-training for gradual domain adaptation. In: International Conference on Machine Learning, pp. 5468–5479. PMLR (2020)
8. Liang, J., Hu, D., Feng, J.: Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In: International Conference on Machine Learning, pp. 6028–6039. PMLR (2020)
9. Liu, H., Long, M., Wang, J., Wang, Y.: Learning to adapt to evolving domains. *Adv. Neural. Inf. Process. Syst.* **33**, 22338–22348 (2020)
10. Ortiz-Jiménez, G., El Gheche, M., Simou, E., Maretic, H.P., Frossard, P.: Forward-backward splitting for optimal transport based problems. In: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5405–5409. IEEE (2020)
11. Petersen, R.C., et al.: Alzheimer's disease neuroimaging initiative (adni): clinical characterization. *Neurology* **74**(3), 201–209 (2010)
12. Shen, J., Qu, Y., Zhang, W., Yu, Y.: Wasserstein distance guided representation learning for domain adaptation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
13. Sun, B., Feng, J., Saenko, K.: Correlation alignment for unsupervised domain adaptation. In: Domain Adaptation in Computer Vision Applications, pp. 153–171 (2017)
14. Villani, C., Villani, C.: The wasserstein distances. In: Optimal Transport: Old and New, pp. 93–111 (2009)
15. Wang, Y., et al.: An empirical study of selection bias in pinterest ads retrieval. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 5174–5183 (2023)
16. Xu, Y., Jiang, Z., Men, A., Liu, Y., Chen, Q.: Delving into the continuous domain adaptation. In: Proceedings of the 30th ACM International Conference on Multimedia, pp. 6039–6049 (2022)

17. Zhao, H., Des Combes, R.T., Zhang, K., Gordon, G.: On learning invariant representations for domain adaptation. In: International Conference on Machine Learning, pp. 7523–7532. PMLR (2019)
18. Zhou, S., Wang, L., Zhang, S., Wang, Z., Zhu, W.: Active gradual domain adaptation: dataset and approach. IEEE Trans. Multimedia **24**, 1210–1220 (2022)
19. Zhou, Z., et al.: A fast screening framework for second-life batteries based on an improved bisecting k-means algorithm combined with fast pulse test. J. Energy Storage **31**, 101739 (2020)

Graphs and Networks



Enhancing Network Role Modeling: Introducing Attributed Multiplex Structural Role Embedding for Complex Networks

Lili Wang¹(✉), Chenghan Huang², Ruiye Yao³, Chongyang Gao⁴,
Weicheng Ma¹, and Soroush Vosoughi¹

¹ Dartmouth College, Hanover, NH 03755, USA

{lili.wang.gr,soroush}@dartmouth.edu

² Millennium Management, LLC, New York, NY 10022, USA

³ University of California, Davis, Davis, CA 95616, USA

⁴ Northwestern University, Evanston, IL 60208, USA

Abstract. Numerous studies have focused on defining node roles within networks, producing network embeddings that maintain the structural role proximity of nodes. Yet, these approaches often fall short when applied to complex real-world networks, such as Twitter, where nodes have varying types of relationships (e.g., following, retweeting, replying) and possess relevant attributes impacting their network role (e.g., user profiles). To address these limitations, this study presents a novel method for attributed (for dealing with attributed nodes) multiplex (for dealing with networks with different types of edges) structural role embedding. This approach uses an autoencoder mechanism to concurrently encode node structure, relationships, and attributes, thus successfully modeling nodes' roles within networks. Our method's effectiveness is shown through quantitative and qualitative analyses conducted on synthetic networks, outperforming established benchmarks in identifying node roles within multiplex and attributed networks. Additionally, we have assembled a robust real-world multiplex network composed of almost all verified Twitter users comprised of retweet, reply, and followership interactions between these users, representing three different layers in our multiplex network. This network serves as a practical environment to evaluate our method's capability to map the structural roles of users within real-world attributed multiplex networks. Using a verified dataset of influential users as a reference, we show our method excels over the existing benchmarks in learning structural roles on large-scale, real-world attributed multiplex networks, exemplified by our Twitter network.

1 Introduction

In recent years, social media has increasingly become a staple component of the public sphere, playing a vital role in a multitude of political, social, and cultural movements. This significant influence mandates the evolution of more robust tools to enhance our understanding of their operations. A particular aspect of

social media that necessitates further exploration is users' various roles within these networks. Effective information retrieval and recommendation in online social networks often depend on strategies that compare users on these platforms. Accurate user similarity computations can notably enhance cold-start recommendation systems. Beyond this, user representations are also crucial in pinpointing influential entities within expansive social networks, which isn't simply about tallying followers. Yet, going beyond rudimentary techniques, such as relying merely on user metadata, poses a challenge. Defining and formalizing user roles on social media is complex, given that these roles organically evolve as users participate in actions like sharing, replying, and following. These roles, in their essence, are multifaceted and determined by a user's position within diverse networks, such as followership, retweeting, and reply networks.

Existing methods, referred to as structural role network embedding, facilitate the identification of nodes' roles in networks. Here, "structural role" refers to a node's functional position, for example, as a hub or a bridge between communities. Distant or even disconnected nodes might exhibit identical or closely related structural roles. Methods such as struc2vec [16] and GraphWave [3] represent nodes as vectors encapsulating role-specific information. Such representations are pivotal for network analysis; for instance, the cosine similarity between two vectors can indicate the structural role similarity of the nodes they represent.

Nevertheless, existing methods primarily focus on singular-relationship networks and overlook node attributes. Contrarily, social media roles rely on a fusion of users' personal information and interactions within networks comprising various types of relationships. In other words, the roles are defined on multiplex and attributed networks. The domain of attributed multiplex structural role embedding remains relatively unexplored. Addressing this void, we introduce a formal definition of attributed multiplex structural roles and advance a novel method for learning their representation. Our paper puts forth the following contributions: (1) We introduce the first method for embedding attributed multiplex structural roles in networks. (2) To gauge the effectiveness of our embedding method both qualitatively and quantitatively, we have created a series of synthetic datasets. Our model consistently outperforms several established benchmarks in all evaluations. (3) We have curated a large real-world multiplex dataset containing nearly every verified Twitter account, summing up to approximately 360K accounts. This dataset encapsulates their posts, followers, retweets, and replies interactions. With this vast dataset, we validate the efficacy of our method in identifying structural roles in expansive, real-world attributed multiplex networks.

2 Related Work

Network embeddings convert nodes in a network into vector representations, with different embeddings capturing unique aspects of the nodes (e.g., see [6, 7, 12, 14, 19, 22–26]). Most of these embeddings rely on local homogeneous methods such as DeepWalk [14], node2vec [6], LINE [19], HOPE [12], GraRep [1], and SDNE [21]. These methods chiefly determine node similarity through their shared neighbors.

In contrast, structural role embedding methods like struc2vec [16], GraphWave [3], SEGK [11], RESD [31], and DRNE [20] aim to capture the specific

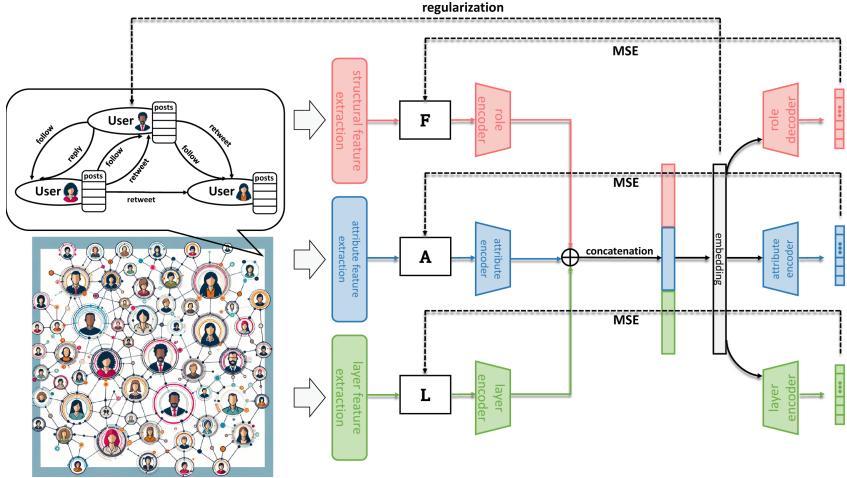


Fig. 1. The framework of our proposed method.

“roles” nodes play within a network. For instance, nodes, even if they are distant or disconnected, might have similar roles, such as serving as the hub of a community or acting as a bridge between two distinct groups. However, a limitation of these methods is their primary design for homogeneous networks, which have a singular edge type and lack node attributes.

Multiplex network embedding methods, such as MNE [29] and MVE [15], were developed to cater to the need for embeddings that can capture diverse relationships. Concurrently, there was the rise of heterogeneous network embedding techniques like metapath2vec [2], HIN2Vec [5], and JUST [8]. The next evolution incorporated node attributes into multiplex embeddings, illustrated by methods like DMGI [13] and URAMN [30]. Notably, these multiplex embedding methods integrated attributes but did not address structural roles.

There is a gap in the existing literature. While there are methods addressing attributed multiplex embedding, they focus on a granular level of embeddings. On the other hand, embeddings that preserve structural roles are still tied to homogeneous networks. This leaves a significant void when analyzing user roles on social media platforms with the nuanced lens of attributed multiplex structural role embedding. Our proposed approach endeavors to fill this crucial gap.

3 Methodology

An attributed multiplex network \mathbf{G} is defined as: $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathcal{T}_E, \mathcal{W}_E, \mathcal{A}_V)$, in which edges E are associated with two mapping functions $\phi(\mathbf{E}) : \mathbf{E} \rightarrow \mathcal{T}_E$, $\mathcal{T}_E \in \mathbb{N}$, $\psi(\mathbf{E}) : \mathbf{E} \rightarrow \mathcal{W}_E$, $\mathcal{W}_E \in \mathbb{R}$, and nodes \mathbf{V} are associated with a mapping function $\eta(\mathbf{V}) : \mathbf{V} \rightarrow \mathcal{A}_V$, $\mathcal{A}_V \in \mathbb{R}^m$. The \mathcal{T}_E denotes the possible set of relation types of the edges, the \mathcal{W}_E denotes the edge weights and the \mathcal{A}_V denotes the possible set of attributes of the nodes. Furthermore, for a node $v \in \mathbf{V}$, $\mathcal{N}_k(v) =$

$\{u \in \mathbf{V} \mid \exists \text{ a path from } v \text{ to } u, \text{with length } k\}$ is the set of its k -hop neighbors. For a node v , we consider the task of learning a structural role preserving d -dimensional embedding, $\mathbf{X}_v \in \mathbb{R}^d, d \ll |\mathbf{V}|$, in an unsupervised fashion.

3.1 Learning Framework

Our learning framework, inspired by RESD and illustrated in Fig. 1, comprises four key components: role-related structural feature extraction, layer feature extraction, attribute feature extraction, and an auto-encoder mechanism. The auto-encoder mechanism jointly encodes these three types of features into a singular embedding.

Role-related Structural Feature Extraction. This module aims to extract the pure structural role identity of nodes. To achieve this, we adopt Forman curvature [4], an essential tool in discrete differential geometry, quantifying the local geometric properties inherent to a network. As a discrete analog to the traditional concept of curvature, it measures the interconnectedness and relationships between vertices, edges, and faces, encapsulating local and global geometric features.

Considering that the mathematical representation of Forman-Ricci curvature is highly technical and less applicable to the current context of networks, we adopt a simplified version [17] suitable for graph-based applications. In this version, the Forman curvature of an edge $e(u_1, u_2)$ is defined as:

$$\begin{aligned} \mathbf{F}(e) = & 2 - \psi(e) \sum_{v_1 \in \mathcal{N}_1(u_1), v_1 \neq u_2} \frac{1}{\sqrt{\psi(e)\psi((u_1, v_1))}} \\ & - \psi(e) \sum_{v_2 \in \mathcal{N}_1(u_2), v_2 \neq u_1} \frac{1}{\sqrt{\psi(e)\psi((u_2, v_2))}} \end{aligned} \quad (1)$$

To extract the role-related structural feature F_v of node v , we generate a histogram that compiles the Forman curvature values of all $\mathbf{F}((v, u)) ((v, u) \in E)$.

Role-related Attribute Feature Extraction. This module is designed to extract attribute features for each node, taking their roles into account. The fundamental concept here is to amalgamate the attributes of adjacent nodes towards the central node. However, considering roles, nodes positioned further from the center should have diminished importance. To account for this, we introduce a decay parameter $p < 1$ in the subsequent equation, which serves to exponentially decrease the significance of nodes situated farther from the center:

$$\mathbf{A}_v = \sum_{k=1}^K p^k \mathcal{A}_u, u \in \mathcal{N}_k(v) \quad (2)$$

In this equation, A_v symbolizes the extracted attribute features of node v . The attributes of its k -hop neighbors u are aggregated in proportion to their distance from v . K denotes the longest distance we consider.

Role-related Layer Feature Extraction. Similar to attribute feature extraction, this module aims to extract layer features for each node. To achieve this, we enumerate all the k -hop edges of each layer surrounding the central node:

$$\mathbf{L}_{v,i,k} = \sum_{u_1 \in \mathcal{N}_{k-1}(v), u_2 \in \mathcal{N}_k(v), (u_1, u_2) \in \mathbf{E}} \mathbb{1}(\phi(u_1, u_2) = i) \quad (3)$$

In this context, $\mathbf{L}_{v,i,k}$ represents the extracted layer features of node v for layer i , considering k -hop neighbors.

Autoencoder. After extracting these three types of features, we utilize an Autoencoder mechanism to learn the multiplex and attributed role embedding for each node. Given the extracted features \mathbf{F} , the structural encoder is defined as:

$$\mathbf{h}_F^i = \tanh(\mathbf{W}_F^i \mathbf{h}_F^{i-1} + \mathbf{b}_F^i), i = 1, 2, \dots, |\text{Layer}| \quad (4)$$

where \mathbf{W}_F^i and \mathbf{b}_F^i represent the weights and bias of i -th layer, respectively, and the \mathbf{h}_F^0 is initialized with \mathbf{F} as the input of the encoder. The attribute and layer encoders are defined similarly. The outputs of the three encoders are then concatenated into a single vector and fed into a fully connected layer to generate the embeddings:

$$\mathbf{h}_C = \mathbf{h}_F^{|\text{Layer}|} \oplus \mathbf{h}_A^{|\text{Layer}|} \oplus \mathbf{h}_L^{|\text{Layer}|} \quad (5)$$

$$\mathbf{X} = \tanh(\mathbf{W}_E \mathbf{h}_C + \mathbf{b}_E) \quad (6)$$

The structural, attribute, and layer feature decoders are defined similarly to their corresponding encoders. For illustration, we provide the structure of the structural decoder as an example:

$$\hat{\mathbf{h}}_F^i = \tanh(\hat{\mathbf{W}}_F^i \hat{\mathbf{h}}_F^{i-1} + \hat{\mathbf{b}}_F^i), i = 1, 2, \dots, |\text{Layer}| \quad (7)$$

where $\hat{\mathbf{W}}_F^i$ and $\hat{\mathbf{b}}_F^i$ represent the weights and bias of i -th layer, respectively, and the $\hat{\mathbf{h}}_F^0$ is initialized with \mathbf{X} as the input of the decoder.

To optimize our autoencoder, we employ four loss functions: structural loss (\mathcal{L}_{str}), attribute loss (\mathcal{L}_{att}), layer loss (\mathcal{L}_{layer}), and regularization loss (\mathcal{L}_r). The first three are employed to ensure that the output of the decoders can accurately reconstruct the three input features \mathbf{F} , \mathbf{A} , and \mathbf{L} :

$$\mathcal{L}_{str} = \sum_{v \in \mathbf{V}} \left\| \mathbf{F}_v - \hat{\mathbf{h}}_{F,v}^{|\text{Layer}|} \right\|_2^2 t \quad [\mathcal{L}_{att} = \sum_{v \in \mathbf{V}} \left\| \mathbf{A}_v - \hat{\mathbf{h}}_{A,v}^{|\text{Layer}|} \right\|_2^2] \quad (8)$$

$$\mathcal{L}_{layer} = \sum_{v \in \mathbf{V}} \left\| \mathbf{L}_v - \hat{\mathbf{h}}_{L,v}^{|\text{Layer}|} \right\|_2^2$$

The regularization loss is designed to mitigate overfitting and to ensure that the learned embeddings \mathbf{X} are able to recover at least some local structural properties of each node:

$$\mathcal{L}_r = \sum_{v \in V} \left\| \mathbf{q}(v) - \mathbf{g}(\mathbf{X}_v) \right\|_2^2 \quad (9)$$

where g is a non-linear function learned by a Multi-Layer perceptron model applied to the embedding of each node, and q is a function that reflects local structural information of each node (e.g., degree, centrality, etc.). This paper uses a simple concatenation of degrees in each layer as q . The overall loss \mathcal{L}_{total} is defined as a weighted combination of all four losses:

$$\mathcal{L}_{total} = \mathcal{L}_r + \lambda_1 \mathcal{L}_{str} + \lambda_2 \mathcal{L}_{att} + \lambda_3 \mathcal{L}_{layer} \quad (10)$$

3.2 Model Training

In optimizing our model, we aim to minimize the global loss \mathcal{L}_{total} to obtain the corresponding embeddings X . The Adam optimizer [9] is utilized for parameter optimization. The hyperparameters are set as follows: an embedding size of 128, learning rate of 0.001, batch size of 32, 500 training epochs, and loss weights λ_1 , λ_2 , and λ_3 all set to 3. For consistency across experiments, the random seed is initialized to 0. Experiments are conducted on an RTX A6000.

4 Experiments

Here, we evaluate our method on node visualization, clustering, and classification tasks.

4.1 Visualization of Attributed Multiplex Role Embeddings

We evaluate generated embeddings through visualization on a synthetic attributed multiplex barbell graph. This graph consists of three barbell graphs in two layers, each consisting of two complete subgraphs connected by a “bridge”. Figure 2 shows the barbell graph used in our experiments. We use different edge colors to denote different layers and node shapes to denote different node attributes. The structurally equivalent nodes have the same color. For baselines, we use node2vec, struc2vec, and DMGI as the representative methods for homogeneous local proximity, structural role, and attributed multiplex local proximity methods, respectively. Other methods in the same category show similar embedding distributions with these three.

Figure 3(a) displays the embeddings generated by node2vec. As observed, node2vec fails to capture structural role identities as well as relational or attribute information. Instead, node2vec arranges the nodes of the six complete subgraphs somewhat haphazardly, resulting in two groups of reds, two groups of blue, and two groups of purple. Figure 3(b) presents the outcomes from struc2vec. Predictably, this method overlooks both relational and attribute data, embedding nodes solely based on their roles in the homogeneous version of the graph. This yields four distinct groups: (1) Subgraph components, depicted

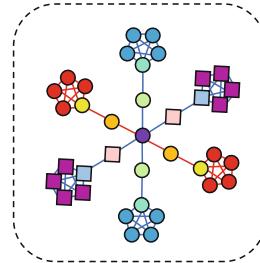


Fig. 2. An attributed multiplex barbell graph with two types of edges (blue and red) and two types of node attributes (square and circle). The colors of nodes represent roles which are not known beforehand. (Color figure online)

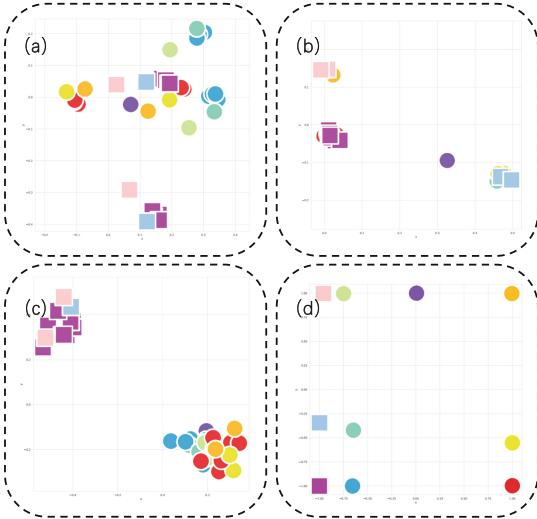


Fig. 3. Embeddings of nodes in the multiplex barbell by (a) node2vec (b) struc2vec (c) DMGI (d) ours. (Color figure online)

by red and blue circles encompassed by purple squares; (2) Subgraph connectors, shown as yellow and cyan circles encased by light sky blue squares; (3) Bridges, represented by green and orange circles surrounded by bisque squares; (4) The center, illustrated by a single indigo circle. Figure 3(c) demonstrates the embeddings of DMGI. While this method adeptly captures attribute information, it neglects other information important for identifying roles. It clusters nodes with similar attributes together. Lastly, Fig. 3(d) showcases the embeddings from our proposed method, which adeptly discerns the attributed multiplex structural roles between the nodes.

4.2 Node Clustering

We next evaluate the embeddings generated for the task of node clustering within intricate synthetic networks. Our analysis builds upon the synthetic datasets and settings employed by GraphWave [3], modifying them to a multiplex and attributed version. These datasets are characterized by a series of cycles adorned with basic shapes (e.g., “house”, “fan”, and “star”) and accompanied by their ground-truth labels. In our adaptation, each graph comprises three layers of edges (colored black, blue, and red) and two distinct node attributes (represented as square and circle) as illustrated in Fig. 4. We also introduce a “perturbed” variant to gauge the robustness of models against noise. This variant augments the original graph with an additional 5% of edges inserted randomly. For the “house” and “house perturbed” configurations, we implement an 18-node cycle, interspersed uniformly with 3 red&circle, 3 blue&circle, and 3 blue&square “houses”. In the scenarios labeled “varied” and “varied perturbed”, an 18-node

Table 1. Evaluation on planted cycle graphs. The asterisk (*) denotes statistically significant differences between a baseline and our method, as determined by the t-test with $p < 0.05$. Results are averages from 10 runs. The bold text highlights the top-performing method.

Shapes along a cycle graph		Method	Homogeneity	Completeness	Rand index
Attributed House		degree	0.620±0.000 *	0.772±0.000 *	0.852±0.000 *
		eigenvector	0.700±0.000 *	0.810±0.000 *	0.898±0.000 *
		node2vec	0.694±0.000 *	0.750±0.000 *	0.922±0.000 *
		DRNE	0.766±0.020 *	0.821±0.032 *	0.938±0.006 *
		struc2vec	0.727±0.012 *	0.771±0.012 *	0.928±0.007 *
		GraphWave	0.753±0.011 *	0.792±0.010 *	0.937±0.004 *
		DMGI	0.700±0.017 *	0.733±0.017 *	0.928±0.004 *
		URAMN	0.712±0.026 *	0.734±0.023 *	0.930±0.005 *
		JUST	0.695±0.004 *	0.750±0.001 *	0.923±0.001 *
		SM2Vec	0.713±0.000 *	0.891±0.000 *	0.885±0.000 *
		hyperstruct	0.747±0.011 *	0.782±0.014 *	0.937±0.003 *
		RESD	0.764±0.018 *	0.804±0.025 *	0.939±0.005 *
		[Our method]	1.000±0.000	1.000±0.000	1.000±0.000
House Perturbed		degree	0.610±0.007 *	0.753±0.011 *	0.856±0.004 *
		eigenvector	0.661±0.025 *	0.748±0.027 *	0.892±0.015 *
		node2vec	0.698±0.004 *	0.744±0.004 *	0.925±0.002 *
		DRNE	0.755±0.030 *	0.803±0.037 *	0.937±0.007 *
		struc2vec	0.769±0.034 *	0.809±0.038 *	0.939±0.008 *
		GraphWave	0.764±0.007 *	0.824±0.001 *	0.935±0.004 *
		DMGI	0.706±0.013 *	0.743±0.011 *	0.928±0.004 *
		URAMN	0.709±0.017 *	0.751±0.022 *	0.927±0.003 *
		JUST	0.701±0.004 *	0.745±0.005 *	0.925±0.002 *
		SM2Vec	0.739±0.020 *	0.828±0.018 *	0.920±0.010 *
		hyperstruct	0.733±0.020 *	0.777±0.019 *	0.932±0.006 *
		RESD	0.764±0.024 *	0.825±0.017 *	0.931±0.011 *
		[Our method]	0.842±0.034	0.921±0.035	0.953±0.006
Attributed Varied		degree	0.574±0.000 *	0.767±0.000 *	0.853±0.000 *
		eigenvector	0.750±0.000 *	0.826±0.000 *	0.939±0.000 *
		node2vec	0.770±0.004 *	0.783±0.004 *	0.958±0.001 *
		DRNE	0.744±0.023 *	0.780±0.020 *	0.939±0.010 *
		struc2vec	0.751±0.008 *	0.805±0.014 *	0.942±0.002 *
		GraphWave	0.781±0.019 *	0.798±0.011 *	0.953±0.006 *
		DMGI	0.629±0.019 *	0.634±0.013 *	0.935±0.004 *
		URAMN	0.688±0.014 *	0.703±0.015 *	0.938±0.005 *
		JUST	0.768±0.003 *	0.781±0.004 *	0.956±0.001 *
		SM2Vec	0.688±0.000 *	0.867±0.000 *	0.893±0.000 *
		hyperstruct	0.746±0.007 *	0.758±0.013 *	0.946±0.002 *
		RESD	0.792±0.005 *	0.838±0.016 *	0.951±0.002 *
		[Our method]	1.000±0.000	1.000±0.000	1.000±0.000
Varied Perturbed		degree	0.557±0.011 *	0.720±0.012 *	0.866±0.007 *
		eigenvector	0.666±0.032 *	0.731±0.017 *	0.923±0.019 *
		node2vec	0.757±0.004 *	0.758±0.007 *	0.956±0.001 *
		DRNE	0.706±0.018 *	0.737±0.016 *	0.937±0.005 *
		struc2vec	0.739±0.013 *	0.791±0.014 *	0.934±0.009 *
		GraphWave	0.725±0.007 *	0.788±0.001 *	0.933±0.002 *
		DMGI	0.628±0.013 *	0.634±0.007 *	0.935±0.003 *
		URAMN	0.687±0.015 *	0.700±0.014 *	0.937±0.004 *
		JUST	0.755±0.005 *	0.757±0.007 *	0.955±0.001 *
		SM2Vec	0.696±0.018 *	0.802±0.014 *	0.916±0.007 *
		hyperstruct	0.758±0.025 *	0.758±0.022 *	0.953±0.005 *
		RESD	0.763±0.013 *	0.842±0.009 *	0.944±0.008 *
		[Our method]	0.873±0.008	0.911±0.008	0.973±0.002

cycle is populated with a blend of “house”, “fan”, and “star” shapes (with 3 red&circle, 3 blue&circle, and 3 blue&square nodes for each form).

For the baselines, we select twelve representative unsupervised methods representing four distinct categories: (1) degree centrality and eigenvector centrality for centrality metrics; (2) node2vec as the representative microscopic homogeneous network embedding; (3) DRNE, struc2vec, GraphWave, SM2Vec [27], hyperstruct [28], and RESD for structural role embedding; (4) JUST, DMGI, and URAMN for multiplex embedding. It is noteworthy that DMGI and URAMN also accommodate attributed networks. Note that many well-known recent works on Graph Neural Networks are supervised or semi-supervised methods, and they are not typically designed for structural roles, so we do not include them as baselines.

We use agglomerative clustering with a “complete” linkage strategy to classify the embeddings and subsequently report on the average homogeneity, completeness, and rand index. Each experiment is repeated 10 times and averaged.

Table 1 shows the results, including the standard deviations and statistical significance. Our method not only bests all 12 baselines in all four tasks across all three metrics but does so with statistical significance ($p < 0.05$). Especially in the “house” and “varied” scenarios, our model attains flawless scores, underscoring its adeptness at accurately distinguishing attributed multiplex roles while also showcasing its innate robustness to disruptive noise.

4.3 Scalability

For a given set of hyper-parameters, the time complexity of our model is linear to the number of nodes. To better illustrate the scalability of our model, we learn node representations using our method with the default parameters on Erdos-Renyi graphs with increasing sizes from 100 to 100,000 nodes with average degrees of 10. For each Erdos-Renyi graph, we uniformly split the edges into three different multiplex layers. We run the tests on an RTX 2080 GPU for 500 epochs. Figure 5 shows the \log_{10} running time vs the \log_{10} nodes. For a network with

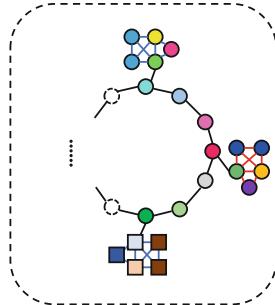


Fig. 4. A cycle planted with multiplex “houses” with three types of edges (black, blue, and red) and two types of node attributes (square and circle). Node colors represent roles (not known beforehand).

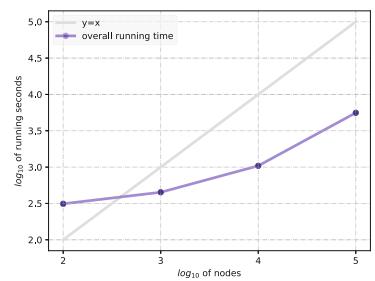


Fig. 5. Scalability of our method on Erdos-Renyi graphs with an average degree of 10. The grey line indicates $y = x$.

1,000,000 nodes and 10,000,000 edges, our method can still finish training in ten hours, which is faster than most of the existing embedding methods, and indicates that our method can easily be scaled to large networks.

5 Studying Influential Roles on Social Media

To test our method’s effectiveness on extensive real-world multiplex networks, we use our multiplex embedding approach to derive representations of a subset of Twitter users. We then analyze the capability of our embeddings in predicting influential nodes or users within our network.

5.1 Data Collection

We used a multiplex network of almost all verified users on Twitter (in 2021). Our network is comprised of 362,055 nodes (users) and has 106,396,154 *follow* edges, 15,454,983 *retweet* edges, and 7,031,786 *reply* edges. It is worth noting that algorithms such as GraphWave, SM2Vec, and RESD are not designed for handling large-scale networks. Due to their reliance on matrix operations like SVD, these algorithms grapple with high spatial and temporal complexities. In the context of our network’s size, using these methods would necessitate over 1 TB of memory. Consequently, we excluded them from our baselines.

5.2 Real-World Multiplex Role Similarity Ranking

We utilized a benchmark dataset of influential Twitter accounts provided by the Statista Research Department [18]. This dataset assigns an influence index score to each account. To construct attributes, we used Doc2Vec [10] on the latest 100 tweets from each user and averaged these embeddings to serve as the user’s attributes (capturing their topics of interest).

Our experiments aimed to assess the correlation between account embeddings derived from our and baseline methods and their respective ground truth influence index scores. For each user in our dataset, we compute its embedding cosine similarity with other users, producing a set of similarity scores. Concurrently, we generate another set of similarity scores based on the ground truth, wherein similarity is gauged by the absolute difference of influence index scores.

Table 2. Average Spearman’s correlations between user similarity lists generated through embeddings and the ground-truth influence scores.

Method	Correlation
degree	0.13
eigenvector	0.16
node2vec	0.09
DRNE	0.12
struc2vec	0.14
DMGI	0.09
URAMN	0.08
JUST	0.08
hyperstruct	0.13
Ours (w/o att)	0.24
Ours (w/ att)	0.26

The relationship between these two sets of scores is then quantified using Spearman's rank correlation. This process is repeated for all accounts in the ground truth dataset.

In Table 2, the average Spearman's correlation for each method is presented. Notably, methods that utilize structural information, including eigenvector centrality, degree centrality, struc2vec, hyperstruct, and our proposed technique, exhibit better performance than local proximity-based approaches like node2vec, URAMN, and JUST. Our approach, both with and without attributes, consistently outperforms other methods, showing an improvement of over 50% relative to the next best performer, eigenvector centrality. This highlights the proficiency of our method in capturing the structural role representations of accounts on platforms such as Twitter, distinguishing it from existing techniques.

5.3 Real-World Multiplex Role Visualization

We demonstrate how our model handles multiplex information by visualizing various networks: the retweet network (Fig. 6 (a)), reply network (Fig. 6 (b)), followership network (Fig. 6 (c)), and an integrated multiplex network (Fig. 6 (d)) that includes all three relationships. The red points represent the most influential accounts as identified by Statista. While our model in single-layer networks (retweet, reply, or followership) struggles to cluster these influential users effectively, the results from the integrated multiplex network successfully group them together (in the bottom-right corner in Fig. 6 (d)). This suggests that individual networks such as reply, retweet, and followership may not be enough to identify the influence of users on such complex networks. Whereas our model manages to effectively capture the influence information, emphasizing the significance of multiplex roles in such networks.

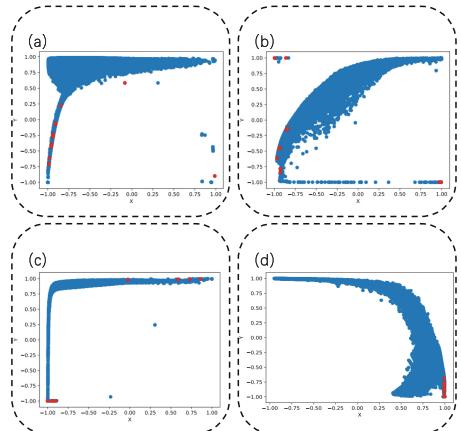


Fig. 6. The embeddings from our verified user Twitter dataset for (a) the retweet, (b) reply, (c) followership, and (d) the integrated networks of our model. The red points represent the most influential accounts. (Color figure online)

6 Conclusion

We present an unsupervised attributed multiplex structural role network embedding method tailored for large-scale networks. This method, utilizing an autoencoder framework, integrates node structures, relationships, and attributes, opti-

mizing embeddings in a unified manner. Our evaluations across various synthetic networks demonstrate our method's superiority over leading baselines in discerning complex node roles in attributed multiplex networks. It also showcases remarkable scalability, efficiently handling large networks. Our analysis of a tri-layered Twitter multiplex network, encompassing verified accounts and their interactions, highlights our method's applicability to extensive social media platforms. A case study using a dataset of influential Twitter users, with similarity ranking and visual assessments, shows our embeddings are more effective at identifying key users than alternatives. These results suggest our approach provides deeper, more nuanced insights into multiplex network structures.

References

1. Cao, S., Lu, W., Xu, Q.: Grarep: learning graph representations with global structural information. In: Proceedings of the 24th ACM International Conference on Information and Knowledge Management, pp. 891–900. ACM (2015)
2. Dong, Y., Chawla, N.V., Swami, A.: metapath2vec: Scalable representation learning for heterogeneous networks. In: Proceedings of the 23rd ACM SIGKDD (2017)
3. Donnat, C., Zitnik, M., Hallac, D., Leskovec, J.: Learning structural node embeddings via diffusion wavelets. In: Proceedings of the 24th KDD (2018)
4. Forman, R.: Discrete and computational geometry (2003)
5. Fu, T., Lee, W.C., Lei, Z.: Hin2vec: explore meta-paths in heterogeneous information networks for representation learning. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 1797–1806 (2017)
6. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks. In: Proceedings of the 22nd KDD, pp. 855–864. ACM (2016)
7. Huang, C., Wang, L., Cao, X., Ma, W., Vosoughi, S.: Learning dynamic graph embeddings using random walk with temporal backtracking. In: NeurIPS 2022 Temporal Graph Learning Workshop (2022)
8. Hussein, R., Yang, D., Cudré-Mauroux, P.: Are meta-paths necessary? Revisiting heterogeneous graph embeddings. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 437–446 (2018)
9. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)
10. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning, pp. 1188–1196. PMLR (2014)
11. Nikolenzos, G., Vazirgiannis, M.: Learning structural node representations using graph kernels. IEEE TKDE **33**(5), 2045–2056 (2019)
12. Ou, M., Cui, P., Pei, J., Zhang, Z., Zhu, W.: Asymmetric transitivity preserving graph embedding. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1105–1114 (2016)
13. Park, C., Kim, D., Han, J., Yu, H.: Unsupervised attributed multiplex network embedding. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 5371–5378 (2020)
14. Perozzi, B., Al-Rfou, R., Skiena, S.: DeepWalk: online learning of social representations. In: Proceedings of the 20th KDD, pp. 701–710. ACM (2014)
15. Qu, M., Tang, J., Shang, J., Ren, X., Zhang, M., Han, J.: An attention-based collaboration framework for multi-view network representation learning. In: Proceedings of the 2017 CIKM (2017)

16. Ribeiro, L.F., Saverese, P.H., Figueiredo, D.R.: struc2vec: learning node representations from structural identity. In: the 23rd KDD, pp. 385–394 (2017)
17. Sreejith, R., Mohanraj, K., Jost, J., Saucan, E., Samal, A.: Forman curvature for complex networks. *J. Stat. Mech. Theory Exp.* **2016**(6), 063206 (2016)
18. Statista: Most popular influential twitter users in 2020. <https://www.statista.com/statistics/1100266/top-influential-twitter-users/> (2022)
19. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: large-scale information network embedding. In: Proceedings of the 24th International Conference on World Wide Web, pp. 1067–1077. International World Wide Web Conferences Steering Committee (2015)
20. Tu, K., Cui, P., Wang, X., Yu, P.S., Zhu, W.: Deep recursive network embedding with regular equivalence. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2357–2366 (2018)
21. Wang, D., Cui, P., Zhu, W.: Structural deep network embedding. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1225–1234. ACM (2016)
22. Wang, L., Gao, C., Huang, C., Liu, R., Ma, W., Vosoughi, S.: Embedding heterogeneous networks into hyperbolic space without meta-path. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 10147–10155 (2021)
23. Wang, L., Huang, C., Cao, X., Ma, W., Vosoughi, S.: Graph-level embedding for time-evolving graphs. In: Companion Proceedings of the ACM Web Conference 2023, pp. 5–8 (2023)
24. Wang, L., Huang, C., Lu, Y., Ma, W., Liu, R., Vosoughi, S.: Dynamic structural role node embedding for user modeling in evolving networks. *ACM Trans. Inf. Syst.* **40**, 1–21 (2021)
25. Wang, L., Huang, C., Ma, W., Cao, X., Vosoughi, S.: Graph embedding via diffusion-wavelets-based node feature distribution characterization. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, pp. 3478–3482 (2021)
26. Wang, L., Huang, C., Ma, W., Liu, R., Vosoughi, S.: Hyperbolic node embedding for temporal networks. *Data Mining Knowl. Disc.* **35**, 1–35 (2021)
27. Wang, L., Huang, C., Ma, W., Lu, Y., Vosoughi, S.: Embedding node structural role identity using stress majorization. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, pp. 3473–3477 (2021)
28. Wang, L., Lu, Y., Huang, C., Vosoughi, S.: Embedding node structural role identity into hyperbolic space. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, pp. 2253–2256 (2020)
29. Zhang, H., Qiu, L., Yi, L., Song, Y.: Scalable multiplex network embedding. *IJCAI* **18**, 3082–3088 (2018)
30. Zhang, R., Zimek, A., Schneider-Kamp, P.: Unsupervised representation learning on attributed multiplex network. In: Proceedings of the 31st ACM International Conference on Information & Knowledge Management, pp. 2610–2619 (2022)
31. Zhang, W., Guo, X., Wang, W., Tian, Q., Pan, L., Jiao, P.: Role-based network embedding via structural features reconstruction with degree-regularized constraint. *Knowl. Based Syst.* **218**, 106872 (2021)



Query-Decision Regression Between Shortest Path and Minimum Steiner Tree

Guangmo Tong^(✉), Peng Zhao^{ID}, and Mina Samizadeh^{ID}

University of Delaware, Newark, USA

{amotong, pzhao, minasmz}@udel.edu

Abstract. Considering a graph with unknown weights, can we find the shortest path for a pair of nodes if we know the minimal Steiner trees associated with some subset of nodes? That is, with respect to a fixed latent decision-making system (e.g., a weighted graph), we seek to solve one optimization problem (e.g., the shortest path problem) by leveraging information associated with another optimization problem (e.g., the minimal Steiner tree problem). In this paper, we study such a prototype problem called *query-decision regression with task shifts*, focusing on the shortest path problem and the minimum Steiner tree problem. We provide theoretical insights regarding the design of realizable hypothesis spaces for building scoring models, and present two principled learning frameworks. Our experimental studies show that such problems can be solved to a decent extent with statistical significance.

Keywords: Statistical Learning · Data-driven Optimization · Combinatorial Optimization

1 Introduction

In its most general sense, a decision-making problem seeks to find the best decision for an input query in terms of an objective function that quantifies the decision qualities [6]. Traditionally, the objective function is given a prior, and we thus focus primarily on its optimization hardness. However, real-world systems are often subject to uncertainties, making the latent objective function not completely known to us [11]; this creates room for data-driven approaches to play a key role in building decision-making pipelines [17].

Query-decision Regression with Task Shifts (QRTS). When facing an unknown objective function, one can adopt the learn-and-optimize framework where we first learn the unknown objective function from data and then solve the target optimization problem based on the learned function [3], which can be dated back to Bengio's work twenty years ago [2]. Nevertheless, the learn-and-optimize framework suffers from the fact that the learning process is often driven by the *average* accuracy while good optimization effects demand *worst-case* guarantees [7]. Query-decision regression (QR), as an alternative decision-making diagram, seeks to infer good decisions by learning directly from successful

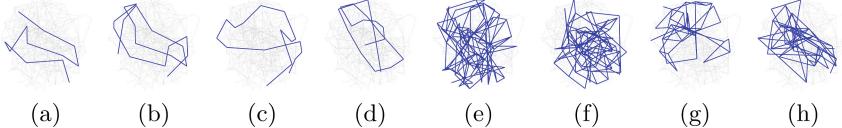


Fig. 1. Associated with a fixed weighted graph, (a)–(d) show the shortest paths of four pairs of nodes, and (e)–(h) show the minimal Steiner trees for four node subsets. Can we leverage the information in (a)–(d) to compute the solutions in (e)–(h), or vice versa?

optimization results. The feasibility of such a diagram has been proved by a few existing works [4]. Such a success points out an interesting way of generalizing QR called query-decision regression with task shifts (QRTS): assuming that there are two query-decision tasks associated with a latent system, can we solve one task (i.e., the target task) by using optimization results associated with the other task (i.e., the source task) – Fig. 1? Proving the feasibility of such problems is theoretically appealing, as it suggests that one can translate the optimal solutions between different optimization problems.

Contribution. This paper presents the first study on QRTS over stochastic graphs for two specific problems, the shortest path problem and the minimum Steiner tree problem. Taking QRTS as a statistical learning problem, we provide theoretical analysis regarding the creation of realizable hypothesis spaces for designing score functions, seeking to integrate the latent decision objective into the learning pipeline for better optimization effects. Based on the proposed hypothesis space, we design two principled methods QRTS-P and QRTS-D, where P stands for **p**oint estimation and D stands for **d**istribution learning. In particular, QRTS-P is designed based on the principle of point estimation that implicitly searches for the best mean graph, while QRTS-D leverages distribution learning to compute the pattern of the edge weights that can best fit the samples. We present empirical studies using graphs of classic families. As one of the main contributions of this paper, our results confirm that QRTS can be solved to a satisfactory extent, which may be the first piece of evidence showing that one can successfully translate knowledge between different optimization problems sharing the same underlying system. The appendix and supplementary materials¹ include technical proofs, more discussions on experiments, source code, and data.

2 Preliminaries

We consider a countable family \mathcal{G} of weighted directed graphs sharing the same graph structure $G = (V, E)$. For each weighted graph $g \in \mathcal{G}$, let $g() : E \rightarrow \mathbb{R}^+$ be its weight function. Without loss of generality, we assume that G has no multiple edge when the edge directions are omitted; therefore, each graph $g \in \mathcal{G}$ can also

¹ <https://github.com/cdslabamotong/QRTS>.

be taken as an undirected graph, without causing confusion regarding the edge weights. Associated with \mathcal{G} , there is an *unknown* distribution $\mathcal{D}_{\mathcal{G}}$ over \mathcal{G} . We consider optimization problems in the following form.

Definition 1. (Query-decision Optimization). Let $\mathcal{X} \subseteq 2^V$ be a query space and $\mathcal{Y} \subseteq 2^E$ be a decision space. In addition, let $f(x, y, g) \in \mathbb{R}^+$ be the decision value associated with a query $x \in \mathcal{X}$, a decision $y \in \mathcal{Y}$, and a graph $g \in \mathcal{G}$ (either directed or undirected). For a given query $x \in \mathcal{X}$, we seek to find the decision that can minimize the expected decision value:

$$\arg \min_{y \in \mathcal{Y}} F_{f, \mathcal{D}_{\mathcal{G}}}(x, y) \text{ where } F_{f, \mathcal{D}_{\mathcal{G}}}(x, y) := \mathbb{E}_{g \sim \mathcal{D}_{\mathcal{G}}} [f(x, y, g)]. \quad (1)$$

Such a task is specified by a three-tuple $(f, \mathcal{X}, \mathcal{Y})$. It reduces to the deterministic case with $|\mathcal{G}| = 1$ (e.g., Fig. 1).

When the distribution $\mathcal{D}_{\mathcal{G}}$ is known to us, the above problems fall into stochastic combinatorial optimization [18]. In addressing the case when $\mathcal{D}_{\mathcal{G}}$ is unknown, query-decision regression emphasizes the scenario when there is no proper data to learn $\mathcal{D}_{\mathcal{G}}$, and it is motivated by the aspiration to *learn directly from successful optimization results*. Since the optimization problem in question (i.e., Eq. (1)) can be computationally hard under common complexity assumptions (e.g., NP \neq P), we assume that an approximate solution is observed. Accordingly, we will utilize samples in the form of

$$D_{f, \alpha} = \left\{ (x_j, y_j) \mid F_{f, \mathcal{D}_{\mathcal{G}}}(x_j, y_j) \leq \alpha \cdot \min_{y \in \mathcal{Y}} F_{f, \mathcal{D}_{\mathcal{G}}}(x_j, y) \right\}, \quad (2)$$

where the quality of the observed decision is controlled by a nominal ratio $\alpha \geq 1$. With such, we formulate query-decision regression with/without task shifts as statistical learning problems.

Definition 2. (Query-decision Regression (QR)). Associated with a query-decision optimization problem $(f, \mathcal{X}, \mathcal{Y})$ and a distribution $\mathcal{D}_{\mathcal{X}}$ over \mathcal{X} , given a collection $D_{f, \alpha} = \{(x_j, y_j)\}$ of query-decision pairs with x_j being iid from $\mathcal{D}_{\mathcal{X}}$, we aim to learn a decision-making model $M : \mathcal{X} \rightarrow \mathcal{Y}$ that can predict high-quality decisions for future queries.

Definition 3. (Query-decision Regression with Task Shifts (QRTS)). Consider a source task $(f_S, \mathcal{X}_S, \mathcal{Y}_S)$ and a target task $(f_T, \mathcal{X}_T, \mathcal{Y}_T)$ sharing the same \mathcal{G} and $\mathcal{D}_{\mathcal{G}}$. Suppose that we are provided with query-decision samples $D_{f_S, \alpha}$ associated with the source task. We aim to learn a decision-making model $M : \mathcal{X}_T \rightarrow \mathcal{Y}_T$ for the target task, with the goal of maximizing the optimization effect:

$$\mathcal{L}(M, \mathcal{D}_{\mathcal{X}_T}) := \mathbb{E}_{x \sim \mathcal{D}_{\mathcal{X}_T}} \left[\frac{\min_{y \in \mathcal{Y}_T} F_{f_T, \mathcal{D}_{\mathcal{G}}}(x, y)}{F_{f_T, \mathcal{D}_{\mathcal{G}}}(x, M(x))} \right], \quad (3)$$

where $\mathcal{D}_{\mathcal{X}_T}$ is the query distribution of the target task.

Remark 1. (Technical Challenge). In principle, QR falls into the setting of supervised learning, in the sense that it attempts to learn a mapping using labeled data. Therefore, standard supervised learning methods can solve such problems with statistical significance more or less, although they may not be the

optimal methods for specific tasks [21]. QRTS reduces to QR when the source task is identical to the target one, but it is arguably more challenging: one can no longer apply standard supervised learning methods because the query-decision mapping we seek to infer is associated with the target task but the samples are of the source task. To the best of our knowledge, no existing method can be directly applied to solve problems like the one in Fig. 1.

3 A Warm-Up Method: QRTS-P

In this section, we present a simple and intuitive method called QRTS-P for solving QRTS. To illustrate the idea, we notice that the latent objective function F_{f, \mathcal{D}_G} can be expressed as a function of the mean weights of the edges, which is due to the linearity of expectation.

Example 1. Suppose that the considered query-decision optimization problem is the stochastic shortest path problem. For each node pair $x = (u, v)$, let \mathcal{Y}_x be the set of all paths from u to v . The latent objective function can thus be expressed as

$$F_{f, \mathcal{D}_G}(x, y) = \begin{cases} \sum_{e \in y} \mathbb{E}_{g \sim \mathcal{D}_G}[g(e)] & \text{if } y \in \mathcal{Y}_x \\ +\infty & \text{otherwise.} \end{cases} \quad (4)$$

Similarly, for the stochastic minimal Steiner tree problem [22], which finds the min-weight subgraph that connects a given set of nodes, we may define \mathcal{Y}_x as the set of valid Steiner trees of a node set x , and with such, the latent objective has the identical form as Eq. (4).

In abstract, let $\mathcal{Y}_{f,x} \subseteq \mathcal{Y}_f$ be the set of the feasible solutions associated with a query x , and $\mathbb{1}_S \in \{0, 1\}$ be the set indicator function, i.e., $\mathbb{1}_S(x) = 1 \iff x \in S$. The query-decision optimization now has the generic form of

$$\arg \min_{y \in \mathcal{Y}_{f,x}} \sum_{e \in y} \mathbb{E}_{g \sim \mathcal{D}_G}[g(e)] = \sum_{e \in E} \mathbb{E}_{g \sim \mathcal{D}_G}[g(e)] \mathbb{1}_y(e) \quad (5)$$

Such an abstraction suggests that it would be sufficient for solving the target task if one can find the mean graph induced by \mathcal{D}_G , which essentially asks for good estimations of $\{\mathbb{E}_{g \sim \mathcal{D}_G}[g(e)] | e \in E\}$ – leading to a point estimation problem [13]. In what follows, we will see how samples $D_{f_S, \alpha}$ associated with the source task can be helpful for such a purpose.

For each $e \in E$, let $w_e \in \mathbb{R}^+$ be the sought-after estimation of $\mathbb{E}_{g \sim \mathcal{D}_G}[g(e)]$. Since each sample (x_j, y_j) in $D_{f_S, \alpha}$ is an α -approximation, in light of Eq. (5), a desired set $\{w_e\} := \{w_e | e \in E\}$ should satisfy the linear constraint

$$\alpha \cdot \min_{y \in \mathcal{Y}_{f_S, x_j}} \sum_{e \in E} w_e \mathbb{1}_y \geq \sum_{e \in E} w_e \mathbb{1}_{y_j}, \quad (6)$$

which means that the sample decision y_j is also an α -approximation in the mean graph induced by $\{w_e\}$. Applying the standard large-margin training to Eq.

(6) [19], a robust estimation can be inferred by solving the following quadratic program

$$\min_{w_e, \eta_j} \sum_{e \in E} w_e^2 + C \cdot \sum_j \eta_j \text{ s.t. } \alpha \cdot \underbrace{\min_{y \in \mathcal{Y}_{f_S, x_j}} \sum_{e \in E} w_e \mathbf{1}_y(e)}_{\text{source inference}} - \sum_{e \in E} w_e \mathbf{1}_{y_j}(e) \geq -\eta_j, \forall j. \quad (7)$$

where C is a hyperparameter. Optimization problems in the above form have been widely discussed for training structured prediction models, and they can be solved efficiently as long as the source inference problem in Eq. (7) can be effectively solved for a given $\{w_e\}$ [15]. We adopt the cutting plane algorithm in our experiments and defer the details to the Appendix. With the learned weights $\{w_e\}$, the inference for a query $x^* \in \mathcal{X}_T$ of the target problem can be computed through

$$\text{target inference: } \min_{y \in \mathcal{Y}_{f_T, x^*}} \sum_{e \in E} w_e \mathbf{1}_y(e) \quad (8)$$

We denote such an approach as QRTS-P. The source and target inferences will be discussed later in Remark 2, as they are special cases of later problems.

4 A Probabilistic Perspective: QRTS-D

In this section, we present a more involved method called QRTS-D for solving QRTS. It turns out that QRTS-D subtly subsumes QRTS-P as a special case.

4.1 Overall Framework

QRTS-D follows the standard scoring model, where we assign each decision a score and make a prediction by selecting the decision with the *lowest score*:

$$\begin{aligned} \text{score function: } & h : \mathcal{X}_T \times \mathcal{Y}_T \rightarrow \mathbb{R} \\ \text{inference: } & \arg \min_{y \in \mathcal{Y}_T} h(x, y). \end{aligned} \quad (9)$$

Such a framework is expected to solve QRTS well, provided that for each pair $(x, y) \in \mathcal{X}_T \times \mathcal{Y}_T$, a low score $h(x, y)$ can imply a small objective value $F_{f_T, \mathcal{D}_G}(x, y)$. Implementing such an idea hinges on three integral parts: **a**) a hypothesis space H of h ; **b**) training methods to search for the best score function within H based on the empirical evidence $D_{f_S, \alpha}$; **c**) algorithms for solving the inference problem. With such a framework, we will first discuss insights for designing a desired hypothesis space and then present training methods.

4.2 Hypothesis Design

In designing a desired score function h , the key observation is that the true objective function F_{f_T, \mathcal{D}_G} of the target task is a perfect score function, in that the inference over $F_{f_T, \mathcal{D}_G}(x, y)$ recovers the exact optimal solution. While \mathcal{D}_G is unknown to us, the technique of importance sampling offers a means of deriving

a parameterized approximation [20]. In particular, for any empirical distribution $\mathcal{D}_{\mathcal{G}}^{em}$ over \mathcal{G} , we have

$$F_{f_T, \mathcal{D}_{\mathcal{G}}}(x, y) = \int_{g \in \mathcal{G}} \frac{\mathcal{D}_{\mathcal{G}}[g]}{\mathcal{D}_{\mathcal{G}}^{em}[g]} f_T(x, y, g) d\mathcal{D}_{\mathcal{G}}^{em}, \quad (10)$$

which immediately implies the function approximation guarantee between $F_{f_T, \mathcal{D}_{\mathcal{G}}}$ and an affine combination of f_T .

Theorem 1. *Let $\|\cdot\|$ denote the function distance with respect to the Lebesgue measure associated with any distribution \mathcal{D} over $\mathcal{X}_T \times \mathcal{Y}_T$. For each $\epsilon \geq 0$, $\lim_{K \rightarrow \infty} \Pr_{g_i \sim \mathcal{D}_{\mathcal{G}}^{em}} \left[\inf_{w_i \in \mathbb{R}} \left\| F_{f_T, \mathcal{D}_{\mathcal{G}}}(x, y) - \sum_{i=1}^K w_i f_T(x, y, g_i) \right\| \leq \epsilon \right] = 1$.*

Theorem 1 justifies the following hypothesis space for the score function of which the complexity is controlled by its dimension $K \in \mathbb{Z}$.

$$H_{K, \mathcal{D}_{\mathcal{G}}^{em}} := \left\{ h_{\mathbf{w}, \{g_i\}}(x, y) := \sum_{i=1}^K w_i f_T(x, y, g_i) \mid g_i \sim \mathcal{D}_{\mathcal{G}}^{em}, \mathbf{w} = (w_1, \dots, w_K) \in \mathbb{R}^K \right\}.$$

These score functions are very reminiscent of the principled kernel machines [8], with the distinction that our kernel function, namely f_T , is inherited from the latent optimization problem rather than standard kernels [16]. For such a score function $h_{\mathbf{w}, \{g_i\}}(x, y)$, the inference process is further specialized as

$$\text{target inference: } M_{\mathbf{w}, \{g_i\}}(x) := \min_{y \in \mathcal{Y}_T} \sum_i w_i f_T(x, y, g_i). \quad (11)$$

With the construction of $H_{K, \mathcal{D}_{\mathcal{G}}^{em}}$, a realizable space can be achieved provided that the dimension K is sufficiently large, which allows us to characterize the generalization loss Eq. (3).

Theorem 2. *Suppose that a β -approximation is adopted to solve the target inference problem Eq. (11). Let $D_\infty \in \mathbb{R}$ be the ∞ -order Rényi divergence between $\mathcal{D}_{\mathcal{G}}$ and $\mathcal{D}_{\mathcal{G}}^{em}$. For each $\epsilon \geq 0$ and $\delta > 0$, there exists*

$$C = O\left(\frac{\ln |\mathcal{X}_T| + \ln |\mathcal{Y}_T|}{\epsilon^2} \cdot \ln \frac{1}{\delta} \cdot \exp(D_\infty)\right)$$

such that when $K \geq C$, with probability at least $1 - \delta$ over the selection of $\{g_i\}$, we have $\sup_{\mathbf{w} \in \mathbb{R}^K} \mathcal{L}(M_{\mathbf{w}, \{g_i\}}, \mathcal{D}_{\mathcal{X}_T}) \geq \beta \cdot \frac{1-\epsilon}{1+\epsilon}$.

Theorem 2 suggests that a high dimension (i.e., K) may be needed when a) the spaces are large and/or b) the deviation between $\mathcal{D}_{\mathcal{G}}$ and $\mathcal{D}_{\mathcal{G}}^{em}$ is high, which is intuitive. The proof of Theorem 2 leverages point-wise concentration to acquire the desired guarantees, while Theorem 1 is proved through concentrations in function spaces.

4.3 QRTS-D

With the design of $H_{K, \mathcal{D}_{\mathcal{G}}^{em}}$, we now present methods for computing a concrete score function $h_{\mathbf{w}, \{g_i\}}$, which is to decide a collection $\{g_i\}$ of subgraphs as well

as the associated weights \mathbf{w} . In light of Eq. (10), \mathbf{w}_i can be viewed as the importance of graph g_i . We will not restrict ourselves to a specific choice of $\mathcal{D}_{\mathcal{G}}^{em}$ and thus assume that a nominal parametric family $\mathcal{D}_{\mathcal{G},\theta}^{em}$ is adopted, with an extra subscription θ added to denote the parameter set. Assuming that the hyperparameter K is given, the framework of QRTS-D loops over three phases: **a) graph sampling**, to sample $\{g_1, \dots, g_K\}$ iid from $\mathcal{D}_{\mathcal{G},\theta}^{em}$; **b) importance learning**, to compute \mathbf{w} for $\{g_i\}$; **c) distribution tuning**, to update $\mathcal{D}_{\mathcal{G},\theta}^{em}$. The first phase is trivial, and we will therefore focus on the other two phases.

Importance Learning. In computing the weights \mathbf{w} for a given $\{g_i\}$, we have reached a key point to attack the challenges mentioned in Remark 1: the function approximation guarantee in Theorem 1 holds not only for the target task but also for the source task. That is, $\sum_{i=1}^K w_i f_T(x, y, g_i)$ is a desired score function (for solving the target task) if and only if $\sum_{i=1}^K w_i f_S(x, y, g_i)$ can well approximate the true objective function $F_{f_S, \mathcal{D}_{\mathcal{G}}}(x, y)$ of the source task. Therefore, since the samples in $D_{f_S, \alpha} = \{(x_j, y_j)\}$ are α -approximations to the source task, the ideal weights \mathbf{w} should satisfy

$$\alpha \min_{y \in \mathcal{Y}_S} \sum_{i=1}^K w_i f_S(x_j, y, g_i) \geq \sum_{i=1}^K w_i f_S(x_j, y_j, g_i).$$

In this way, we have been able to leverage the samples from the source task to decide the best \mathbf{w} associated with $\{g_i\}$. This owes to the fact that our design $H_{K, \mathcal{D}_{\mathcal{G}}^{em}}$ allows us to separate $\mathcal{D}_{\mathcal{G}}$ from the task-dependent kernels (i.e., f_S and f_T), which is otherwise not possible if we parametrized the score function (i.e., Eq. 9) using naive methods (e.g., neural networks). Following the same logic behind the translation from Eq. (6) to Eq. (7), the above constraints lead to a similar optimization program:

$$\min_{\mathbf{w}, \eta_i} \|\mathbf{w}\|^2 + C \cdot \sum_i \eta_i \text{ s.t. } \underbrace{\min_{y \in \mathcal{Y}_S} \alpha \sum_{i=1}^K w_i f_S(x_j, y, g_i) - \sum_{i=1}^K w_i f_S(x_j, y_j, g_i)}_{\text{source inference}} \geq -\eta_j, \forall j. \quad (12)$$

The above program shares the same type with Eq. (7), and we again defer the optimization details to the appendix.

Distribution Tuning. With the importance vector \mathbf{w} learned based on the subgraphs $\{g_i\}$ sampled from the current θ , we seek to fine-tune $\mathcal{D}_{\mathcal{G},\theta}^{em}$ to make it aligned more with the latent distribution $\mathcal{D}_{\mathcal{G}}$, which is desired as suggested by the proof of Theorem 1. Inspired by Eq. (10), the true likelihood associated with g_i is approximated by $w_i^* := w_i \mathcal{D}_{\mathcal{G},\theta}^{em}[g_i]$. Consequently, one possible way to reshape $\mathcal{D}_{\mathcal{G},\theta}^{em}$ is to find the θ^* that can minimize the discrepancy between $\mathcal{D}_{\mathcal{G},\theta^*}^{em}$ and $\mathcal{D}_{\mathbf{w}^*}$, i.e., $\theta^* = \arg \min_{\theta} D(\mathcal{D}_{\mathcal{G},\theta}^{em} || \mathcal{D}_{\mathbf{w}^*})|_{\{g_i\}}$, where $\mathcal{D}_{\mathbf{w}^*}$ is the discrete distribution over $\{g_i\}$ defined by normalizing (w_1^*, \dots, w_K^*) , and the distance measure $D(||)$ can be selected at the convenience of the choice of the $\mathcal{D}_{\mathcal{G},\theta}^{em}$ – for example, cosine similarity or cross-entropy. For such problems, standard methods can be directly applied when $\mathcal{D}_{\mathcal{G},\theta}^{em}$ is parameterized by common distribution families; first- and second-order methods can be readily used if $\mathcal{D}_{\mathcal{G},\theta}^{em}$ has a complex form such as neural networks.

Algorithm 1. QRTS-D

-
- 1: **Input:** $D_{f_S, \alpha} = \{x_y, y_i\}, C, K, T, \alpha, \mathcal{D}_{\mathcal{G}, \theta}^{em}$;
 - 2: **Output:** $\mathbf{w} = (w_1, \dots, w_K)$ and $\{g_1, \dots, g_K\}$
 - 3: Initialize $\theta, t = 0$;
 - 4: **repeat**
 - 5: $\{g_1, \dots, g_K\}$ iid from $\mathcal{D}_{\mathcal{G}, \theta}^{em}$;
 - 6: Compute \mathbf{w} via Eq. (12) based on $D_{f_S, \alpha}$ and C ;
 - 7: Update θ via $\theta^* = \arg \min_{\theta} D(\mathcal{D}_{\mathcal{G}, \theta}^{em} || \mathcal{D}_{\mathbf{w}^*})_{|\{g_i\}}$;
 - 8: $t = t + 1$
 - 9: **until** $t = T$
 - 10: **Return** $\{g_1, \dots, g_K\}$ and \mathbf{w}
-

The QRTS-D method is conceptually simple, as summarized in Alg. 1. Similar to QRTS-P, using such a method requires algorithms for solving the source and target inferences in Eqs. (11) and (12). In what follows, we discuss such issues as well as the possibility of enhancing QRTS-D using QRTS-P.

Remark 2 (Source and Target Inferences). For QRTS-P, the source (resp., target) inference problem is nothing but to solve the source (resp., target) query-decision optimization task in its deterministic case. For QRTS-D, the inference problems are to solve the source and target tasks over a weighted combination of deterministic graphs. For the shortest path problem, such inference problems can be solved in polynomial time; for the minimum Steiner tree problem, such inference problems admit 2-approximation [22].

Remark 3 (QRTS-P vs QRTS-D). As one may have noticed, QRTS-P is a natural special case of the importance learning phase of QRTS-D, in the sense that each w_e in QRTS-P corresponds to the importance of the subgraph with one edge (i.e., e). In other words, QRTS-P can be viewed as the QRTS-D where the support of $\mathcal{D}_{\mathcal{G}, \theta}^{em}$ is the span of single-edge subgraphs with unit weights. Notably, the dimension of QRTS-P is fixed and thus limited by the number of edges, while the dimension K of QRTS-D can be made arbitrarily large. For this reason, QRTS-P may be preferred if the sample size is small, while QRTS-D can better handle large sample sets, which is evidenced by our experimental studies.

Remark 4 (QRTS-PD). In QRTS-D, the initialization of $\mathcal{D}_{\mathcal{G}, \theta}^{em}$ is an open issue, and this creates the possibility of integrating QRTS-P into QRTS-D by initializing $\mathcal{D}_{\mathcal{G}, \theta}^{em}$ using the weights $\{w_e\}$ learned from QRTS-P. This leads to another approach called QRTS-PD consisting of three steps: **a**) run QRTS-P to acquire the estimations $\{w_e\}$; **b**) stabilize θ based on $\{w_e\}$ through maximum likelihood estimation, i.e., $\min_{\theta} - \sum_{e \in E} \log \sum_{g \in \mathcal{G}} \mathcal{D}_{\mathcal{G}, \theta}^{em}[g | g(e) = w_e]$; **c**) run QRTS-D. From such a perspective, QRTS-PD can be taken as a continuation of QRTS-P to further improve the generalization performance by building models that are more expressive.

5 Empirical Studies

In this section, we present empirical studies demonstrating that QRTS can be solved with statistical significance using the presented methods.

5.1 Experimental Settings

Source and Target Tasks. We specifically focus on two query-decision optimization tasks: shortest path and minimum Steiner tree [9]. Depending on the selection of the source and target tasks, we have two possible settings: *Path-to-Tree* and *Tree-to-Path*. The source and target inferences can be approximated effectively, as discussed in Remark 2. These algorithms are also used to generate samples of query-decision pairs (i.e., Eq. (2)).

Table 1. Results for Path-to-Tree on Kro, Col and BA. Each cell shows the mean ratio together with the standard deviation (std). The top three results in each column are highlighted.

Train Size	Kro			Col			BA		
	60	240	2400	60	240	2400	60	240	2400
QRTS-P	4.0 _(0.3)	3.4 _(0.5)	2.4 _(0.3)	291 ₍₇₈₎	150 ₍₂₉₎	128 _(7.4)	181 ₍₂₈₎	144 ₍₄₁₎	63 ₍₃₅₎
QRTS-PD-	60	4.4 _(0.3)	3.4 _(0.8)	2.3 _(0.4)	250 ₍₈₂₎	367 ₍₅₆₎	123 _(2.3)	209 ₍₂₂₎	195 ₍₂₂₎
	240	4.6 _(0.7)	3.7 _(0.1)	2.7 _(0.2)	330 ₍₆₅₎	227 ₍₁₄₎	117 _(6.1)	129 ₍₂₂₎	149 ₍₁₉₎
	2400	4.3 _(0.9)	3.2 _(0.5)	2.4 _(0.1)	214 ₍₆₈₎	183 ₍₆₉₎	88 ₍₁₂₎	139 ₍₃₈₎	131 ₍₄₄₎
QRTS-PD-1	60	3.9 _(0.7)	3.4 _(0.8)	2.3 _(0.4)	361 ₍₅₅₎	225 ₍₁₁₎	115 _(5.1)	177 ₍₃₁₎	130 ₍₃₄₎
	240	4.2 _(0.4)	3.2 _(0.5)	2.4 _(0.2)	350 ₍₈₈₎	245 ₍₃₁₎	129 ₍₁₆₎	166 ₍₃₄₎	132 ₍₄₉₎
	2400	4.3 _(0.5)	3.1 _(0.3)	2.3 _(0.4)	261 ₍₉₃₎	186 ₍₂₄₎	106 _(6.2)	160 ₍₂₉₎	119 ₍₃₎
QRTS-PD-3	60	4.3 _(1.1)	3.2 _(0.6)	2.6 _(0.5)	431 ₍₂₇₎	167 ₍₉₉₎	110 ₍₁₅₎	391 ₍₄₆₎	144 ₍₁₃₎
	240	4.3 _(0.9)	3.3 _(0.6)	2.3 _(0.4)	317 ₍₈₇₎	183 ₍₃₅₎	126 ₍₁₁₎	202 ₍₃₈₎	138 ₍₁₇₎
	2400	4.0 _(0.4)	3.2 _(0.4)	2.2 _(0.2)	324 ₍₃₈₎	107 ₍₁₂₎	113 _(6.1)	184 ₍₄₉₎	120 _(2.4)
Unit & Rand	5.2 _(0.2) & 10.4 _(0.3)			990 ₍₆₈₎ & 2231 ₍₄₅₎			349 _(4.7) & 749 ₍₁₃₎		

Graph, True Distribution, and Samples. We adopt a collection of graphs of classic types: a Kronecker graph (**Kro**) [14], a road network of Colorado (**Col**) [5], a Barabasi-Albert graph (**BA**) [1], and two Watts-Strogatz graphs with different densities (**WS-dense** and **WS-sparse**) [23]. The statistics of these graphs can be found in the appendix. To have a diverse graph pattern, we generate the ground truth distribution \mathcal{D}_G by assigning each edge a Weibull distribution [12] with parameters randomly selected from $\{1, \dots, 20\}$. For each graph and each problem instance, we generate a pool of 10,000 query-decision pairs.

QRTS Methods. We use QRTS-PD-1 (resp., QRTS-PD-3) to denote the QRTS-PD method when one (resp., three) iterations over the three phases are

used. Based on QRTS-PD-1, we implement QRTS-PD⁻ that foregoes the distribution tuning phase. For these methods, the model dimensions K are selected from {60, 240, 2400}. We utilize the one-slack cutting plane algorithm [10] for the large-margin training (i.e., Eqs. (7) and (12)). The empirical distribution \mathcal{D}_G^{em} is parameterized by assigning each edge an exponential distribution.

Baselines. We set up two baselines **Unit** and **Rand**. Unit computes the predictions based on the graph with unit-weight edges. Rand computes the decision based on the graph with random weights. Unit essentially leverages only the graph structure to compute predictions. We note that none of the methods in existing papers can be directly applied to QRTS (Remark 1).

Training and Testing. In each run, the training size is selected from {60, 240, 2400}, and the testing size is 1000, where samples are randomly selected from the sample pool. Given a testing set $\{(x_i, y_i)\}$ of the target task, the performance is measured by the ratio $\sum_i F_{f_T, \mathcal{D}_G}(x_i, y_i^*) / \sum_i F_{f_T, \mathcal{D}_G}(x_i, y_i)$, where y_i^* is the predicted decision associated with x_i ; a lower ratio implies better performance. We report the average ratios and the standard deviations over five runs for each method.

Table 2. Results on Tree-to-Path. Each cell shows the mean ratio together with the standard deviation (std). Small stds (< 0.1) are denoted as 0.0. The top three results in each column are highlighted.

Train Size		Kro			Col			BA		
		60	240	2400	60	240	2400	60	240	2400
QRTS-P		1.46 (0.0)	1.37 (0.0)	1.60 (0.0)	9.8 (0.2)	6.6 (0.4)	6.1 (0.1)	1.6 (0.1)	1.4 (0.2)	1.4 (0.1)
QRTS-PD ⁻	60	1.44 (0.1)	1.41 (0.1)	1.39 (0.0)	11 (5.8)	8.6 (1.5)	7.1 (0.6)	1.9 (0.4)	1.4 (0.2)	1.5 (0.1)
	240	1.42 (0.1)	1.46 (0.0)	1.39 (0.0)	9.9 (4.2)	6.8 (3.6)	6.5 (0.4)	1.7 (0.3)	1.5 (0.2)	1.5 (0.1)
	2400	1.48 (0.1)	1.45 (0.0)	1.38 (0.0)	8.9 (3.1)	6.0 (1.3)	6.2 (0.9)	1.5 (0.2)	1.3 (0.1)	1.3 (0.1)
QRTS-PD-1	60	1.55 (0.0)	1.42 (0.1)	1.37 (0.0)	6.6 (0.6)	6.3 (2.4)	6.8 (0.6)	1.6 (0.3)	1.5 (0.0)	1.4 (0.1)
	240	1.56 (0.1)	1.44 (0.1)	1.36 (0.0)	11 (3.2)	7.6 (1.4)	6.6 (0.0)	1.6 (0.3)	1.5 (0.3)	1.5 (0.1)
	2400	1.52 (0.1)	1.39 (0.1)	1.37 (0.0)	14 (2.4)	7.7 (3.9)	7.2 (0.3)	1.7 (0.1)	1.5 (0.3)	1.7 (0.1)
QRTS-PD-3	60	1.53 (0.1)	1.41 (0.1)	1.34 (0.1)	13 (4.2)	5.9 (1.5)	5.5 (0.9)	1.7 (0.2)	1.5 (0.1)	1.4 (0.1)
	240	1.50 (0.1)	1.42 (0.0)	1.36 (0.0)	9.4 (1.4)	6.6 (0.2)	5.9 (0.1)	1.6 (0.1)	1.4 (0.2)	1.2 (0.1)
	2400	1.47 (0.1)	1.41 (0.1)	1.32 (0.0)	7.8 (0.6)	8.5 (1.4)	7.7 (2.0)	1.3 (0.1)	1.3 (0.1)	1.3 (0.2)
Unit & Rand		1.57 (0.1) & 1.78 (0.1)			9.2 (0.1) & 19 (0.1)			1.57(0.0) & 2.2(0.3)		

5.2 Analysis

The results on Kro, Col, and BA are given in Tables 1 and 2. The results on WS-sparse and WS-dense can be found in the Appendix. The main observations are listed below, and the minor observations are given in the appendix.

O1: The Proposed Methods Behave Reasonably with Promising Performance. First, we observe that the proposed methods perform significantly better when more samples are given, which suggests that they are able to

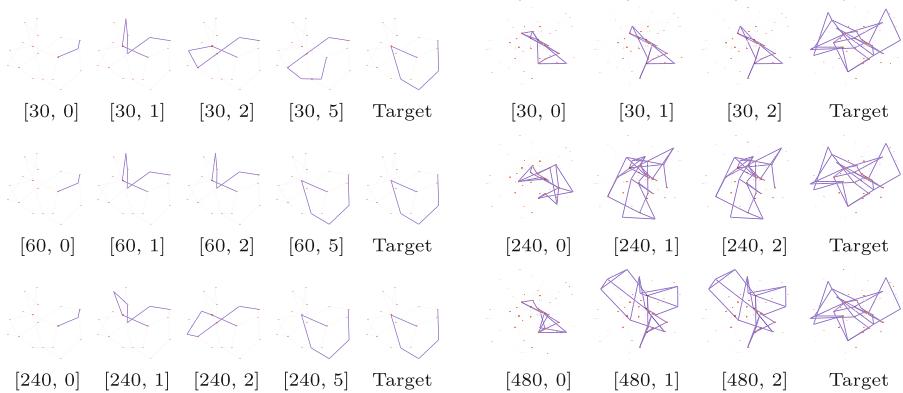


Fig. 2. The left (resp., right) shows the visualizations of the solution to one testing query for the Tree-to-Path (resp., Path-to-Tree) problem under QRTS-P on Kro. The figure labeled by $[a, b]$ shows the result with training size a after b iterations in the cutting plane algorithm. Each row shows the results under one training size, where the last figure shows the optimal solution.

infer meaningful information from the samples toward solving the target task. On the other hand, all the proposed methods are clearly better than Rand, implying that the model efficacy is non-trivial. In addition, they easily outperform Unit by an evident margin in most cases. For example, for Path-to-Tree on BA in Table 1, the best ratio achieved by QRTS-PD-3 is 23, while Unit and Random cannot produce a ratio smaller than 300.

O2: QRTS-P Offers an Effective Initialization for QRTS-D. With very few exceptions, QRTS-PD performs much better than QRTS-P under the same sample size, which confirms that QRTS-P can indeed be improved by using importance learning through re-sampling, which echos Remark 4. This is especially true when the sample size is large; for example, for Path-to-Tree on BA with 2400 samples, methods based on QRTS-PD with a dimension of 2400 are at least twice better than QRTS-P in terms of the performance ratio, demonstrating that QRTS-PD of a high dimension can better consume large datasets.

O3: Distribution Tuning is Helpful After Multiple Iterations. Since QRTS-D can be used without the distribution tuning phase, we are wondering if the distribution turning phase is necessary. By comparing QRTS-PD-1 with QRTS-PD⁻, we see that the distribution tuning phase can be useful in many cases, but its efficacy is not very significant. However, combined with the results of QRTS-PD-3, we observe that the distribution turning phase can better reinforce the optimization effect when multiple iterations are used. Finally, by comparing QRTS-PD-1 and QRTS-PD-3, we find that training more iterations is useful mostly when the model dimension is large, which is especially the case for Tree-to-Path (Table 2).

O4: The Learning Process is Smooth. Fig. 2 visualizes the learning process of QRTS-P for two example testing queries. One can see that QRTS-P tends to select solutions with fewer edges under the initial random weights \mathbf{w} , and it gradually finds better solutions (possibly with more edges) when better weights are learned. We have such observations for most of the samples, which suggest that the proposed method works the way it is supposed to. More visualizations can be found in the appendix.

6 Future Directions

Although we observed that the approximation ratio becomes better with the increase in training size and training iteration, it is not always the case that the solution converges to the optimal one – for example, Fig. 2-Right and the visualizations in the Appendix. This is reasonable because two solutions may have similar costs but with very different edge sets. Depending on the needs of the agents, other metrics can be adopted, which may require new designs of the hypothesis space and training methods. Another important future direction is to enhance the proposed method through (deep) representation learning.

Acknowledgement. This project is supported in part by National Science Foundation under Award Career IIS-2144285.

References

1. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. *Science* **286**(5439), 509–512 (1999)
2. Bengio, Y.: Using a financial training criterion rather than a prediction criterion. *Int. J. Neural Syst.* **8**(04), 433–443 (1997)
3. Bertsimas, D., Kallus, N.: From predictive to prescriptive analytics. *Manage. Sci.* **66**(3), 1025–1044 (2020)
4. Chen, C., Seff, A., Kornhauser, A., Xiao, J.: Deepdriving: learning affordance for direct perception in autonomous driving. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2722–2730 (2015)
5. Demetrescu, C., Goldberg, A.V., Johnson, D.S.: Implementation challenge for shortest paths. *Encyclopedia Algorithms* **15**, 54 (2008)
6. Edwards, W.: The theory of decision making. *Psychol. Bull.* **51**(4), 380 (1954)
7. Ford, B., Nguyen, T., Tambe, M., Sintov, N., Fave, F.D.: Beware the soothsayer: from attack prediction accuracy to predictive reliability in security games. In: Khouzani, M.H.R., Panaousis, E., Theodorakopoulos, G. (eds.) GameSec 2015. LNCS, vol. 9406, pp. 35–56. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25594-1_3
8. Hofmann, T., Schölkopf, B., Smola, A.J.: Kernel methods in machine learning (2008)
9. Hwang, F.K., Richards, D.S.: Steiner tree problems. *Networks* **22**(1), 55–89 (1992)
10. Joachims, T., Finley, T., Yu, C.N.J.: Cutting-plane training of structural svms. *Mach. Learn.* **77**(1), 27–59 (2009)

11. Kochenderfer, M.J.: Decision making under uncertainty: theory and application. MIT press (2015)
12. Lai, C., Murthy, D., Xie, M.: Weibull distributions. Wiley Interdisciplinary Reviews: Computational Statistics **3**(3), 282–287 (2011)
13. Lehmann, E.L., Casella, G.: Theory of point estimation. Springer Science & Business Media (2006)
14. Leskovec, J., Chakrabarti, D., Kleinberg, J., Faloutsos, C., Ghahramani, Z.: Kronecker graphs: an approach to modeling networks. J. Mach. Learn. Res. **11**(2) (2010)
15. Lucchi, A., Li, Y., Fua, P.: Learning for structured prediction using approximate subgradient descent with working sets. In: Proceedings of the CVPR (2013)
16. Murphy, K.P.: Machine learning: a probabilistic perspective. MIT press (2012)
17. Provost, F., Fawcett, T.: Data science and its relationship to big data and data-driven decision making. Big Data **1**(1), 51–59 (2013)
18. Rajkumar, A., Agarwal, S.: Online decision-making in general combinatorial spaces. Advances in Neural Information Processing Systems **27** (2014)
19. Suthaharan, S., Suthaharan, S.: Support vector machine. Machine learning models and algorithms for big data classification: thinking with examples for effective learning, pp. 207–235 (2016)
20. Tokdar, S.T., Kass, R.E.: Importance sampling: a review. Wiley Interdisciplinary Reviews: Computational Statistics **2**(1), 54–60 (2010)
21. Tong, G.: Usco-solver: solving undetermined stochastic combinatorial optimization problems. NeurIPS **34**, 1646–1659 (2021)
22. Vazirani, V.V.: Approximation algorithms, vol. 1. Springer (2001)
23. Watts, D.J., Strogatz, S.H.: Collective dynamics of ‘small-world’ networks. Nature **393**(6684), 440–442 (1998)



Enhancing Policy Gradient for Traveling Salesman Problem with Data Augmented Behavior Cloning

Yunchao Zhang¹, Kewen Liao² , Zhibin Liao³, and Longkun Guo^{1,4}

¹ School of Computer Science, Qilu University of Technology
(Shandong Academy of Sciences), Jinan 250316, China

² HilstLab, Peter Faber Business School, Australian Catholic University,
North Sydney 2060, Australia

Kewen.Liao@acu.edu.au

³ Australian Institute for Machine Learning, University of Adelaide,
Adelaide 5005, Australia

zhibin.liao@adelaide.edu.au

⁴ School of Mathematics and Statistics, Fuzhou University, Fuzhou 350116, China
lkguo@fzu.edu.cn

Abstract. The use of deep reinforcement learning (DRL) techniques to solve classical combinatorial optimization problems like the Traveling Salesman Problem (TSP) has garnered considerable attention due to its advantage of flexible and fast model-based inference. However, DRL training often suffers low efficiency and scalability, which hinders model generalization. This paper proposes a simple yet effective pre-training method that utilizes behavior cloning to initialize neural network parameters for policy gradient DRL. To alleviate the need for large amounts of demonstrations in behavior cloning, we exploit the symmetry of TSP solutions for augmentation. Our method is demonstrated by enhancing the state-of-the-art policy gradient models Attention and POMO for the TSP. Experimental results show that the optimality gap of the solution is significantly reduced while the DRL training time is greatly shortened. This also enables effective and efficient solving of larger TSP instances.

Keywords: Traveling salesman problem · behavior cloning · policy gradient · deep reinforcement learning

1 Introduction

The traveling salesman problem (TSP) is a well-known classical combinatorial optimization problem in which a salesman aims to visit a list of cities once, starting from an origin city and returning to it, intending to minimize the tour length. TSP problems and variants are commonly computationally NP-hard even in the Euclidean space [18] which requires exponential time to find an exact solution. TSP and its variants have seen numerous real-world applications [16] in logistics,

transportation, and circuit design. Many traditional algorithms have been developed for solving TSPs, including exact algorithm [14], heuristic algorithm [7], and approximate algorithm [30]. Similar to the TSP, traditional algorithms for solving combinatorial optimization problems usually require expert knowledge of mathematical modeling and optimization associated with the structure and constraints of the problem. In addition, these algorithms are limited to solving specific types of problems and are not generalizable to a variety of combinatorial optimization problems. In practical applications, it is often not necessary to obtain an exact solution to a combinatorial optimization problem, but instead, practitioners are seeking approximate feasible solutions that are efficient.

In recent years, deep reinforcement learning (DRL) has emerged as a promising approach [4] to tackle combinatorial optimization. It leverages the capability of neural networks to automatically learn intricate patterns and mappings, bypassing the need for manually designed processes. DRL-based methods demonstrated effectiveness in problem-solving along with generalizability to related combinatorial optimization problems. Meanwhile, there are several challenges when using DRL to solve combinatorial optimization problems. DRL training poses significant overhead such as high computational cost and low sample efficiency, resulting in a considerably prolonged training time that prohibits scalability. Policy gradient [29] has been a popular reinforcement learning method for combinatorial optimization due to its simplicity of integration and property of faster convergence. Its policy network is designed to directly optimize the policy of an agent by computing gradients of expected rewards. Imitation learning [9] such as behavior cloning is a type of machine learning algorithm that learns behavior policies from demonstrations. It has been adopted for robot control, autonomous driving, and game playing [27]. AlphaGo [24] utilized imitation learning to learn behaviors from human players and then it can play against itself to make improved sequential decisions through reinforcement learning. The algorithm proposed in [21] also leveraged imitation learning and reinforcement learning and demonstrated the effectiveness in its application of controlling robotic arms. These successful examples represent innovative combinations of two learning approaches to overcome the limitation of using each approach alone. Inspired by their works, we improve the state-of-the-art policy gradient DRL methods for the TSP by proposing an effective pre-training method with behavior cloning to initialize the policy network. We also exploit the symmetry of TSP solutions in pre-training. Our main contributions are summarized as follows:

- We propose to employ behavior cloning with generated demonstration data to pre-train a policy network before it gets optimized by the policy gradient algorithm. Our pre-training method significantly enhances the state-of-the-art policy gradient DRL model’s performance and convergence.
- We utilize the symmetry of TSP solutions to conduct augmentation on the demonstration data. This addresses the issue of lacking sufficient demonstration data for achieving effective behavior cloning.

- Our method enables the use of a pre-trained model developed from a smaller TSP instance to initialize the learning model for a larger TSP instance with quality solution and faster convergence.

2 Related Work

Machine learning methods for solving the TSP are concentrated on policy gradient reinforcement learning. In the following, we briefly cover the traditional combinatorial optimization methods and tools, early-developed ML methods, and finally the policy gradient.

Traditional Approaches. The solvers Concorde [6] and LKH3 [8] use linear programming and heuristics to exactly solve the TSP. OR Tools [19] is a software library that provides optimization tools to solve a range of optimization problems including TSP variants. Unlike Concorde and LKH3, solutions from OR Tools are not guaranteed to be optimal. Farthest Insertion is a naive greedy algorithm that is suitable for small to medium-sized problem instances but also does not guarantee solution optimality. 2-opt [1] is a popular local search-based approximation algorithm that is both simple and efficient.

Early Development. Vinyals et al. [28] proposed a Pointer Network (Ptr-Net) model based on a sequence-to-sequence architecture [25] that utilizes attention pointers to select input to form output. The solution is constructed in an autoregressive approach, and the model is trained using supervised learning. Dai et al. [10] utilized a combination of Deep Q-Network (DQN) [22, 26] and graph embedding Structure2Vec [5]. The graph embedding network provides an output of the Q value based on the current state and then selects the action greedily according to the Q value, gradually constructing a solution.

Policy Gradient. Bello et al. [3] proposed to use reinforcement learning to further train the Ptr-Net. They suggested using the negative tour length as the reward and the policy gradient algorithm REINFORCE [29] to optimize the parameters of the Ptr-Net. Kool et al. [12] proposed a Transformer-based model – Attention Model (AM) for the TSP. The model maps the graph embedding and selected node embeddings to the context embedding. They also utilize the REINFORCE algorithm but in addition with a simple baseline and a deterministic greedy rollout strategy to train the model. The AM is currently one of the most effective deep learning models in terms of both solution quality and efficiency. Different from the full attention mechanism adopted by the above-mentioned Transformer-based model, the sparse attention mechanism takes up significantly less memory and trains notably faster [35]. Ma et al. [15] proposed a novel model called the Graph Pointer Network (GPN) by combining the Graph Neural Network (GNN) [33] and the Ptr-Net. They introduced a graph embedding layer on the input to build the Pointer Network, which captures the relationship between nodes. Kwon et al. [13] introduced Policy Optimization with Multiple Optima (POMO), which is another state-of-the-art method for the TSP. The POMO

improves the solution representation of the TSP, adapts the REINFORCE algorithm, and mandates distinct rollouts for each optimal solution. The POMO’s low-variance baseline also enhances the speed and stability of the DRL training, while increasing its resilience to local minima. Paulo et al. [17] proposed using the policy gradient algorithm to learn a local search heuristic algorithm based on 2-opt operators. Xin et al. [31] proposed a Multi-Decoder Attention Model (MDAM) based on AM to train various policies, effectively increasing the chances of finding optimal solutions. They also separately discovered in [32] that the existing methods did not fully comply with the Bellman optimality principle [2]. Therefore, a solution was proposed to remove visited nodes and recalculate node embeddings at each selection step. Yang et al. [34] proposed a new Random baseline in RL, which stands out for its contribution of reducing the average training time by 16%.

3 Behavior Cloning Enhanced Policy Gradient

The input of the TSP in two-dimensional Euclidean space consists of the coordinates of n nodes, which are denoted as $s = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$. The aim is to determine a permutation π of n nodes that results in the shortest tour by visiting each node once except the starting node. Let $\pi(i)$ represent the i -th node of a TSP tour, and $\pi(< i)$ represent the partial solution before reaching the i -th node. In addition, tour length is denoted as $L(\pi|s)$, which is the distance traveled from the starting node to all other nodes and then back to the starting node. The reinforcement learning algorithm [3] proposed by Bello et al. utilizes the parameter θ of a neural network for defining the policy network $p_\theta(\pi|s)$. The policy network represents the probability of π for a given TSP instance s . Using the chain rule, the neural network factorizes the probability into a product of conditional probabilities:

$$p_\theta(\pi | s) = \prod_{i=1}^n p_\theta(\pi(i) | \pi(< i), s) \quad (1)$$

In the following, we will provide a detailed explanation of how to perform data augmentation on the demonstration data required by behavior cloning and how to pre-train the policy network $p_\theta(\pi|s)$ using behavior cloning. Then, we will elaborate on how to further train the policy network with reinforcement learning.

3.1 Behavior Cloning Pre-training

Behavior cloning [20] is a supervised imitation learning method that utilizes expert demonstrations to train policy networks. This method does not require random exploration or reward design, and it can produce relatively good results quickly. Demonstrations are composed of corresponding tuples of (states, actions), that is $D = \{(s_1, \pi_1), \dots, (s_m, \pi_m)\}$ where m denotes the number of

instances. The downside of behavior cloning is that it requires a substantial amount of demonstrations to be effective. However, for the TSP, even the most efficient solver takes up substantial time, especially for solving large-scale problem instances (e.g., n in the magnitude above thousands). As a consequence, sufficient demonstrations would be costly or impossible to obtain.

Data Augmentation. In fact, TSP solutions themselves possess some interesting properties. Specifically, every TSP solution/tour takes the form of a closed loop on a two-dimensional plane, whereas, in deep learning, this solution is encoded in a vector structure. Consequently, when constructing the demonstration data, we only designate a single node as the starting point for the solution, while each node in the loop can be the starting point. For instance, in behavior cloning, the initial node (x_1, y_1) is consistently designated as the starting point $\pi(1)$ for the partial solution π in solving the TSP. The autoregressive model will then be influenced by this choice, with the trained policy showing a tendency to select the initial node (x_1, y_1) as the starting point $\pi(1)$ for π in subsequent solution iterations. To address the above issue while augmenting the demonstration data, we utilize the symmetry of any TSP solution (i.e., let each node be a starting node $\pi(1)$) as follows:

$$\pi = \{j, \dots, n, 1, \dots, j-1\} \Leftrightarrow \pi' = \{j+1, \dots, n, 1, \dots, j\}, \quad (2)$$

where for $j \in (1, 2, \dots, n)$. As a result, we can generate up to n different representations of a solution, so the size of the demonstration data can be significantly augmented.

Pre-training. After obtaining sufficient demonstration data, the policy network can then be effectively initialized in a supervised way. We employ an expert policy p_E from behavior cloning and train a learning policy $p_\theta(\pi|s)$, with the objective to approximate the expert policy p_E by minimizing the difference between the predictions generated by the learning policy $p_\theta(\pi|s)$ and the expert actions from p_E . As displayed in the left part of Fig. 1, we use the coordinates s of n nodes and a partial solution $\pi(< i)$ as a demonstration input to the policy network $p_\theta(\pi|s)$. The policy network then outputs probability values of n nodes. We select a node as $\pi(i)$ based on the demonstrations and record the corresponding probability value p_i . After an episode ends, we obtain the probabilities (p_1, \dots, p_n) of n nodes. We use maximum likelihood as the loss function to maximize the probability of selecting the nodes based on the demonstrations. The loss function of the policy network is represented as follows:

$$\mathcal{L}(\theta) = - \sum_{i=1}^n \log p_\theta(\pi(i)|\pi(< i), s). \quad (3)$$

Using Adam optimization [11] on the above loss function can direct the policy $p_\theta(\pi|s)$ to be closer to the expert policy p_E .

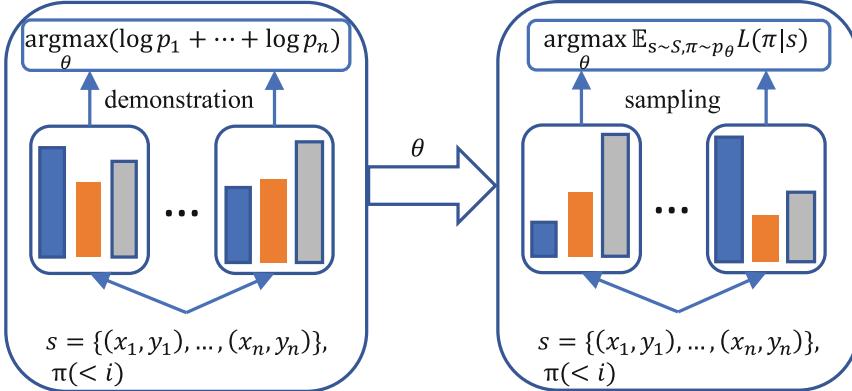


Fig. 1. Behavior cloning enhanced policy gradient.

3.2 Policy Gradient Training

Due to the limited collection of state-action pairs through expert demonstration and the non-explorative nature of behavior cloning, the pre-trained model is unable to explore new solutions or policies on its own. Therefore, the policy network trained through behavior cloning often cannot generalize well to new instances, i.e., distribution drift [23] exists. Although behavior cloning converges faster than the policy gradient algorithm, its prediction accuracy is often much lower due to limited solution exploration. To address the aforementioned problems, we utilize the advantage of behavior cloning to initialize better policy parameters, and further train the policy network model through policy gradient, which can continuously improve the policy through interaction with the environment. As shown in the right part of Fig. 1, after the policy network is initialized, we use it to generate the probability values of unselected nodes each time and then sample the nodes. At the end of the episode, we obtain solution π and its corresponding probability values. Our objective is to obtain the optimal policy by maximizing the expected return, which is equivalent to minimizing the tour length $L(\pi | s)$. The algorithm relies on refining the policy network $p_{\theta}(\pi | s)$ to achieve the optimal policy. Regarding the TSP, instances (i.e., node coordinates) are randomly sampled from S , thus the objective function can be represented as $J(\theta) = \mathbb{E}_{s \sim S, \pi \sim p_{\theta}(\pi | s)} L(\pi | s)$, and the policy gradient is defined as:

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{s \sim S, \pi \sim p_{\theta}(\pi | s)} [(L(\pi | s) \nabla_{\theta} \log p_{\theta}(\pi | s))]. \quad (4)$$

We use the REINFORCE algorithm [29] with a baseline to optimize the objective function. The REINFORCE algorithm uses the Monte Carlo method to approximate the policy gradient (4). Here, $b(s)$ represents the baseline that is independent of the solution π and is used to estimate the tour length to reduce the variance of the gradient and speed up the training process. The algorithm generates B TSP instances from distribution S , and samples a solution π_j in each instance:

Algorithm 1. Behavior Cloning Enhanced Policy Gradient

```

1: procedure TRAINING(demonstrations  $D$ , training set  $S$ , steps per epoch  $T$ , batch
   size  $B$ , number of nodes of the instance  $N$ )
2:   Perform data augmentation on  $D$  with formula (2)
3:   Initialize pre-training network parameter  $\theta$ :
4:   for step = 1,...,T do
5:     Set  $s_j, \pi_j \leftarrow \text{SAMPLEINPUT}(D)$  for  $j \in \{1, \dots, B\}$ 
6:      $\mathcal{L}(\theta) = - \sum_{j=1}^B \sum_{i=1}^N \log p_\theta(\pi_j(i) | \pi_j(< i), s_j)$ 
7:      $\theta = \text{ADAM}(\theta, \mathcal{L}(\theta))$ 
8:   end for
9:   Policy gradient with pre-trained network parameter  $\theta$ :
10:  for step = 1,...,T do
11:     $s_j \leftarrow \text{SAMPLEINPUT}(S)$  for  $j \in \{1, \dots, B\}$ 
12:     $\pi_j \leftarrow \text{SAMPLESOLUTION}(p_\theta(\cdot | s_j))$  for  $j \in \{1, \dots, B\}$ 
13:    Use (5) as the loss function  $\nabla_\theta J(\theta)$ 
14:     $\theta = \text{ADAM}(\theta, \nabla_\theta J(\theta))$ 
15:  end for
16:  return  $\theta$ 
17: end procedure

```

$$\nabla_\theta J(\theta) \approx \frac{1}{B} \sum_{j=1}^B (L(\pi_j | s_j) - b(s_j)) \nabla_\theta \log p_\theta(\pi_j | s_j). \quad (5)$$

The details of the entire algorithm with behavior cloning pre-training and policy gradient are described in Algorithm 1. Finally, as the number of nodes n increases, generating sufficient demonstrations according to the problem size becomes virtually impossible, hence making the solution of large-scale TSP highly challenging. Fortunately, we can use a TSP model pre-trained on smaller TSP instances to be further trained on larger TSPs via policy gradient training. This is possible because the pre-training phase has already initialized the training model parameters with expert policies, thereby the time-consuming random exploration steps in reinforcement learning can be greatly reduced.

4 Experiments

Experiments were conducted on a server with an Intel Xeon Processor (Icelake) CPU, an Nvidia A100 GPU (40 G), and CentOS 7.9 64bit. By convention, TSP node coordinates are uniformly sampled within a square of unit side length. For each TSP of size n , algorithms (covered in the related work) were tested on 10,000 randomly generated TSP instances (as the test set w.r.t. n), with their average output tour length serving as the evaluation metric for model effectiveness. The pre-training method presented in this paper was applied to two state-of-the-art policy gradient algorithms AM [12] and POMO [13], which are the most effective deep learning models in terms of both solution quality and

Table 1. A comprehensive testing results on the TSP

Method	n=20		n=50		n=100		n=250		n=500	
	Len.	Gap	Len.	Gap	Len.	Gap	Len.	Gap	Len.	Gap
Concorde	3.83	0.00%	5.69	0.00%	7.77	0.00%	11.89	0.00%	16.55	0.00%
LKH3	3.83	0.00%	5.69	0.00%	7.77	0.00%	11.89	0.00%	16.55	0.00%
OR Tools	3.86	0.94%	5.85	2.87%	8.06	3.86%	12.86	8.15%	17.45	5.43%
Farthest Insertion	3.92	2.28%	6.01	5.55%	8.35	7.59%	13.03	9.55%	18.29	10.50%
2-opt	3.96	3.37%	6.13	7.66%	8.52	9.69%	13.25	11.46%	18.60	12.39%
Ptr-Net [28]	3.88	1.28%	7.66	34.53%	10.81	39.18%	-	-	-	-
Rl Ptr-Net [3]	3.89	1.54%	5.95	4.50%	8.30	6.86%	-	-	-	-
S2V-DQN [10]	3.89	1.42%	5.99	5.20%	8.31	6.99%	13.08	10.00%	18.43	11.35%
GPN [15]	3.87	0.94%	5.95	4.55%	8.34	7.42%	13.54	13.88%	19.53	17.99%
2-OPT DL [17]	3.84	0.23%	5.70	0.21%	7.83	0.87%	-	-	-	-
MDAM [31]	3.84	0.29%	5.73	0.63%	7.93	2.10%	-	-	-	-
ASW-TAM [32]	3.84	0.23%	5.76	1.16%	8.01	3.13%	-	-	-	-
Rand Baseline [34]	3.84	0.23%	5.78	1.58%	7.87	1.29%	-	-	-	-
AM [12]	3.85	0.50%	5.80	1.86%	8.12	4.54%	13.10	10.18%	19.85	19.94%
*AM-Ours	3.84	0.23%	5.77	1.33%	7.99	2.87%	12.66	6.48%	18.30	10.57%
POMO [13]	3.83	0.04%	5.70	0.21%	7.80	0.46%	12.42	4.46%	20.51	23.92%
*POMO-Ours	3.83	0.00%	5.69	0.04%	7.78	0.17%	12.07	1.51%	17.63	6.53%

efficiency. We also conducted ablation studies for both pre-training and reinforcement learning phases to evaluate the contributions of different components in the model.

4.1 Experimental Results

For testing on the TSP, we conducted a comprehensive study involving multiple traditional algorithms and state-of-the-art deep learning models. Due to the randomness of TSP instances, we report the average tour length (Len) to two decimal places, but for calculating the optimality gap (Gap), we use the average tour length rounded to three decimal places. Note that traditional methods such as Concorde and LKH3 produced optimal solutions from the solver. However, when $n = 500$, the solver takes approximately 38.6 h to solve 10,000 instances, whereas the trained POMO takes only 6 min for its inference. This demonstrates that the traditional methods often cannot achieve both solution speed and accuracy simultaneously. For TSP sizes up to $n = 100$, most models were able to attain a satisfactory solution. Due to the training complexity, Ptr-Net, Rl Ptr-Net, MDAM, and ASW-TAM cannot be generalized to larger TSP instances (i.e., $n = 250$ or 500). In addition, S2V-DQN, GPN, AM, and POMO can only generalize using the TSP100-trained RL models due to the excessive time in training larger models. However, with our pre-training that results in faster RL, we were able to obtain TSP250-trained RL models AM-Ours and POMO-Ours within the time limit and we can then generalize them to $n = 500$. As shown

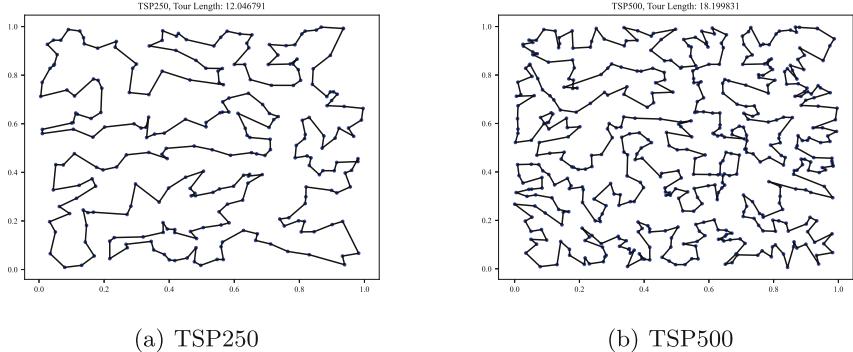


Fig. 2. The two graphics respectively display the solutions when $n = 250/500$. The tour lengths are marked above each graphic.

in Table 1, we obtained significantly improved results on both AM and POMO, especially on the larger TSPs. In particular, the POMO with our pre-training method (i.e., POMO-Ours) reached a solution that is close to optimal. Figure 2 displays the solutions to our large-scale TSP instances.

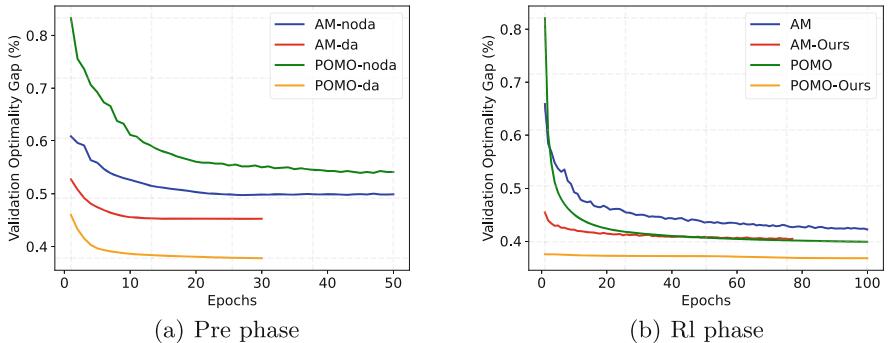


Fig. 3. Training performances of AM and POMO in both pre-training (pre) and reinforcement learning (rl) phases.

4.2 Ablation Study

During the pre-training phase, models that use data augmentation demonstrate faster convergence speed and better performance compared to those that do not use data augmentation. We first performed behavior cloning pre-training alone (as in Sect. 3.1) on AM and POMO. Specifically, we used the Concorde solver [6] to generate exact demonstrations by

collecting state-action pairs. We compared the training performance and convergence speed of two models with pre-training using data augmentation (da) or not. As shown in Figure 3(a), we present the validation optimality gap (It exhibits the same trend across all problem sizes in the experiment.) as epochs increase, which proves the conclusion we made at the beginning of this paragraph. Furthermore, when both models employed data-augmented behavior cloning, the performance of POMO is superior to AM because it achieved a lower validation optimality gap while they both quickly converged after 30 epochs of training.

During the reinforcement learning phase, using pre-trained AM and POMO for reinforcement learning can achieve convergence with fewer epochs compared to directly using AM and POMO for reinforcement learning, and can also achieve better performance. We then applied data-augmented pre-training to AM and POMO and contrasted the performances of policy gradient training with and without pre-training. As shown in Fig. 3(b), we further showcase the comparative RL training, wherein pre-trained models have shown a lower validation optimality gap compared to their respective models. Notably, the POMO with pre-training already reached a good performance from the start and then it converged slowly. In addition, as shown in Table 2, we present the RL training time of models trained with and without our pretraining method, as well as the Random Baseline [34], to further demonstrate its advantage. Trainings taking longer than a week were terminated and their training times (in Table 2) and testing results (in Table 1) were not recorded. If not specified otherwise, we use greedy decoding to obtain the testing results.

Table 2. RL training times comparison (in hours)

Method	TSP20	TSP50	TSP100	TSP250
Random Baseline	4.37	9.50	17.50	-
AM	5.20	11.25	20.83	-
*AM-Ours	0.39	2.79	15.64	67.52
POMO	1.42	11.70	170.23	-
*POMO-Ours	0.35	3.60	56.10	127.50

5 Conclusion

In this paper, we first introduced pre-training with data-augmented behavior cloning for solving combinatorial optimization problems based on policy gradient reinforcement learning. We conducted a specific study on the classical TSP and demonstrated that our method has significantly progressed in narrowing the optimality gap, reducing the training time, and solving larger instances. For future work, we will apply our method for solving other important TSP variants

to demonstrate model generalizability. We will also investigate more scalable solutions and conduct comprehensive experiments to validate the scalability.

Acknowledgments. This work is supported by the Taishan Scholars Young Expert Project of Shandong Province (No. tsqn202211215) and the National Science Foundation of China (Nos. 12271098 and 61772005). The corresponding author is Longkun Guo (lkguo@fzu.edu.cn).

References

1. Aarts, E.H., Lenstra, J.K.: Local search in combinatorial optimization. Princeton University Press (2003)
2. Bellman, R.: Dynamic programming treatment of the travelling salesman problem. *J. ACM (JACM)* **9**(1), 61–63 (1962)
3. Bello, I., Pham, H., Le, Q.V., Norouzi, M., Bengio, S.: Neural combinatorial optimization with reinforcement learning. In: Proceedings of International Conference on Learning Representations (ICLR) (2017)
4. Bengio, Y., Lodi, A., Prouvost, A.: Machine learning for combinatorial optimization: a methodological tour d’horizon. *Eur. J. Oper. Res.* **290**(2), 405–421 (2021)
5. Dai, H., Dai, B., Song, L.: Discriminative embeddings of latent variable models for structured data. In: International Conference on Machine Learning, pp. 2702–2711. PMLR (2016)
6. Applegate, D., Robert Bixby, V.C., Cook, W.: Concorde TSP Solver (2006). <https://www.math.uwaterloo.ca/tsp/concorde/index.html>
7. Halim, A.H., Ismail, I.: Combinatorial optimization: comparison of heuristic algorithms in travelling salesman problem. *Arch. Comput. Methods Eng.* **26**, 367–380 (2019)
8. Helsgaun, K.: An extension of the lin-kernighan-helsgaun TSP solver for constrained traveling salesman and vehicle routing problems: Technical report (2017)
9. Hussein, A., Gaber, M.M., Elyan, E., Jayne, C.: Imitation learning: a survey of learning methods. *ACM Comput. Surv. (CSUR)* **50**(2), 1–35 (2017)
10. Khalil, E., Dai, H., Zhang, Y., Dilkina, B., Song, L.: Learning combinatorial optimization algorithms over graphs. *Adv. Neural. Inf. Process. Syst.* **30** (2017)
11. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: Proceedings of International Conference on Learning Representations (ICLR) (2015)
12. Kool, W., van Hoof, H., Welling, M.: Attention, learn to solve routing problems! In: International Conference on Learning Representations (2019)
13. Kwon, Y.D., Choo, J., Kim, B., Yoon, I., Gwon, Y., Min, S.: Pomo: Policy optimization with multiple optima for reinforcement learning. *Adv. Neural. Inf. Process. Syst.* **33**, 21188–21198 (2020)
14. Lawler, E.L., Wood, D.E.: Branch-and-bound methods: a survey. *Oper. Res.* **14**(4), 699–719 (1966)
15. Ma, Q., Ge, S., He, D., Thaker, D., Drori, I.: Combinatorial optimization by graph pointer networks and hierarchical reinforcement learning. In: AAAI Workshop on Deep Learning on Graphs: Methodologies and Applications (2020)
16. Matai, R., Singh, S.P., Mittal, M.L.: Traveling salesman problem: an overview of applications, formulations, and solution approaches. *Traveling Salesman Problem, Theory and Applications* **1** (2010)

17. d O Costa, P.R., Rhuggenaath, J., Zhang, Y., Akcay, A.: Learning 2-opt heuristics for the traveling salesman problem via deep reinforcement learning. In: Asian Conference on Machine Learning, pp. 465–480. PMLR (2020)
18. Papadimitriou, C.H.: The euclidean travelling salesman problem is np-complete. *Theoret. Comput. Sci.* **4**(3), 237–244 (1977)
19. Perron, L., Furnon, V.: Or-tools (2022). <https://developers.google.com/optimization/>
20. Pomerleau, D.A.: Alvinn: an autonomous land vehicle in a neural network. *Adv. Neural. Inf. Process. Syst.* **1** (1988)
21. Rajeswaran, A., et al.: Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. In: Proceedings of Robotics: Science and Systems. Pittsburgh, Pennsylvania (June 2018)
22. Riedmiller, M.: Neural fitted Q iteration – first experiences with a data efficient neural reinforcement learning method. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) ECML 2005. LNCS (LNAI), vol. 3720, pp. 317–328. Springer, Heidelberg (2005). https://doi.org/10.1007/11564096_32
23. Ross, S., Gordon, G., Bagnell, D.: A reduction of imitation learning and structured prediction to no-regret online learning. In: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, pp. 627–635. JMLR Workshop and Conference Proceedings (2011)
24. Silver, D., et al.: Mastering the game of go with deep neural networks and tree search. *Nature* **529**(7587), 484–489 (2016)
25. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. *Adv. Neural. Inf. Process. Syst.* **27** (2014)
26. Thrun, S., Littman, M.L.: Reinforcement learning: an introduction. *AI Mag.* **21**(1), 103–103 (2000)
27. Torabi, F., Warnell, G., Stone, P.: Recent advances in imitation learning from observation. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, pp. 6324–6331 (2019)
28. Vinyals, O., Fortunato, M., Jaitly, N.: Pointer networks. *Adv. Neural. Inf. Process. Syst.* **28** (2015)
29. Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Reinforc. Learn.*, 5–32 (1992)
30. Williamson, D.P., Shmoys, D.B.: The design of approximation algorithms. Cambridge University Press (2011)
31. Xin, L., Song, W., Cao, Z., Zhang, J.: Multi-decoder attention model with embedding glimpse for solving vehicle routing problems. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 12042–12049 (2021)
32. Xin, L., Song, W., Cao, Z., Zhang, J.: Step-wise deep learning models for solving routing problems. *IEEE Trans. Industr. Inf.* **17**(7), 4861–4871 (2021)
33. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? In: International Conference on Learning Representations (2019)
34. Yang, H., Gu, M.: A new baseline of policy gradient for traveling salesman problem. In: 2022 IEEE 9th International Conference on Data Science and Advanced Analytics (DSAA), pp. 1–7. IEEE (2022)
35. Zaheer, M., et al.: Big bird: transformers for longer sequences. *Adv. Neural. Inf. Process. Syst.* **33** (2020)



Leveraging Transfer Learning for Enhancing Graph Optimization Problem Solving

Hui-Ju Hung¹(✉), Wang-Chien Lee¹, Chih-Ya Shen², Fang He¹, and Zhen Lei¹

¹ The Pennsylvania State University, University Park, PA 16802, USA
 {hzhl131,wu12,fxh35,zlei}@psu.edu

² National Tsing Hua University, Hsinchu 300044, Taiwan
chihya@cs.nthu.edu.tw

Abstract. Reinforcement learning to solve graph optimization problems has attracted increasing attention recently. Typically, these models require extensive training over numerous graph instances to develop generalizable strategies across diverse graph types, demanding significant computational resources and time. Instead of tackling these problems one by one, we propose to employ transfer learning to utilize knowledge gained from solving one graph optimization problem to aid in solving another. Our proposed framework, dubbed the *State Extraction with Transfer-learning (SET)*, focuses on quickly adapting a model trained for a specific graph optimization task to a new but related problem by considering the distributional differences among the objective values between the graph optimization problems. We conduct a series of experimental evaluations on graphs that are both synthetically generated and sourced from real-world data. The results demonstrate that SET outperforms other algorithmic and learning-based baselines. Additionally, our analysis of knowledge transferability provides insights into the effectiveness of applying models trained on one graph optimization task to another. Our study is one of the first studies exploring transfer learning in the context of graph optimization problems.

Keywords: graph optimization problem · reinforcement learning · transfer learning · node representation

1 Introduction

Proven to be NP-hard, many graph optimization problems do not have polynomial-time algorithms to solve them. Yet, the quest to solve these complex problems is of great importance, as they find practical applications in diverse areas like transportation, scheduling, and social networking [14, 30]. Traditionally, research efforts in solving graph optimization problems are mostly garnered in the theory and algorithm design communities. In recent years, however, research interests in solving graph optimization problems with various *learning* approaches such as supervised, unsupervised, or reinforcement learning have been growing [6, 13, 19, 28, 32]. In the following, we informally describe the learning task.

Table 1. Comparison between *No-Transfer* and *Transfer* in solving MC

Approach	<i>No-Transfer</i>	<i>Transfer</i>
Average objective	4.2578	4.2734
Number of training iterations to converge	4673	1443

Learning to Solve Graph Optimization Problems. To solve a graph optimization problem, a typical algorithmic approach is to develop a greedy algorithm that begins with an empty solution and sequentially incorporates the node with the highest score from the candidates until a predefined termination criterion is met. Instead of adopting a manually designed scoring function, a machine learning model may be trained to serve as the scoring function in the greedy algorithm.

Many models have been introduced to tackle various graph optimization problems and achieve commendable results [3, 6, 9, 13, 36]. These models typically undergo intensive training on numerous graph instances to automatically derive effective strategies for a variety of graph types, making the extensive training duration a challenge. An important research question arises in exploring ideas to address the challenge: *Do models designed for distinct graph optimization problems share common knowledge that can be used in other problem settings?* In other words, *is it possible to transfer the knowledge, e.g., inherent structural information of the graph instances, acquired from solving one graph optimization problem to another?*

To investigate whether *knowledge* acquired from one optimization problem can be beneficial for another, we introduce a motivating study, under the context of reinforcement learning, to examine the transferability of solutions between distinct problems. Specifically, we explore if insights gained from solving the Dense Subgraph (DSG) problem can aid in addressing the Maximum Clique (MC) problem. For clarity, we differentiate between two methodologies: *Transfer* and *No-transfer*, which share the same model structure constructed with GCN [34]. *Transfer* trains a model with the data collected for solving DSG first and adjusts the model with a few data collected for solving MC. In contrast, *No-transfer* trains the MC model from scratch with only the data collected for solving MC. Our comparative analysis assesses the *objective value* for the MC problem across a test set of 128 graphs, each with 20 nodes, achieved by the converged *Transfer* and *No-transfer* models.

The results averaged over three trials is shown in Table 1. *Transfer* achieves objective value of 4.2734, slightly better than the objective value of 4.2578 under *No-transfer*. More importantly, *Transfer* converges much faster, with 1443 training iterations, than *No-transfer* (4673 iterations). These findings suggest that parameters optimized for DSG can indeed enhance the training efficiency for MC, demonstrating the potential of transferring knowledge learned from one optimization challenge to benefit another.¹

¹ Notably, the efficacy of *knowledge transfer* extends beyond this specific pair of graph optimization problems and has been observed in several other problem combinations.

The aforementioned example compellingly illustrates that transfer learning can facilitate solving one optimization problem by applying knowledge from another within the context of reinforcement learning frameworks. This success prompts a deeper inquiry: *What specific information is learned and subsequently utilized to address a new optimization challenge?* Most models claim that they are able to accurately encapsulate structural information, which contributes to their superior performance. However, the precise nature of the information, i.e., what facets of structure or patterns are captured and how they are leveraged to navigate graph optimization problems, remains largely unexplored.

In this work, we introduce the State Extraction with Transfer-learning (SET) framework, designed to generalize a model across different graph optimization problems. The core idea of SET is to train a model predominantly on one optimization problem, which can then swiftly adapt to solve another, thereby indicating the generalizability of the structural information captured. SET achieves rapid adaptation by acknowledging and resolving the distributional differences among the objective values across multiple graph optimization problems. We validate the effectiveness of SET on various graph optimization problems. We also conduct an analysis to show the knowledge transferability of applying models trained on one graph optimization task to another.

The contributions of this paper are summarized as follows.

- Existing work exploiting reinforcement learning requires a large amount of data (or a large number of training iterations) to train a model or solve a graph optimization problem. To alleviate this issue, we propose to exploit transfer learning to reduce the number of training iterations while maintaining a comparable performance.
- We introduce the State Extraction with Transfer-learning (SET) framework, which transfers the useful structural features learned from one graph optimization problem to another via transferring model parameters. Moreover, SET is able to quickly adapt the model to address another graph optimization problem by considering the distributional differences of objective values across multiple graph optimization problems.
- Our evaluation demonstrates the efficacy of SET in finding near-optimal solutions efficiently in significantly less number of training iterations. SET outperforms the algorithmic and learning-based baselines in various graph optimization problems, where learning-based ones are fully trained by the data collected from the target problem.

The rest of this paper is organized as follows. Section 2 reviews the related work. Section 3 proposes the SET framework. Section 4 reports the evaluation. Finally, Sect. 5 concludes the paper.

2 Related Work

2.1 Transfer Learning

Transfer learning, aiming to boost model performance by drawing on knowledge from related domains or datasets, has been widely adopted in various fields such

as healthcare, natural language processing, computer vision, and robotics [12, 16, 20, 37]. To facilitate a task *transfer* from a source domain to a target domain, the most commonly employed method involves the transferal of model parameters, i.e., the parameters fine-tuned on the source task are utilized as the starting point for training the model on the target task.

Generally speaking, a model is composed of two parts [11, 29, 33]: the feature extractor, which identifies crucial features for the tasks, and the predictor, which bases its predictions on these extracted features. There are two primary forms of parameter adoption [29, 33]: 1) Linear Probing: where only the parameters in the predictor are modified using the target task’s training data and the parameters in the feature extractor are not. 2) Fine-tuning: where adjustments are made to the parameters in both the feature extractor and the predictor with the target task’s training data. Dai et al. have employed adversarial domain adaptation to bridge the distributional gap between source and target domains, thereby aiding the transfer of knowledge [7]. Similarly, Evci et al. have taken advantage of intermediate representations from all layers of the source model to construct a classification head for the target domain [10]. It should be noted, however, that the integration of these techniques into graph optimization problems has been relatively limited.

2.2 Reinforcement Learning for Graph Optimization Problems

Reinforcement learning (RL) to solve optimization problems has seen a notable surge in interest. These graph optimization problems are often delineated by the nature of their solutions, which may be either sequences or subsets of nodes. Among the variety of problems, those requiring a sequence of nodes, such as the Traveling Salesman Problem (TSP) and its variants, have garnered significant attention in the literature [3, 9].² For instance, Bello et al. leverage a *pointer network*, an architecture incorporating recurrent neural networks with an attention mechanism, to tackle TSP [3]. Dorigo et al. introduce a multi-agent system based on reinforcement learning to approach the TSP [9]. Expanding beyond the TSP, Nazari et al. address the Capacitated Vehicle Routing Problem, another routing problem that optimizes the paths of several vehicles in tandem, using RL techniques [24].

RL has also been adopted to address graph optimization problems where the goal is to identify a subset of nodes that meet certain criteria, such as the Minimum Vertex Cover (MVC) problem [6, 13, 35]. Dai et al. introduced a Q-learning-based method that leverages instances without optimal solutions [6]. Expanding on Dai et al.’s work, Yang and Shen enhanced the RL paradigm with graph augmentation techniques for tackling both the Maximum Independent Set (MIS) and MVC problems [35].³ Furthermore, Groshev et al. utilized an alternative approach, namely *imitation learning*, to train a scoring function that assesses the combination of a state and an action [13].

² TSP finds the shortest route that visits each node once and returns to the origin.

³ MIS involves selecting the largest set of vertices where no two are adjacent.

In summary, there has been considerable work leveraging RL to tackle graph optimization problems, yet these efforts typically train models for one specific problem at a time. In contrast, our study explores a new direction that employs transfer learning in the context of graph optimization problems.

3 The State Extraction with Transfer-Learning Framework

3.1 Framework Structure

Several studies have applied RL to solve various graph optimization problems, training independent models for each specific problem [3, 6, 8, 13]. Achieving good solution qualities, these models capture structural and state information helpful to different graph optimization problems. Building on this, we hypothesize that *models trained for different graph optimization problems may capture some common structural and state information*. By testing this hypothesis through Centered Kernel Alignment (CKA) analysis [22], we find that models independently trained from scratch for the DSG, MVC, and MC problems indeed capture shared structural and state information (detailed in Sect. 4.2), which motivates us to explore the feasibility of transfer learning in solving graph optimization problems.

To effectively implement transfer learning in graph optimization problems, understanding the model’s objective is crucial. As detailed later in Sect. 3.2, in an RL model developed from scratch, the model’s predictions need to be aligned with the accumulated rewards, which is a reflection of the objective values of the tackled graph optimization problem. Given that accumulated rewards are distinct across different graph optimization problems, transferring knowledge from a source graph optimization problem (*source problem* for short) to another target graph optimization problem (*target problem* for short) necessitates the model’s proficiency in obtaining not only the graph’s structural and state information but also the distributional differences between the source and target problems’ accumulated reward functions. Capturing the distributional differences is critical for a model to re-calibrate its predictions from being attuned to the source problem’s reward structure to aligning with that of the target problem. To expedite this reward adjustment, we introduce the State Extraction with Transfer-learning (SET) framework. SET exploits the power of transfer learning to extract pivotal features from the resolution of a different graph optimization problem. The essence of this approach is to incorporate the above distributional differences into a loss function, thus facilitating a more efficient transition in learning applicable information to the target problem.

The SET framework is illustrated in Fig. 1. The model structure has two components, *feature extractor* and *predictor*. The feature extractor, responsible for capturing graph and state information as would be done in a training-from-scratch scenario, is constructed using GCN [34].⁴ The predictors, comprising

⁴ Alternative layer-wise GNN methodologies such as GAT [31], GraphSage [15], or GBP [5] could also be used to build the feature extractors.

stacked fully connected layers, are tasked to make predictions based on the extracted features. For both the source and target problems, we utilize the same feature extractor and predictor architecture. Despite this consistency, the notable differences between the source and target problems can lead to a mismatch in the accumulated reward distributions, which leads to differences in the model's parameters, especially those in the predictor. Consequently, a model fine-tuned on data from the source problem may not accurately estimate the accumulated rewards for the target problem. The SET framework is designed to rectify this issue and facilitate problem alignment.

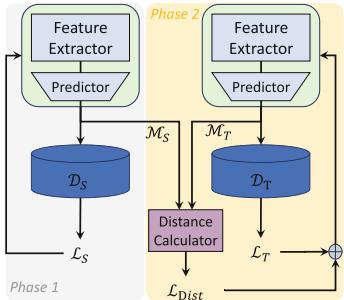


Fig. 1. SET framework illustration

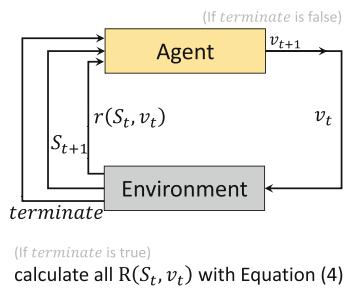


Fig. 2. RL process illustration

SET has two distinct phases. During the first phase, the feature extractor and the predictor for the source problem (i.e., the left green box in Fig. 1) are trained to address the source problem. To align the model to approximate the accumulated rewards of the source problem, the training of the first phase leverages a loss function which measures the *mean square error* (MSE) between the accumulated reward and the model's predictions. The loss function is mathematically represented as follows.

$$\mathcal{L}_S = \sum_{(S_t, v_t) \in \mathcal{D}_S} MSE(R(S_t, v_t), s^S(S_t, v_t)), \quad (1)$$

where \mathcal{D}_S denotes the dataset of (state, action) pairs compiled through the RL process, MSE is the mean square error function, $R(S_t, v_t)$ is the accumulated reward associated with state S_t and action v_t , and $s^S(S_t, v_t)$ is the score predicted by the model.

In the second phase, the parameters of the feature extractor for the source problem are transferred to initialize the feature extractor for the target problem (i.e., the right green box in Fig. 1). Then, we fine-tune both the feature extractor and the predictor for the target problem with the data collected in solving the target problem. The loss function for this phase (shown below) is similar to that used for the source problem, focusing on the *mean square error* (MSE) between the accumulated reward and the prediction made by the model.

$$\mathcal{L}_T = \sum_{(S_t, v_t) \in \mathcal{D}_T} MSE(R(S_t, v_t), s^T(S_t, v_t)), \quad (2)$$

where \mathcal{D}_T represents the training data generated from the target problem's RL process, and $s^T(S_t, v_t)$ denotes the prediction of the target model.

Additionally, the learning process needs to account for \mathcal{L}_{Dist} , which is for the distributional differences of model parameters between the source and target problems, as defined below.

$$\mathcal{L}_{Dist} = WS(dist(\mathcal{M}_S), dist(\mathcal{M}_T)), \quad (3)$$

where WS denotes the Wasserstein distance [21] that quantifies the divergence in distributions, and \mathcal{M}_S and \mathcal{M}_T denote the model parameters trained for the source and target problems, respectively. In SET, we train the source and target problems' models separately for training efficiency. Specifically, for addressing a new graph optimization problem, only the second phase of training is required.

3.2 Reinforcement Learning Process

Reinforcement learning (RL) is a paradigm of learning where an agent is trained to make sequential decisions within an environment, targeting on maximize obtained rewards [17]. When applying RL to the domain of graph optimization problems [3, 6, 13, 35], three foundational components are imperative: 1) The *environment setting* corresponds to the definitions of an action, like the *next state* and *obtained reward* after performing an action, and whether an agent reaches the *termination condition*. 2) The *framework* refers to how the learned algorithm operates. For example, a commonly used greedy framework performs one action at a time and never reverses any previous decisions. Each time a new action is performed, a new *intermediate solution* is created based on the previous intermediate solution. Here, an intermediate solution is a subset of nodes $S \subseteq V$, and the action is to add a node to the current intermediate solution. 3) The *model* is responsible for encoding the information of the graph structure and intermediate solution in the embeddings, which serves as the basis for the learned algorithm to make decisions. In the interaction with the environment, the agent's sequence of actions progresses until a termination criterion is met. The obtained feedback, in the form of rewards, informs and propels the agent towards optimal decision strategies.

The training data is collected during the interaction process with the environment. The training data comprises pairs of trials and their associated rewards, where a trial is a sequence of intermediate solutions derived from the actions taken, and the rewards serve as indicators of the efficacy of each solution. Formally, we represent a trial as $\langle S_1, v_1, S_2, v_2, \dots, S_n, v_n \rangle$, where S_t denotes the state or intermediate solution at step t , and v_t represents the action taken or the node added at step t . Consequently, a trial of length n encompasses n state-action pairs.

Utilizing the training data, our objective is to calibrate the model such that it effectively functions as an evaluative scoring function. This function takes a state S and an action v as inputs and produces a corresponding score $Q(S, v)$, with higher scores indicating more advantageous actions. To facilitate the training

of SET, we distinguish between two types of rewards: the immediate rewards and the accumulated rewards. The immediate rewards, expressed as $r(S_t, v_t)$, represent the instantaneous feedback received from the environment subsequent to executing an action. This immediate feedback serves as an indicator of the action's utility in the given state.

Training the model solely to approximate immediate rewards can result in a myopic perspective that neglects the potential benefits of subsequent actions. To overcome this limitation, it is imperative that the model is trained to estimate the accumulated rewards rather than just the immediate ones. For a given state S and action v , the accumulated reward $R(S_t, v_t)$ is a cumulative function of the immediate rewards $r(S_i, v_i)$ from the current step $i = t$ to the end of the trial $i = n$. The accumulated reward $R(S_t, v_t)$ is formulated as follows.

$$R(S_t, v_t) = r(S_t, v_t) + \sum_{i=1}^{n-t} \gamma^i \cdot r(S_{t+i}, v_{t+i}), \quad (4)$$

where S_t represents the state at step t , v_t denotes the action executed at step v , and n is the trial's length. In Eq. (4), the term $r(S_t, v_t)$ is the immediate reward at step t , while the summation $\sum_{i=1}^{n-t} \gamma^i \cdot r(S_{t+i}, v_{t+i})$ accounts for the weighted rewards of future steps, with the discount factor γ ranging between 0 and 1. A higher value of γ reflects a stronger emphasis on the rewards of later actions. The RL process is illustrated in Fig. 2.

Table 2. Formulations and RL settings for the explored graph optimization problems

Problem	Formulation	State S	Action v	Immediate Reward	Termination
Minimum Vertex Cover (MVC)	Given: a graph $G = (V, E)$ Objective: find a minimal $S \subseteq V$ such that every $(u, v) \in E$ has at least u or v in S	The subset of selected nodes	Add v into S	-1	All edges are adjacent to S
Maximal Clique (MC)	Given: a graph $G = (V, E)$ Objective: find a maximal subset $S \subseteq V$, such that the subgraph induced by S is a clique	The subset of selected nodes	Add v into S	+1 if S is a clique, -1 otherwise	S is not a clique
Dense Sub-Graph (DSG)	Given: a graph $G = (V, E)$, an integer k Objective: find a k -node $S \subseteq V$, such that the subgraph induced by S has the most edges.	The subset of selected nodes	Add v into S	Number of newly-added edges among S	S involve k nodes

The training of our model hinges on minimizing the *mean square error* (MSE) between the true accumulated rewards $R(S_{t+i}, v_{t+i})$ in Eq. (4) and the predicted scores by the model $Q(S_{t+i}, v_{t+i})$. We utilize the Adam optimizer for training. The specific reward configurations and formulations for the graph optimization problems we address are detailed in Table 2. In the cases of the Dense Subgraph

(DSG) and Minimum Vertex Cover (MVC) problems, nodes are incrementally incorporated into the intermediate solution S until S qualifies as a feasible solution of the graph optimization problem. For DSG, feasibility is attained when the solution includes k nodes, whereas for MVC, it is when all edges in the graph are covered. Regarding the Maximum Clique (MC) problem, nodes are added to S until adding any further node makes S infeasible, i.e., S no longer constitutes a clique. Consequently, the training dataset comprises $n - 1$ node additions with a reward of +1, indicating the maintenance of a clique, and one final node addition with a reward of -1, reflecting the formation of a non-clique.

4 Experiments

Graph Optimization Problems. To evaluate the efficacy of our SET framework, we conduct experiments targeting two graph optimization problems: the *Minimum Vertex Cover* (MVC) problem and the *Maximal Clique* (MC) problem. In the evaluation, we use the *Dense Sub-Graph* (DSG) problem as the source graph optimization problem. Their problem formulations are detailed in Table 2.

Compared Baselines. To benchmark the SET framework’s performance, we compare it against a suite of established algorithmic and learning-based approaches: 1) The optimal (OPT) approach: The optimal solution is obtained by solving a Mixed Integer Linear Programming with CPLEX. 2) The approximation (APPR) algorithms: For MVC, APPR follows the 2-approximation algorithm in guarantee [18]. For MC, APPR follows the $O(|V|/(\log |V|)^2)$ -approximation algorithm in [4]. 3) DQN [6]: DQN is an RL method with a message-passing mechanism for state representation. 4) LIGD [35]: LIGD is an RL method that enhances training by diversifying the graph instances encountered during training. 5) LwD [1]: LwD reduces the search space by assessing the likelihood of each node belonging to the optimal solution, applicable to only locally decomposable problems. Note that LwD is only applicable to MC, as DSG and MVC do not meet the locality criteria. 6) ERD [19]: ERD is an unsupervised learning method inspired by Erdos’ probabilistic method. 7) HAM [27]: HAM is an unsupervised learning method that exploits the Hamiltonian cost function. 8) REL [32]: REL is an unsupervised learning approach built upon relaxing the objective and constraint functions. 9) NT: NT shares the same structure as SET but trains the target model from scratch.

Metrics. The solution quality is measured by *objective value*. For DSG, the objective value is the number of edges (u, v) with u and v both in S . For MVC and MC, the objective value is the number of nodes within S , i.e., $|S|$. Since our experiments involve both maximization and minimization problems, we report the *approximation ratio* rather than the objective value. For both problem types, an approximation ratio is a value greater than 1, with a lower ratio indicative of superior solution quality. The approximation ratio is 1 if the optimal solution is obtained. To assess training efficiency, we report the *number of training iterations*

required to obtain a converged model. In each iteration, solutions for 128 training graphs are generated and utilized in the training process.

Default Settings. Throughout our experiments, we use the following early-stop condition: training is halted if there is no improvement in the objective value or validation loss for w validation iterations. Unless otherwise stated, the default settings are as follows: the dimensionality of input embeddings is set to 64; the number of GCN layers in the feature extractor is 5; the number of fully connected layers in the predictor is 4; the batch size is 16, and the early-stop window size w is 50.⁵ We also apply L2 regularization with the weight λ as 10^{-5} . The reported results are the mean of 3 trials. All experiments are conducted on an HP DL580 G9 server equipped with dual six-core Intel i7-8700K CPUs, a pair of GeForce RTX 2080 Graphics Cards, and 64GB of RAM.

4.1 Effectiveness and Efficiency

Testing on Real Networks. We compare the performance of SET and other baselines on several real networks: 1) Facebook, which comprises 4039 nodes and 88234 edges [23]; 2) GitHub, which includes 37700 nodes and 289003 edges, though we down-sample it to 5000 nodes [25]; and 3) LastFM, which consists of 7624 nodes and 27806 edges [26]. For learning-based methods, we utilize training data collected from 100-node graphs generated by the Barabási-Albert (BA) model [2]. The results are presented in Table 3. Due to the computational limitations on larger graphs, the optimal (OPT) solution is not always obtainable. Hence, we calculate the approximation ratio using the best-known objective values. LIGD and REL do not return results within a day and thus are marked with '*' in the table. LwD and HAM, being specific to the MC problem, have their fields for MVC indicated as '-'. As Table 3 demonstrates, SET and NT reach similar performance and consistently outperform all other baselines across these networks, showcasing SET's ability to capture the structural and state information that could be generalized to large-scale graphs. Also, SET requires fewer training iterations than NT, as shown in the next experiment.

Table 3. Testing on large and real networks ('*': out-of-time, '-': not applicable)

		SET	NT	DQN	LIGD	ERD	REL	HAM	APPR	LwD
Facebook	MVC	1.0000	1.0011	1.2135	*	1.2118	*	-	1.1356	-
	MC	1.0009	1.0000	1.2649	*	1.2647	*	∞	1.6152	1.6583
GitHub	MVC	1.0000	1.0032	1.2226	*	1.2235	*	-	1.4165	-
	MC	1.0000	1.0041	1.5192	*	1.5166	*	∞	1.7278	1.7268
LastFM	MVC	1.0021	1.0000	1.2552	*	1.2589	*	-	1.4531	-
	MC	1.0015	1.0000	1.8633	*	1.8654	*	∞	1.7316	1.7265

⁵ We perform sensitivity tests on those parameters, and select the best-performing settings as the defaults. The sensitivity tests are eliminated for the sake of space.

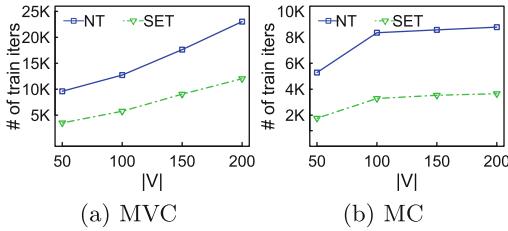


Fig. 3. Number of training iterations needed for models trained with various sizes of graphs

Testing on Synthetic Graphs. In addition, we evaluate the performance of SET and NT on graphs of varying sizes generated using the BA model, where both training and testing graphs are of the same size. For each graph size, the testing dataset has 128 graphs generated by BA. While the objective values of SET and NT are closely aligned, mirroring the results observed with real networks, we opt not to detail these values here for the sake of space. Figure 3 shows the number of training iterations required for both SET and NT when the target graph optimization problem is MVC and MC, respectively. Notably, SET consistently requires fewer training iterations than NT, underscoring its efficiency in reducing both training time and iterations.

4.2 Transferability of Features

We utilize Centered Kernel Alignment (CKA) [22] to analyze models that have been independently trained for various graph optimization problems. CKA serves as an effective tool for measuring the similarity between sets of high-dimensional data representations, where higher CKA values signify greater similarity or a higher degree of shared features. Specifically, we use CKA to evaluate node representations learned from scratch by NT on 32 sampled graphs, each comprising 20 nodes, for each examined graph optimization problem. The corresponding CKA values for pairs of these problems are detailed in Table 4. Furthermore, we compare the node embeddings from each model against a *random* baseline RAN, specifically, a set of node embeddings filled with randomly generated values. This comparison is to establish a benchmark against two unrelated sets of node embeddings. As shown in Table 4, the CKA scores among the graph optimization problems (DSG, MVC, and MC) are markedly higher than those between the problems and RAN, suggesting that the models for DSG, MVC, and MC share common structural and state information. Particularly, the highest CKA score between DSG and MC can be attributed to their similar objectives, both focusing on the retrieval of k -node dense subsets.

5 Conclusion

The utilization of RL in graph optimization problems has seen rising interest, but the requirement for extensive training remains a challenge. In this work, we

Table 4. Measuring similarities of node embeddings of models trained for various problems

	DSG	MVC	MC	RAN
DSG	—	0.31	0.42	0.0001
MVC	—	—	0.29	0.0003
MC	—	—	—	0.0004

harness the power of transfer learning to tackle this issue. The proposed SET framework is adept at rapidly adapting to different graph optimization problems by accounting for the distributional differences of their objective values. Through extensive experimentation, SET has proven to outclass various algorithmic and learning-based competitors. Moreover, our analysis of transferring knowledge across different tasks provides valuable insights into the adaptability of SET. This evidence suggests that models trained under the SET framework are not only effective for specific problems but can also be efficiently repurposed for new, related problems.

References

1. Ahn, S., Seo, Y., Shin, J.: Learning what to defer for maximum independent sets. In: ICML (2020)
2. Albert, R., Barabási, A.L.: Statistical mechanics of complex networks. *Rev. Mod. Phys.* (2002)
3. Bello, I., Pham, H., Le, Q.V., Norouzi, M., Bengio, S.: Neural combinatorial optimization with reinforcement learning. arXiv preprint [arXiv:1611.09940](https://arxiv.org/abs/1611.09940) (2016)
4. Boppana, R., Halldórsson, M.M.: Approximating maximum independent sets by excluding subgraphs. *BIT Numerical Mathematics* (1992)
5. Chen, M., Wei, Z., Ding, B., Li, Y., Yuan, Y., Du, X., Wen, J.R.: Scalable graph neural networks via bidirectional propagation. In: NeurIPS (2020)
6. Dai, H., Khalil, E., Zhang, Y., Dilkina, B., Song, L.: Learning combinatorial optimization algorithms over graphs. In: NeurIPS (2017)
7. Dai, Q., Wu, X.M., Xiao, J., Shen, X., Wang, D.: Graph transfer learning via adversarial domain adaptation with graph convolution. *IEEE Trans. Knowl. Data Eng.* (2022)
8. Deudon, M., Courtnut, P., Lacoste, A., Adulyasak, Y., Rousseau, L.M.: Learning heuristics for the TSP by policy gradient. In: CPAIOR (2018)
9. Dorigo, M., Gambardella, L.M.: Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Trans. Evolutionary Comput.* (1997)
10. Evci, U., Dumoulin, V., Larochelle, H., Mozer, M.C.: Head2toe: utilizing intermediate representations for better transfer learning. In: ICML (2022)
11. Ganin, Y., et al.: Domain-adversarial training of neural networks. *J. Mach. Learn. Res.* (2016)
12. Gopalakrishnan, K., Khaitan, S., Choudhary, A., Agrawal, A.: Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection. *Construct. Building Mater.* (2017)
13. Groshev, E., Tamar, A., Goldstein, M., Srivastava, S., Abbeel, P.: Learning generalized reactive policies using deep neural networks. In: ICAPS (2018)
14. Guze, S.: Graph theory approach to the vulnerability of transportation networks. *Algorithms* (2019)
15. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: NeurIPS (2017)
16. Hua, J., Zeng, L., Li, G., Ju, Z.: Learning for a robot: deep reinforcement learning, imitation learning, transfer learning. *Sensors* (2021)
17. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: a survey. *J. Artifi. Intell. Res.* (1996)

18. Karakostas, G.: A better approximation ratio for the vertex cover problem. In: ICALP (2005)
19. Karalias, N., Loukas, A.: Erdos goes neural: an unsupervised learning framework for combinatorial optimization on graphs. In: NeurIPS (2020)
20. Kim, H., Cosa-Linan, A., Santhanam, N., Jannesari, M., Maros, M.E., Ganslandt, T.: Transfer learning for medical image classification: a literature review. BMC Med. Imaging (2022)
21. Kolouri, S., Nadjahi, K., Simsekli, U., Badeau, R., Rohde, G.: Generalized sliced wasserstein distances. In: NeurIPS (2019)
22. Kornblith, S., Norouzi, M., Lee, H., Hinton, G.: Similarity of neural network representations revisited. In: ICML (2019)
23. Leskovec, J., Mcauley, J.: Learning to discover social circles in ego networks. In: NeurIPS (2012)
24. Nazari, M., Oroojlooy, A., Snyder, L., Takáć, M.: Reinforcement learning for solving the vehicle routing problem. In: NeurIPS (2018)
25. Rozemberczki, B., Allen, C., Sarkar, R.: Multi-scale attributed node embedding. J. Complex Netw. (2021)
26. Rozemberczki, B., Sarkar, R.: Characteristic functions on graphs: Birds of a feather, from statistical descriptors to parametric models. In: CIKM (2020)
27. Schuetz, M.J., Brubaker, J.K., Katzgraber, H.: Combinatorial optimization with physics-inspired graph neural networks. Nat. Mach. Intell. (2022)
28. Selsam, D., Lamm, M., Bünz, B., Liang, P., de Moura, L., Dill, D.L.: Learning a sat solver from single-bit supervision. arXiv preprint [arXiv:1802.03685](https://arxiv.org/abs/1802.03685) (2018)
29. Shen, Z., Liu, Z., Qin, J., Savvides, M., Cheng, K.T.: Partial is better than all: revisiting fine-tuning strategy for few-shot learning. In: AAAI (2021)
30. Staastny, J., Skorpil, V., Balogh, Z., Klein, R.: Job shop scheduling problem optimization by means of graph-based algorithm. Appli. Sci. (2021)
31. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. In: ICLR (2018)
32. Wang, H.P., Wu, N., Yang, H., Hao, C., Li, P.: Unsupervised learning for combinatorial optimization with principled objective relaxation. In: NeurIPS (2022)
33. Wang, H., Yue, T., Ye, X., He, Z., Li, B., Li, Y.: Revisit finetuning strategy for few-shot learning to transfer the emdeddings. In: ICLR (2023)
34. Welling, M., Kipf, T.N.: Semi-supervised classification with graph convolutional networks. In: ICLR (2017)
35. Yang, C.H., Shen, C.Y.: Enhancing machine learning approaches for graph optimization problems with diversifying graph augmentation. In: SIGKDD (2022)
36. Zhang, W., Dietterich, T.G.: Solving combinatorial optimization tasks by reinforcement learning: a general methodology applied to resource-constrained scheduling. J. Artifi. Intell. Res. (2000)
37. Zhu, Z., Lin, K., Jain, A.K., Zhou, J.: Transfer learning in deep reinforcement learning: A survey. IEEE Trans. Pattern Anal. Mach. Intell. (2023)



SD-Attack: Targeted Spectral Attacks on Graphs

Xianren Zhang¹, Jing Ma², Yushun Dong³, Chen Chen³, Min Gao⁴,
and Jundong Li^{3(✉)}

¹ The Pennsylvania State University, State College, PA, USA
xzz5508@psu.edu

² Case Western Reserve University, Cleveland, OH, USA
jing.ma5@case.edu

³ University of Virginia, Charlottesville, VA, USA
{yd6eb,zrh6du,jundong}@virginia.edu

⁴ Chongqing University, Chongqing, China
gaomin@cqu.edu.cn

Abstract. Graph learning (GL) models have been applied in various predictive tasks on graph data. But, similarly to other machine learning models, GL models are also vulnerable to adversarial attacks. As a powerful attack method on graphs, spectral attack jeopardizes the eigenvalues or eigenvectors of the graph topology-related matrices (e.g., graph adjacency matrix and graph Laplacian matrix) due to their inherent connections to certain structural properties of the underlying graph. However, most existing spectral attack methods focus on damaging the global graph structural properties and can hardly perform effective attacks on a target node. In this paper, we propose a novel targeted spectral attack method that can perform model-agnostic attacks effectively on the local structural properties of a target node. First, we define a novel node-specific metric—spectral density distance, which measures the difference of the local structural properties for the same target node between two different graph topologies. Then, we conduct attacks by maximizing the spectral density distance between the graphs before and after perturbation. Additionally, we also develop an effective strategy to improve attack efficiency by using the eigenvalue perturbation theory. Experimental results on three widely used datasets demonstrate the effectiveness of our proposed approach.

Keywords: Graph learning · Adversarial attack · Target attack

1 Introduction

Graphs have been widely used in many high-impact fields such as social media and infrastructure systems [9]. Graphs contain rich structural information and many powerful graph learning (GL) models [6, 17] have been designed to exploit the graph structure for various predictive tasks (e.g., node classification and link prediction). However, recent works [18] have shown that GL models are

vulnerable to adversarial attacks. Specifically, malicious attackers can mislead GL models to make incorrect predictions by making carefully crafted perturbations on the input graph data. Among the existing attack methods on graph data, spectral attack [1, 2, 7] has recently attracted much attention because they are effective and need less information. Compared with conventional attacks, the spectral attack focuses on altering the spectral properties of the graph and does not need any information from victim models. It is usually implemented by changing the spectrum (eigenvalues) of graph topology-related matrices (e.g., Laplacian matrix) [12]. The graph spectrum implies a plethora of important graph structural properties (e.g., the number of connected components [12], algebraic connectivity [4], and degree distribution). Spectral attacks can effectively damage these structural properties by altering the spectrum of a graph. Moreover, the spectral attack is especially powerful in attacking GL models with spectral filters [11], such as Graph Convolution Networks (GCNs) [6].

Despite the effectiveness of spectral attacks, most studies [1, 2, 7] focus on jeopardizing the global structural properties. They design an attack loss function based on the eigenvalues to reflect the changes in structural properties and then flip edges to maximize the loss. However, the attack loss is designed for the whole graph and the attacker can only select from the same set of edges to flip even for different target nodes instead of making adjustments for different nodes. As a consequence, they lack explorations on attacking the local structural properties of a specific node, which limits their effectiveness. On the other hand, many studies have pointed out that eigenvectors can provide information on the local structural properties of nodes, e.g., node membership in clustering [4] and return probabilities of random walks starting from a specific node [3]. For example, the eigenvector of the second smallest eigenvalue, called the Fiedler vector [4], can be used to divide the nodes into two groups. Nodes with the same sign in the Fiedler vector are similar and can be split into the same cluster. Recent studies [3] have shown that the eigenvalues and eigenvectors of the normalized adjacency matrix can be used to calculate the return probabilities of random walks with different steps starting from a node. Various steps of return probabilities reflect interactions between a node and its neighbors with different resolutions. These pieces of evidence demonstrate that eigenvalues indicate the global structural properties of a graph, and eigenvectors can supplement by providing important local (node-level) information. Thus, spectral attacks can be further improved to disrupt the local structural properties of a specific node by perturbing both eigenvalues and eigenvectors.

In this paper, we study the novel problem of targeted spectral attacks on GL models. However, there remain two challenges: 1) The first challenge is to design a proper node-specific attack loss function to measure the attack damage w.r.t. a target node after graph perturbation. This loss function can provide guidance for altering the graph structure (e.g., selecting and flipping edges) to achieve targeted attacks. Nevertheless, it is difficult to bridge the gap between the eigenvalues and eigenvectors of the graph topology-related matrices (e.g., adjacency matrix and Laplacian matrix) and local structural properties (to attack).

2) The second challenge is to achieve such spectral attacks with high efficiency. With a node-specific attack loss function, a straightforward attack strategy is to calculate the attack loss after flipping each edge and then select the edges that lead to high losses. In this process, it is necessary to perform eigendecomposition for each edge flip, which is very time-consuming, as the time complexity is $O(n^3)$ in the worst case, where n is the number of nodes.

To address these challenges, we propose a novel targeted spectral attack method called Spectral Density Attack (**SD-Attack**). To design a node-specific attack loss function (the first challenge), we take advantage of a node-specific metric named Point-wise Density of States (PDOS) [3]. Specifically, the rationale of PDOS is to measure the local structural properties of a target node. For example, the return probabilities of random walks with any steps starting from the target node can be measured by PDOS [3]. We propose a novel node-specific metric named spectral density distance, which measures the Wasserstein distance of a target node's PDOS between the original graph and the perturbed one. We can achieve model-agnostic targeted attacks by flipping edges to maximize the spectral density distance. To avoid frequent computations of eigenvalues and eigenvectors (the second challenge), we leverage eigenvalue perturbation theory [15] to approximate the change of eigenvalues and eigenvectors. With such an approximation strategy, we only need to perform eigendecomposition *once* for the original graph, and the eigenvalues and eigenvectors of the perturbed graph can then be approximated for each edge flip. In experiments, our proposed method is evaluated on widely-used benchmarks (Cora [8], Citeseer [5], and Pubmed [14]) and also tested on two GL models (GCN [6] and GAT [17]). Empirical results show that our method can achieve effective and efficient attacks on target nodes. The main contributions of our work are summarized as follows.

- We study a novel problem: spectral attacks on the local structural properties of a target node. To the best of our knowledge, this is the first study on this important problem.
- We propose a novel node-specific metric spectral density distance to measure the attack effectiveness for any target node. Based on such a metric, we propose a SD-Attack to address the problem of targeted spectral attack.
- We evaluate the proposed method on three well-known datasets with extensive experiments. Experimental results indicate that SD-Attack can generate more effective attacks on a target node than state-of-the-art baselines.

2 Preliminaries

Let $G(V, E)$ represent the structure of an undirected graph, where $V = \{v_1, \dots, v_n\}$ is the set of nodes. Let $A \in \{0, 1\}^{n \times n}$ be the adjacency matrix where $A_{ij} = A_{ji} = 1$ if there is an edge between node i and j . The normalized adjacency matrix is $\hat{A} = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$, where D is a diagonal matrix with $D_{ii} = d_i = \sum_{j=1}^n A_{ij}$.

2.1 Adversarial Attack on Graphs

Given a finite budget M , which is the maximum number of edges the attacker can flip, the targeted attack on node k can be formulated as the following problem:

$$\arg \max_{A'} \mathcal{D}_k(A, A'), \quad (1)$$

subject to $|\Delta A| \leq 2M$.

Here \mathcal{D}_k is defined as the attack loss measuring the attack damage on the node k . $\Delta A = A' - A$ contains the total flipped edges, where A and A' are the original and perturbed adjacency matrices, respectively. ΔA_{ij} is 1 if the attacker adds an edge between node i and node j , and -1 if the edge is deleted. The goal of the attack is to find the best-attacked graph A' , which disrupts the local structural properties of node k most.

2.2 Density of States

A real symmetric matrix H can be diagonalized as $H = U\Lambda U^T$, where $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$ is a diagonal matrix whose diagonal elements are eigenvalues, and $U = [q_1, \dots, q_n]$ is a square matrix whose i -th column is the eigenvector q_i . The *density of states* (DOS) [3] induced by H is defined as a generalized function:

$$\mu(\lambda) = \frac{1}{n} \sum_{i=1}^n \delta(\lambda - \lambda_i), \quad (2)$$

where δ is the Dirac delta function. $\mu(\lambda)$ is also called the spectral density, because it describes the distribution of eigenvalues that relate to the global structural properties of a graph. For a specific node k , the Point-wise Density of State (PDOS) is a weighted version of DOS:

$$\mu_k(\lambda) = \sum_{i=1}^n |e_k^T q_i|^2 \delta(\lambda - \lambda_i), \quad (3)$$

where e_k is the k -th standard basis vector with the k -th entry equal to 1 and 0 for other entries. q_i is the i -th eigenvector. $\mu_k(\lambda)$ describes the local information of the node k .

The PDOS is related to the local structural properties of a specific node. For example, the PDOS encodes return probabilities of any steps starting from the target node. We propose to attack the target node by perturbing its PDOS in our proposed framework.

3 The Proposed Method SD-Attack

In this section, we propose a novel spectral attack strategy to disrupt the local structural properties of a target node. We leverage PDOS to establish a connection between the local structural properties and the eigenvalues and eigenvectors of the normalized adjacency matrix. Then, we design a node-specific metric

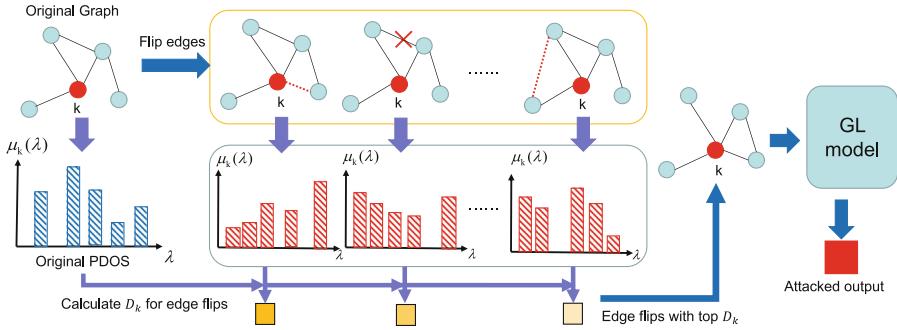


Fig. 1. The framework of Spectral Density Attack. SD-Attack aims to attack the local structural properties of the target node k by flipping edges to maximize the spectral density distance. The GL block can be any general GL models.

called spectral density distance to measure the damage of an attack on the target node. We flip edges that maximize the spectral density distance between the original graph and the perturbed version. To improve the efficiency, we employ eigenvalue perturbation theory [15] to efficiently compute the spectral density distance. The overall framework of our proposed method is shown in Fig. 1.

3.1 Spectral Density Distance

We first propose a novel metric—spectral density distance, to measure the distance between the original graph and the perturbed one w.r.t. their PDOS.

PDOS is a distribution of eigenvalues. One choice to evaluate the difference between two distributions is the Wasserstein distance (a.k.a. the Earth-mover distance). We use this metric to measure the distance between the original PDOS and the perturbed one. The spectral density distance between the original graph and the perturbed one on node k is defined as the W_1 distance (1-Wasserstein distance) of their PDOS:

$$\mathcal{D}_k(A, A') = W_1(\mu_k, \mu'_k) = \int_0^1 |F(q) - F'(q)| dq \quad (4)$$

where μ_k and μ'_k are PDOS of the original graph and perturbed graph, respectively. $F(q)$ and $F'(q)$ are the quantile functions of μ_k and μ'_k (e.g., $F(0.5)$ is the median value of the distribution μ_k).

3.2 Why Wasserstein Distance

We use an example to further explain why we select the Wasserstein distance rather than other popular metrics. We first give an example and then compare how different metrics will behave in that example.

Example. Here, we use two simple PDOS as an example for illustration. We let the original PDOS be $\mu_k(\lambda) = \delta(\lambda - x)$ and the perturbed PDOS be $\mu'_k(\lambda) = \delta(\lambda - y)$. This means $\mu_k(\lambda)$ has a value of 1 when $\lambda = x$ and 0 when λ takes other values. $\mu'_k(\lambda)$ has a value of 1 when $\lambda = y$ and 0 when λ takes other values. Obviously, in this example, $\mu_k(\lambda)$ and $\mu'_k(\lambda)$ are close to each other if x and y are close to each other. Then we demonstrate how different metrics will behave.

Kullback-Leibler divergence (KL-divergence). The KL-divergence $D_{KL}(\mu'_k(\lambda) || \mu_k(\lambda)) = +\infty$ if $x \neq y$ and $D_{KL}(\mu'_k(\lambda) || \mu_k(\lambda)) = 0$ if $x = y$.

Jensen-Shannon divergence (JS-divergence). In our example, the JS $D_{JS}(\mu'_k(\lambda), \mu_k(\lambda)) = \log 2$ if $x \neq y$ and $D_{JS}(\mu'_k(\lambda), \mu_k(\lambda)) = 0$ if $x = y$.

Total Variation distance (VT-distance). In our example, the TV-distance $TV(\mu'_k(\lambda), \mu_k(\lambda)) = 1$ if $x \neq y$ and $TV(\mu'_k(\lambda), \mu_k(\lambda)) = 0$ if $x = y$.

Wasserstein distance. The Wasserstein distance is $W_1(\mu'_k(\lambda), \mu_k(\lambda)) = |x - y|$.

By comparing different distances and divergences in our example, the Wasserstein distance is the best metric to measure the distance between two PDOS. W_1 will become smaller when μ_k and μ'_k are closer and bigger when they are further away. However, other distances or divergences do not follow this trend and result in an infinite value or remain constant.

3.3 SD-Attack

With the spectral density distance defined in Eq. (4), we take advantage of it in our loss function for the attack. A straightforward approach to achieve the attack is to flip the edges to maximize the loss. However, such an approach requires frequent eigendecomposition operations, which can be prohibitively expensive when the graph is large. To efficiently calculate the spectral density distance, we propose to approximate the change in eigenvalues and eigenvectors with the eigenvalue perturbation theory [15].

Eigenvalues and Eigenvectors Approximation. To improve the efficiency, we approximate the perturbed eigenvalues and eigenvectors. For an edge flip ΔA_{ij} , we estimate the change in eigenvalues and eigenvectors.

Theorem 1. *Considering a generalized eigenvalue problem $Au_k = \lambda_k Du_k$ and a flip of a single edge $\Delta A_{ij} = \Delta A_{ji}$, the change in the k -th generalized eigenvalue $\Delta \lambda_k$ can be approximated as follows:*

$$\Delta \lambda_k \approx \Delta A_{ij}(2u_{ki}u_{kj} - \lambda_k(u_{ki}^2 + u_{kj}^2)). \quad (5)$$

We can also approximate the change of eigenvectors. For a target node v_c , the PDOS takes every c -th value from all eigenvectors and the change of it can be calculated as follows:

Theorem 2. *Considering a generalized eigenvalue problem $Au_k = \lambda_k Du_k$ and a flip of a single edge $\Delta A_{ij} = \Delta A_{ji} = 1 - 2A_{ij}$, the change of the c -th value of eigenvector u_k is:*

$$\Delta u_{kc} \approx \Delta A_{ij}((\lambda_k u_{ki} - u_{kj})(A - \lambda_k D)_{ci}^+ + (\lambda_k u_{kj} - u_{ki})(A - \lambda_k D)_{cj}^+), \quad (6)$$

where $(\cdot)^+$ is the pseudo-inverse.

The Eq. (5) and Eq. (6) are proposed by [15]. The pseudo-inverse term in Eq. (6) is independent of any edge flip and can be precomputed. In conventional attack strategies that do eigendecomposition after each edge flip, the time complexity is $O(n^3)$ in the worst case. With our approximation method, the time complexity of calculating the spectral density distance after each edge flip can be reduced to $O(n)$.

The Overall Attack Algorithm. Given a target node k , candidate set C , and budget M , the problem is to flip M edges from the candidate set C so that the spectral density distance $D_k(A, A')$ in Eq. (4) is maximized. There are $\binom{|C|}{M}$ different numbers of different combinations of edges. Even though we can efficiently calculate the spectral density distance, it is still time-consuming to try every possible combination of edges and calculate the spectral density distance for each one. As a result, instead of exhausting every possible combination of edges, we calculate the spectral density distance for each single edge flip with Eq. (5) and Eq. (6), and then we can select the final perturbations based on the calculated spectral density distances. A straightforward way to conduct attacks is to use a greedy strategy to select these candidate edges. The greedy strategy in our setting is to calculate the spectral density distance for each edge flip, select the edge with the highest spectral density distance, and repeat this process until the budget is reached. However, the greedy strategy still leads to a high time complexity because we have to repeat the calculation of the spectral density distance for the remaining edges. As a result, we choose to directly select edges with the top- M spectral density distance on the original graph as the final perturbations. The rationale for such an approximation is that the perturbation budget M is often very small, therefore choosing the candidate edges on the original graph all at once and choosing the candidate edges greedily does not yield much difference.

Time Complexity. The overall time complexity of the algorithm is $O(n^3)$ in the worst case, where n is the number of nodes in the whole graph. The time complexity of the overall algorithm comes mainly from the eigendecomposition of the original normalized adjacency matrix. The time complexity for a single node attack is $O(n * |C|)$, where $|C|$ is the number of edges in the candidate set.

4 Experiments

In this section, we evaluate our proposed attack method SD-Attack with extensive experiments on three widely-used graph datasets [5, 8, 14]. Specifically, to validate the effectiveness of SD-Attack, we aim to answer the following two research questions:

- **RQ1:** How well can SD-Attack generate effective attacks for a target node compared to baselines?
- **RQ2:** How well does the spectral density distance help SD-Attack to attack a target node?

4.1 Setup

Datasets. We evaluate our proposed method on three well-known datasets: Cora [8], Citeseer [5], and Pubmed [14]. As it is time-consuming to perform targeted attacks on every unlabeled node, we randomly select 20% nodes as test nodes in Cora and Citeseer and 1.5% nodes as test nodes in Pubmed. We follow the data preprocessing setting in [2], where nodes' features are all normalized, and only the largest connected component is considered.

Baselines. Across all three datasets, we evaluate the proposed attack model SD-Attack against four baselines:

- **Degree** [16]: This method flips edges (inserting or removing) based on degree centrality (the sum of degrees at two ends of the edge). Edges with higher degree centralities are considered to be more important and flipped.
- **A_{class}** [1]: A_{class} focuses on attacking network representation learning models. A_{class} formulates the representation learning process as a matrix factorization process and flips edges to damage that process.
- **GF-attack** [2]: GF-Attack (Graph Filter Attack) attacks the graph filters. They formulate GNN models as an approximation process. They then attack filters by maximizing the loss function for the approximation.
- **SPAC** [7]: SPAC focuses on attacking the eigenvalues of the normalized graph Laplacian. It perturbs the filters of the GCN models by maximizing the spectral distance which is defined as the L2 distance of the eigenvalues.

Variants of SD-Attack. We conduct ablation study with following variants:

- **SD-DOS:** To demonstrate the importance of local information of the target node, SD-DOS replaces the PDOS (Eq. 3) with DOS (Eq. 2) which indicates the global structural properties of a graph [3].
- **SD-Eigenval:** SD-Eigenval only considers the change of eigenvalues.
- **SD-Eigenvec:** SD-Eigenvec only considers the change of eigenvectors.
- **SD-KL:** SD-KL replaces the Wasserstein distance with KL divergence.

Configurations. We use two widely-used graph learning models as victim models: GCN [6] and GAT [17]. In our experiments, we conduct experiments on different layers of GCN models (2,3, and 4 layers), and the output dimension of each layer is 32 (except the output layer). GAT has two layers, and the output dimension of the first layer is set to 32. In our setting, the attack is an evasion attack which means that attacks occur after the victim model is trained.

For a target node k , the candidate set is defined as: $C = \{(v, u) | v \in N_k \cup \{k\}, u \in V\}$, where N_k is the set of neighbors of node k . For each correctly predicted node in the test set, we take it as the target node and carry out the attack. We calculate the decrease in accuracy as the attack performance.

Attacker Knowledge and Capacity. In many real-world scenarios, it is hard for attackers to obtain all information (e.g., victim models and training data labels), and we assume that such information is not accessible. Attackers are only allowed to access the graph topology. For capacity, attackers can add or remove edges within a limited budget. Similarly to [18], since high-degree nodes are more difficult to attack, the budget is set as the degree of the target node.

Table 1. The change in classification accuracy after different attack methods (lower is better). Results are averaged for ten trials.

Model	Attack	Cora	Citeseer	Pubmed
GCN	Degree	-1.65 ± 0.12	-1.93 ± 0.29	-1.05 ± 0.03
	A _{class}	-4.59 ± 0.18	-4.95 ± 0.34	-9.12 ± 0.21
	GF-Attack	-4.63 ± 0.29	-3.42 ± 0.31	-5.57 ± 0.14
	SPAC	-5.13 ± 0.27	-4.62 ± 0.34	-4.73 ± 0.12
	SD-Attack	-10.56 ± 0.33	-7.76 ± 0.38	-10.34 ± 0.14
GAT	Degree	-5.73 ± 0.83	-3.44 ± 0.31	-2.09 ± 0.19
	A _{class}	-5.29 ± 0.19	-4.55 ± 0.24	-11.32 ± 0.42
	GF-Attack	-6.84 ± 0.76	-4.67 ± 0.57	-4.76 ± 0.23
	SPAC	-5.19 ± 1.02	-3.11 ± 0.40	-3.07 ± 0.21
	SD-Attack	-9.24 ± 0.71	-6.20 ± 0.22	-13.72 ± 0.41

4.2 Attack Performance

To answer **RQ1**, we evaluate our proposed attack strategy with other baselines. We first train the GCN [6] and GAT [17] models on the original graph. Then the models are fixed, and the attackers perform attacks for each target node. We repeat experiments for ten trials and get the average results. The averaged changes of accuracy and standard deviation are shown in Table 1.

In general, our attack strategy achieves the best performance on three datasets. Among the baselines, the method based on degree centrality [16] can hardly perform effective attacks. Its performance is worse than the baselines of the spectral attacks [1, 2, 7]. This indicates that flipping edges with a high degree of centrality would not be enough to mislead GL models. Spectral attacks have better results than the method based on degree centrality, which demonstrates the effectiveness of attacking graph spectra. However, they ignore the local structural properties of a target node, which limits the attack performance when the objective is to attack a target node. SD-Attack damages the local structural properties of the target node by maximizing the spectral density distance. Compared with other baselines, SD-Attack can generate more effective attacks on three well-known datasets.

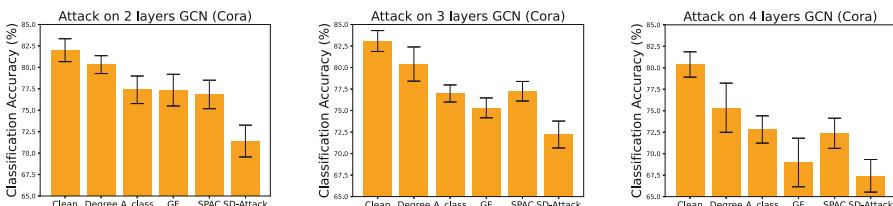


Fig. 2. Comparison of different attacks on different layers of GCN on Cora dataset.

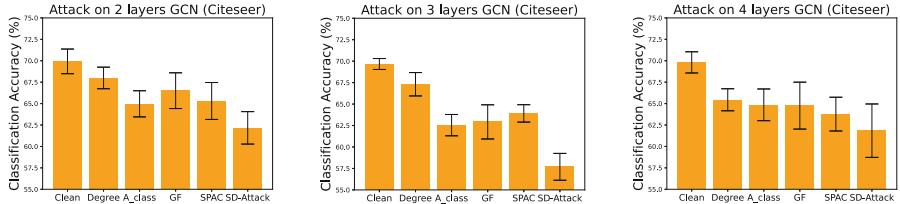


Fig. 3. Comparison of different attack strategies on different layers of GCN on Citeseer dataset.

To further evaluate the effectiveness of our approach, we compare its attack performance with baselines in victim GCN models with different numbers of layers. The experiment is repeated ten times and the averaged results with standard deviations are shown in Fig. 2 and Fig. 3. Compared to other baselines, the degree method [16] has little influence on the accuracy of the victim model. Other baselines [1,2,7] perform better and are more effective in attacking the victim model. In different cases, our method always outperforms other baselines, further indicating the effectiveness of SD-Attack.

Table 2. The change in classification accuracy under different variants (lower is better). Results are averaged for ten trials.

Model	Attack	Cora	Citeseer	Pubmed
GCN	SD-DOS	-6.52 ± 0.36	-4.03 ± 0.34	-8.14 ± 0.17
	SD-Eigenval	-4.95 ± 0.23	-4.55 ± 0.21	-7.43 ± 0.17
	SD-Eigenvec	-9.05 ± 0.30	-7.43 ± 0.34	-6.86 ± 0.10
	SD-KL	-8.59 ± 0.41	-5.90 ± 0.65	-9.86 ± 0.08
	SD-Attack	-10.56 ± 0.33	-7.76 ± 0.38	-10.34 ± 0.14
GAT	SD-DOS	-5.61 ± 0.50	-3.21 ± 0.11	-8.55 ± 0.20
	SD-Eigenval	-5.92 ± 0.43	-4.43 ± 0.27	-7.97 ± 0.25
	SD-Eigenvec	-8.29 ± 0.61	-5.92 ± 0.27	-6.08 ± 0.35
	SD-KL	-9.50 ± 0.52	-6.14 ± 0.28	-15.24 ± 0.50
	SD-Attack	-9.24 ± 0.71	-6.20 ± 0.22	-13.72 ± 0.41

4.3 Variants of SD-Attack

To answer **RQ2**, we compare different variants of our method and report the results in Table 2. By comparing SD-Attack with SD-DOS, we observe that the performance of attacks improves. The reason is that PDOS can provide more local and node-specific information about the structural properties of the target node [3]. SD-Attack has better results than SD-Eigenval and SD-Eigenvec, which

demonstrates that both eigenvalues and eigenvectors can provide important information in exploiting the vulnerability of the target node. SD-KL replaces the Wasserstein distance with the KL divergence to validate the effectiveness of the spectral density distance in our attack method. Overall, SD-Attack generates better results than SD-KL, indicating that the spectral density distance can better measure the damage caused by attacks on the local structural properties.

5 Related Works

Spectral Adversarial Attack. Recently, several works focus on spectral attacks because most of them only need the network structure without access to the information about the victim models. Network representation learning can be viewed as solving a matrix factorization problem, and the learning process can be viewed as performing singular value decomposition (SVD) on certain graph topology-related matrices [13]. The loss function can be viewed as the square of the sum of small singular values. To damage the quality of representation, A_{class} [1] flips the edges and maximizes the loss function of the victim models. Inspired by the observation that output embeddings of GNN models have low-rank properties [10], GF-Attack [2] formulates GNN models as an approximation problem. GF-Attack then flips edges to maximize the loss function for the approximation problem. SPAC [7] focuses on attacking the graph filter of GCN models. SPAC proposes to maximize the changes on the eigenvalues of the graph Laplacian, so that the spectral filters and graph convolution can be influenced. SPAC defines the spectral distance as the L_2 distance of eigenvalues of the Laplacian matrix and then flips the edges to maximize it. These methods ignore the local structural properties of a specific node. In this paper, we introduce a novel spectral attack strategy that attacks the local structural properties of a target node.

6 Conclusion

In this paper, we propose a novel method SD-Attack to attack the local structural properties of a target node. More specifically, we first leverage the Pointwise Density of State (PDOS) for the target node to connect the local structural properties of that node with the eigenvalues and eigenvectors of the normalized adjacency matrix. Based on PDOS, we define a node-specific metric called spectral density distance to measure the difference between two graphs on the target node. We then attack the local structural properties of the target node by flipping edges to maximize the spectral density distance between the original graph and the graph after perturbation. Furthermore, we improve the attack efficiency by approximating the perturbed eigenvalues and eigenvectors with the eigenvalue perturbation theory. We also conduct experiments with baselines and variants on widely used datasets. By comparing the results of our model and the baselines, we validate the effectiveness of our method in targeted attacks on different victim models.

References

1. Bojchevski, A., Günnemann, S.: Adversarial attacks on node embeddings via graph poisoning. In: International Conference on Machine Learning, pp. 695–704. PMLR (2019)
2. Chang, H., et al.: A restricted black-box adversarial framework towards attacking graph embedding models. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 3389–3396 (2020)
3. Dong, K., Benson, A.R., Bindel, D.: Network density of states. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1152–1161 (2019)
4. Fiedler, M.: Algebraic connectivity of graphs. Czechoslov. Math. J. **23**(2), 298–305 (1973)
5. Giles, C.L., Bollacker, K.D., Lawrence, S.: Citeseer: an automatic citation indexing system. In: Proceedings of the third ACM Conference on Digital Libraries, pp. 89–98 (1998)
6. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint [arXiv:1609.02907](https://arxiv.org/abs/1609.02907) (2016)
7. Lin, L., Blaser, E., Wang, H.: Graph structural attack by perturbing spectral distance. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 989–998 (2022)
8. McCallum, A.K., Nigam, K., Rennie, J., Seymore, K.: Automating the construction of internet portals with machine learning. Inf. Retrieval **3**(2), 127–163 (2000)
9. Milanović, J.V., Zhu, W.: Modeling of interconnected critical infrastructure systems using complex network theory. IEEE Trans. Smart Grid **9**(5), 4637–4648 (2017)
10. Nar, K., Ocal, O., Sastry, S.S., Ramchandran, K.: Cross-entropy loss and low-rank features have responsibility for adversarial examples. arXiv preprint [arXiv:1901.08360](https://arxiv.org/abs/1901.08360) (2019)
11. Nt, H., Maehara, T.: Revisiting graph neural networks: all we have is low-pass filters. arXiv preprint [arXiv:1905.09550](https://arxiv.org/abs/1905.09550) (2019)
12. Oellermann, O.R., Schwenk, A.J.: The Laplacian spectrum of graphs. Graph Theory, c, Appl. **2**, 871–898 (1991)
13. Qiu, J., Dong, Y., Ma, H., Li, J., Wang, K., Tang, J.: Network embedding as matrix factorization: unifying DeepWalk, LINE, PTE, and node2vec. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, pp. 459–467 (2018)
14. Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., Eliassi-Rad, T.: Collective classification in network data. AI Mag. **29**(3), 93–93 (2008)
15. Stewart, G., Sun, J.: Matrix Perturbation Theory. Elsevier Science, Computer Science and Scientific Computing (1990)
16. Tong, H.E.A.: Gelling, and melting, large graphs by edge manipulation. In: Proceedings of 21st ACM International Conference on Information and Knowledge Management, pp. 245–254 (2012)
17. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint [arXiv:1710.10903](https://arxiv.org/abs/1710.10903) (2017)
18. Zügner, D., Akbarnejad, A., Günnemann, S.: Adversarial attacks on neural networks for graph data. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2847–2856 (2018)



Improving Structural and Semantic Global Knowledge in Graph Contrastive Learning with Distillation

Mi Wen¹, Hongwei Wang^{1(✉)}, Yunsheng Xue¹, Yi Wu², and Hong Wen³

¹ Shanghai University of Electric Power, Shanghai 201306, China
wanghongwei@mail.shiep.edu.cn

² State Grid Shanghai Electric Power Co., Ltd., Shanghai 200122, China

³ University of Electronic Science and Technology of China, Chengdu 611731, China

Abstract. Graph contrastive learning has emerged as a pivotal task in the realm of graph representation learning, with the primary objective of maximizing mutual information between graph-augmented pairs exhibiting similar semantics. However, existing unsupervised graph contrastive learning approaches face a notable limitation in capturing both structural and semantic global information. This issue poses a substantial challenge, as nodes in close geographical proximity do not consistently possess similar features. To tackle this issue, this study introduces a simple framework for Distillation Node and Prototype Graph Contrastive Learning (DNGCL). The framework enables contrastive learning by harnessing similar knowledge distillation to obtain more valuable structural and semantic global indications. Experimental results demonstrate that DNGCL outperforms existing unsupervised learning methods across a range of diverse graph datasets.

Keywords: Contrastive learning · Self-supervised learning · Graph representation learning · Graph neural network

1 Introduction

Graph contrastive learning (GCL) [17] stands as a potent pretext task within the realm of self-supervised learning (SSL) [8], designed to enhance representations by maximizing the agreement between two augmented views of a given graph through contrastive loss in latent space. Despite the current success of unsupervised graph contrastive learning, many existing GCL methods in this domain rely on partial domain information, largely due to the limited depth of traditional Graph Neural Network (GNN) [13] layers. However, in practical scenarios, as depicted in Fig. 1, it is evident that “nodes with similar characteristics are not always geographically close”, emphasizing the need for algorithms to possess a global perspective. Achieving this goal poses a formidable challenge for existing GCL methods based on shallow GNNs. These models face inherent

limitations in capturing both structural global knowledge and semantic global knowledge. Specifically, at the structural level, due to the complexity of the graph data, capturing structural global knowledge requires long interactions between nodes. While deepening GNN layers may seem like a straightforward solution, it introduces a potential bottleneck in information propagation [18], potentially hindering the model’s ability to effectively capture the global graph structure. At the seminal level, existing approaches focus on modelling instance-level structural similarities, but fail to discover the underlying global structure of the entire data distribution.

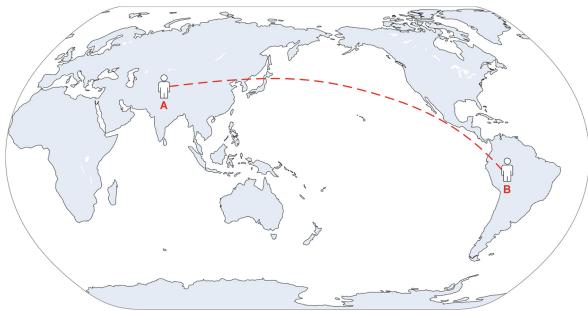


Fig. 1. Nodes with similar characteristics are not always geographically close: in a social network with similar interests, node A and node B may be from different cities or even different countries, but they still share common interests and communicate with each other.

In this paper, we propose a simple and effective graph contrastive learning method for Distillation Node and Prototype Graph Contrastive Learning (DNPGL) to address the above limitations. The framework proposes two new contrastive learning algorithms. In order to obtain interactions between distant nodes, distillation node contrastive learning, which can be interpreted as a kind of unlabelled knowledge distillation. It passes input graphs with two different data augments to the student network and the teacher network, and improves the differentiation of nodes by directly predicting the output of the teacher network using the standard cross-entropy loss function. Meanwhile, the distillation prototype contrastive learning algorithm further enhances intra-cluster compactness and inter-cluster divisibility to model the underlying semantic structure of graph data by clustering semantically similar graphs into the same group for better consolidation of the semantic structure from a global perspective. By jointly optimizing the two contrasting learning losses, the pre-trained encoder network can learn representations suitable for a variety of downstream tasks without using any manually labeled labels. We summarize our contributions as follows:

- We develop a novel GCL framework, we call DNPGL, which learns node representations by employing a simple, parameter-free encoder network in an unsupervised manner.

- We propose distillation node contrastive learning algorithms and distillation prototype contrastive learning algorithms that significantly acquire global structural and semantic information of the input graph.
- We conduct extensive experiments to demonstrate that our approach outperforms other state-of-the-art GCL methods on a variety of downstream tasks.

2 Related Work

Graph contrastive learning aims to maximize the Mutual Information (MI) between instances with similar semantic information. InfoGraph [14] emphasizes maximizing mutual information between graph-level representations and patch representations to learn, obtaining good performance graph representations and achieving competitive results on graph classification tasks. GraphCL [17] designs four types of graph data augmentation and proposes a graph contrastive learning framework for pre-training of GNNs, which allows learning of a variety of perturbation-independent graph structural data representation. GCA [19] designs a joint adaptive data augmentation model to provide multiple contexts for nodes at the topology level by removing edge and node attribute perspectives for feature masking at the topology level, respectively. SimGRACE [16] introduces perturbations at the encoder level, compares the semantic similarity between the views obtained after perturbing two encoders, and learns from the comparison using vector representations of semantically similar vectors in the approximate vector space. While these methods have focused on modeling instance-level feature similarity, they often ignores the feature that “nodes with similar features are not always geographically close” despite maintaining local smoothness within individual instances. Therefore, our proposed DNPGCL strives to address this limitation by incorporating feature similarity modeling at both the instance and prototype levels. This comprehensive approach enables our method to uncover the underlying semantic structure of the entire dataset.

3 Methodology

In this section, we elaborate our DNPGCL framework, as depicted in Fig. 2. Our proposed framework consists of three main components: (1) a simple (e.g., 1-layer) encoder network (2) a distillation node comparison learning algorithm (3) a distillation prototype comparison learning algorithm.

3.1 Distillation Node Contrastive Learning

Existing graph contrastive learning methods aim to maximize the MI between different augmented views to achieve node-wise discrimination. While they have achieved some accomplishments, they usually ignore global structural information. In this section, we propose the distillation node contrastive learning algorithm, which shares similarities with knowledge distillation [7]. We approach

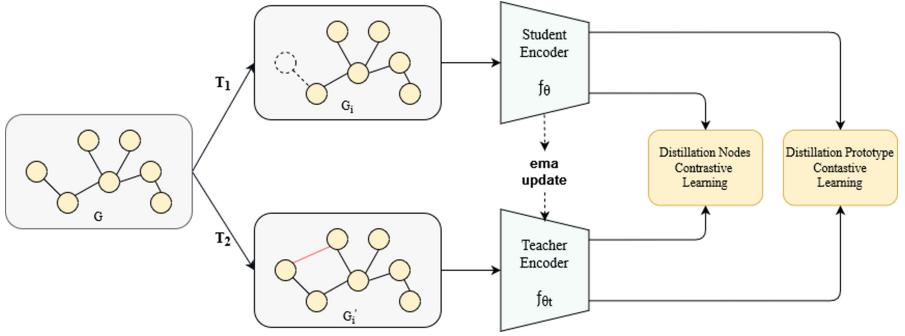


Fig. 2. Overview of DNPGCL. Two graph data augmentations T_1 and T_2 are applied to the input graph G . Subsequently, two views are sent to the student-teacher network. These two networks have the same architecture but different parameters; the parameters of the teacher network are updated with exponential moving average (ema) as the student network's parameters are updated. This process yields graph representations, z_i and z'_i .

distillation node graph contrastive learning from a different perspective. We illustrate the distillation node contrastive learning method in the left of Fig. 3.

Knowledge distillation is a learning paradigm where we train the student network $f_{\theta s}$ to match the output of the teacher network $f_{\theta t}$, parameterized by θs and θt , respectively. Given an input graph G , after undergoing two different data augmentation approaches, the outputs of both networks are represented by P_s and P_t . The probability P is obtained by normalizing the output of the network f_θ with a softmax function. More precisely,

$$P(x)^{(i)} = \frac{\exp(g_{\theta s}(x)^{(i)}/\tau)}{\sum_{k=1}^K \exp(g_{\theta s}(x)^{(k)}/\tau)} \quad (1)$$

where $\tau > 0$ is the temperature parameter controlling the sharpness of the output distribution, and similar formulas apply to P_t with temperature τ_t . Given a fixed teacher network $f_{\theta t}$, we match these distributions by learning to minimize the cross-entropy loss. The parameters of the student network θ_s :

$$\mathcal{L}_{\text{node}} = \min_{\theta_s} H(P_t(x), P(x)), \text{ where } H(a, b) = -a \log b \quad (2)$$

In the following, we will provide a detailed explanation of how to adjust the self-supervised learning problem in Eq. 4. We construct different augmented views of the graph using a perturbation-based strategy [17]. This strategy includes node perturbation, edge perturbation, subgraph sampling. More precisely, from a given graph, we generate a set of different views V . This set contains two global views, G_1 and G'_1 . We minimize this loss:

$$\min_{\theta_s} \sum_{x \in \{G_1, G'_1\}} H(P_t(x), P_s(x)) \quad (3)$$

Both networks share the same architecture with different sets of parameters f_{θ_s} and f_{θ_t} . We learn the parameters θ s by minimizing Eq. (6) using stochastic gradient descent.

Teacher Network. In contrast to traditional knowledge distillation where the teacher network f_{θ_t} is pre-trained, our approach circumvents this step by constructing the teacher network from earlier versions of the student network. Notably, we employ Exponential Moving Average (EMA) on student weights, specifically a momentum encoder [5], which is well-suited for our framework. Originally introduced as an alternative to queues in contrast learning [5], the update rule $\theta_t \leftarrow \lambda\theta_t + (1 - \lambda)\theta_s$, utilizes a cosine schedule for λ ranging from 0.996 to 1 during training [3]. It is crucial to emphasize that, in our framework, the momentum encoder serves a distinct purpose without relying on a queue or a traditional contrast loss such as InfoNCE. Instead, it aligns more closely with the role of the average teacher used in self-supervised training [15]. Remarkably, we observe that this teacher network exhibits a form of model averaging with exponential decay, akin to Polyak-Ruppert averaging. Leveraging Polyak-Ruppert averaging mitigates oscillations during training, enhancing stability and generalization performance. Our findings indicate that this teacher network consistently outperforms the student network throughout training, guiding the training of student network by providing superior-quality target features.

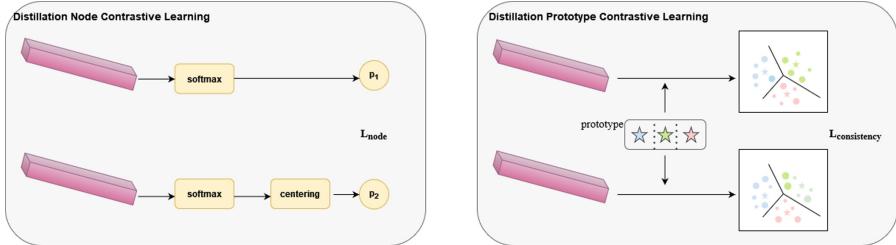


Fig. 3. In the distillation node contrastive learning, the output of the teacher networks is decentralized, then softmax normalized, and subsequently measured for their similarity using cross-entropy loss. In the distillation prototype contrastive learning, the representations are learned by encouraging clustering consistency between relevant views, where prototype vectors are also updated with the encoder parameters via backpropagation.

3.2 Distillation Prototype Contrastive Learning

The distillation node contrastive learning algorithm enhances node-level discrimination by utilizing global structural knowledge acquired from the student-teacher network. However, akin to existing graph contrastive learning methods, it faces a common limitation in explicitly describing the semantic structure of the input graph. To address this shortfall and further capture semantic global

knowledge, we propose the distillation prototype contrastive learning algorithm. This novel algorithm aims to cluster graphs with semantic similarity and promote consistency in clustering between different augmented views, fostering cohesion among semantically similar nodes around their prototypes (cluster centroids). As illustrated in the right side of Fig. 3, this approach facilitates the formation of meaningful clusters.

Representing a graph neural network as $z_i = f_\theta(G_i)$, where f_θ maps the graph sample G_i to the representation vectors $z_i \in \mathbb{R}^D$, we aggregate these representations z_i to form K clusters. These clusters are represented by the trainable prototype vector collection in matrix $C \in \mathbb{R}^{K \times D}$, with column vectors c_1, \dots, c_K serving as the prototype vectors. Unlike traditional clustering methods such as K-means with fixed mean cluster centers, our approach employs prototype vectors as cluster centers. These vectors are the trainable weight matrices of the feedforward network, initialized with He initialization [6]. Given a graph G_i , we compute the similarity between the graph G_i and the K prototypes using Eq. (4):

$$p(y|z_i) = \text{softmax}(C \cdot f_\theta(G_i)) \quad (4)$$

Similarly, z'_i can be used to represent the prototypical assignment for G'_i . To encourage clustering consistency between different augmented views G_i and G'_i of the same graph, i.e., we would like the clustering assignment of G'_i to result in a z_i rather than the original z'_i . We define the clustering consistency objective by minimizing the average cross-entropy loss:

$$\ell(p_i, q_{i'}) = -\frac{1}{N} \sum_{y=1}^N q(y | z'_i) \log p(y | z_i) \quad (5)$$

In Eq. (5), $q(y | z'_i)$ represents the prototype assignment for view G'_i . The clustering consistency goal acts as a regularizer to encourage the clustering similarity between related views. If we exchange the positions of z_i and z'_i in Eq. (5), we can obtain another similar objective:

$$\mathcal{L}_{\text{consistency}} = \sum_{i=1}^N [\ell(p_i, q_{i'}) + \ell(p_{i'}, q_i)] \quad (6)$$

The consistency regularizer can be interpreted as comparing the clustering distributions of multiple views, rather than their specific representations to perform comparisons for inter-view comparison. However, in practice, clustering algorithms such as deep cluster [1] may produce degenerated solutions, where all data is assigned to a single cluster, leading to issues with the distribution q . To avoid this, we add the equipartition constraint to q , aiming to evenly distribute data points among the K clusters:

$$\min_{p,q} \mathcal{L}_{\text{consistency}} \text{s.t. } \forall y: q(y | z_i) \in [0, 1] \text{ and } \sum_{i=1}^N q(y | z_i) = \frac{N}{K} \quad (7)$$

These constraints can be understood as introducing a prior assumption, assuming a uniform label distribution when we do not know the true distribution of labels.

We denote that $P \in \mathbb{R}^{K \times N}$ and $Q \in \mathbb{R}^{K \times N}$, where $P = \frac{1}{N}p(y | z_i)$ and $Q = \frac{1}{N}q(y | z_i)$. The multiplication by $1/N$ ensures that P and Q are probability matrices, i.e., $\sum_{p \in P} p = 1$. We define the feasible solution space T for Q , which satisfies the equipartition constraint :

$$T = \{Q \in \mathbb{R}_+^{K \times N} \mid Q\mathbb{1}_N = r, Q^T\mathbb{1}_K = c\} \quad (8)$$

where $r = \frac{1}{K}\mathbb{1}_K$, $c = \frac{1}{N}\mathbb{1}_N$. The loss function in Eq. (7) can then be rewritten as:

$$\min_{p,q} \mathcal{L}_{\text{consistency}} = \min_{Q \in T} \langle Q, -\log P \rangle - \log N \quad (9)$$

where $\langle \cdot \rangle$ denotes the Frobenius dot-product between two matrices, i.e., element-wise multiplication between matrices. By moving $\log N$ to the other side, we rephrase Eq. (9) as:

$$\min_{p,q} \mathcal{L}_{\text{consistency}} + \log N = \min_{Q \in T} \langle Q, -\log P \rangle \quad (10)$$

For $\min_{Q \in T} \langle Q, -\log P \rangle$, this is a standard optimal transport problem, where $-\log P$ serves as the cost matrix in the optimal transport problem, and during the optimization process is fixed. To make the problem easier to solve, we add the regular term $\text{KL}(Q \parallel rc^T)$, which transforms the Wasserstein distance into the Sinkhorn distance, which we solve iteratively by the Sinkhorn-Knopp algorithm [2]:

$$\min_{Q \in T} \langle Q, -\log P \rangle + \frac{1}{\gamma} \text{KL}(Q \parallel rc^T) \quad (11)$$

where KL represents the Kullback-Leibler divergence, and rc^T can be viewed as a $K \times N$ probability matrix. For Eq. (11), the Sinkhorn-Knopp algorithm shows that its optimal solution satisfies the following form:

$$Q = \text{Diag}(\alpha) P^\eta \text{Diag}(\beta) \quad (12)$$

where α and β are two vectors of selected scaling coefficients. Exponentiation is meant element-wise vectors, α and β can be obtained by simple matrix scaling iterations as follows:

$$\forall y: \alpha_y \leftarrow [P^\lambda \beta]_y^{-1} \forall i: \beta_i \leftarrow [\alpha^T P^\lambda]_i^{-1} \quad (13)$$

3.3 Model Learning

Overall Loss. In order to train our model and learn the encoder $f_\theta(\cdot)$ in an end-to-end manner, we jointly optimize the distillation node and distillation prototype contrastive learning loss. The overall objective function is defined as:

$$\mathcal{L} = \gamma \mathcal{L}_{kno} + (1 - \gamma) \mathcal{L}_{pro} \quad (14)$$

Here, our objective is to minimize \mathcal{L} during training, where γ is a balancing parameter to control the weight of each contrastive learning loss. In our work, γ is a learnable parameter, which enables our model to progressively optimize its performance during training to approximate the optimal solution. Compared to a fixed parameter γ , our model exhibits strong adaptability and expressive power.

4 Experiments

This section is dedicated to demonstrating the effectiveness of the DNPGLC method. In our experiments, we compare with state-of-the-art competitors in the unsupervised graph classification task [14, 17]. The results demonstrate our method outperform the baseline model in several different datasets to achieve optimal results.

4.1 Experimental Setup

Evaluation Datasets and Compared Methods. We evaluate DNPGLC on eight publicly available benchmark datasets, including four bioinformatics datasets (MUTAG, PTC, PROTEINS, NCI1, NCI109) and one social network dataset (COLLAB). We compare with six unsupervised methods including Graph2Vec [11], InfoGraph [14], MVGRL [4], GCC [12], GraphCL [17] and PGCL [9].

4.2 Experimental Results

Baseline Comparisons. For unsupervised graph classification, we adopt the methodology outlined in [14, 17] to access the performance of DNPGLC. The experimental results are presented in Table 1, demonstrating the efficacy of our approach in graph-level representation for downstream graph classification tasks. Overall, from the table, it is evident that our method establishes state-of-the-art performance across all eight datasets when compared to other unsupervised models. DNPGLC consistently outperforms the unsupervised baseline, showcasing its superiority. Notably, on the MUTAG dataset, our method attains an accuracy of 92.8%, marking an absolute improvement of 1.7% over the previous state-of-the-art method (PGCL [9]). These results affirm the robustness and effectiveness of DNPGLC in unsupervised graph classification tasks.

Table 1. Graph classification accuracy (%) for supervised, kernel, and unsupervised methods. We report the average 10-fold cross-validation accuracy over 5 runs.

Datasets	MUTAG	PTC	PROTEINS	NCI1	NCI109	COLLAB
# graphs	188	344	1113	4110	4127	5000
Avg # nodes	17.9	14.3	39.1	29.9	29.7	74.5
Unsupervised						
GRAPH2VEC [11]	83.2 ± 9.3	60.2 ± 6.9	73.3 ± 2.1	73.2 ± 1.8	72.1 ± 0.4	–
INFOGRAPH [14]	89.0 ± 1.1	61.7 ± 1.7	74.4 ± 0.3	73.8 ± 0.7	71.3 ± 0.9	67.6 ± 1.2
MVGRL [4]	89.7 ± 1.1	62.5 ± 1.7	–	75.0 ± 0.7	74.5 ± 1.4	68.9 ± 1.9
GCC [12]	86.4 ± 0.5	58.4 ± 1.2	72.9 ± 0.5	66.9 ± 0.2	67.5 ± 0.3	75.2 ± 0.3
GRAPHCL [17]	86.8 ± 1.3	58.4 ± 1.7	74.4 ± 0.5	77.9 ± 0.4	77.1 ± 1.1	71.4 ± 1.2
PGCL [9]	91.1 ± 1.2	63.3 ± 1.3	75.7 ± 0.2	78.8 ± 0.8	76.4 ± 1.2	76.0 ± 0.3
DNPGLC(ours)	92.8 ± 1.3	64.5 ± 1.5	76.0 ± 0.6	78.2 ± 0.9	78.4 ± 1.7	76.5 ± 0.7

Ablation Studies. In order to access the efficacy of distillation node contrastive learning and distillation prototype contrastive learning within DNPGLC, we conducted ablation studies in MUTAG, NCI1 and PROTEINS datasets. Specifically, we systematically removed one of the contrastive learning components from either variant of DNPGLC and examined the impact on performance in terms of node classification, as detailed in Table 2. The results reveal a noticeable decline in the performance of DNPGLC when either the distillation node contrast or distillation prototype contrast component is omitted. Notably, the comprehensive DNPGLC, incorporating both components, exhibits superior performance. This finding underscores the synergy between distillation node contrast and distillation prototype contrast, as their combined use proves most effective. Thus, our ablation studies provide compelling evidence of the individual effectiveness and mutual complementarity of each contrastive learning component within DNPGLC.

Table 2. Ablation study on contrastive components

Method	MUTAG	NCI1	PROTEINS
Our DNPGLC(\mathcal{L}_{node})	90.5 ± 0.5	78.0 ± 0.9	76.0 ± 0.6
Our DNPGLC($\mathcal{L}_{consistency}$)	91.1 ± 1.2	77.5 ± 0.8	78.3 ± 0.7
Our final DNPGLC($\mathcal{L}_{node} + \mathcal{L}_{consistency}$)	92.8 ± 1.3	78.2 ± 0.9	79.1 ± 0.5

Sensitivity to the Number of GNN Layers. In this section, we discuss the crucial consideration of selecting the number of GNN layers (L). Figure 4(a) presents the performance of DNPGLC across different number of layers ranging from 1 to 20, focusing on MUTAG and PTC datasets. Observations indicate that the optimal performance is achieved when the GNN employs 2 to 3 layers. Notably, while there is a slightly performance decrease with an increase in the number of layers, it avoids severe overfitting - a common concern in traditional GNN models, where an excessive number of layers may lead to a drastic decline in performance. Overall, our DNPGLC demonstrates adaptability and stability to variations in the number of GNN layers.

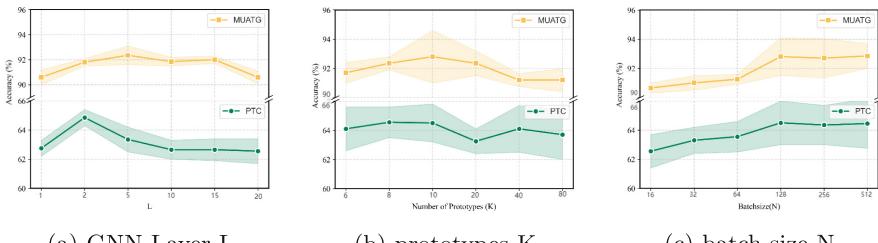


Fig. 4. Sensitivity analysis

Sensitivity to the Number of Prototypes. In this experiment, we evaluate the impact of prototypes count (K) on model performance. Figure 4(b) illustrates the classification accuracy of our DNPGLC on the MUTAG and PTC datasets across a range of prototype counts, varying from 6 to 80. The line plots depict a discernible trend, indicating that initially increasing the number of prototypes contributes to improve performance. However an excessive number of prototypes leads to a marginal decline in performance. In general, our DNPGLC exhibits robustness in response to variations in the choice of prototype count (K), showcasing its stability and adaptability in handling different prototype configurations.

Sensitivity to Batch Size. Figure 4(c) presents the classification accuracy of our model on the MUTAG and PTC datasets across different batch sizes ranging from 16 to 512. The line plots reveal a notable trend where larger batch sizes lead to better performance. In our experiments, for fairness and comparability with other competitors, we adapt a consistent setup, training the GNN encoder with a batch size of 128 and an epoch number of 20.

Visualization Results. In order to demonstrate the high-quality graph representations learned by DNPGLC, we use T-SNE [10] to visualize graph representations and prototype vectors for the number of clusters $K = 10$ on the three datasets. The visualization results, depicted in Fig. 5, represent each dot as a graph representation, with point color indicating its true label. The model excels in uncovering the underlying linguistic structure of the entire data distribution. The visualization highlights the superior intra-class compactness and inter-class distinctiveness achieved by DNPGLC. These results underscore the model’s capacity to generate meaningful representations, substantiating its effectiveness in capturing intricate graph relationships and semantic structures.

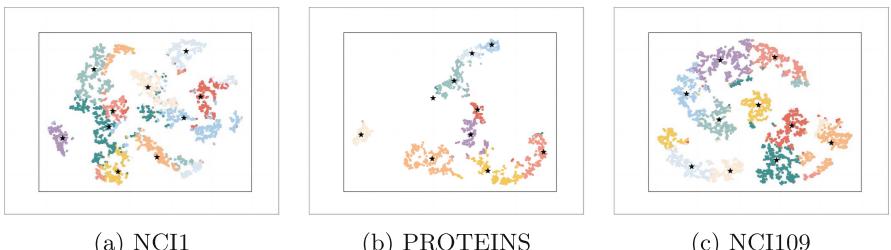


Fig. 5. Visualization of T-SNE for learning representations on three datasets, “ \star ” denotes the prototype vector.

5 Conclusion

In this paper, we propose a novel GCL framework called DNPGCL. Our framework effectively captures global knowledge in unsupervised graph learning, addressing both structural and semantic levels. Through the synergistic optimization of distillation nodes and distillation prototypes contrastive learning loss, we effectively train an encoder network employing simple graph neural networks. This enables the learning of excellent representations without the need for manually labeled data. Our comprehensive experiments demonstrate the superiority of the DNPGCL method across multiple publicly available benchmark datasets. Notably, DNPGCL outperforms current state-of-the-art unsupervised GCL methods, showcasing its efficacy in advancing the field of unsupervised graph representation learning.

Acknowledgments. This work was supported by the National Natural Science Foundation of China under Grant No. U1936213, Program of Shanghai Academic Research Leader No. 21XD1421500, Shanghai Science and Technology Commission Project No. 20020500600.

References

1. Caron, M., Bojanowski, P., Joulin, A., Douze, M.: Deep clustering for unsupervised learning of visual features. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 132–149 (2018)
2. Cuturi, M.: Sinkhorn distances: lightspeed computation of optimal transport. In: Advances in Neural Information Processing Systems, vol. 26 (2013)
3. Grill, J.B., et al.: Bootstrap your own latent-a new approach to self-supervised learning. *Adv. Neural. Inf. Process. Syst.* **33**, 21271–21284 (2020)
4. Hassani, K., Khasahmadi, A.H.: Contrastive multi-view representation learning on graphs. In: International Conference on Machine Learning, pp. 4116–4126. PMLR (2020)
5. 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)
6. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1026–1034 (2015)
7. 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)
8. Lin, B., Luo, B., He, J., Gui, N.: Self-supervised adaptive aggregator learning on graph. In: Karlapalem, K., et al. (eds.) PAKDD 2021. LNCS (LNAI), vol. 12714, pp. 29–41. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-75768-7_3
9. Lin, S., et al.: Prototypical graph contrastive learning. *IEEE Trans. Neural Netw. Learn. Syst.* **35**, 2747–2758 (2022)
10. Van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**(11), 2579–2605 (2008)
11. Narayanan, A., Chandramohan, M., Venkatesan, R., Chen, L., Liu, Y., Jaiswal, S.: graph2vec: learning distributed representations of graphs. arXiv preprint [arXiv:1707.05005](https://arxiv.org/abs/1707.05005) (2017)

12. Qiu, J., et al.: GCC: graph contrastive coding for graph neural network pre-training. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1150–1160 (2020)
13. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE Trans. Neural Networks **20**(1), 61–80 (2008)
14. Sun, F.Y., Hoffmann, J., Verma, V., Tang, J.: Infograph: unsupervised and semi-supervised graph-level representation learning via mutual information maximization. arXiv preprint [arXiv:1908.01000](https://arxiv.org/abs/1908.01000) (2019)
15. Tarvainen, A., Valpola, H.: Mean teachers are better role models: weight-averaged consistency targets improve semi-supervised deep learning results. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
16. Xia, J., Wu, L., Chen, J., Hu, B., Li, S.Z.: SimGRACE: a simple framework for graph contrastive learning without data augmentation. In: Proceedings of the ACM Web Conference 2022, pp. 1070–1079 (2022)
17. You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., Shen, Y.: Graph contrastive learning with augmentations. Adv. Neural. Inf. Process. Syst. **33**, 5812–5823 (2020)
18. Zhang, W., et al.: Evaluating deep graph neural networks. arXiv preprint [arXiv:2108.00955](https://arxiv.org/abs/2108.00955) (2021)
19. Zhu, Y., Xu, Y., Yu, F., Liu, Q., Wu, S., Wang, L.: Graph contrastive learning with adaptive augmentation. In: Proceedings of the Web Conference 2021, pp. 2069–2080 (2021)



DEGNN: Dual Experts Graph Neural Network Handling both Edge and Node Feature Noise

Tai Hasegawa^{1,2} , Sukwon Yun³ , Xin Liu^{2(\square)} , Yin Jun Phua¹ , and Tsuyoshi Murata^{1,2}

¹ Department of Computer Science, Tokyo Institute of Technology, Tokyo, Japan
hasegawa.t@net.c.titech.ac.jp, {phua,murata}@c.titech.ac.jp

² Artificial Intelligence Research Center, AIST, Tokyo, Japan
xin.liu@aist.go.jp

³ Industrial and Systems Engineering, KAIST, Daejeon, Republic of Korea
swyun@kaist.ac.kr

Abstract. Graph Neural Networks (GNNs) have achieved notable success in various applications over graph data. However, recent research has revealed that real-world graphs often contain noise, and GNNs are susceptible to noise in the graph. To address this issue, several Graph Structure Learning (GSL) models have been introduced. While GSL models are tailored to enhance robustness against edge noise through edge reconstruction, a significant limitation surfaces: their high reliance on node features. This inherent dependence amplifies their susceptibility to noise within node features. Recognizing this vulnerability, we present DEGNN, a novel GNN model designed to adeptly mitigate noise in both edges and node features. The core idea of DEGNN is to design two separate experts: an edge expert and a node feature expert. These experts utilize self-supervised learning techniques to produce modified edges and node features. Leveraging these modified representations, DEGNN subsequently addresses downstream tasks, ensuring robustness against noise present in both edges and node features of real-world graphs. Notably, the modification process can be trained end-to-end, empowering DEGNN to adjust dynamically and achieves optimal edge and node representations for specific tasks. Comprehensive experiments demonstrate DEGNN’s efficacy in managing noise, both in original real-world graphs and in graphs with synthetic noise.

Keywords: Graph Neural Networks · Graph Structure Learning · Graph Self-Supervised Learning

1 Introduction

Graphs are essential data structures for modeling a wide range of real-world phenomena, such as social networks, transportation networks, and chemical molecules. Graph Neural Networks (GNNs) have emerged as a powerful paradigm for modeling such graphs, primarily due to their message-passing

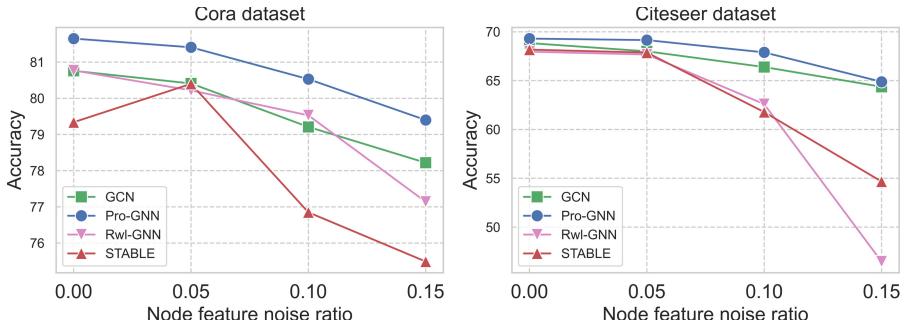


Fig. 1. A comparison of semi-supervised node classification between GCN and GSL models on Cora and Citeseer dataset when there is noise in the node features.

mechanism that aggregates node representations via edges. These GNNs can be applied to various tasks, including node classification [11, 24], link prediction [37], node ranking [3, 7], community detection [29], and graph classification [4].

Despite their success, it is well known that the quality of real-world graph data is often unreliable [8]. In other words, real-world graphs are known to contain noise. For instance, in citation networks, references to unrelated papers can introduce noise in the form of inaccurate edges. Recent studies on adversarial attacks and defenses have highlighted the susceptibility of GNNs to noise within graphs [9, 10]. In response, Graph Structure Learning (GSL) has been developed as a method to optimize graph structure, thus improving graph representations and ensuring more resilient predictions amidst edge noise.

Besides edge noise, node features can also contain noise. For instance, in social networks, users might provide inconsistent, overstated, or even false information about their interests or attributes, which introduces noise into the node features. GSL models often struggle with such node feature noises. The vulnerability of GSL models against node feature noise arises as these models disseminate noisy node features via message passing scheme. Additionally, their reliance on node features to rewire edges compounds the challenge. As illustrated in Fig. 1 (refer to Sect. 4 for the experiment details), recent GSL models such as Pro-GNN [18], Rwl-GNN [35], and STABLE [20] tend to underperform in prediction accuracy when compared to traditional GNN model, GCN [11] when the node feature noise increases. Although Pro-GNN marginally surpasses the performance of GCN, there remains ample scope for improvements.

In this paper, we introduce Dual Experts Graph Neural Network (DEGNN)¹, a novel GNN model crafted to offer robust predictions irrespective of the presence or absence of noise in edges, nodes, or both. At the heart of DEGNN is its distinctive architecture, which employs specialized branches, termed “experts”, to individually learn and refine node features and edges. Using a self-supervised learning approach, these experts are seamlessly integrated and co-trained end-

¹ Codes are available at: <https://github.com/TaiHasegawa/DEGNN>.

to-end, ensuring task-specific optimization. Our contributions are summarized as follows:

- We highlight the susceptibility of GSL models to node feature noise based on preliminary experiments.
- We introduce DEGNN, a novel GNN that offers robust predictions irrespective of the presence or absence of noise in nodes, edges, or both, by individually addressing these noises through a self-supervised learning approach.
- Through comprehensive experiments on real-world datasets, we establish that DEGNN consistently delivers stable predictions, outperforming state-of-the-art models in the presence of either type of noise.

2 Related Work

2.1 Graph Neural Networks

GNNs have emerged as a powerful tool for learning from graph-structured data, effectively capturing the complex relationships and interdependencies between nodes [1]. Their successful development across numerous practical fields underscores their extensive applicability and effectiveness [16, 17, 26–28]. Generally, GNNs can be classified into two categories: spectral-based methods [6, 11] and spatial-based methods [12, 13]. Spectral-based GNNs hinge upon spectral graph theory [5] and employ spectral convolutional neural networks. To streamline the intricacies of spectral-based GNNs, various techniques have emerged, including ChebNet [6] and GCN [11].

On the other hand, spatial-based GNNs are engineered to tackle challenges related to efficiency, generality, and flexibility, as highlighted in [15]. They achieve graph convolution operation through neighborhood aggregation. For instance, GraphSAGE [12] selectively samples a subset of neighbors to grasp local features, while GAT [13] employs an attention mechanism for adaptive neighbor aggregation. However, these models are susceptible to edge noise.

2.2 Graph Structure Learning

The purpose of GSL is to improve prediction accuracy by modifying the given graph into an optimal structure. In this paper, we primarily focus on GNN-based graph structure learning models. LDS [14] jointly optimizes the probability for each node pair and the parameters of GNNs in a bilevel way. Pro-GNN [18] aims to learn the optimal graph structure by incorporating several regularizations, such as low-rank sparsity and feature smoothness. Gaug-M [19] directly computes the edge weights by taking the inner product of node embeddings. STABLE [20] utilizes self-supervised learning to acquire node embeddings and then modifies the graph structure based on their similarities. These obtained node embeddings and the modified graph structure are employed in downstream tasks. Each of these models demonstrates the ability to robustly predict against edge noise, as shown in their respective papers. However, as elucidated in Sect. 1, their pronounced reliance on node features during edge rewiring inherently exposes them to vulnerabilities in situations characterized by noisy node features.

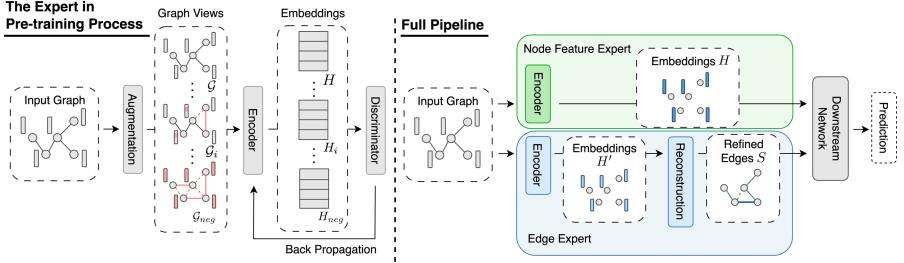


Fig. 2. The overview of DEGNN and the expert in its pre-training process.

3 The Proposed Model

In this section, we introduce our proposed approach, DEGNN. We begin by formulating the problem definition, followed by an overview of the model, and then provide a detailed description of its architecture and learning procedure.

3.1 Problem Definition

Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, X\}$ represent an undirected graph, where $\mathcal{V} = \{v_i\}_{i=1}^N$ is the set of N nodes, \mathcal{E} is the set of edges, $X = [x_1, \dots, x_N] \in \mathbb{R}^{N \times D}$ indicates the node feature matrix and each $x_i \in \mathbb{R}^D$ is the feature vector of node v_i . The set of edges is represented by an adjacency matrix $A \in \{0, 1\}^{N \times N}$, where A_{ij} denotes the connection between nodes v_i and v_j . Following the common semi-supervised node classification setting, only a small portion of nodes $\mathcal{V}_L = \{v_i\}_{i=1}^l$ are associated with the corresponding labels $\mathcal{Y}_L = \{y_i\}_{i=1}^l$ while the rest of the nodes $\mathcal{V}_U = \{v_i\}_{i=l+1}^N$ are unlabeled.

Given graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, X\}$ and the available labels \mathcal{Y}_L , the goal of graph structure learning, aimed at refining node features and graph structure, is to learn optimal node embeddings $H \in \mathbb{R}^{N \times D'}$ with hidden dimension D' , a modified adjacency matrix $S \in \mathbb{R}^{N \times N}$, and the GNN parameters θ in order to improve the predictive accuracy of $\hat{\mathcal{Y}}_L$. The objective function can be formulated as

$$\min_{\theta, S, H} \mathcal{L}(A, X, \mathcal{Y}_L) = \sum_{v_i \in V_L} \ell(f_\theta(H, S)_i, y_i), \quad (1)$$

where $f_\theta : \mathcal{V}_L \rightarrow \mathcal{Y}_L$ is a function learned by downstream GNNs that maps nodes to the set of labels, $f_\theta(H, S)_i$ is the prediction of node v_i , and ℓ is the loss measuring difference between prediction and true label such as cross-entropy.

3.2 Overview

The overview of DEGNN and the expert in its pre-training process are illustrated in Fig. 2. The input graph is passed through both the node feature expert and the edge expert. The node feature expert outputs node embeddings H , while the edge

expert produces the modified adjacency matrix S . Using the obtained H and S , the downstream network makes predictions for a specific task. Traditionally, models have primarily leaned on learned node embeddings for edge reconstruction [20]. However, when these embeddings are derived from noisy node features, the resulting in reconstructed edges often produce sub-optimal outcomes, potentially that are unintended dependencies. Our proposed dual experts design aims to eliminate these dependencies and learn both node representations and edges to be optimal. These two experts can be trained with the downstream network end-to-end, allowing each of them to acquire representations suitable for the specific task. The details of the model are described in the following sections.

3.3 Node Feature Expert

A straightforward approach to predict robustly against node feature noise is to generate node embeddings H that capture node features more effectively than the input node features X . To obtain the node embeddings H , we utilize self-supervised learning, since it enables the model to achieve better performance, generalization, and robustness across a wide range of downstream tasks [32].

Graph Augmentation. Generating views is a key component of self-supervised learning methods. For instance, in the field of computer vision [21], views are created by rotating or cropping images. This allows for mitigating the impact of differences in angles and scales on the model’s predictions. Similarly, for graph data, we presume that by generating views and exposing the model to modified edges and nodes, it can achieve enhanced generalizability, equipping it with a stronger capacity to manage noise emanating from both edges and nodes during predictions. Therefore, we generate graphs of positive view $\mathcal{G}_1 = (\tilde{A}, X)$ with noise added to the edges, $\mathcal{G}_2 = (A, \tilde{X})$ with noise added to the node features, and $\mathcal{G}_3 = (\tilde{A}, \tilde{X})$ with noise added to both edges and node features, where $\tilde{A} \in \{0, 1\}^{N \times N}$ and $\tilde{X} \in \mathbb{R}^{N \times D}$ denote the noisy adjacency matrix and noisy node features, respectively.

Our proposed model augments edges by randomly rewiring them. Formally, we first create a mask matrix $P \in \{0, 1\}^{N \times N}$ to rewire edges, where P is obtained from a Bernoulli distribution $P_{ij} \sim \mathcal{B}(p)$ with a hyper-parameter p that controls the rewiring probability. And then, \tilde{A} is calculated as follows:

$$\tilde{A} = (1 - A) \odot P + A \odot (1 - P), \quad (2)$$

where \odot is the Hadamard product. For node feature augmentation, unlike other studies that use node feature masking [22, 23], in this paper, we shuffle elements in each row of X randomly with the probability q to replicate scenarios where there is noise in the node features.

Besides positive graphs, we generate a negative graph $\mathcal{G}_{neg} = (A_{neg}, X_{neg})$ that is entirely different from the original graph \mathcal{G} , where $A_{neg} \in \{0, 1\}^{N \times N}$ and $X_{neg} \in \mathbb{R}^{N \times D}$ denote the negative adjacency matrix and negative node features, respectively. A_{neg} is given by $A_{neg} = (1 - A) \odot P_{neg}$, where $P_{neg} \in \{0, 1\}^{N \times N}$ is a

mask matrix that is obtained from a Bernoulli distribution $P_{negij} \sim \mathcal{B}(\frac{\|\mathcal{E}\|}{N^2 - \|\mathcal{E}\|})$. X_{neg} is obtained through row-wise shuffling of X .

Encoder. The encoder f_ϕ parameterized by ϕ learns embeddings for each of the generated views. In this paper, we use a 1-layer GCN [11] as the encoder, which is formulated as follows:

$$f_\phi(X, A) = \sigma(\hat{D}^{-1/2} \hat{A} \hat{D}^{-1/2} X W), \quad (3)$$

where $\hat{A} = A + I_N$, $\hat{D} = D + I_N$, D is the degree matrix of A , I_N is the identity matrix, σ is a non-linear activation function and W is weight matrix transforming raw features to embeddings. The embeddings for the original graph and each view $H, H_1, H_2, H_3, H_{neg} \in \mathbb{R}^{N \times D'}$ are obtained using the encoder as follows:

$$H_2 = f_\phi(\tilde{X}, A) \quad (6)$$

$$H = f_\phi(X, A) \quad (4) \qquad H_3 = f_\phi(\tilde{X}, \tilde{A}) \quad (7)$$

$$H_1 = f_\phi(X, \tilde{A}) \quad (5) \qquad H_{neg} = f_\phi(X_{neg}, A_{neg}) \quad (8)$$

Objective Function. Following other graph contrastive learning methods [20, 31], we train the encoder to maximize the mutual information between the original graph \mathcal{G} and positive graphs \mathcal{G}_1 , \mathcal{G}_2 and \mathcal{G}_3 while minimizing agreement between original graph \mathcal{G} and negative graph \mathcal{G}_{neg} . Formally, the objective function for node feature expert can be formulated via using binary cross-entropy loss between positive samples and negative samples as follows:

$$\mathcal{L}_N = -\frac{1}{2N} \sum_{i=1}^N \left(\frac{1}{3} \sum_{j=1}^3 (\log \mathcal{D}(h_i, h_i^j) + \log(1 - \mathcal{D}(h_i, h_i^{neg}))) \right), \quad (9)$$

where h_i, h_i^j, h_i^{neg} are the embeddings of node v_i in \mathcal{G} , \mathcal{G}_j and \mathcal{G}_{neg} , and \mathcal{D} is the discriminator built upon an inner product, i.e., $\mathcal{D}(a, b) = ab^T$. In conclusion, by training the node feature expert, f_ϕ , we derive a noise-robust node embedding H that subsequently serves as input for the downstream network.

3.4 Edge Expert

To differentiate between the impacts of node feature noise and edge noise, and to adeptly address each situation, we further implement an edge expert. The edge expert learns node embeddings H' using self-supervised learning as with the

node feature expert. More precisely, H' is obtained through an encoder f_ψ with different parameters from the node feature expert f_ϕ , and the objective function for edge expert using the binary cross-entropy loss L_E can be formulated as follows:

$$\mathcal{L}_E = -\frac{1}{2N} \sum_{i=1}^N \left(\frac{1}{3} \sum_{j=1}^3 (\log \mathcal{D}(h'_i, h_i^{j'}) + \log(1 - \mathcal{D}(h'_i, h_i^{neg'}))) \right), \quad (10)$$

where h'_i , $h_i^{j'}$, $h_i^{neg'}$ are the embeddings of node v_i in \mathcal{G} , \mathcal{G}_j and \mathcal{G}_{neg} obtained with encoder f_ψ .

Reconstruction. Once the high-quality embedding H' from the edge expert is obtained, it is used to reconstruct the graph structure. This reconstruction process rewrites the edges using the pairwise similarity in the embeddings H' under the homophily assumption [2], which posits that nodes with similar features are more likely to be connected.

In the beginning, we compute the cosine similarity matrix $B \in \mathbb{R}^{N \times N}$, where B_{ij} represents the cosine similarity between H'_i and H'_j . Next, we remove the $k\%$ (k is a hyper-parameter) of the edges with the smallest cosine similarity between pairs of nodes from the original edge set \mathcal{E} , resulting in a sparser adjacency matrix $\tilde{S} \in \{0, 1\}^{N \times N}$ as follows:

$$\tilde{S} = A \odot M_1, \quad (11)$$

where $M_1 \in \{0, 1\}^{N \times N}$ represents the mask matrix for edge deletion, where M_{1ij} is 1 if its cosine similarity ranks over the smallest $k\%$, and 0 otherwise.

Finally, to obtain the modified adjacency matrix S , we add the same number of edges that were removed i.e., $k * |\mathcal{E}|$ with the highest cosine similarity between pairs of nodes from the set of node pairs $\mathcal{E}' = V \times V \setminus \mathcal{E}$ that are not included in the edge set \mathcal{E} . Formally, S is obtained as follows:

$$S = \tilde{S} + B \odot M_2, \quad (12)$$

where $M_2 \in \{0, 1\}^{N \times N}$ is the mask matrix for edge addition, where M_{2ij} is 1 for the edges within top cosine similarity ($k * |\mathcal{E}'|$), and 0 otherwise. Here, it is important that the encoder in the edge expert can be trained through backpropagation from the downstream network, so the reconstruction needs to be differentiable. Consequently, the value associated with newly introduced edges in S is not clamped and retains its cosine similarity. To sum up, through training edge expert, f_ψ , we obtain a modified adjacency matrix, S , which is then utilized as input for the downstream network.

3.5 Downstream Network

Once we obtain the node embeddings H and the modified adjacency matrix S , we now use them as input for the downstream network (i.e., GNNs). It is worth noting that our proposed method allows the use of any GNNs such as GCN [11]

or GAT [13], and also can be applied to various tasks beyond node classification, including link prediction [37] and graph classification [4]. In this paper, we use a 2-layer GCN f_θ as the downstream network. To tackle node classification, the model is trained to minimize the cross-entropy loss:

$$\mathcal{L}_{GNN} = \sum_{v_i \in V_L} \ell(f_\theta(H, S)_i, y_i), \quad (13)$$

where $\ell(f_\theta(H, S)_i, y_i)$ is the cross-entropy between the prediction and the ground-truth label for node v_i .

3.6 Training Methodology

In this paper, we propose two variants of DEGNN with different training methods: (i) the pre-training and fine-tuning model (referred to as DEGNN-I), and (ii) the modular learning model (referred to as DEGNN-II). DEGNN-I first pre-trains the node feature expert and edge expert separately. Then, all components are jointly fine-tuned in an end-to-end manner. This is to enable each expert to obtain the optimal node embeddings and graph structure for downstream tasks. During the fine-tuning process, it is trained to minimize the following objective function:

$$\mathcal{L} = \mathcal{L}_{GNN} + \alpha \mathcal{L}_N + \beta \mathcal{L}_E, \quad (14)$$

where α and β are hyper-parameters to balance the contributions of node embedding generation and edge reconstruction, respectively.

In contrast, DEGNN-II follows a two-step approach: initially training the node feature expert and edge expert and subsequently freezing them during the training of the downstream network. This methodology empowers each expert to attain task-agnostic representations. This allows robust predictions in contexts with limited labels or large biases, given that the approach does not depend on the provided label during training—essentially, a self-supervised paradigm.

4 Experiments

In this section, we evaluate our proposed method, DEGNN, on a variety of noisy graphs in the context of the semi-supervised node classification task.

4.1 Experimental Setup

Datasets. We used four open datasets, including two citation networks (i.e., Cora [19], Citeseer [19]) and two co-purchasing networks (i.e., Photo [33], Computer [33]). Regarding the train/validation/test split, we prepared 20 training labels for each class in all datasets. For validation and test, we used 500 and 1000 nodes, respectively.

Noisy Graphs. To assess the robustness of our model across various graph settings with noise, we compared models on graphs that included the following types of noise:

- **Clean Graphs:** The original graphs of the datasets which may contain inherent node feature noise and edge noise.
- **Edge Noisy Graphs:** We randomly remove a certain number of edges and insert the same number of fake edges.
- **Node Feature Noisy Graphs:** As in [36], we added independent Gaussian noise to the node features. Specifically, we obtained the reference amplitude r by calculating the mean of the maximum value across each node’s features. For each feature dimension of each node, we introduced independent Gaussian noise $\lambda \cdot r \cdot \epsilon$, where $\epsilon \sim N(0, 1)$, and λ represents the feature noise ratio.
- **Edge and Node Feature Noisy Graphs:** We introduced both the edge noise and node feature noise described above.

When adding these noises, we employed a poisoning attack, which initially prepares a graph with noise added, and used it for both training and evaluation.

Baselines. We compare the proposed DEGNN-I and DEGNN-II with two categories of baselines: classical GNN models (i.e., GCN [11], GAT [13] and RGCN [34]) and graph structure learning methods (i.e., Pro-GNN [18], Rwl-GNN [35] and STABLE [20]).

Implementation Details. All hyper-parameters are tuned on the clean graph. All models are trained using Adam optimizer with a default learning rate of 1e-2 and a weight decay of 5e-4 when not explicitly specified. GCN [11], GAT [13], and RGCN [34] have a fixed number of layers at 2. For GCN and RGCN, the hidden dimension is chosen from {16, 32, 64, 128}. GAT’s number of heads and head dimensions are selected from {1, 2, 4, 8, 16} and {8, 16, 32, 64, 128}, respectively, with a total hidden dimension ranging from 16 to 128. Other baselines follow hyper-parameter combinations specified in their respective papers. For DEGNN, α and β are tuned from {0, 0.1, 1.0, 10}, the hidden dimension D' is tuned from {128, 256, 512}. k is searched in {1, 5, 10, 15, 20, 25}, p and q are searched in {0.2, 0.4, 0.6}. The learning rate in the pre-training process is tuned from {1e-2, 5e-3, 1e-3}.

4.2 Semi-supervised Node Classification

Performance Comparison. In this section, we evaluate the proposed DEGNN on semi-supervised node classification on original graphs. The average accuracy and standard deviation of the model across 10 runs are summarized in Table 1. OOM indicates out of memory. Based on the experiment, our proposed approach outperformed other models in citation networks (Cora, Citeseer), demonstrating the highest predictive accuracy. This suggests that the experts successfully

Table 1. The results (accuracy(%) \pm std) of semi-supervised node classification on clean graphs. The top two performance is highlighted in bold and underline.

Dataset	GCN	GAT	RGCN	Pro-GNN	Rwl-GNN	STABLE	DEGNN-I	DEGNN-II
Cora	80.8 ± 0.9	79.5 ± 0.8	79.6 ± 0.7	81.7 ± 0.7	80.8 ± 0.7	79.3 ± 1.1	83.0 ± 0.9	81.6 ± 1.0
Citeseer	68.8 ± 0.7	66.7 ± 1.7	65.7 ± 1.5	69.3 ± 0.6	68.0 ± 0.6	68.2 ± 0.5	70.9 ± 1.9	71.6 ± 1.4
Photo	91.8 ± 0.2	85.5 ± 17.3	91.7 ± 0.5	91.8 ± 0.7	86.4 ± 2.9	OOM	91.3 ± 0.4	91.6 ± 0.6
Computers	85.0 ± 0.9	77.2 ± 22.7	81.6 ± 2.6	OOM	74.6 ± 1.4	OOM	83.5 ± 0.9	82.8 ± 0.8

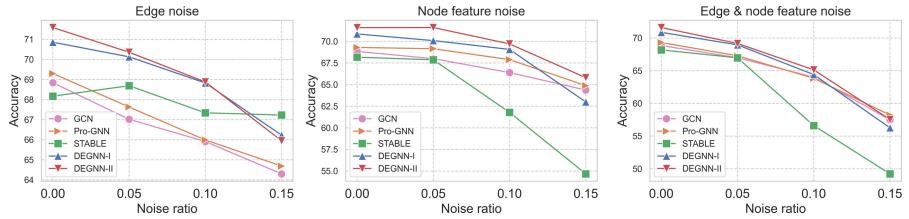


Fig. 3. Accuracy on Citeseer with noise added to either or both of edge and node features.

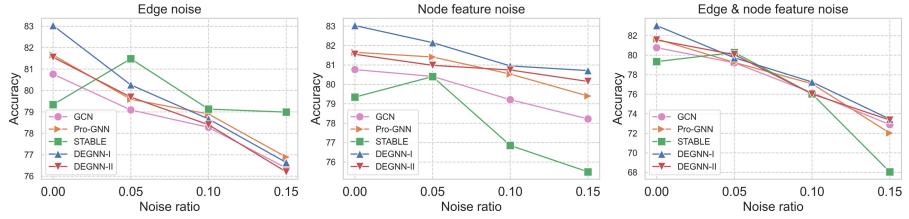


Fig. 4. Accuracy on Cora with noise added to either or both of edge and node features.

obtain superior representations for both edges and node features to eliminate latent noise within these graphs. However, in co-purchasing networks (Photo, Computer), our model achieved competitive results but was marginally surpassed by the traditional GCN. Considering that other GSL models also lagged behind GCN in accuracy and the fact that GCN's predictive accuracy in co-purchasing networks is significantly higher than in citation networks, it is conceivable that the co-purchasing networks may contain less inherent noise within the original graphs, and the GSL models might be unnecessarily altering the graph.

Robustness Evaluation. In this experiment, we evaluate the robustness of the models by comparing their performance on graphs with noise added to either or both of the edge and node features. The nature of these noise is described in Sect. 4.1. Specifically, we added edge noise with noise ratio 0.05, 0.1 and 0.15, while the parameter λ for node feature perturbation is also set to 0.05, 0.1, and 0.15. All experiments were conducted 10 times, and the average accuracy on

Citeseer and Cora dataset are shown in Figs. 3 and Fig. 4, respectively. From this experiment, we can obtain the following observation:

- When node feature noise is added, DEGNN-I or DEGNN-II demonstrated the highest accuracy among the compared models in all settings.
- When noise is introduced in only the edge and in both edge and node features, DEGNN demonstrated the best or competitive results compare to the baselines especially when the noise ratio is 0, 0.05 or 0.1.
- The smallest accuracy gap between edge noise ratio of 0 and 0.15 was observed in STABLE. However, it is highly vulnerable when node features are perturbed.
- GCN and Pro-GNN follow a similar trend, with Pro-GNN slightly outperforming GCN by a small margin. In many settings, their accuracy fell below that of DEGNN.

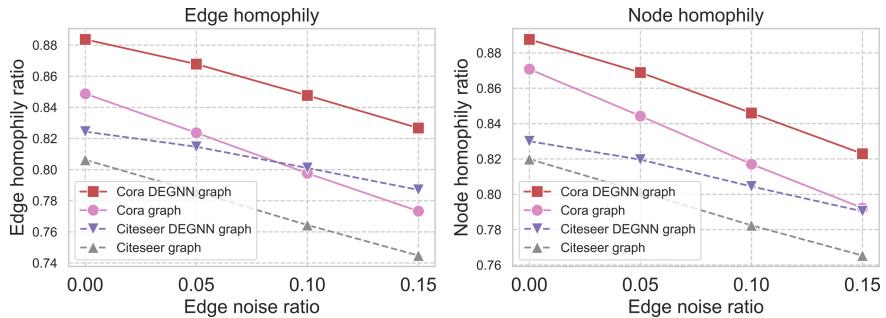


Fig. 5. Comparison of edge homophily ratio and node homophily ratio between noisy graphs and refined graphs by the edge expert on Cora and Citeseer.

4.3 Edge Expert Analysis

To evaluate the edge expert, we compared the edge homophily ratio [25] and node homophily ratio [30] between the noisy graph and the graphs refined by the edge expert (referred to as the DEGNN graph) on various edge noise added settings. In this experiment, we used DEGNN-II, and we set the hyper-parameter k , which controls the number of edge rewrites, to 10%. Figure 5 shows the edge homophily ratio and node homophily ratio as edge noise is gradually added. The solid lines represent the results for the Cora dataset, while the dashed lines represent the results for the Citeseer dataset. This experiment reveals that the edge expert consistently promotes high edge homophily and node homophily in all scenarios.

4.4 Node Feature Expert Analysis

In this experiment, we compared the effectiveness of the node feature expert by comparing the models' accuracy when node features have noise added. Figure 6 shows the average of accuracy of 10 runs of GCN and DEGNN-I only having node feature expert (without edge expert) on Cora and Citeseer dataset with added node feature noise. Empirical results confirm that employing node embeddings derived from the node feature expert markedly elevates prediction accuracy across a majority of scenarios. This underscores both the indispensability and efficacy of the node feature expert.

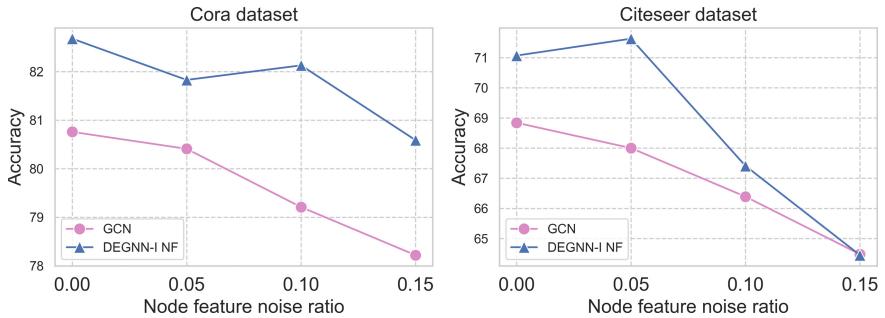


Fig. 6. Comparison of GCN and DEGNN-I without edge expert on Cora and Citeseer with added node feature noise.

5 Conclusion

In this paper, we identified the vulnerability of recent GSL methods to node feature noise and proposed a novel GNN model, DEGNN, to address this issue. DEGNN refines both edges and node features using two carefully designed experts via self-supervised learning, allowing it to robustly perform predictions in the presence of both edge and node feature noise. Extensive experiments verify the effectiveness of the experts in handling graphs with various noise scenarios.

In this paper, we employed GCN as both the encoder and downstream network. Additionally, simple methods were used for augmentation and edge reconstruction techniques. As future work, we plan to explore more specialized networks in the encoder and downstream network, as well as investigate different augmentation and edge reconstruction techniques.

Acknowledgements. This work is partly supported by JSPS Grant-in-Aid for Scientific Research (grant number 23H03451, 21K12042) and the New Energy and Industrial Technology Development Organization (Grant Number JPNP20017).

References

1. Zhou, J., et al.: Graph neural networks: a review of methods and applications. *AI Open* **1**, 57–81 (2020)
2. McPherson, M., Smith-Lovin, L., Cook, J.M.: Birds of a feather: homophily in social networks. *Ann. Rev. Sociol.* **27**(1), 415–444 (2001)
3. Maurya, S.K., Liu, X., Murata T.: Graph neural networks for fast node ranking approximation. In: TKDD (2021)
4. Zhang, M., et al.: An end-to-end deep learning architecture for graph classification. In: AAAI (2018)
5. Chung, F.R.K.: Spectral Graph Theory. number 92. American Mathematical Soc (1997)
6. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: NeurIPS (2016)
7. Maurya, S.K., Liu, X., Murata, T.: Fast approximations of betweenness centrality with graph neural networks. In: CIKM (2019)
8. Marsden, P.V.: Network data and measurement. *Ann. Rev. Sociol.* **16**(1), 435–463 (1990)
9. Dai, H., et al.: Adversarial attack on graph structured data. In: ICML (2018)
10. Jin, W., et al.: Adversarial attacks and defenses on graphs. In: SIGKDD (2021)
11. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: ICLR (2017)
12. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: NeurIPS (2017)
13. Veličković, P., Cucurull, G.: Arantxa Casanova. Pietro Lio, and Yoshua Bengio. Graph attention networks. In ICLR, Adriana Romero (2018)
14. Franceschi, L., Niepert, M., Pontil, M., He, X.: Learning discrete structures for graph neural networks. In: ICML (2019)
15. 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)
16. Jin, R., Xia, T., Liu, X., Murata, T.: Predicting emergency medical service demand with bipartite graph convolutional networks. *IEEE Access* **9**, 9903–9915 (2021)
17. Fan, W., et al.: Graph neural networks for social recommendation. In: WWW, pp. 417–426 (2019)
18. Jin, W., et al.: Graph structure learning for robust graph neural networks. In: SIGKDD (2020)
19. Zhao, T., et al.: Data augmentation for graph neural networks. In: AAAI (2021)
20. Li, K., et al.: Reliable representations make a stronger defender: unsupervised structure refinement for robust GNN. In: SIGKDD (2022)
21. Berthelot, D., et al.: MixMatch: a holistic approach to semi-supervised learning. In: NeurIPS (2019)
22. Zhu, Y., Xu, Y., Yu, F., Liu, Q., Wu, S., Wang, L.: Deep graph contrastive representation learning. In: ICML (2020)
23. You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., Shen, Y.: Graph contrastive learning with augmentations. In: NeurIPS (2020)
24. Maurya, S.K., Liu, X., Murata, T.: Simplifying approach to node classification in graph neural networks. *J. Comput. Sci.* **62**, 101695 (2022)
25. Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L., Koutra, D.: Beyond homophily in graph neural networks: current limitations and effective designs. In: NeurIPS (2020)

26. Marcheggiani, D., Titov, I.: Encoding sentences with graph convolutional networks for semantic role labeling. In: EMNLP, pp. 1506–1515 (2017)
27. Rakhimberdina, Z., Liu, X., Murata, T.: Population graph-based multi-model ensemble method for diagnosing autism spectrum disorder. Sensors **20**(21), 6001 (2020)
28. Djenouri, Y., Belhadi, A., Srivastava, G., Lin, J.C.: Hybrid graph convolution neural network and branch-and-bound optimization for traffic flow forecasting. *Futur. Gener. Comput. Syst.* **139**, 100–108 (2023)
29. Choong, J.J., Liu, X., Murata, T.: Learning community structure with variational autoencoder. In: ICDM, pp. 69–78 (2018)
30. Pei, H., Wei, B., Kevin, C.-C.C., Yu, L., Yang, B.: Geom-GCN: Geometric graph convolutional networks. In: ICLR (2020)
31. Suresh, S., Li, P., Hao, C., Neville, J.: Adversarial graph augmentation to improve graph contrastive learning. In: NeurIPS (2021)
32. Liu, Y., et al.: Graph self-supervised learning: a survey. *IEEE Trans. Knowl. Data Eng.* **35**(6), 5879–5900 (2022)
33. Shchur, O., Mumme, M., Bojchevski, A., Günnemann, S.: Pitfalls of graph neural network evaluation. In: NeurIPS Workshop (2018)
34. Zhu, D., Zhang, Z., Cui, P., Zhu, W.: Robust graph convolutional networks against adversarial attacks. In: SIGKDD (2019)
35. Runwal, B., Kumar, S.: Robust graph neural networks using weighted graph Laplacian (2022). arXiv preprint [arXiv:2208.01853](https://arxiv.org/abs/2208.01853)
36. Wu, T., Ren, H., Li, P., Leskovec, J.: Graph information bottleneck. In: NeurIPS (2020)
37. Zhang, M., Chen, Y.: Link prediction based on graph neural networks. In: NeurIPS (2018)



Alleviating Over-Smoothing via Aggregation over Compact Manifolds

Dongzhuoran Zhou^{1,2(✉)}, Hui Yang³, Bo Xiong⁴, Yue Ma³,
and Evgeny Kharlamov^{1,2}

¹ Bosch Center for AI, Karlsruhe, Germany

{dongzhuoran.zhou, evgeny.kharlamov}@de.bosch.com

² University of Oslo, Oslo, Norway

³ LISN, CNRS, Université Paris-Saclay, Gif Sur Yvette, France

{yang,ma}@linsn.fr

⁴ University of Stuttgart, Stuttgart, Germany

bo.xiong@ipvs.uni-stuttgart.de

Abstract. Graph neural networks (GNNs) have achieved significant success in various applications. Most GNNs learn the node features with information aggregation of its neighbors and feature transformation in each layer. However, the node features become indistinguishable after many layers, leading to performance deterioration: a significant limitation known as over-smoothing. Past work adopted various techniques for addressing this issue, such as normalization and skip-connection of layer-wise output. After the study, we found that the information aggregations in existing work are all contracted aggregations, with the intrinsic property that features will inevitably converge to the same single point after many layers. To this end, we propose the aggregation over compacted manifolds method (ACM) that replaces the existing information aggregation with aggregation over compact manifolds, a special type of manifold, which avoids contracted aggregations. In this work, we theoretically analyze contracted aggregation and its properties. We also provide an extensive empirical evaluation that shows ACM can effectively alleviate over-smoothing and outperforms the state-of-the-art. The code can be found in <https://github.com/DongzhuoranZhou/ACM.git>.

Keywords: Graph Neural Network · Over-smoothing · Manifold

1 Introduction

Graph neural networks (GNNs) [42] are potent tools for analyzing graph-structured data, including biochemical networks [38], social networks [13], and academic networks [9]. Most GNNs employ a message-passing mechanism for

D. Zhou and H. Yang—Equal contribution.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-981-97-2253-2_31.

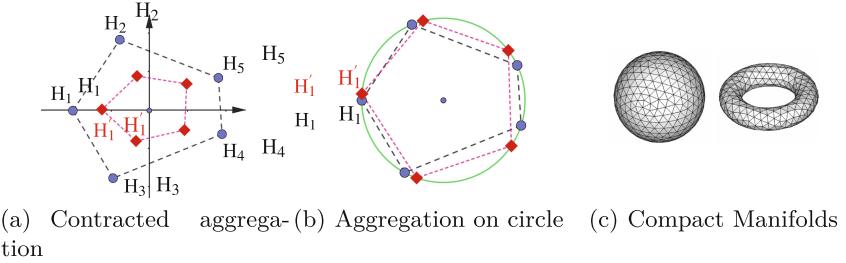


Fig. 1. (a) node features with *contracted aggregation* become closer and inevitably converge after many layers. (b) Our approach avoids contracted aggregation by aggregating on compact manifolds. (c) Compact manifolds: sphere and torus.

learning node features, involving information aggregation from neighbors and feature transformation in each layer [12]. This mechanism enables GNNs to effectively capture detailed information in graph data.

Though potential in various tasks, existing GNNs with many layers cause node representations to be highly indistinguishable, thus leading to significant performance deterioration in downstream tasks - a phenomenon known as over-smoothing [26]. Increasing efforts have been devoted to addressing the over-smoothing issue, such as batch norm [17], pair norm [41], group norm [43], drop edges or nodes [5, 32], and regularize the nodes [15] or feature distance [18]. Most of existing studies focus on alleviating the over-smoothing based on enhancing the distinction in node dimension [17, 41] or feature dimension [18] through normalization, adding regularization items [18], skip-connection [25, 39], etc.

In this work, we show that past works on over-smoothing are confined to *contracted aggregation* on Euclidean spaces, where features inevitably converge to a single point after multiple layers of information aggregations. The intuition of *contracted aggregation* is illustrated in Fig. 1a, using the plane \mathbb{R}^2 as the embedding space and the mean of vectors as the aggregation function. The aggregation results (red points) become closer than the initial embedding (blue points), causing nodes to converge to a single point after several steps. In contrast, Fig. 1b shows that after *aggregation over compact manifold* (compact manifold see Fig. 1(c)), node features do not become closer, thus avoiding the *contracted aggregation* issue¹. Even after numerous information aggregation steps, the aggregation results (red points) remain distinguishable and do not converge to a single point, unlike the Euclidean case.

In light of this, we perform an in-depth study on the over-smoothing in GNNs from the information aggregation perspective. We aim to bring the benefits of information aggregation on manifolds to alleviate over-smoothing. We develop a general theory to explain the reason for over-smoothing in Riemannian manifolds (including Euclidean spaces). Based on this theory, we propose *aggregation over compact manifolds* (ACM) to address the over-smoothing issue.

¹ In Fig. 1b, the unit circle is used as the embedding space. The aggregation of points is defined under *polar coordinates*. For example, the aggregation of two points in the unit circle with polar coordinate $(1, \theta), (1, \phi)$ is $(1, \frac{\theta+\phi}{2})$.

Our contributions can be summarized as follows:

- We propose the notion of *contracted aggregation* (Sect. 4) and prove that node features will converge to the same single point after many layers of contracted aggregation, leading to over-smoothing. We also prove that standard GNNs, e.g., SGC, GCN, GAT, are either *contracted aggregation* or mathematically equal. Our claim holds for embedding space of any Riemannian manifolds, including Euclidean Space.
- We propose *aggregation over compact manifold* (ACM) (Sect. 5) that can be integrated into each layer of GNNs models to prevent contracted aggregation, effectively mitigating over-smoothing across various GNN architectures.
- We provide extensive empirical evidence (Sect. 6) that shows the proposed ACM can alleviate over-smoothing and improve deeper GNNs with better performance compared to the state-of-the-art (SotA) methods. The improvement becomes more significant in a more complex experiment setting (i.e., missing feature) that requires massive layers for GNNs to achieve a good performance.

2 Related Work

Over-Smoothing in GNNs. Recent studies have shown that deep stacking of GNN layers can result in a significant performance deterioration, commonly attributed to the problem of over-smoothing [5, 41]. The over-smoothing issue was initially highlighted in [26], where the authors show that node embeddings will converge to a single point or lose structural information after an infinite number of random walk iterations. Their result has been extended to more general cases considering transformation layers and different activate functions by [3, 16, 29].

To address the over-smoothing issue and enable deeper GNNs, various methods have been proposed [13, 25, 26, 32]. One approach is to employ skip connections for multi-hop message passing, such as GraphSAGE [13] JKNet [26]. There are also several approaches developed based on the existing methods in other areas. For instance, the study of [25] leveraged concepts from ResNet [14] to incorporate both residual and dense connections in the training of deep GCNs; GCNII [6] utilizes initial residual and identity mapping. [32] proposed Dropedge that alleviates over-smoothing through a reduction in message passing by removing edges inspired by the use of Dropout [34]. Another mainstream approach is to normalize output features of each layer, such as Batch-Norm, PairNorm [41] and DGN [43]. DeCorr [18] introduced over-correlation as one reason for significant performance deterioration. APPNP [22] use Personalized PageRank and GPRGNN [7] use Generalized PageRank to mitigate over-smoothing. DAGNN [27] was introduced to create deep GNNs by decoupling graph convolutions and feature transformation. However, these works keep using the original information aggregation function and focus only on alleviating distinguishable after feature transformation.

GNNs on Manifolds. Recent attention in GNN research has been directed towards neural networks on *manifolds*, impacting various domains like knowledge

graph embedding, computer vision, and natural language processing [2, 11, 19]. Many works focus on the *hyperbolic space*, a Riemannian manifold with constant negative curvature, with [4] introducing Hyperbolic Graph Neural Networks (HGNN). [1] proposes a general GNN on manifolds applicable to hyperbolic space and hyperspheres. This work concentrates on GNN over compact manifolds, encompassing hyperspheres but not hyperbolic spaces. We introduce a novel GNN framework over compact manifolds. Unlike existing methods, our model doesn't depend on special non-linear functions, such as the *exponential* and *logarithmic* maps [8] defined in hyperbolic space and hyperspheres, making it computationally more straightforward. Additionally, our model is more general, defined on more general kinds of manifold spaces, including hyperspheres.

3 Preliminaries

Compact Metric Space. We start with a brief introduction to basic topology (more details can be found in literature [28]). A *metric space* \mathcal{S} is a set equipped with a distance function $d_{\mathcal{S}} : \mathcal{S} \times \mathcal{S} \rightarrow \mathbb{R}_{\geq 0}$ (i.e., a function that is positive, symmetric, and satisfying the triangle inequality). A sequence $p_1, p_2, \dots \in \mathcal{S}$ converge if $\lim_{k \rightarrow \infty} \max\{d_{\mathcal{S}}(p_i, p_j) \mid i, j \geq k\} = 0$. Metric space \mathcal{S} is *closed* if for every convergent sequence $p_1, p_2, \dots \in \mathcal{S}$ there exists $p \in \mathcal{S}$ such that $\lim_{i \rightarrow \infty} d_{\mathcal{S}}(p_i, p) = 0$. Metric space \mathcal{S} is *compact* if it is closed and *bounded*—that is, there exists a $r \in \mathbb{R}$ such that $d_{\mathcal{S}}(p, q) < r$ for every $p, q \in \mathcal{S}$. For example, the open interval $\mathcal{S}_{\text{oint}} = (0, 1) \subseteq \mathbb{R}$ is a metric space associated with distance $d_{\mathcal{S}_{\text{oint}}}(x, y) = |x - y|$. Then $2^{-1}, 2^{-2}, 2^{-3}, \dots \in \mathcal{S}_{\text{oint}}$ is a convergent sequence, but there is no point in $\mathcal{S}_{\text{oint}}$ that is the limit of this sequence. Therefore, $\mathcal{S}_{\text{oint}}$ is not closed. In contrast, a unit circle $\mathcal{S}_{\text{circ}}$ in \mathbb{R}^2 with $d_{\mathcal{S}_{\text{circ}}}(x, y) = \|x - y\|$ is closed, since $d_{\mathcal{S}_{\text{circ}}}(p, q) < 3$ for all $p, q \in \mathcal{S}_{\text{circ}}$, and compact.

Compact Manifolds. Next we provide a brief introduction to manifolds with the notions necessary for this paper (see further details in literature [24]). A d -dimensional *manifold* \mathcal{M} is a hyper-surface in the Euclidean space \mathbb{R}^n with $n \geq d$ such that each point has an (open) neighbourhood that is homeomorphic to an open subset of \mathbb{R}^d (i.e., locally looks like \mathbb{R}^d). A *Riemannian manifold* \mathcal{M} is a manifold along with a Riemannian metric, from which one can derive a distance function $d_{\mathcal{M}}(\mathbf{x}, \mathbf{y})$ for points $\mathbf{x}, \mathbf{y} \in \mathcal{M}$ thus making \mathcal{M} a metric space (so, we can talk about closed and compact Riemannian manifolds). A *compact manifold* is a Riemannian manifold that is also being a compact metric space. Some examples of compact manifolds in \mathbb{R}^3 are shown in Fig. 1(c). A differentiable map $f : \mathcal{M} \rightarrow \mathcal{M}$ is *diffeomorphism* if it is bijective and its inverse f^{-1} is a differentiable map.

Graph Neural Networks. A (undirected) graph $G = (V, E)$ is a pair of a nodes set V and a edges set E . Let $\mathcal{N}(u) = \{v \mid \{u, v\} \in E\}$ and $\tilde{\mathcal{N}}(u) = \mathcal{N}(u) \cup \{u\}$, $\mathbf{D} = \text{diag}(d_1, \dots, d_m)$ with $d_i = \sum_{j=1}^m \mathbf{A}_{i,j}$, where $\mathbf{A}_{i,j}$ denotes the elements of the adjacency matrix \mathbf{A} , as well as the augmented matrices $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ and $\tilde{\mathbf{D}} = \mathbf{D} + \mathbf{I}$, for $\mathbf{I} = \text{diag}(1, \dots, 1)$. We denote the collection of all (undirected) graphs as \mathcal{G} .

Given a graph G and an n -dimensional node embedding $\mathbf{H}^{(0)}$ of G , A Graph Neural Network (GNN) \mathcal{A} updates the embedding for ℓ layers as follows:

$$\mathbf{H}^{(k)} = \text{Tran}^{(k)} \left(\text{Agg}(\mathbf{H}^{(k-1)}) \right); \quad (1)$$

The layer of vanilla-GCN [21] is defined by $\text{Agg}(\mathbf{H}) = \tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \cdot \mathbf{H}$, $\text{Tran}^{(k)}(\mathbf{H}) = \sigma(\mathbf{H} \cdot \mathbf{W}^{(k)})$, where σ is activation function and $\mathbf{W}^{(k)} \in \mathbb{R}^{n \times n}$. GNN \mathcal{A} is over manifold \mathcal{M} in \mathbb{R}^n if its aggregation and transformation functions are over \mathcal{M} .

In this work, we consider arbitrary aggregation functions that are universally defined over the embedding of all neighborhoods $\tilde{\mathcal{N}}(u_i)$ in any graph $G \in \mathcal{G}$. Therefore, all aggregate-transform GNNs with specific aggregations are included, such as GCN [21] and GAT [36], as well as SGC [37].

4 Contracted Aggregation Problem

This section is organized as follows. First, in Sect. 4.1, we identify a special property, called *contracted*, of aggregation functions. We show that aggregation functions in GCNs, GATs, and SGCs are contracted or mathematically equivalent. Then, in Sect. 4.2, we demonstrate that contracted aggregations cause over-smoothing by means of Theorem 1. Inspired by this, in Sect. 4.3, we develop a general approach for constructing non-contracted aggregations for alleviating over-smoothing using aggregation over compact Riemannian manifolds (Theorem 2). Finally, in Sect. 4.4, we motivate and introduce our aggregation function defined over a specific kinds of compact Riemannian manifolds denoted by \mathcal{M}_U .

Next, we let $\mathcal{M} \subset \mathbb{R}^n$ be a d -dimensional closed Riemannian manifold, and let Agg be an aggregation function over \mathcal{M} . For each graph $G \in \mathcal{G}$ with nodes $\{u_1, \dots, u_m\}$, we denote by $\text{Agg}^G : \mathcal{M}^m \rightarrow \mathcal{M}^m$ the *restriction of Agg on G* defined by restricting Agg over embeddings of G on \mathcal{M} (i.e., $\mathbf{H} \in \mathcal{M}^m$).

4.1 Contracted Aggregation

The notion of contracted aggregation, which we introduce in this section, generalizes the usual mean (or avg) aggregation by extracting two center properties of the averaging function over Euclidean space \mathbb{R}^n : for all $\mathbf{x}, \mathbf{x}_1, \dots, \mathbf{x}_h \in \mathbb{R}^n$ we have (i) $\text{Mean}(\mathbf{x}_1, \dots, \mathbf{x}_h) = \mathbf{x}$ if $\mathbf{x}_1 = \dots = \mathbf{x}_h = \mathbf{x}$; and (ii) $d_{\mathbb{R}^n}(\mathbf{x}, \text{Mean}(\mathbf{x}_1, \dots, \mathbf{x}_h)) \leq \max(d_{\mathbb{R}^n}(\mathbf{x}, \mathbf{x}_1), \dots, d_{\mathbb{R}^n}(\mathbf{x}, \mathbf{x}_h))$. Figure 2 illustrates the second property.

Definition 1 (Contracted aggregation). For a graph G with nodes $\{u_1, \dots, u_m\}$, Agg^G is contracted if for any $1 \leq i \leq m$, and for any embedding $\mathbf{H} \in \mathcal{M}^m$ of G with $\bar{\mathbf{H}} = \text{Agg}^G(\mathbf{H})$, the following results hold:

1. If $\mathbf{H}_{j,:} = \mathbf{x}, \forall u_j \in \tilde{\mathcal{N}}(u_i)$, then $\bar{\mathbf{H}}_{i,:} = \mathbf{x}$.
2. For any $\mathbf{x} \in \mathcal{M}$, we have $d_{\mathcal{M}}(\mathbf{x}, \bar{\mathbf{H}}_{i,:}) \leq \max_{u_j \in \tilde{\mathcal{N}}(u_i)} \{d_{\mathcal{M}}(\mathbf{x}, \mathbf{H}_{j,:})\}$ and the equality holds if and only if Case 1 happens.

Moreover, we say Agg^G is equivalently contracted if there exists a diffeomorphism $g : \mathcal{M}^m \rightarrow \mathcal{M}^m$ s.t. $g^{-1} \circ \text{Agg}^G \circ g$ is contracted, where \circ means the map composition, i.e., $f \circ g(x) = f(g(x))$. Finally, an aggregation Agg is (equivalently) contracted if Agg^G is (equivalently) contracted for any graph $G \in \mathcal{G}$.

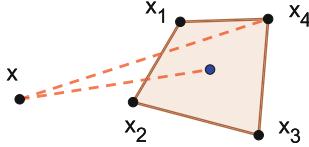


Fig. 2. Distance between \mathbf{x} and the mean point (blue) is smaller than $d_{\mathbb{R}^2}(\mathbf{x}, \mathbf{x}_4)$
(Color figure online)

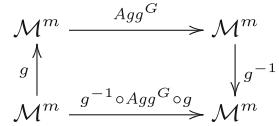


Fig. 3. Equivalently contracted aggregation

Intuitively, Agg^G is equivalently contracted if Agg^G is contracted after changing the “reference frame” by the equivalent transformation g on \mathcal{M}^m , which is the space of embedding matrix of G , as shown in Fig. 3. By the following result, we show that the standard aggregations used in SGC, GCN, and GAT are all contracted or equivalently contracted.

Proposition 1. Let $\mathcal{M} = \mathbb{R}^n$ ($n > 0$), assume Agg is a aggregation function on \mathcal{M} s.t. the restriction of Agg on a graph $G \in \mathcal{G}$ with node $\{u_1, \dots, u_n\}$ is defined by $\text{Agg}^G(\mathbf{H}) = \mathbf{L} \cdot \mathbf{H}$, $\forall \mathbf{H} \in \mathbb{R}^{m \times n}$, where $\mathbf{L} \in \mathbb{R}^{m \times m}$. Then we have the following results:

1. If $\mathbf{L} = (1 - \lambda)\mathbf{I} + \lambda\tilde{\mathbf{D}}^{-1}\tilde{\mathbf{A}}$ for some $\lambda \in (0, 1]$, then Agg is contracted;
2. If $\mathbf{L} = (1 - \lambda)\mathbf{I} + \lambda\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$ for some $\lambda \in (0, 1]$, then Agg is equivalently contracted;
3. If $\mathbf{L} = \text{Att}(\mathbf{H})$ is defined by a function $\text{Att} : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^{n \times n}$ s.t. $\mathbf{L}_{i,j} > 0$ and $\sum_k \mathbf{L}_{i,k} = 1$ for any $\mathbf{H} \in \mathcal{M}^m$ and $1 \leq i, j \leq m$, then Agg is contracted.

4.2 Over-Smoothing Due to Contracted Aggregations

Now, we show that contracted aggregations lead to over-smoothing by Theorem 1. This generalizes Theorem 1 of [26] to arbitrary equivalently contracted aggregations over Riemannian manifolds, from aggregations defined by *Laplace smoothing* (aggregations of Case 1 and 2 in Proposition 1) over Euclidean spaces.

Theorem 1. Let $\{\mathbf{H}^{(k)}\}_{k \geq 0}$ be an infinite sequence of embeddings of G over \mathcal{M} s.t. $\mathbf{H}^{(k+1)} = \text{Agg}^G(\mathbf{H}^{(k)})$, for all $k \geq 0$. Assume G is connected. If Agg^G is equivalently contracted wrt a diffeomorphism $g : \mathcal{M}^m \rightarrow \mathcal{M}^m$ (i.e., $g^{-1} \circ \text{Agg}^G \circ g$ is contracted), then there exist $\mathbf{X}_0 = (\mathbf{x}_0, \dots, \mathbf{x}_0) \in \mathcal{M}^m$ s.t. $\lim_{k \rightarrow \infty} \mathbf{H}^{(k)} = g(\mathbf{X}_0)$.

Note that Theorem 1 (proof in Appendix B.1) can be easily extended to non-connected graphs by considering their connected components. Theorem 1 shows that by repeatedly applying an equivalently contracted aggregation, the embedding points of nodes in the given graph converge to the same point (modulo the impact of the diffeomorphism g). That is, equivalently contracted aggregation leads to over-smoothing.

4.3 Constructing Non-contracted Aggregations

Next, we call an aggregation *non-contracted* if it is neither contracted nor equivalently contracted. Inspired by Theorem 1 and the example in Fig. 1, we propose to alleviate the over-smoothing problem by constructing non-contracted aggregations on manifolds. Indeed, by the following result, we show that any aggregation over a compact manifold is non-contracted. This result provides us with a general approach to constructing non-contracted aggregations.

Theorem 2. *Assume $\mathcal{M} \subset \mathbb{R}^n$ is a compact Riemannian manifold with dimension $d > 0$. If a graph G contains a node u_0 s.t. $|\tilde{\mathcal{N}}(u_0)| > 2$, then for any aggregation Agg over \mathcal{M} , the restricted aggregation Agg^G is non-contracted.*

For instance, the unit circle is compact, thus, all aggregation over the unit circle is non-contracted. An example of non-contracted aggregation that could avoid over-smoothing problems is shown in Fig. 1. In contrast, Euclidean spaces \mathbb{R}^n ($n > 0$) are not compact and thus can not benefit from the result of Theorem 2. The proof of Theorem 2 is in Appendix B.2.

4.4 Our Non-contracted Aggregation

To the best of our knowledge, there are mainly two kinds of aggregation defined over manifolds beyond Euclidean spaces. The *tangential aggregations* [4] and the *κ -Left-Matrix-Multiplication* [1]. However, the former aggregation is defined on *hyperbolic space*, which is not compact. In contrast, the second aggregation could be applied to *hyperspheres*, which are compact Riemannian manifolds. Their definition is based on the *gyromidpoint* introduced in [35] and uses *exponential* and *logarithmic* [8] maps defined on hyperspheres. In the following, we propose a simple way to construct aggregations over the specific family of compact manifolds denoted by $\mathcal{M}_{\mathbf{U}}$, which include hypersphere as a special case. Here, $\mathbf{U} \in \mathbb{R}^{n \times n}$ is a *positive-definite matrix*. That is, $\mathbf{x}\mathbf{U}\mathbf{x}^T > 0, \forall \mathbf{x} \in \mathbb{R}^n$.

Definition 2 (ACM). *Let $\mathbf{U} \in \mathbb{R}^{n \times n}$ be a positive-definite matrix. Consider the Riemannian manifold $\mathcal{M}_{\mathbf{U}} = \{\mathbf{x} = (x_1, \dots, x_n) \in \mathbb{R}^n \mid \mathbf{x}\mathbf{U}\mathbf{x}^T = 1\} \subset \mathbb{R}^n$. ACM aggregation function Agg over $\mathcal{M}_{\mathbf{U}}$ is defined as below. For any graph $G \in \mathcal{G}$ with m nodes, the restricted aggregation Agg^G has the form:*

$$\text{Agg}^G(\mathbf{H}) = P_{\mathbf{U}}(\mathbf{L} \cdot \mathbf{H}), \quad \forall \mathbf{H} \in \mathcal{M}^m,$$

where $P_{\mathbf{U}}(\mathbf{x}) = \frac{\mathbf{x}}{\sqrt{\mathbf{x}\mathbf{U}\mathbf{x}^T}}$, $\forall \mathbf{x} \in \mathbb{R}^n \setminus \{0\}$ (0 is the original point of \mathbb{R}^n), and $\mathbf{L} \in \mathbb{R}^{m \times m}$ takes one of the three forms as introduced in Proposition 1.

Since $\mathcal{M}_{\mathbf{U}}$ is compact, the above aggregation Agg is non-contracted (Theorem 2). For any graph $G \in \mathcal{G}$, Agg^G satisfies condition 1 of Definition 1 since $P_{\mathbf{U}}(k \cdot \mathbf{x}) = \mathbf{x}$ for any $k > 0, \mathbf{x} \in \mathcal{M} \subset \mathbb{R}_n$. Therefore, Agg can be regarded as a generalization of mean aggregations on compact manifold $\mathcal{M}_{\mathbf{U}}$. Moreover, $\mathcal{M}_{\mathbf{U}}$ is a hypersphere when \mathbf{U} is the identity matrix. Thus, our aggregation also works for hyperspheres. The complexity of computing $P_{\mathbf{U}}(\mathbf{x})$ is $O(n^2)$.

5 Aggregation over Compact Manifolds

Here, we introduce ACM (**A**ggregation over **C**ompact **M**anifolds), which integrates our aggregation function over compact manifolds $\mathcal{M}_{\mathbf{U}}$ (Definition 2) into standard SGC, GCN and GAT. Let $G \in \mathcal{G}$ be a graph with nodes $\{u_1, \dots, u_m\}$.

SGC. We integrate our aggregation into SGC by setting $\mathbf{L} = \tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}}$. Then, the embedding is updated by $\mathbf{H}^{(k)} = P_{\mathbf{U}} \left(\tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \cdot \mathbf{H}^{(k-1)} \right)$.

GCN. The adaptation of our aggregation on GCN is defined by Eqs. (2) below. The aggregation step of GCN is defined the same as above. Next, we show how to build a transformation function between the compact manifold $\mathcal{M}_{\mathbf{U}} \subset \mathbb{R}^n$.

Let $N_b = \{\mathbf{x} = (x_1, \dots, x_n) \in \mathbb{R}^n \mid x_1 = b\}$ be a hyperplane in \mathbb{R}^n . Then, our transformation function from $\mathcal{M}_{\mathbf{U}}$ to $\mathcal{M}_{\mathbf{U}}$ takes three steps:

$$\mathcal{M}_{\mathbf{U}} \xrightarrow{\text{Step 1}} N_b \xrightarrow{\text{Step 2}} \mathbb{R}^n \xrightarrow{\text{Step 3}} \mathcal{M}_{\mathbf{U}}.$$

In Step 1, we map $\mathcal{M}_{\mathbf{U}}$ to the hyperplane N_b using *push forward* (*PF*) function as illustrated in Fig. 4; Then, at Step 2, we map N_b to \mathbb{R}^n by the standard transformation functions (i.e., $\mathbf{x} \mapsto \sigma(\mathbf{x} \cdot \mathbf{W})$); In the last step, we map \mathbb{R}^n to the manifold $\mathcal{M}_{\mathbf{U}}$ by the *push back* (*PB*) function as shown in Fig. 4. Finally, our GCN layer update $\mathbf{H}^{(k-1)}$ to a new embedding $\mathbf{H}^{(k)}$ as below.

$$\overline{\mathbf{H}}^{(k)} = P_{\mathbf{U}} \left(\tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \cdot \mathbf{H}^{(k-1)} \right), \quad \mathbf{H}_u^{(k)} = PB \left(\sigma \left(PF(\overline{\mathbf{H}}^{(k)}) \cdot \mathbf{W}^{(k)} \right) \right), \quad (2)$$

where σ is an activate function, $\mathbf{W}^{(k)} \in \mathbb{R}^{n \times n}$. The functions *PF* and *PB* is a generalization of *Stereographic projection* and its inverse map from $\mathcal{M}_{\mathbf{U}}$ to N_b with center $\mathbf{x}_0 = (a_0, 0, \dots, 0) \in \mathcal{M}_{\mathbf{U}}$, where $a_0 = \mathbf{U}_{11}^{-\frac{1}{2}}$, as in Fig. 4.

GAT Our GAT layer is defined similarly to our GCN layer but with the weighted $\mathbf{L} = Att(\mathbf{H}^{(k-1)})$, where *Att* is the attention function introduced in [36].

Finally, following the works of [14, 26], for node classification task, we use *softmax* classifier in last layer of GNN to predict the labels of nodes in the given graph G .

6 Experiments

This section validates the following hypotheses: (1) ACM successfully mitigates over-smoothing in deep layers within GNNs, as evidenced by a reduced decline in

performance; (2) ACM effectively alleviate over-smoothing in a more challenging context, specifically the scenario of missing features.

6.1 Experiments Setup

Datasets. We use four homophilic datasets (Cora, Citeseer, Pubmed [40], and CoauthorCS [33]) and five heterophilic datasets (Texas, Cornell, Wisconsin, Actor, and Chameleon [30]). In the heterophilic datasets, the assumption of neighboring feature similarity is not applicable. Our focus is on standard transductive node classification tasks across these datasets. Details can be found in Appendix C.1.

Experiment Settings. We conducted node classification experiments in two settings [41]: the standard scenario and the *missing feature scenario*. The latter aims to highlight deep Graph Neural Networks' performance by assuming that initial node embeddings in the test and validation sets are absent, replaced with zero vectors. This reflects real-world scenarios, such as in social networks, where new users may have limited connections [31], requiring more information aggregation steps for effective representation.

Implementation. We consider three standard GNN models, namely GCN [21], GAT [36], and SGC [37], and augment them with techniques alleviating over-smoothing: PairNorm [41], BatchNorm [17], DropEdge [32], DGN [43], and DeCorr [18]. Moreover, we incorporate recent SOTA models, including GPRGNN [7], GCNII [6], DAGNN [27], and APPNP [23]. To analyze the impact of different methods, we vary the number of layers (5/60/120 for SGC, and 2/15/30/45/60 for GCN and GAT). By convention, SGC typically has more layers than GCN and GAT due to its simpler design and fewer parameters, requiring more layers for best performance and to effectively capture data details. We hence evaluate deeper models. Hyperparameter settings for GNN models and optimizers follow previous works [18, 41, 43]. The hidden layer size is set to 16 units for GCN, GAT, and SGC. GAT has 1 attention head. Training involves a maximum of 1000 epochs with Adam optimizer [20] and early stopping. GNN weights are initialized using the Glorot algorithm [10]. Each experiment is repeated 5 times, and mean values are reported. For ACM, we employ \tanh as the activation

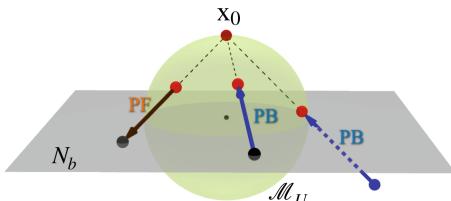


Fig. 4. Illustration of PB (orange) and PF (blue), points in \mathcal{M}_U (resp. N_b) are in red (resp. black). The formal definitions can be found in the supply material. (Color figure online)

function in hidden layers. We introduce two distinct variants of the ACM, differentiated by the configuration of the parameter matrix \mathbf{U} . (i) the first variant, still denoted as ACM, sets the matrix \mathbf{U} to the identity matrix \mathbf{I} ; (ii) the second variant, denoted as ACM*, defines the matrix \mathbf{U} as a positive definite diagonal matrix, where each diagonal element is a trainable parameter. Therefore, for ACM, the underlying manifold is the unit hypersphere, while ACM* offers a variable manifold (e.g., ellipsoid) that can be tailored through training to better suit the underlying data distribution.

6.2 Experiment Results

In this section, we compare the performance of ACM with previous methods that tackle the over-smoothing issue on SGC, GCN, and GAT. Refer to [18, 43] for all the results of the methods other than ACM unless stated otherwise.

Mitigate the Performance Drop in Deep SGC. SGC is a proper model to show the effectiveness of ACM in relieving over-smoothing. As SGC has only one feature transformation step, the performance of SGC highly depends on the information aggregation steps.

We present the performances of SGCs with 5/60/120 layers. Table 1 summarizes the results of applying different over-smoothing alleviation approaches to SGC. “None” indicates the vanilla SGC without over-smoothing alleviation. As shown in Table 1, ACM outperforms the other over-smoothing alleviation methods in most of the cases. In particular, on the Citeseer dataset with 120 layers, the improvements over None, BatchNorm, PairNorm, and DGN achieved by ACM are by a margin of 57.5%, 19%, 19.8%, and 14.3%, respectively.

To show the detailed performance differences on each layer, we illustrate in Fig. 5(a,b) the test accuracy of SGC trained with different methods with layers from 1 to 120 on datasets Cora and CoauthorCS. We can see that ACM behaved the best in slowing down the performance drop on Cora and comparably with the best system DGN on CoauthorCS. The same conclusion holds for other datasets as on Cora (see Appendix D.1).

Table 1. Node classification accuracy (%) on SGC. The two best performing methods are highlighted in **blue** (First), **violet** (Second).

Datasets	Cora			Citeseer			Pubmed			CoauthorCS		
#Layers	5	60	120	5	60	120	5	60	120	5	60	120
None	75.8	29.4	25.1	69.6	66.3	9.4	71.5	34.2	18.0	89.8	10.2	5.8
BatchNorm	76.3	72.1	51.2	58.8	46.9	47.9	76.5	75.2	70.4	88.7	59.7	30.5
PairNorm	75.4	71.7	65.5	64.8	46.7	47.1	75.8	77.1	70.4	86.0	76.4	52.6
DGN	77.9	77.8	73.7	69.5	53.1	52.6	76.8	77.4	76.9	90.2	81.3	60.8
ACM	78.5	80.0	77.0	65.0	67.4	66.9	76.9	78.3	78.6	90.9	80.1	57.4

Mitigation of the Performance Drop in Deeper GCN, GAT. We integrated our ACM method into GCN and GAT and measured the performance of

different GNNs with 30/45/60 layers in Table 2. The full table is in D.2. The results of None, BatchNorm, PairNorm, DGN at 45/60 layers were obtained by running the source code supplied by [43]. The results of Dropedge and DeCorr at 45/60 layers were collected by running the source code from [18].

From Table 2, we can see that ACM can greatly improve the performance of deeper GNNs on these datasets. In particular, given the same number of layers, ACM consistently achieves the best performance for almost all cases, dramatically outperforms recent baselines on deep models (30/45/60 layers), and keeps the comparable performance in the lower-layers model. For instance, on the Cora dataset, at 60 layers of GCN setting, ACM improves the classification accuracy of None, BatchNorm, PairNorm, Dropedge, DGN and DeCorr by 56.5% 11.1%, 22.5%, 56.5%, 16.9%, 56.5%, respectively.

Table 2. Node classification accuracy (%) on GCN, GAT.

Datasets		Cora			Citeseer			Pubmed			CoauthorCS		
Methods	#Layers	L30	L45	L60									
GCN	None	13.1	13	13.0	9.4	7.7	7.7	18.0	18	18.0	3.3	3.3	3.3
	BN	67.2	60.2	58.4	47.9	38.7	36.5	70.4	72.9	67.1	84.7	80.1	79.1
	PN	64.3	54.5	47.0	47.1	43.1	37.1	70.4	63.4	60.5	64.5	70	66.5
	DropEdge	45.4	13	13.0	31.6	7.7	7.7	62.1	18	18.0	31.9	3.3	3.3
	DGN	73.2	67.8	52.6	52.6	45.8	40.5	76.9	73.4	72.8	84.4	83.7	82.1
	DeCorr	73.4	38.9	13.0	67.3	37.1	7.7	77.3	32.5	13.0	84.5	29	3.3
	ACM	73.5	71.6	69.5	55.1	56.5	53.5	75.7	74.3	74.4	85.5	84.6	82.4
	ACM*	72.4	71.6	70.3	56.5	57.0	53.4	74.4	76.3	74.8	84.3	84.7	70.7
GAT	None	13.0	13	13.0	7.7	7.7	7.7	18.0	18	18.0	3.3	3.3	3.3
	BN	25.0	21.6	16.2	21.4	21.1	18.1	46.6	45.3	29.4	16.7	4.2	2.6
	PN	30.2	28.8	19.3	33.3	30.6	27.3	58.2	58.8	58.1	48.1	30.4	26.6
	DropEdge	51.0	13	13.0	36.1	7.7	7.7	64.7	18	18.0	52.1	3.3	3.3
	DGN	51.3	44.2	38.0	45.6	32.8	27.5	73.3	53.7	60.1	75.5	20.9	44.8
	DeCorr	54.3	18.3	13.0	46.9	18.9	7.7	74.1	48.9	18.0	77.3	19.2	3.3
	ACM	67.4	53.5	48.5	49.5	38.8	38.4	75.4	72.0	68.4	84.8	79.8	74.2
	ACM*	71.4	53.6	61.3	56.2	48.3	47.2	76.7	75.0	68.5	85.0	82.6	76.3

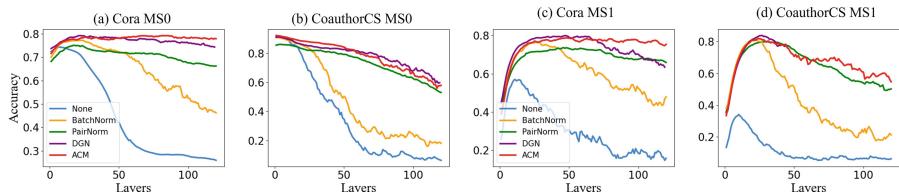


Fig. 5. Test accuracy of SGC with different methods and layers on Cora and CoauthorCS datasets under miss rate 0 (MS0) and miss rate 1 (MS1) scenario.

Table 3. Test accuracy (%) on missing feature setting.

Model	SGC				GCN				GAT			
Methods	Cora	Citeseer	Pubmed	CCS	Cora	Citeseer	Pubmed	CCS	Cora	Citeseer	Pubmed	CCS
None	63.4(5)	51.2(40)	63.7(5)	71(5)	57.3(3)	44(6)	36.4(4)	67.3(3)	50.1(2)	40.8(4)	38.5(4)	63.7(3)
BN	78.5(20)	50.4(20)	72.3(50)	84.4(20)	71.8(20)	45.1(25)	70.4(30)	82.7(30)	72.7(5)	48.7(5)	60.7(4)	80.5(6)
PN	73.4(50)	58(120)	75.2(30)	80.1(10)	65.6(20)	43.6(25)	63.1(30)	63.5(4)	68.8(8)	50.3(6)	63.2(20)	66.6(3)
DGN	80.2(5)	58.2(90)	76.2(90)	85.8(20)	67(6)	44.2(8)	69.3(6)	68.6(4)	67.2(6)	48.2(6)	67.2(6)	75.1(4)
ACM	80.7(90)	59.5(109)	76.3(91)	86.5(26)	76.3(20)	50.2(30)	72(30)	83.7(25)	75.8(8)	54.5(5)	72.3(20)	83.6(15)
ACM*	-	-	-	-	73.8(20)	49.1(30)	73.3(15)	84.3(20)	72.8(15)	46.5(6)	72.4(15)	83.7(15)

Table 4. Node classification accuracy (%) for eight datasets

Methods	Cora	Citeseer	Pubmed	Texas	Cornell	Wisconsin	Actor	Chameleon
GCN	82.3	71.7	78.6	67.2	57.4	56.0	26.7	43.3
GAT	81.2	69.5	77.4	71.6	63.6	64.1	27.8	48.1
DAGNN	84.4	73.3	80.3	78.6	68.1	71.3	32.4	47.9
APPNP	83.3	71.8	80.1	73.5	54.3	65.4	34.5	54.3
GPRGNN	83.0	71.3	71.5	78.7	76.7	71.2	37.1	48.8
GCNII	84.5	71.2	79.1	67.2	80.8	72.1	33.4	51.2
GPR+ACM	84.7	71.9	80.4	80.3	87.3	88.7	38.8	50.8

Enabling Deeper GNNs Under Missing Feature Setting. Table 1 and 2 show that deeper GNNs often perform worse than shallower ones. The “missing feature setting” experiments (see Sect. 6.1) executed the node classification task in more complex scenario. Such complex setting requires massive layers for GNNs to achieve good performance, for which ACM shows particular benefits since over-smoothing issue becomes severer with increasing layers.

In Table 3, Acc is the best test accuracy of a model with the optimal layer number #K. We observe that ACM outperforms the other over-smoothing alleviation methods in almost all cases. The average accuracy improvements achieved by ACM over None, BatchNorm, PairNorm, and DGN are 19.62%, 5.4%, 9.3%, and 1.18%, respectively. Moreover, the best performances were achieved with large layers. For instance, on the Cora dataset, ACM behaved the best with 90, 34, and 11 layers for SGC, GCN, and GAT, respectively, evidently higher than those widely-used shallow vanilla models, i.e., with 2 or 3 layers.

To better compare different methods with various layers, we run the experiments for all the layers from 1 to 120 on SGC. The result is illustrated by Fig. 5(c,d). We can see that ACM achieved best performance on deep SGCs on Cora and Citeseer. The same conclusion holds for other datasets (see Appendix D.1).

Combining with Other Deep GNN Methods. To ascertain whether ACM can function as a supplementary technique for alleviating over-smoothing, we choose to integrate it with one of the strongest baselines, GPRGNN. Our experiments on eight benchmark datasets, with the average accuracy reported for 10 random seeds in Table 4, shows that ACM consistently enhances GPRGNN

across all datasets. For example, ACM enhance the GPRGNN by 10.6% on Cornell and 17.5% on Wisconsin, respectively. This supports the idea that ACM can enhance model performance.

7 Conclusion

This paper proposed a general theory for explaining over-smoothing in Riemannian manifolds, which attributes the over-smoothing to the contracted aggregation problem. Inspired by this theoretical result, we proposed a general framework ACM—by constructing non-contracted aggregation functions on compact Riemann manifold—to effectively mitigate the over-smoothing and encourage deeper GNNs to learn distinguishable node embeddings. The effectiveness of ACM has been demonstrated through the extensive experimental results.

In the future, we aim to extend our theoretical findings to encompass transformation functions. Additionally, we intend to employ the ACM framework in tasks requiring deeper GNNs, such as multi-hop question answering.

References

1. Bachmann, G., Bécigneul, G., Ganea, O.: Constant curvature graph convolutional networks. In: ICML, pp. 486–496. PMLR (2020)
2. Balazevic, I., Allen, C., Hospedales, T.M.: Multi-relational poincaré graph embeddings. In: NeurIPS, pp. 4465–4475 (2019)
3. Cai, C., Wang, Y.: A note on over-smoothing for graph neural networks. CoRR abs/ [arXiv: 2006.13318](https://arxiv.org/abs/2006.13318) (2020)
4. Chami, I., et al.: Hyperbolic graph convolutional neural networks. In: NeurIPS, pp. 4869–4880 (2019)
5. Chen, D., Lin, Y., et al.: Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. CoRR abs/ [arXiv: 1909.03211](https://arxiv.org/abs/1909.03211) (2019)
6. Chen, M., Wei, Z., Huang, Z., Ding, B., Li, Y.: Simple and deep graph convolutional networks. In: ICML, pp. 1725–1735. PMLR (2020)
7. Chien, E., Peng, J., Li, P., Milenkovic, O.: Adaptive universal generalized pagerank graph neural network. In: ICLR (2021)
8. Ganea, O., Bécigneul, G., Hofmann, T.: Hyperbolic neural networks. In: NeurIPS, pp. 5350–5360 (2018)
9. Gao, H., Wang, Z., Ji, S.: Large-scale learnable graph convolutional networks. In: KDD, pp. 1416–1424. ACM (2018)
10. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: AISTATS, pp. 249–256 (2010)
11. Gülcöhre, Ç., et al.: Hyperbolic attention networks. In: ICLR (2019)
12. Hamilton, W.L.: Graph Representation Learning. Synthesis Lect. Artifi. Intell. Mach. Learn. (2020)
13. Hamilton, W.L., et al.: Inductive representation learning on large graphs. In: NIPS, pp. 1024–1034 (2017)

14. He, K., et al.: Deep residual learning for image recognition. In: CVPR, pp. 770–778. IEEE Computer Society (2016)
15. Hou, Y., Zhang, J., et al.: Measuring and improving the use of graph information in graph neural networks. In: ICLR (2020)
16. Huang, W., et al.: Tackling over-smoothing for general graph convolutional networks. CoRR (2020)
17. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: ICML, pp. 448–456 (2015)
18. Jin, W., Et al.: Feature overcorrelation in deep graph neural networks: a new perspective. In: KDD, pp. 709–719. ACM (2022)
19. Khrulkov, V., Et al.: Hyperbolic image embeddings. In: CVPR, pp. 6417–6427. Computer Vision Foundation/IEEE (2020)
20. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: ICLR (2015)
21. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. CoRR abs/ [arxiv: 1609.02907](https://arxiv.org/abs/1609.02907) (2016)
22. Klicpera, J., Et al.: Predict then propagate: graph neural networks meet personalized pagerank. In: ICLR (2019)
23. . Klicpera, J., et al.: Predict then propagate: graph neural networks meet personalized pagerank. In: ICLR (2019)
24. Lee, J.: Introduction to Smooth Manifolds. Graduate Texts in Mathematics
25. Li, G., Müller, M., et al.: Deepgcns: can gcns go as deep as cnns? In: ICCV, pp. 9266–9275. IEEE (2019)
26. Li, Q., Han, Z., Wu, X.M.: Deeper insights into graph convolutional networks for semi-supervised learning. In: AAAI, pp. 3538–3545 (2018)
27. Liu, M., Gao, H., Ji, S.: Towards deeper graph neural networks. In: KDD, pp. 338–348. ACM (2020)
28. Mendelson, B.: Introduction to topology (1990)
29. Oono, K., Suzuki, T.: Graph neural networks exponentially lose expressive power for node classification. In: ICLR (2020)
30. Pei, H., et al.: Geom-gcn: Geometric graph convolutional networks. In: ICLR (2020)
31. Rashid, A.M., Karypis, G., et al.: Learning preferences of new users in recommender systems: an information theoretic approach. SIGKDD Explor., 90–100 (2008)
32. Rong, Y., Huang, W., Xu, T., Huang, J.: Dropedge: towards deep graph convolutional networks on node classification. In: ICLR (2020)
33. Shchur, O., Mumme, M., Bojchevski, A., Günnemann, S.: Pitfalls of graph neural network evaluation. CoRR abs/ [arXiv: 1811.05868](https://arxiv.org/abs/1811.05868) (2018)
34. Srivastava, N., et al.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
35. Ungar, A.A.: Barycentric calculus in Euclidean and hyperbolic geometry: a comparative introduction (2010)
36. Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. CoRR abs/ [arXiv: 1710.10903](https://arxiv.org/abs/1710.10903) (2017)
37. Wu, F., et al.: Simplifying graph convolutional networks. In: ICML, vol. 97, pp. 6861–6871. PMLR (2019)
38. Xu, K., Hu, W., et al.: How powerful are graph neural networks? In: ICLR (2019)
39. Xu, K., Li, C., Tian, Y., et al.: Representation learning on graphs with jumping knowledge networks. In: ICML, pp. 5449–5458. PMLR (2018)
40. Yang, Z., et al.: Revisiting semi-supervised learning with graph embeddings. In: ICML. JMLR Workshop and Conference Proceedings, vol. 48, pp. 40–48 (2016)

41. Zhao, L., Akoglu, L.: Pairnorm: tackling oversmoothing in gnns. In: ICLR (2020)
42. Zhou, J., Cui, G., et al.: Graph neural networks: a review of methods and applications. *AI Open* **1**, 57–81 (2020)
43. Zhou, K., et al.: Towards deeper graph neural networks with differentiable group normalization. In: NeurIPS (2020)



Are Graph Embeddings the Panacea?

An Empirical Survey from the Data Fitness Perspective

Qiang Sun^(✉), Du Q. Huynh, Mark Reynolds, and Wei Liu^(✉)

The University of Western Australia, 35 Stirling Hwy, Crawley, WA 6009, Australia
pascal.sun@research.uwa.edu.au,
{du.huynh,mark.reynolds,wei.liu}@uwa.edu.au

Abstract. Graph representation learning has emerged as a machine learning go-to technique, outperforming traditional tabular view of data across many domains. Current surveys on graph representation learning predominantly have an algorithmic focus with the primary goal of explaining foundational principles and comparing performances, yet the natural and practical question “Are graph embeddings the panacea?” has been so far neglected. In this paper, we propose to examine graph embedding algorithms from a data fitness perspective by offering a methodical analysis that aligns network characteristics of data with appropriate embedding algorithms. The overarching objective is to provide researchers and practitioners with comprehensive and methodical investigations, enabling them to confidently answer pivotal questions confronting node classification problems: 1) Is there a potential benefit of applying graph representation learning? 2) Is structural information alone sufficient? 3) Which embedding technique would best suit my dataset? Through 1400 experiments across 35 datasets, we have evaluated four network embedding algorithms – three popular GNN-based algorithms (GraphSage, GCN, GAE) and node2vec – over traditional classification methods, namely SVM, KNN, and Random Forest (RF). Our results indicate that the cohesiveness of the network, the representation of relation information, and the number of classes in a classification problem play significant roles in algorithm selection.

Keywords: Network Characteristics · Graph embedding

1 Introduction

Graphs are vital for representing non-IID (Independent and Identically Distributed) data, such as social, citation, and traffic networks. Graph embedding techniques effectively interpret these structures through low-dimensional representations, thus becoming indispensable in contemporary machine learning pipelines.

Existing surveys [2,3,5,12,15] on graph embedding techniques primarily examine the following key areas: theoretical foundations of embedding algorithms (including deep learning, matrix factorization, and edge reconstruction

approaches), preservation of graph properties (such as first or second proximity) in the embedding space, the types of outputs produced by these algorithms (be it node, edge, whole-graph or sub-graph embeddings), the downstream problems (e.g., classification and link prediction), and the application domains (e.g., social networks, citation networks). With the sole focus on comparing graph embedding techniques, these surveys, to some extent, overlooked an important question: *Are graph embeddings the “panacea” for network data?* Intuitively, while some network data may respond well to graph embedding techniques, others may not.

The study by Makaro et al. [12] can be considered as an exception. However, the work was on limited datasets, all for citation networks (Cora and Citeseer) and their focus was on performance comparison over different downstream tasks such as node clustering, link prediction, and node classification. To the best of our knowledge, there are no systematic investigations on embedding techniques from the dataset fitness perspective despite the urgent needs in answering the following real-world applicability questions:

1. Is there a potential benefit of applying graph representation learning?
2. Would structural information alone be sufficient?
3. Which embedding technique would best suit my dataset?

In this research, we selectively collected a diverse range of public datasets used extensively in literature [4,6,8,10,11] and propose to use network characteristics as proxies to measure dataset fitness, so as to quantify the applicability of graph representation learning algorithms. Through 1400 experiments across 35 datasets, we have evaluated four network embedding algorithms: three popular GNN-based algorithms (GraphSage [8], GCN [13], GAE [10]) and a widely-adopted random-walk based technique (node2vec [6]), over traditional classification methods, namely SVM, KNN, and Random Forest (RF).

Our systematic analysis confirmed that the effectiveness of graph embedding techniques for node classification tasks is highly dependent on the network characteristics. In general, *sparsely connected networks* that have longer average shortest paths, relative large numbers of classes and fewer edges, graph representation learning is not beneficial. On the contrary, *dense and cohesive networks* with moderate numbers of classes see significant benefits from graph representation learning. Graphs that effectively represent connection information can perform well in node classification tasks by utilizing node2vec [6] without incorporating node attribute information. Notably, node2vec offers good performance for networks that lack node features, while supervised Graph Neural Networks (GNNs) are preferred in feature-rich networks. The number of edges, diversity of class labels, and average shortest path length of networks are identified as pivotal determinants in the effectiveness of graph embeddings techniques.

The novel and core contribution of this research therefore lies in the proposed data fitness measurement framework. Based on network characteristics matching with performance profiles, the framework facilitates the quantification of how applicable a graph representation learning algorithm over a dataset is, which yields critical insights and interpretable decision trees.

2 Literature Review

Node classification in homogeneous graphs is a task that assigns labels to nodes based on the graph topology and node attributes [2]. Graph representation learning is essential for node classification, as it effectively transforms the complex structure of graphs into low-dimensional representations, thereby converting the challenge of classifying nodes in graph structures into a more tractable tabular data classification problem. Node-level graph embedding techniques for homogeneous graphs have evolved from matrix factorization to deep learning-based approaches, driven by advancements in deep learning and the increasing complexity of graphs [2, 5, 15]. A typical categorisation for node embedding techniques is illustrated in Fig. 1. In the context of deep learning-based graph embeddings, methods are primarily categorized into two groups: *feature-less random walk-based* and *neighbourhood feature-based* techniques. Within the *neighborhood feature-based* category, depending on the downstream task (*extrinsic tasks* such as node classification, or *intrinsic tasks* such as edge reconstruction or graph reconstruction), these methods can be further divided into three distinct subcategories [2]. Note *intrinsic* tasks are also known as *in-graph tasks*, can be handled by *unsupervised learning* techniques because no external labels are required.

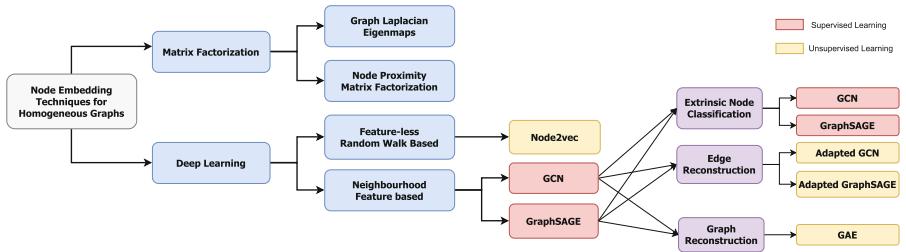


Fig. 1. Node Embedding Techniques for Homogeneous Graphs (Adapted from [2]).

Node2vec, GCN, and GraphSAGE listed in Fig. 1 are prominent deep learning models in node classification tasks. Based on random walks and inspired by SkipGram [7], node2vec [6] can effectively capture the structural context of nodes by balancing breadth-first (BFS) and depth-first searches (DFS) on the graph. *Graph Convolutional Networks* (GCNs) [11] use convolutional neural networks to generate node embeddings, aggregating features from the immediate neighbours of nodes. *Graph Sample and AggreGate* (GraphSAGE) extends this approach by sampling a fixed number of neighbours and employing techniques like averaging, LSTM, or pooling to aggregate features [12]. *Graph Autoencoders* (GAEs) [10] use an encoder-decoder framework, with GCN or GraphSAGE being the encoding layers, to learn node embeddings that minimize the graph reconstruction loss.

If we treat each node as an independent observation, focus solely on node attributes, and ignore network topology, then the classification task becomes one

that can be learned using a traditional method like Random Forest (RF), KNN, and SVM. In contrast, Node2vec primarily focuses on graph topology structure to learn node embeddings that can be used in the subsequent classification tasks. Meanwhile, GCN, GraphSAGE and GAE integrate both node attributes and structural topological information in their representation learning. GCN and GraphSAGE can directly perform node classification in a supervised manner, whereas GAE, similar to Node2vec, prioritize learning on node embeddings. GCN and GraphSAGE, on the other hand, possess the flexibility to adapt their loss functions to align with the approaches of Node2vec and GAE.

These models are ideal for our empirical study as they provide diversity for analyzing node features (attributes) and structure. We exclude matrix factorization methods due to scalability issues and the superior performance of deep learning models. We also limit our scope to node-level graph embedding techniques.

3 Network Characteristics of Datasets

Datasets with intrinsic connections, such as social networks which capture relationships between people, citation networks which map scholarly references, are typical examples of network datasets and can be described using the standard definition of graphs. There is also an increasing trend of processing non-networked data into graphs to capture relations between data points and attributes, e.g. for anomaly detection [14].

Definition 1. A **graph** is defined as $\mathcal{G} = \langle V, E \rangle$, where a node $v \in V$ and an edge $e \in E$. Edge $e_{ij} \in E$ stands for an edge between node $v_i, v_j \in V$.

In **graph theory**, diverse measures have been used to analyze structural characteristics and connectivity of networks and to capture network topological features from multiple perspectives. Let k be the random variable that describes the degrees of nodes in a graph. Then the *average node degree* (\bar{k}) represents the average number of edges for all the nodes. Other statistics of k that offer insights to the graph's density and degree distribution are also commonly used. The *distance* between two nodes in a graph is defined to be the number of edges in a shortest path connecting them. The *average shortest path length* (L) therefore quantifies the average node distance; in terms of information flow, this measure is important for measuring network efficiency. The *diameter* (D), which gives the longest distance between all node pairs, provides the upper bound on the reachability of nodes in a graph. The *betweenness centrality*, based on the number of shortest paths passing through a node, measures its criticality in forming sub-networks. The *average clustering coefficient* (C) measures how many nodes, on average, cluster together, while *transitivity* (T) looks at triangle connections among nodes. In scale-free networks [1], the distribution of node degrees follows a power law: $P(k) \sim k^{-\gamma}$, where the *degree exponent* γ quantifies whether the network has large hubs or not (i.e., having a few nodes with large degrees (for small γ) or many nodes with small degrees (for large γ)). The definitions of these measures are summarised in Table 1.

Table 1. Network characteristics: notations and definitions

$ V $	Total number of nodes	$ E $	Total number of edges
d	Dimension of node features	K	Number of classes
\bar{k}	Average degree of nodes ($= 2 E / V $)	k_{var}	Second moment of degree distribution
k_{\min}	Minimum node degree	k_{\max}	Maximum node degree
L	Average shortest path between all node pairs	D	Diameter (The maximum distance between all possible pairs of nodes)
T	Transitivity (measuring likelihood of triangle formation)	C	Average clustering (quantifying the tendency of nodes to cluster together)
γ	Degree exponent of the power-law degree distribution, $P(k) \sim k^{-\gamma}$		

Table 2. Dataset Context, where each number inside the parentheses denotes the number of datasets for that given network type or dataset topic.

Network Type	Dataset topic	Description
Co-existence Network (7)	Actor (1)	Wikipedia co-occurrence of actors; Classify into categories.
	Amazon (3)	Product co-purchases, bag-of-words from reviews; Product/review categories.
	Coauthor (CS, Physics) (2)	Authorship network, paper keywords; Study fields classification.
	Tolokers (1)	Toloka worker data, shared tasks; Banned worker prediction.
Citation Network (15)	Cora, etc. (12)	Publications, citations, bag-of-words; Publication classes.
	WebKG (3)	University web pages, hyperlinks, bag-of-words; Page categories.
Social Network (8)	Twitch (6)	Streamers, mutual follows, game embeddings; Language use classification.
	BlogCatalog (1)	Blog platform users and friendships, TF-IDF from blogs; User categories.
	Github (1)	Developer relationships, locations, repos; Web/ML developer classification.
Knowledge Graph (2)	Wiki (1)	Wikidata graph, item relations, one-hot vectors; Item categories.
	Roman Empire (1)	Wikipedia articles, word connections, embeddings; Syntactic roles classification.
Social Knowledge Network (2)	Questions (1)	Yandex Q dataset, answered questions, user profile embeddings; Active user prediction.
	Flickr (1)	Users, images, metadata, text annotations; Tag-based classification.
Grid (1)	Minesweeper (1)	Grid cells, adjacent mines; Mine presence prediction.

3.1 Dataset Selection

The datasets commonly used in the four graph embedding techniques discussed in Sect. 2 are mostly citation networks, e.g. Cora, CiteSeer, and PubMed, as well as knowledge and social networks including BlogCatalog, Wikipedia, NELL, and Reddit [6, 8, 10, 11], which are lack diversity. In contrast, our study adopts a **data first** perspective, examining a broader spectrum of public datasets.

After a comprehensive review of the literature, we compiled 35 datasets for node classification tasks, excluding larger ones like Reddit and focusing on homogeneous networks for ease of analysis. These datasets encompass various types and sizes of networks. These network types (see Table 2) include: (i) Co-existence networks (such as Actor and Amazon), where nodes are connected by shared contexts, (ii) Citation networks (including CiteSeer where connections are formed through citations, and WebKG which associates nodes via hyperlink references), (iii) Social networks e.g. BlogCatalog, Github, and Twitch, based on social interactions such as friendships or followings), (iv) Knowledge Graph networks (represented by Roman Empire and Wiki datasets, illustrating structured knowledge in graph form), and (v) a Grid network (Minesweeper, where nodes are connected based on grid adjacency).

3.2 Summary

The network characteristics for the 35 datasets are shown in Table 3. We can see that the characteristics span a wide range of values: $|V|$ (183 to 48,921), $|E|$ (298 to 519,000), d (1 to 12,047), K (2 to 70), \bar{k} (2.52 to 88.28), and k_{var} (9.50 to

Table 3. Dataset Statistics, where the figures in bold under each column denote the smallest or largest number in that column (see Table 2 for the colour codes).

Dataset	$ V $	$ E $	d	K	\bar{k}	k_{var}	k_{min}	k_{max}	L	C	T	D	γ
Actor	7,600	30,019	932	5	7.90	400.56	1	1,304	4.11	0.05	0.04	12	2.81
AZ_COMPUTERS	13,752	245,861	767	10	35.76	6,221.40	0	2,992	3.38	0.34	0.10	10	2.83
AZ_PHOTO	7,650	119,081	745	8	31.13	3,204.10	0	1,434	4.05	0.40	0.17	11	2.92
AGD_BlogCatalog	5,196	171,743	8,189	6	66.11	7,376.53	5	769	2.51	0.12	0.08	4	4.06
AGD_CiteSeer	3,312	4,715	3,703	6	2.85	19.81	1	100	1.08	0.07	0.03	28	3.31
AGD_Cora	2,708	5,429	1,433	7	4.00	44.23	1	169	1.17	0.13	0.02	19	3.05
AGD_Flickr	7,575	239,738	12,047	9	63.30	21,303.96	1	1,881	2.41	0.33	0.10	4	2.76
AGD_Pubmed	19,717	44,338	500	3	4.50	75.51	1	171	6.34	0.03	0.01	18	4.20
AGD_Wiki	2,405	16,523	4,973	17	13.74	499.01	1	281	3.65	0.32	0.44	9	3.82
CF_CiteSeer	4,230	5,337	602	6	2.52	20.43	1	85	1.35	0.11	0.08	26	2.83
CF_Cora	19,793	63,421	8,710	70	6.41	118.33	1	297	1.16	0.26	0.13	23	3.38
CF_Cora_ML	19,793	63,421	8,710	70	6.41	118.33	1	297	1.16	0.26	0.13	23	3.38
CF_DBLP	17,716	52,867	1,639	4	5.96	123.03	1	339	1.16	0.13	0.10	34	3.23
CF_PubMed	19,717	44,324	500	3	4.49	75.43	1	171	6.34	0.06	0.05	18	4.17
CiteSeer	3,327	4,552	3,703	6	2.73	18.92	0	99	1.08	0.14	0.13	28	2.63
Coauthor_CS	18,333	81,894	6,805	15	8.93	162.75	1	136	5.42	0.34	0.18	24	5.18
Coauthor_Physics	34,493	247,962	8,415	5	14.38	449.23	1	382	5.16	0.37	0.18	17	4.88
Cora	2,708	5,278	1,433	7	3.89	42.53	1	168	1.17	0.24	0.09	19	2.98
Cora_Full	19,793	63,421	8,710	70	6.41	118.33	1	297	1.16	0.26	0.13	23	3.38
GitHub	37,700	289,003	128	2	15.33	6,761.61	1	9,458	3.25	0.17	0.01	11	2.54
HGD_AZ_Ratings	24,492	93,050	300	5	7.60	93.39	5	132	16.24	0.29	0.15	46	3.60
HGD_Minesweeper	10,000	39,402	7	2	7.88	62.44	3	8	46.67	0.22	0.33	99	2.79
HGD_Questions	48,921	153,540	301	2	6.28	774.50	1	1,539	4.29	0.02	0.01	16	1.83
HGD_Roman_empire	22,662	32,927	300	18	2.91	9.50	2	14	2331.56	0.19	0.28	6,824	6.34
HGD_Tolokers	11,758	519,000	10	2	88.28	33,978.74	1	2,138	2.78	0.27	0.11	11	3.08
PubMed	19,717	44,324	500	3	4.50	75.43	1	171	6.34	0.06	0.05	18	4.18
TWITCH_DE	9,498	162,636	128	2	34.25	8,363.44	3	4,261	2.72	0.20	0.05	7	2.58
TWITCH_EN	7,126	42,450	128	2	11.91	634.28	3	722	3.68	0.13	0.04	10	2.79
TWITCHLES	4,648	64,030	128	2	27.55	3,198.91	3	1,024	2.88	0.22	0.08	9	2.58
TWITCH_FR	6,551	119,217	128	2	36.40	7,328.28	2	2,042	2.68	0.22	0.05	7	2.61
TWITCH_LPT	1,912	33,211	128	2	34.74	4,324.69	3	769	2.53	0.32	0.13	7	2.53
TWITCH_RU	4,385	41,689	128	2	19.01	2,098.57	3	1,231	3.02	0.17	0.05	9	2.58
WEBKB_Cornell	183	298	1,703	5	3.26	60.52	1	94	3.20	0.10	0.01	8	3.09
WEBKB_Texas	183	325	1,703	5	3.55	75.22	1	104	3.03	0.11	0.01	8	2.62
WEBKB_Wisconsin	251	515	1,703	5	4.10	81.85	1	122	3.26	0.11	0.01	8	2.50

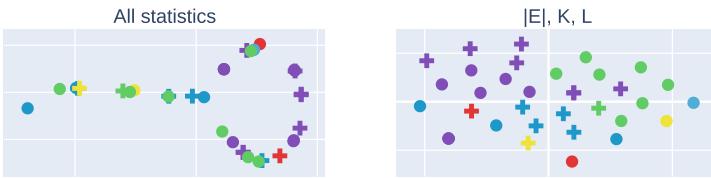


Fig. 2. t-SNE plots of network types (see Table 2) with 13 graph metrics and specific metrics (—E—, K, L). The “+” symbols indicate datasets that do not benefit from graph representation learning.

33,978.74). Among these, HGD_Roman_Empire has the lowest k_{var} but highest D , L , and γ . In contrast, HGD_Questions has the most number of nodes, yet fewest classes, smallest C , and lowest γ . The statistics of each dataset, represented as vectors and visualized through a t-SNE plot (Fig. 2), highlight the relationships and cluster groups of the network characteristics. Both co-existence networks

and social networks have more similar network characteristics and separated from the clusters of Knowledge Graph and Citation Networks.

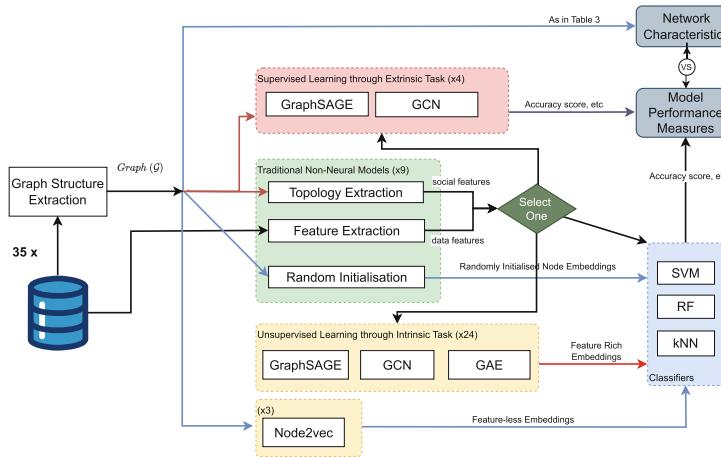


Fig. 3. Experiment Architecture

4 Experiments, Results and Discussions

4.1 Experiments

To answer the dataset fitness questions raised in the Abstract and Sect. 1, we designed a comprehensive set of experiments. Figure 3 shows that each dataset undergoes a Graph Structure Extraction phase, wherein the network characteristics (see Table 1) are computed. The Topology Extraction phase then extracts the *social features* [*betweenness*, *degree*] centrality for each node in the graph, while the Feature Extraction phase retrieves the d -dimensional attribute-based data features from the dataset. In addition, we initialize nodes within the graph using random vectors and labels, feed them into SVM, RF and KNN to establish the baseline for the classification task (**Random Guess**). The graph G that contains the node and edge information is combined with either the *data features* (node attributes) or the *social features*, [*betweenness*, *degree*]. These combined inputs are then passed to either a supervised graph embedding technique (**Supervised GCN/GraphSAGE**) or an unsupervised (**Node2vec; Unsupervised GCN/GraphSAGE/GAE**) graph embedding technique. The former techniques learn the embeddings specifically from the extrinsic task of node classifications, where the class labels are provided by the datasets. For the latter techniques, the learned node embeddings are used directly as inputs for the traditional classification models, namely, SVM, RF, and KNN. The data

features (**Feature Only**) and social features [**betweenness**, **degree**] are also independently fed into these classification models.

Overall, our experiments are categorized into three main types: 4 supervised learning graph embedding learning experiments through an **extrinsic task** (with class labels used in the training), 9 experiments of traditional non-neural models (using data features, centrality features, or random vectors as graph embeddings for SVM, RF, or KNN), and 27 experiments of unsupervised graph embedding learning through intrinsic tasks. The total is 1400 experiments, with 40 experiments per dataset. Classification measures from the 40 experiments per dataset and the network characteristics across the 35 datasets are analyzed to assess the models' performances and efficacies. We first find the optimal hyperparameters for SVM, KNN, and RF models using the **Cora** dataset, then apply these consistently across all datasets to ensure uniform experiment process. Datasets and Codes are available: <https://github.com/PascalSun/PAKDD-2024>.

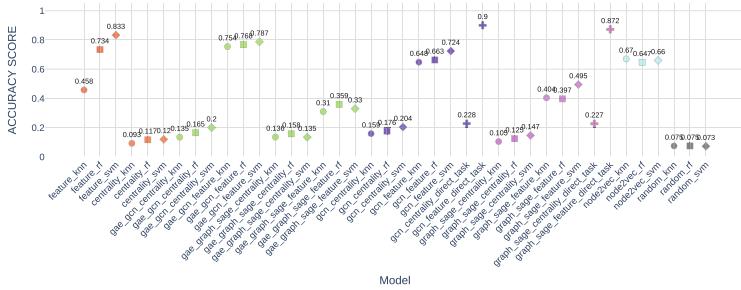


Fig. 4. The performance vector on accuracy for the **Coauthor_CS** dataset.

4.2 Results and Discussions

For each dataset, the models' performance measures are compiled into a 40-dimensional vector, giving $\mathbf{v}_{\text{per}} \triangleq (\text{per}_1, \dots, \text{per}_{40})$, where all the elements in \mathbf{v}_{per} are either F1 scores or accuracies. For example, Fig. 4 shows the performance vector for accuracy for the **Coauthor_CS** Dataset. In addition, the 13-dimensional vector, $\mathbf{v}_{\text{net}} \triangleq (\text{NC}_1, \dots, \text{NC}_{13})$, that captures the network characteristics (see Table 1) for each dataset is also extracted. Analyzing these vectors across all datasets provides insights into the interplay between network characteristics and the efficacy of graph embedding techniques.

Q1: Do all datasets benefit from graph structure representation learning?

To determine the benefits of graph structure in representation learning, we focus on comparing the components of \mathbf{v}_{per} for each dataset. We compared **Feature Only** models with those integrating node features and graph structure, noting that a superior performance by the latter would indicate a benefit from graph structures. Our study showed that 21 of 35 datasets, including **Minesweeper**,

Questions, **Roman Empire**, **Tolokers**, **Twitch**, **WebKG**, and **Github**, as well as most citation networks, benefited from this approach, with supervised learning models often outperforming others. In contrast, datasets such as **Actor**, **Amazon Ratings**, **Wiki**, **BlogCatalog**, **Flickr**, and **PubMed**, found the **Feature Only** models to be more effective. Specifically, the **PubMed** dataset, tasked with classifying diabetes-related literature into one of three classes, displayed the largest average shortest path (L) in the citation networks (a very sparse network), which may explain its unique status as an exception. In summary, our findings reveal that graph structure representation learning is not always advantageous; for certain datasets, “Feature Only” models suffice and outperform.

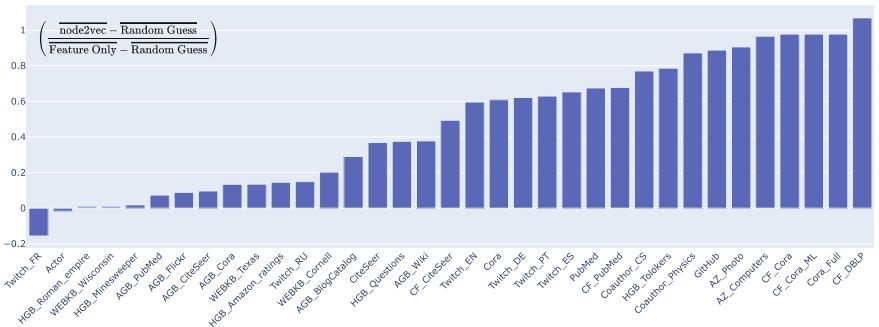


Fig. 5. Comparison of classification accuracies of the **node2vec** models versus the **Feature Only** models for all the datasets (the bar symbol represents the average accuracy of the model over the SVM, RF, and KNN classifiers).

Q2: Is structural information alone sufficient?

If we compare models that use only structural information (the graph \mathcal{G} and/or [betweenness, degree], **Structure Only**) with those that integrate data features and structural information, the latter consistently outperforms the former, signifying the need for both feature and structural data. This finding aligns with our intuitive expectations. If we adopt a different perspective and compare **Feature Only** models with **Structure Only** models, we then found that for certain datasets, the performance of node2vec models, which solely rely on structural information, is comparable to that of feature-only models. To quantify this observation, we calculate the ratio of accuracy improvement of node2vec models over **Random Guess** compared to feature-only models against the same baseline, using the average accuracy from three classical classifiers (see Fig. 5). For **Cora series**, **Amazon Photo**, **Amazon Computers**, **Github**, the performances of **Feature Only** and **node2vec** models are comparable and both significantly outperform the **Random Guess**. However, for the datasets **TWITCH_FR**, **Actor**, **Roman Empire**, **WebKG_Wisconsin** and **Minesweeper**, **Structure Only** models show no improvement over **Random Guess**. While in the remaining datasets, **node2vec** models exhibit reasonable performance gains over **Random Guess**,

but not to the extent of **Feature Only** models. Interestingly, **Amazon Ratings**, **Amazon Photos**, and **Amazon Computers**, all from the same co-existence network category, show varied diameters of 46, 10, and 11 respectively. The latter two networks, having shorter diameters, achieve notably better results for **Structure Only** models. Conclusively, our analysis shows that structural information alone is not sufficient for optimal model performance. However, the random walk-based node2vec approach can achieve comparable results to **Feature Only** models for dense networks. So, node2vec would be useful for dense but feature-less networks.

Q3: Which model(s) suit my dataset?

A classical approach to answer this question is to run various models on a given dataset and pick the best performing model according to a set of performance metrics. We have done this for each of the 35 datasets over 40 models. However, this is an expensive exercise and not viable for large complex datasets. On the other hand, computing the network characteristics is more attainable in terms of computational costs. This is especially useful at the data exploration stage, multiple graph models representing the same dataset can be constructed and compared without having to go through the modelling stage. Therefore we propose to find the best combination of network characteristics (i.e., a subset \mathbf{v}'_{net} of \mathbf{v}_{net}) that correlate well with the model performance. We can then infer model performance based on the dominate network characteristics.

Computationally, we formulate this as searching for \mathbf{v}'_{net} that minimise the differences between *network characteristic similarity matrix* M_{net} and the *performance similarity matrix* M_{per} , where M_{net} in this case is a 35×35 matrix calculated based on pairwise cosine similarity of the dataset network characteristics vectors $\{\mathbf{v}_{\text{net}}\}$, and M_{per} is a pairwise cross dataset 35×35 similarity matrix based on 35 performance vectors, $\{\mathbf{v}_{\text{per}}\}$, one for each of the 35 datasets. Figure 6 shows the heatmaps (M_{per}) of pairwise similarities across all 35 datasets on the 40-dimensional vectors of F1 score, accuracy, and one of the network characteristics similarity matrices (M_{net}) for the $\mathbf{v}'_{\text{net}} \triangleq (|E|, K, L)$ subset.

To identify a subset of 13 network characteristics, represented by the vector \mathbf{v}'_{net} with a dimension n ($0 < n \leq 13$), that generates a 35×35 similarity matrix closely resembling the performance similarity matrix, we first need to measure the closeness between two heatmap matrices. Instead of going for numerical distance and similarity measures, the heatmaps' visual structural similarity is more important, we therefore choose to utilize the Structural Similarity Index Measure (SSIM) [9], a metric developed for image quality assessment. SSIM, by evaluating structural information of the heatmap as images, luminance, and contrast variations, is apt for this analysis as it prioritizes structural similarities over individual value discrepancies. An SSIM value close to 1 indicates higher similarity.

We conducted a search of all combinations of the 13 network characteristics, and found that the following combinations emerged as the most prominent: $(|E|, K, L)$ in Fig. 6 (right), with an SSIM of approximately 0.77. Similarly, the combination of $(|E|, K, D)$ also yields an SSIM score of 0.77. Comparing the



Fig. 6. Similarity matrices for: F1 score (left), Accuracy score (middle), Network Parameters ($|E|$, K , L) (right).

F1 score heatmap and the Accuracy heatmap, we get an SSIM of 0.95. Visually, the heatmaps in Fig. 6 (left) and (middle) do closely resemble each other.

To deepen our understanding of the previous two questions within the framework of ($|E|$, K , L), we classify the datasets into two categories respectively based on the observations in **Q1** and **Q2**. For **Q2**, datasets are categorized based on the threshold value of 0.5 in Fig. 5. This division distinguishes between datasets that achieve relatively good performance solely with structural information and those that do not. We then utilize the decision tree classifier, with ($|E|$, K , L) as inputs, for binary classification. All datasets were used in the training process. The findings from this analysis are also depicted in Fig. 7. For Q1, the accuracy score is 0.942, with feature importances 0.13, 0.45, and 0.41 for $|E|$, K , and L , respectively. We can see that K and L play a more crucial role when deciding whether a dataset benefits from graph representation learning or not. For Q2, the accuracy score is 0.88, with feature importances 0.895, 0, and 0.1 for $|E|$, K , and L . As shown in Fig. 7 (b), a graph with adequately represented connections alone can be sufficient for node classification.

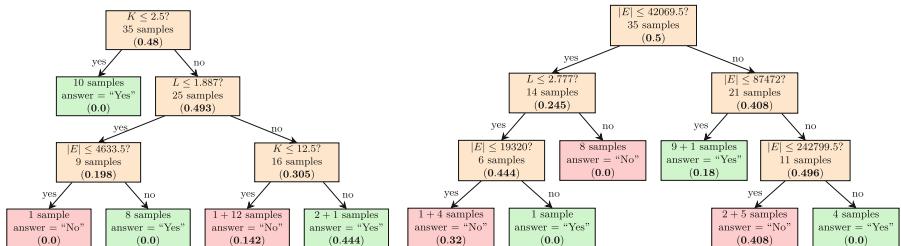


Fig. 7. Decision Trees for Q1 and Q2 based on $\mathbf{v}'_{\text{net}} = (|E|, K, L)$. The boldface numbers inside parentheses denote the Gini indices. Each leaf node is coloured in green (or pink) for the “Yes” (or “No”) answer to the question. (Color figure online)

5 Conclusion and Future Work

This paper proposes to use network characteristics to quantify data fitness in graph representation learning algorithm selection. We have methodically and empirically identified a clear relationship between network characteristics and the performance of graph embedding techniques. We observe that not all datasets benefit from graph representation in node classification tasks. Additionally, even for datasets that do benefit, relying solely on structural information is insufficient, as the contribution of structural information tends to be relatively minor compared to node attribute information. In particular, sparse networks with long average shortest path lengths, large number of classes and limited number of edges are generally unsuitable for classification tasks via graph representation learning. Whereas for densely well-connected networks with rich edge information, random walk-based models (e.g. node2vec) demonstrate notable performance compared to attribute-only models. However, the most effective ones are neighbour-attribute-based supervised learning methods, which consistently outperform others by leveraging both structural and attribute information.

To proceed into the future, firstly, the dominance of citation networks urgently call for wider domains to improve **dataset diversity**. Secondly, the reliance on bag-of-words and one-hot encoding for textual attributes could benefit from semantic enriched content embeddings through Large Language Models (LLM). Last but not least, while random walk-based node2vec demonstrates impressive performance, our non-random walk-based models utilize only two layers. Further study on **deeper random-walk inspired networks** is needed.

References

1. Barabási, A.L.: Network Science by Albert-László Barabási. Cambridge University Press (2016). <http://networksciencebook.com/>
2. Cai, H., Zheng, V.W., Chang, K.C.C.: A comprehensive survey of graph embedding: problems, techniques, and applications. *IEEE Trans. Knowl. Data Eng.* **30**(9), 1616–1637 (2018) <https://doi.org/10.1109/TKDE.2018.2807452>
3. Chen, S., Huang, S., Yuan, D., Zhao, X.: A survey of algorithms and applications related with graph embedding. In: Proceedings of the 2020 International Conference on Cyberspace Innovation of Advanced Technologies, pp. 181–185. ACM, Guangzhou China (2020). <https://doi.org/10.1145/3444370.3444568>
4. Fey, M., Lenssen, J.E.: Fast graph representation learning with PyTorch Geometric. In ICLR Workshop on Representation Learning on Graphs and Manifold [arXiv:1903.02428](https://arxiv.org/abs/1903.02428) (2019)
5. Goyal, P., Ferrara, E.: Graph embedding techniques, applications, and performance: a survey. *Knowl.-Based Syst.* **151**, 78–94 (2018). <https://doi.org/10.1016/j.knosys.2018.03.022>
6. Grover, A., Leskovec, J.: Node2vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864. ACM (2016). <https://doi.org/10.1145/2939672.2939754>

7. Guthrie, D., Allison, B., Liu, W., Guthrie, L., Wilks, Y.: A closer look at skip-gram modelling. In: Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06) (2014)
8. Hamilton, W.L., Ying, R., Leskovec, J.: Inductive representation learning on large graphs (2018) [arXiv:1706.02216](https://arxiv.org/abs/1706.02216) [cs, stat]
9. Hore, A., Ziou, D.: Image quality metrics: PSNR vs SSIM. In: 2010 20th International Conference on Pattern Recognition, pp. 2366–2369. IEEE (2010) <https://doi.org/10.1109/ICPR.2010.579>
10. Kipf, T.N., Welling, M.: Variational graph Auto-encoders (2016) <https://doi.org/10.48550/arXiv.1611.07308>
11. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: ICLR (2017)
12. Makarov, I., Kiselev, D., Nikitinsky, N., Subelj, L.: Survey on graph embeddings and their applications to machine learning problems on graphs. PeerJ Comput. Sci. **7**, e357 (2021). <https://doi.org/10.7717/peerj-cs.357>
13. Pei, H., Wei, B., Chang, K.C.C., Lei, Y., Yang, B.: Geom-GCN: geometric graph convolutional networks (2020) <https://doi.org/10.48550/arXiv.2002.05287>
14. Senaratne, A., Christen, P., Williams, G.J., Omran, P.G.: Rule-based knowledge discovery via anomaly detection in tabular data. In: Make (2023) <https://api.semanticscholar.org/CorpusID:260356626>
15. Xu, M.: Understanding graph embedding methods and their applications. SIAM Rev. **63**(4), 825–853 (2021). <https://doi.org/10.1137/20M1386062>



Revisiting Link Prediction with the Dowker Complex

Jae Won Choi¹(✉), Yuzhou Chen², José Frías³, Joel Castillo³, and Yulia Gel⁴

¹ Department of Computer Science, University of Texas at Dallas, Richardson, USA
jaewon.choi@utdallas.edu

² Department of Computer and Information Sciences, Temple University,
Philadelphia, USA

³ Center for Research in Mathematics (CIMAT), Mexico, Mexico

⁴ Department of Mathematical Sciences, University of Texas at Dallas, Richardson,
USA

Abstract. We propose a novel method to study properties of graph-structured data by means of a geometric construction called Dowker complex. We study this simplicial complex through the use of persistent homology, which has shown to be a prominent tool to uncover relevant geometric and topological information in data. A positively weighted graph induces a distance in its sets of vertices. A classical approach in persistent homology is to construct a filtered Vietoris-Rips complex with vertices on the vertices of the graph. However, when the size of the set of vertices of the graph is large, the obtained simplicial complex may be computationally hard to handle. A solution The Dowker complex is constructed on a sample in the set of vertices of the graph called landmarks. A way to guaranty sparsity and proximity of the set of landmarks to all the vertices of the graph is by considering ϵ -nets. We provide theoretical proofs of the stability of the Dowker construction and comparison with the Vietorrips-Rips construction. We perform experiments showing that the Dowker complex based neural networks model performs good with respect to baseline methods in tasks such as link prediction and resilience to attacks.

Keywords: Graph Neural Networks · Link Prediction · Dowker Complex

1 Introduction

The emerging findings increasingly more often suggest that higher order interactions among multiple constituents of complex systems such as generators, buses, and substations in power grid networks or proteins in molecular reactions tend to play a fundamental role behind the system organization and response to adverse external events [26]. Such higher-order or polyadic relationships among nodes are ubiquitous and vital for the functioning of a wide range of complex systems. For example, in the ecosystem, a prey may react to a predator's presence

and decrease its foraging for other species, while human interference (or lack of there off) can further drastically alter the dynamics of these interactions, and as the recent results suggest, the higher order relationships among multiple species may be the key driving factor behind species coexistence and the ecosystem biodiversity. In turn, recent findings highlight the crucial role of social group interactions, like those among colleagues and family members, in improving contagion tracking and intervention strategies. This is not to mention the critical role of higher-order relationships in formation and evolution of criminal gangs. However, as the existing graph learning methods still predominantly focus only on pair-wise node relationships, our understanding of the key mechanisms of these higher-order relationship phenomena remains largely in its nascentcy.

One emerging direction to address this fundamental challenge is to describe the mechanism of message propagation on higher-order network substructures such as simplicial complexes. In the context of graph neural networks (GNNs), this idea advances the notion of convolution from the conventional combinatorial graph Laplacian to a higher-order Hodge-Laplacian, leveraging the concepts of the Hodge-de Raam theory. This direction allows for describing diffusion not only at a pairwise node level as the prevailing GNNs do, but also at multi-node level such as among edges via triangles and other higher-order graph substructures. This approach results in the notion of simplicial neural networks which are arguably one of the most recent techniques in geometric deep learning. The alternative, yet closely related approach is to capitalize on the ideas of persistent homology (PH), in particular, by extracting topological signatures from the observed graph and constructing various forms of topological layers. Despite their success in graph tasks like classification and forecasting, the utility of both approaches for link prediction is less explored. This is largely due to excessive computational costs of higher-order graph learning, particularly, PH, and albeit the intuitive premise that higher-order relationships may be one of the most critical factors behind the link formation.

In this paper we take the first step toward bridging these two directions for higher-order graph learning, capitalizing on their complementary strengths not only to better capture diffusion on graphs but also to enhance robustness of link prediction against perturbations. In particular, we introduce a notion of Dowker complexes to graph machine learning, allowing us to substantially improve computational costs, thereby addressing one of the key bottlenecks in the way of the wider applicability of PH for link prediction and, more generally, graph learning. We develop a fully trainable Dowker persistence representation layer and integrate it into a simplicial convolutional network, then design a topological convolutional neural network over topological features of different Dowker filtrations. Note that, for topological signature representation learning, we consider global shape descriptor and local shape descriptor of Dowker complexes respectively. We also prove the persistence stability guarantees for Dowker complexes and offer the in-depth theoretical exploration of the relationship between the Dowker complex and its currently most widely used but computationally costly counterpart, the Vietoris-Rips complex. Our extensive experiments suggest that

the proposed DC-NETs leads to substantial gains both in link prediction performance and robustness.

Significance of our contributions can be summarized as follows:

- We propose a new summary of Dowker complex, namely, Dowker complex based topological summary. DC-NETs is the first approach introducing the concepts of Dowker complex to graph learning.
- We validate DC-NETs with link prediction on 5 datasets, showcasing superior performance over state-of-the-art baselines and robustness to noise.

2 Related Work

Link Prediction. Graph Convolutional Networks (GCNs) are recognized as powerful tools for addressing link prediction tasks. In particular, the Graph Autoencoder (GAE) [20] and its alteration version, Variational Graph Autoencoder (VGAE), are initially utilized to link prediction on citation networks. SEAL [30] adopts a distinctive approach by extracting local enclosing subgraphs surrounding target links and subsequently learning a function that maps subgraph patterns to link existence. Along with, the Hyperbolic Graph Convolutional Neural Networks (HGCN) [7] leverages both hyperbolic geometry and the GCN framework to derive node representations. Another recent strategy involves utilizing pairwise topological features with GCN to uncover latent representations of a graph’s geometrical structure [29].

Simplicial Complexes in Machine Learning and Persistent Homology. Persistent homology is a key tool in topological data analysis to deliver invaluable and complementary information on the intrinsic properties of data that are inaccessible with conventional methods [9, 18]. In the past decade, PH has become quite popular in various ML tasks, ranging from manifold learning to medical image analysis, and material science to finance. Modeling higher-order interactions on graphs is an emerging direction in graph representation learning. While the role of higher-order structures for graph learning has been documented for a number of years [1, 13, 19] and involves such diverse applications as graph signal processing in image recognition dynamics of disease transmission and biological network, integration of higher-order graph substructures into DL on graphs has emerged only in 2020. As shown by [4, 11, 26], higher-order network structures can be leveraged to boost graph learning performance. Indeed, several most recent approaches [10, 12, 17, 25] propose to leverage simplicial information to perform neural networks on graphs.

3 Background

Along the present manuscript, all graphs are non directed and weighted. We start describing the metric spaces induced by weighted graphs. These spaces are very important given their strong combinatorial foundations and have demonstrated relevance in data analysis [3, 22]. A positively weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$ is

determined by set of vertices \mathcal{V} ($|\mathcal{V}| = N$), an edge set $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ ($|\mathcal{E}| = M$), and a weight function $\omega : \mathcal{E} \rightarrow \mathbb{R}_+$. The graph \mathcal{G} is simple if it does not contain self-edges nor multiple edges. Given a path γ in a weighted graph \mathcal{G} , the length of γ is the sum of the weights of the edges in γ . A connected positively weighted graph \mathcal{G} is then endowed with the geodesic distance $d_{\mathcal{G}} : \mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}_{\geq 0}$, defined on a pair of vertices $u, v \in \mathcal{V}$ as the minimum length among all the paths connecting u and v . The set $(\mathcal{V}, d_{\mathcal{G}})$ is a finite metric space. In this case, the graph \mathcal{G} is also referred as a metric graph. If we have a sequence of nested simplicial complexes $\mathcal{K}_0 \subseteq \mathcal{K}_1 \subseteq \dots \subseteq \mathcal{K}_n = \mathcal{K}$, we say that the simplicial complex \mathcal{K} is filtered. The aim of Persistent Homology is to track the geometric evolution of the simplicial complexes along the filtration when we increase the parameter α by means of the homology groups $H_p(\mathcal{K}_i)$, for every $i, p \geq 0$ [14]. A non-trivial element $\beta \in H_p(\mathcal{K}_i)$ is usually called a topological feature of dimension p and can be interpreted in dimension 0 as a connected component in the complex K_i , in dimension 1 is the number of loops in the complex. In general, a topological feature in dimension p represents a p -dimensional “hole” and has associated two levels of the filtration b_β and d_α , the filtration level at which the topological feature is born (when it appears) and dies (when it disappears or is merged with another topological feature that was born before), respectively. The history of all topological features along the filtration can be summarized in a persistence diagram $PD(\mathcal{K}) = \{(b_\beta, d_\beta) \mid \beta \in H_p(\mathcal{K})\}$ [14]. When we restrict to a particular dimension $n \geq 0$, we denote the persistence diagram as $PD^n(\mathcal{K})$.

One of the most widely used simplicial complexes in topological data analysis (TDA) is the Vietoris-Rips simplicial complex [8]. This complex has the important property of being a flag complex, namely, it is completely determined by its 1-skeleton, which makes it suitable for computations. To implement any computation in persistent homology, to construct the Vietoris-Rips complex, we only need to compute the distance between any pair of points.

Definition 1 (Vietoris-Rips Complex). Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$ be a weighted simple graph with induced geodesic distance $d_{\mathcal{G}}$. For $\alpha \in \mathbb{R}_{\geq 0}$, we define the Vietoris-Rips complex $VR_\alpha(\mathcal{G})$ as the abstract simplicial complex with vertices in \mathcal{V} and, for $k \geq 2$, a k -simplex $\sigma = [x_0, x_1, \dots, x_k] \in VR_\alpha(\mathcal{G})$ if and only if $d_{\mathcal{G}}(x_i, x_j) \leq \alpha$ for $0 \leq i \leq j \leq k$.

Starting with a sequence of non-decreasing scale parameters, $0 \leq \alpha_0 \leq \alpha_1 \leq \dots \leq \alpha_{n-1} \leq \alpha_n$, we can construct a filtration of Vietoris-Rips complexes constructed on a geometric graph \mathcal{G} satisfying $VR_{\alpha_0}(\mathcal{G}) \subseteq VR_{\alpha_1}(\mathcal{G}) \subseteq \dots \subseteq VR_{\alpha_n}(\mathcal{G})$, and then apply persistent homology. One problem with the Vietoris-Rips complex is that if the cardinality of \mathcal{V} is considerable, the complex $VR_\alpha(\mathcal{G})$ may be excessively large to be analyzed using computational tools. A natural alternative to deal with the scalability problem is to construct another simplicial complex with set of vertices $L \subset \mathcal{V}$, where $|L| < |\mathcal{V}|$. One of such simplicial complexes is the witness complex [3, 6, 15, 16].

Definition 2 (Witness Complex). Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$ be a weighted simple graph with induced geodesic distance $d_{\mathcal{G}}$, and take subsets $L, W \subset \mathcal{V}$. For $\alpha \in$

$\mathbb{R}_{\geq 0}$, let $Wit_\alpha(W, L)$ be the abstract simplicial complex with set of vertices L , and a simplex $\sigma \in Wit_\alpha(W, L)$ if and only if for every $\tau \subseteq \sigma$ there exists $w \in W$ such that $d_{\mathcal{G}}(w, l) \leq d_{\mathcal{G}}(w, l') + \alpha$ for all $l \in \tau$ and $l' \in L \setminus \tau$. The complex $Wit_\alpha(W, L)$ is called the witness complex of \mathcal{G} with set of witnesses W and landmarks L .

Witness complex has important geometric properties. For instance, in the case of a point cloud in an Euclidean space it is related to the Delaunay triangulation, namely, the sequence of scale values define a filtered simplicial complex. Note that if we select sets of landmarks and witnesses $L, W \subset \mathcal{V}$, there exists also a filtration of witness complexes.

4 Dowker Complex

The witness complex is used as an alternative to deal with the scalability problem of computations of Vietoris-Rips constructions. As mentioned before, the witness complex has important geometric implications for point clouds in Euclidean spaces. However, in the case of metric graphs, those geometric properties are not inherited and it is possible to restrict the constructions of witness complex to obtain the Dowker complex:

Definition 3 (Dowker Complex). Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$ be a weighted simple graph with induced geodesic distance $d_{\mathcal{G}}$, and let $L, W \subset \mathcal{V}$. For $\alpha \in \mathbb{R}_{\geq 0}$, let $Dow_\alpha(W, L)$ be the abstract simplicial complex with set of vertices L and a simplex $\sigma \in Dow_\alpha(W, L)$ if and only if there exists $w \in W$ such that $d_{\mathcal{G}}(w, l) \leq \alpha$ for all $l \in \sigma$. The simplicial complex $Dow_\alpha(W, L)$ is called the Dowker complex of \mathcal{G} with sets of witnesses and landmarks W and L , respectively.

Note that Dowker complexes construction strongly depend on the selection of the sets L and W . In [15], two algorithms to define the set of landmarks L are developed: random and maxmin algorithms. An interesting approach to define the a set of landmarks is via ϵ -nets [2,3]. This selection of landmarks satisfies two important properties: sparcity and proximity to the complete set of vertices, thereby reducing both, computational complexity and potential information loss.

Definition 4 (ϵ -Net). Let $(\mathcal{V}, d_{\mathcal{G}})$ be the finite metric space obtained from the weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$. Given $L = \{u_1, u_2, \dots, u_l\} \subset \mathcal{V}$ and $\epsilon \geq 0$ then:

- (i) The set L is an ϵ -sample of \mathcal{G} if the collection $\{\mathcal{N}(u_i)\}_{i=1}^l$ of closed ϵ -neighborhoods of points in L covers \mathcal{V} , i.e. for any $v \in \mathcal{V}$ there exists $u_j \in L$ such that $d_{\mathcal{G}}(v, u_j) \leq \epsilon$.
- (ii) L is ϵ -sparse if for any two points $u_i, u_j \in L$, their distance $d_{\mathcal{G}}(u_i, u_j) > \epsilon$.
- (iii) The set L is an ϵ -net of \mathcal{G} if it is an ϵ -sample of \mathcal{G} and is ϵ -sparse.

Following the definition of an ϵ -net, natural questions arise, for instance, given the metric graph \mathcal{G} and ϵ : (i) how many points does it contain an ϵ -net of \mathcal{G} ?, or (ii) what is the most suitable way to select the points in an ϵ -net? Concerning the later question, some algorithms were presented and compared in

[3], however the suitability of each algorithm will depend on the context in which they are compared. With respect to the number of points of an ϵ -net, an upper bound in terms of the diameter of the graph and ϵ is presented in Theorem 3 of [3]. As an example, consider the unweighted star graph S_n with central vertex v_0 . It is clear that there exist two 1-nets, $L_1 = \{v_0\}$ and $L_2 = \mathcal{V}(S_n) \setminus \{v_0\}$, with cardinalities 1 and $N - 1$ (see Fig. 1). However, if $\epsilon \geq 2$, every ϵ -net has a single vertex, showing the difficulty of provide bounds for the cardinality of ϵ -nets for general graphs. We describe the construction of ϵ -nets with bounded cardinality for arbitrary unweighted graphs:



Fig. 1. Two 1-nets of an unweighted star graph. Marked points are the ϵ -net.

Lemma 1. *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a connected unweighted graph and $\epsilon > 0$. If $N = |\mathcal{V}|$, then there is an ϵ -net of \mathcal{G} whose cardinality is bounded by $N/(\epsilon+1) + 2N/d$, where d is the diameter of the graph.*

Proof. Let T be a spanning tree of \mathcal{G} , which is rooted at one of its centers, say v_0 (remember that a tree has at most two centers). Given a vertex $v \in \mathcal{V}$, let $\text{depth}(v)$ be the depth of v with respect to v_0 , and d_{\max} the maximum depth among the vertices in \mathcal{V} . Since the graph \mathcal{G} is unweighted (every edge has weight 1), we may assume that ϵ is a positive integer. Define the level sets $N_i = \{v \in \mathcal{V} \mid \text{depth}(v) = i\}$, and $M_j = \cup N_i$, where the union is taken on the indexes $i \equiv j \pmod{\epsilon+1}$, for $j = 0, 1, \dots, \epsilon$. Note that each set $M_j \cup \{v_0\}$ is an ϵ -sample of \mathcal{G} , and each of such sets contains at most $d_{\max}/(\epsilon+1) + 1$ of the level sets N_i . On the other hand, there exists at least one set M_k such that every level set contained in M_k has in average N/d_{\max} points. Then the total number of points in M_k is upperly bounded by $(d_{\max}/(\epsilon+1) + 1)(N/d_{\max}) = N/(\epsilon+1) + N/d_{\max}$. Since $d \leq d_{\max}$, the result follows.

5 Methodology

Recall that a k -simplex $\sigma_k = (v_0, \dots, v_k)$ is a k -dimensional convex hull of nodes v_0, \dots, v_k . Given the set of nodes v_0, \dots, v_k of σ_k , define an orientation on the simplex σ_k by choosing some (arbitrary) ordering of its nodes. Consider a real-valued vector space C^k with basis from the oriented k -simplices. Elements C^k are called *k-chains*. The *boundary* operator ∂_k is a homomorphism $\partial_k : C^k \rightarrow C^{k-1}$. The adjoint of ∂_k is the *co-boundary* operator $\partial_k^T : C^k \rightarrow C^{k+1}$. Let B_k and B_k^\top be the matrix representations of ∂_k and ∂_k^T , respectively.

Definition 5. Linear operator \mathbf{L}_k over oriented k -simplices $\mathbf{L}_k : C^k \rightarrow C^k$ is called the k -Hodge Laplacian, and its corresponding matrix representation is

$$\mathbf{L}_k = \mathbf{B}_k^\top \mathbf{B}_k + \mathbf{B}_{k+1} \mathbf{B}_{k+1}^\top, \quad (1)$$

where $\mathbf{B}_k^\top \mathbf{B}_k$ and $\mathbf{B}_{k+1} \mathbf{B}_{k+1}^\top$ are often referred to \mathbf{L}_k^{down} and \mathbf{L}_k^{up} , respectively.

That is, the standard graph Laplacian $\mathbf{L}_0 = \mathbf{B}_1 \mathbf{B}_1^\top \in \mathbb{R}^{N \times N}$ is a special case of the above k -th combinatorial Hodge Laplacian and the matrix $\mathbf{L}_1 \in \mathbb{R}^{M \times M}$ is the Hodge 1-Laplacian. While the Hodge theory allows us to systematically describe diffusion across higher-order graph substructures, or k -simplices of any k , all current studies are restricted solely to Hodge 1-Laplacian \mathbf{L}_1 [25, 26]. In this paper, we propose a new random walk-based block Hodge-Laplacian operator that enables us to simultaneously integrate knowledge on interactions among higher-order substructures of various orders into graph learning. The K -block Hodge-Laplacian \mathfrak{L}_K is related to the Dirac operator in differential geometry. As such, \mathfrak{L}_K has multiple implications for analysis of synchronization dynamics and coupling of various topological signals on graphs.

Furthermore, we propose the random walk-based block Hodge-Laplacian, i.e., r -th power of block Hodge-Laplacian representation (where $r \in \mathbb{Z}_{>0}$). Our random walk-based block Hodge-Laplacian is inspired by the recent success in random walk based graph embeddings and simplicial complexes but is designed to conduct informative *joint* random walks on higher-order Hodge Laplacians instead of limited powering Hodge 1-Laplacian [4, 5]. Indeed, successfully travelling through higher-order topological features will provide us with additional feature information which is valuable for learning edge embeddings.

Adaptive Simplicial Complex Based Block. Our innovative method for understanding the complex topological structure of higher-order graphs is driven by a central inquiry: Is it possible to encapsulate the relational dynamics between k -simplices within a graph \mathcal{G} that exhibit a separation beyond the immediate adjacency of orders $k-1$ and $k+1$? Although exploring this issue extends beyond the conventional scope of the Hodge-de Rham framework applied to simplicial Hodge Laplacians, we present a solution via the development of an Adaptive Simplicial Complex (ASC) block as follows

$$\mathbb{L}_2^{ASC} = \left[\begin{array}{c|c} \mathbf{L}_{k_1} & g(\mathbb{P}_{d_2} \mathbf{L}_{k_1}, \mathbf{L}_{k_2}) \\ \hline g(\mathbb{P}_{d_2} \mathbf{L}_{k_1}, \mathbf{L}_{k_2})^\top & \mathbf{L}_{k_2} \end{array} \right]. \quad (2)$$

Note that in general, the linear operator $\mathbb{L}_2^{ASC} \in \mathbb{R}^{(d_1+d_2) \times (d_1+d_2)}$ is no longer a Laplacian. For example, while \mathbb{L}_2^B is symmetric (by construction), it may not satisfy the condition of positive semidefiniteness. However, as shown below, ASC block opens multiple opportunities to better describe higher-order interactions on graphs. Armed with the ASC block, we can utilize the non-local message operation $g(\cdot, \cdot)$ to capture long-range relations and intrinsic higher-order connectivity among entities in the 2-block (reduced) Hodge-Laplacian \mathfrak{L}_2 , i.e., $g(\mathbb{P}_{d_2} \mathbf{L}_{k_1}, \mathbf{L}_{k_2})$ placed in the off-diagonal. We now describe the choices of non-local message passing functions which can be used for $g(\cdot, \cdot)$ in the following.

Specifically, given two higher-order Hodge Laplacians \mathbf{L}_{k_1} and \mathbf{L}_{k_2} , we define two types of the non-local message passing functions to capture the relations between $[\mathbb{P}_{d_2} \mathbf{L}_{k_1}]_i \in \mathbb{R}^{d_2}$ and $[\mathbf{L}_{k_2}]_j \in \mathbb{R}^{d_2}$ (i.e., topological embedding of the i -th graph substructure in $\mathbb{P}_{d_2} \mathbf{L}_{k_1}$ and topological embedding of the j -th graph substructure in \mathbf{L}_{k_2} respectively) as $g([\mathbb{P}_{d_2} \mathbf{L}_{k_1}]_i, [\mathbf{L}_{k_2}]_j) = <[\mathbb{P}_{d_2} \mathbf{L}_{k_1}]_i, [\mathbf{L}_{k_2}]_j>$. Finally, the adaptive Hodge block can be formulated as $\tilde{\mathbb{L}}_2^B = \text{softmax}(\text{ReLU}(\mathbb{L}_2^B))$, where the softmax function is employed to standardize the block operator \mathbb{L}_2^B , the ReLU activation function, defined as $\text{ReLU}(\cdot) = \max(0, \cdot)$, serves to filter out both the weak pairwise interactions among higher-dimensional simplices and weak connections. Consequently, the proposed ASC block $\tilde{\mathbb{L}}_2^{\text{ASC}}$ is adept at dynamically learning the interplay between simplices across varying dimensions. We now turn our attention to formulating the adaptive simplicial complex block convolutional layer (ASC-CL), which facilitates the learning of distances between nodes within the embedding space and can be formulated as $\mathbf{Z}^{(\ell+1)} = (\tilde{\mathbb{L}}_2^B \mathbf{Z}^{(\ell)} \Theta_1^{(\ell)}) \Theta_2^{(\ell)}$, where $\mathbf{Z}^{(0)} = \mathbf{X} \in \mathbb{R}^{n \times d}$ is node features matrix, $\Theta_1^{(\ell)} \in \mathbb{R}^{d \times (d_1+d_2)}$ and $\Theta_2^{(\ell)} \in \mathbb{R}^{(d_1+d_2) \times d_{\text{out}}}$ are two trainable weight matrices at layer ℓ , and d_{out} denotes the dimension of node embedding at the (ℓ) -th layer through the ASC-CL operation. For the link prediction task, we use the Fermi-Dirac decoder to compute the distance between the two nodes. Formally, $\text{dist}_{uv}^{\text{ASC-CL}} = (\mathbf{Z}_u^{(\ell+1)} - \mathbf{Z}_v^{(\ell+1)})^2$ where $\text{dist}_{uv}^{\text{ASC-CL}} \in \mathbb{R}^{1 \times d_{\text{out}}}$ is the distance between nodes u and v in a local spatial domain.

Dowker Complex Based Topological Layer. To learn critical information on the Dowker complex based topological features, we propose the Dowker complex based topological layer (DC-TL) denoted by Ψ . DC-TL is a Dowker complex based topological representation learning layer. Given Dowker complex based topological features Ξ , the DC-TL operator will output the latent Dowker complex based topological representation $\tilde{\mathbf{Z}}$ with shape d_c as follows (d_c is the number of channels in output), i.e., $\mathbf{Z}_{\text{DC-TL}} = \Psi(\Xi) = \phi_{\text{MAX}}(f_{\text{CNN}}^{(\ell)}(\Xi))$ where $f_{\text{CNN}}^{(\ell)}$ is the convolutional neural network (CNN) in the ℓ -th layer, and ϕ_{MAX} denotes global max-pooling layer. Similar to the process of computing distances between learnable node embeddings with the adaptive simplicial complex block convolutional layer, we conduct convolution operation to evaluate the distance between the node embeddings of nodes u and v as

$$\mathbf{H}^{(\ell+1)} = [\tilde{\mathbf{L}} \mathbf{H}^{(\ell)} \Theta_3^{(\ell)}, \mathbf{Z}_{\text{DC-TL}}], \quad \text{dist}_{uv}^{\text{DC-TL}} = (\mathbf{H}_u^{(\ell+1)} - \mathbf{H}_v^{(\ell+1)})^2, \quad (3)$$

where $[\cdot, \cdot]$ denotes the concatenation operation, $\tilde{\mathbf{L}} = \mathbf{D}_v^{-1/2}(\mathbf{A} + \mathbf{I})\mathbf{D}_v^{-1/2}$ (where \mathbf{D}_v is the degree matrix of $\mathbf{A} + \mathbf{I}$, i.e., $[\mathbf{D}_v]_{ii} = \sum_j [\mathbf{A} + \mathbf{I}]_{ij}$), $\mathbf{H}^{(0)} = \mathbf{X} \in \mathbb{R}^{N \times d}$ is node features matrix, $\Theta_3^{(\ell)} \in \mathbb{R}^{d \times d_{\text{out}}}$ is the trainable weight matrix, d_{out} denotes the dimension of node embedding at the (ℓ) -th layer through the graph convolution operation, $\text{dist}_{uv}^{\text{DC-TL}} \in \mathbb{R}^{1 \times d_{\text{out}}}$ is the distance between nodes u and v in global spatial domain. Finally, we wrap the concatenation of Hodge-style adaptive convolution and graph convolution operation outputs into the Multi-layer Perceptron (MLP) $\text{dist} =$

$\sigma(f_{\text{MLP}}([\alpha_{\text{DC-TL}} \times \text{dist}^{\text{DC-TL}}, \alpha_{\text{SC}} \times \text{dist}^{\text{ASC-CL}}]))$ where $[\cdot, \cdot]$ denotes the concatenation of the outputs from the ASC-CL operator and Dowker complex based topological layer, $\sigma(\cdot)$ denotes the activation function, f_{MLP} is an MLP layer that maps the concatenated embedding to a d_o -dimensional space, and $\alpha_{\text{DC-TL}}$ and $\alpha_{\text{ASC-CL}}$ are the hyperparameters representing the weight of each distance in the $(\ell+1)$ -th layer. Lastly, the edges connection probability can be computed as $P_{u,v} = [\exp((\text{dist}_{uv} - \delta)/\eta) + 1]^{-1}$, where δ and η are hyperparameters. Then, training via standard backpropagation is performed via binary cross-entropy loss function.

Table 1. ROC AUC and standard deviations for link prediction. Bold numbers denote the best results.

Model	Cora	Citeseer	Cornell	Texas	Wisconsin
GCN	89.14 ± 1.20	87.89 ± 1.48	79.14 ± 8.89	67.42 ± 9.39	72.77 ± 6.96
GAT	72.80 ± 0.20	74.80 ± 1.50	68.83 ± 4.34	70.42 ± 3.98	70.41 ± 4.03
GIC	91.42 ± 1.24	92.99 ± 1.14	57.26 ± 8.77	65.16 ± 7.87	75.24 ± 8.45
GAE	90.21 ± 0.98	88.42 ± 1.13	60.58 ± 3.71	68.67 ± 6.95	75.10 ± 8.69
GVAE	92.17 ± 0.72	90.24 ± 1.10	68.25 ± 4.21	74.61 ± 8.61	74.39 ± 8.39
SEAL	90.29 ± 1.89	88.12 ± 0.85	68.99 ± 3.77	71.68 ± 6.85	77.96 ± 10.37
TLC-GCN	94.22 ± 0.78	93.30 ± 0.60	63.49 ± 10.43	58.12 ± 5.67	58.57 ± 3.28
BScNets	95.09 ± 0.65	92.06 ± 0.66	72.50 ± 11.64	72.29 ± 14.49	84.83 ± 4.80
DC-NETs (ours)	95.12 ± 0.54	93.87 ± 0.83	83.35 ± 6.12	82.35 ± 6.99	87.14 ± 5.63

Table 2. Performance comparison between dowker complex and Vietoris-Rips complex.

Model	Cora	Cornell	Texas	Wisconsin
DC-NETs (ours)	95.12 ± 0.54	83.35 ± 6.12	82.35 ± 6.99	87.14 ± 5.63
RC-NETs	94.85 ± 0.39	84.55 ± 6.18	81.49 ± 2.25	86.23 ± 7.50

6 Experiments

Datasets and Baselines. We experiment on three types of networks (i) citation networks: Cora and Citeseer [27]; (ii) webpage datasets: Cornell, Texas, and Wisconsin [24]. We compare against eight state-of-the-art (SOA) baselines, including (i) Graph convolution network (GCN) [21] which multiplies the adjacency matrix to simplify the multi-layer graph convolutional networks; (ii) Graph Attention Networks (GAT) [28] which adopts attention mechanism to aggregate the information from neighbors, considering relative weights between two connected nodes; (iii) Graph InfoClust (GIC) [23] which enhances node representations by leveraging cluster-level content; (iv) Graph Autoencoder (GAE) and (v) its variational version, Variational Graph Autoencoder (GVAE) [20] which

Table 3. ROC AUC and standard deviations for link prediction.

Model	Cora	Cornell	Texas	Wisconsin
SEAL (0.05)	91.70 ± 0.73	76.02 ± 5.80	80.66 ± 5.54	67.91 ± 4.69
TLC-GCN (0.05)	92.24 ± 0.68	58.94 ± 7.77	56.47 ± 13.67	56.82 ± 5.77
BscNets (0.05)	94.95 ± 0.64	71.54 ± 10.08	72.74 ± 11.30	84.07 ± 5.88
DC-NETs (0.05)	94.10 ± 1.62	79.15 ± 5.32	77.01 ± 4.48	87.04 ± 3.88
SEAL (0.10)	90.91 ± 2.36	74.78 ± 4.678	79.83 ± 4.29	66.93 ± 4.22
TLC-GCN (0.10)	91.48 ± 2.13	54.96 ± 7.43	55.71 ± 8.83	53.62 ± 2.72
BscNets (0.10)	94.52 ± 0.70	69.63 ± 9.11	64.77 ± 13.01	83.71 ± 6.20
DC-NETs (0.10)	92.75 ± 3.56	78.86 ± 7.26	76.94 ± 4.84	86.92 ± 4.43
SEAL (0.15)	89.93 ± 0.46	73.78 ± 4.13	79.39 ± 4.53	66.28 ± 3.13
TLC-GCN (0.15)	88.98 ± 3.89	51.94 ± 4.50	54.90 ± 6.62	50.10 ± 5.86
BscNets (0.15)	93.95 ± 1.03	68.53 ± 12.66	71.55 ± 11.47	81.62 ± 7.86
DC-NETs (0.15)	92.24 ± 1.15	76.55 ± 7.07	76.61 ± 4.26	86.63 ± 4.08

is employing latent variables and GCN encoder to generate interpretable representations; (vi) SEAL [30] which predicts the relationship on graph-structure data based on graph convolution on local subgraph around each link; (vii) Topological Loop-Counting Graph Neural Network (TLC-GNN) [29] which injects the pairwise topological information into the latent representation of a GCN; (viii) BScNet [10] which revolutionize graph learning by incorporating multi-level interactions among higher-order graph structures using the Block Hodge-Laplacian.

Hyperparameter Settings and Training Details. DC-NETs consists of a one adaptive block Hodge convolution layer ($nhid1$) and two graph convolution layers ($nhid2$, $nhid3$), where $nhid1 \in \{1, 8, 16, 32, 64, 128\}$, $nhid2 \in \{16, 32, 64, 128\}$, and $nhid3 \in \{4, 16, 32\}$. In addition, we perform an extensive grid search for learning rate among $\{0.001, 0.005, 0.008, 0.01, 0.05\}$, the dropout rate among $\{0.1, 0.2, \dots, 0.9\}$, power r for random walk among $\{2, 3, 4, 5\}$, and importance weights $\alpha_{\text{DC-TL}}$ among $\{0.01, 0.1, 1.0, 10.0, 100.0\}$ and $\alpha_{\text{ASC-CL}}$ among $\{1.0, 10.0\}$ respectively. The model is trained for 1,000 epochs with early stopping applied when the metric (i.e., validation loss) starts to drop.

6.1 Results

The evaluation results on 5 datasets are summarized in Tables 1 and 2. Table 1 shows the performance comparison among 8 baselines on 2 citation networks (i.e., Cora and Citeseer) and 3 webpage datasets (i.e., Cornell, Texas, and Wisconsin) where nodes indicate web pages, and edges show hyperlinks between them. On all 5 datasets, we find that our DC-NETs consistently perform far better than baseline models. In particular, the average relative gain of DC-NETs over the

runner-ups is 3.812%, demonstrating the proposed DC-NETs' effectiveness on all common link prediction benchmarks. Regarding baseline methods, although GCN which is two spectral-based convolutional GNN can reflect the relationship between nodes, they fail to consider topological structure information. Furthermore, three spatial-based convolutional GNNs, such as GAT and SEAL, capture the weights of neighboring nodes, representing node-level information. This approach has limitations in expressing broader availability, making it more difficult to extract information from nodes located at a significant distance and to identify high-order spatial dependencies. Table 2 shows the performance comparison between the Dowker and Vietoris-Rips complex. We can observe that, for 3 out of 4 datasets, DC-NETs achieves an average of 0.24% higher performance than RC-NETs. The results of resilience analysis are summarized in Table 3. Table 3 shows performance comparisons among SEAL, TLC-GCN, BscNets, and DC-NETs with different perturbation rates from 0.05 to 0.15 on 1 citation network and 3 web page datasets. From Table 3, we can observe that (i) our proposed DC-NETs significantly outperform all baselines on 3 web page datasets across all 3 perturbation rates and (ii) although BScNets outperform DC-NETs on Cora dataset, our Dowker complex based topological layer improves the robustness with a slight performance degradation.

7 Conclusion

By introducing the concepts of Dowker complex to graph learning, we have addressed one of the primary bottlenecks in the way of wider adoption of topological methods – that is, computational complexity. We have developed DC-NETS, a novel approach for link prediction harnessing topological information on a graph in the form of the Dowker layer. Our numerical experiments have indicated that DC-NETS outperforms state-of-the-art competitors while requiring noticeably fewer computational resources than other topological approaches. This opens up new possibilities for future research in topological graph learning such as computationally efficient pooling, topological attacks, and topological defences.

Acknowledgements. This project has been supported in part by NASA AIST 21-AIST21_2-0059, NSF ECCS 2039701, TIP-2333703, and ONR N00014-21-1-2530 grants. The paper is based upon work supported by (while Y.R.G. was serving at) the NSF. The views expressed in the article do not necessarily represent the views of NSF, ONR or NASA.

References

1. Agarwal, S., Branson, K., Belongie, S.: Higher order learning with graphs. In: ICML, pp. 17–24 (2006)
2. Aksoy, S.G., et al.: Seven open problems in applied combinatorics. arXiv preprint [arXiv:2303.11464](https://arxiv.org/abs/2303.11464) (2023)

3. Arafat, N.A., Basu, D., Bressan, S.: ϵ -net induced lazy witness complexes on graphs (2020)
4. Benson, A.R., Abebe, R., Schaub, M.T., Jadbabaie, A., Kleinberg, J.: Simplicial closure and higher-order link prediction. *PNAS* **115**(48), E11221–E11230 (2018)
5. Bodnar, C., et al.: Weisfeiler and Lehman go topological: message passing simplicial networks. In: ICML, pp. 1026–1037 (2021)
6. Boissonnat, J.D., Guibas, L., Oudot, S.: Manifold reconstruction in arbitrary dimensions using witness complexes. In: SoCG, pp. 194–203 (2007)
7. Chami, I., Ying, R., Ré, C., Leskovec, J.: Hyperbolic graph convolutional neural networks (2019)
8. Chazal, F., De Silva, V., Oudot, S.: Persistence stability for geometric complexes. *Geom. Dedicata.* **173**(1), 193–214 (2014)
9. Chazal, F., Michel, B.: An introduction to topological data analysis: fundamental and practical aspects for data scientists. *Front. Artif. Intell.* **4** (2021)
10. Chen, Y., Gel, Y.R., Poor, H.V.: BScNets: block simplicial complex neural networks. *Proc. AAAI Conf. Artif. Intell.* **36**(6), 6333–6341 (2022)
11. Chen, Y., Gel, Y.R., Marathe, M.V., Poor, H.V.: A simplicial epidemic model for COVID-19 spread analysis. *Proc. Natl. Acad. Sci.* **121**(1), e2313171120 (2024)
12. Chen, Y., Jacob, R.A., Gel, Y.R., Zhang, J., Poor, H.V.: Learning power grid outages with higher-order topological neural networks. *IEEE Trans. Power Syst.* **39**(1), 720–732 (2024)
13. Chen, Y., Jiang, T., Gel, Y.R.: H^2 -nets: hyper-Hodge convolutional neural networks for time-series forecasting. In: Koutra, D., Plant, C., Gomez Rodriguez, M., Baralis, E., Bonchi, F. (eds.) ECML PKDD 2023. LNCS, vol. 14713, pp. 271–289. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-43424-2_17
14. Cohen-Steiner, D., Edelsbrunner, H., Harer, J.: Stability of persistence diagrams. In: SoCG, pp. 263–271 (2005)
15. De Silva, V., Carlsson, G.: Topological estimation using witness complexes. In: PBG, pp. 157–166 (2004)
16. Dey, T.K., Fan, F., Wang, Y.: Graph induced complex on point data. In: SoCG, pp. 107–116 (2013)
17. Ebli, S., Defferrard, M., Spreemann, G.: Simplicial neural networks. In: NeurIPS 2020 Workshop on TDA and Beyond (2020)
18. Hensel, F., Moor, M., Rieck, B.: A survey of topological machine learning methods. *Front. Artif. Intell.* **4**, 52 (2021)
19. Johnson, J.L., Goldring, T.: Discrete Hodge theory on graphs: a tutorial. *Comput. Sci. Eng.* **15**(5), 42–55 (2013)
20. Kipf, T.N., Welling, M.: Variational graph auto-encoders (2016)
21. Kipf, T.N., Welling, M.: Semi-supervised classification with gcn. In: ICLR (2017)
22. Liu, X., Feng, H., Wu, J., Xia, K.: Dowker complex based machine learning (DCML) models for protein-ligand binding affinity prediction. *PLoS Comp. Biol.* **18**(4), e1009943 (2022)
23. Mavromatis, C., Karypis, G.: Graph InfoClust: maximizing coarse-grain mutual information in graphs. In: Karlapalem, K., et al. (eds.) Advances in Knowledge Discovery and Data Mining. PAKDD 2021, Part I. LNCS, vol. 12712, pp. 541–553. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-75762-5_43
24. Pei, H., Wei, B., Chang, K.C.C., Lei, Y., Yang, B.: Geom-GCN: geometric graph convolutional networks. In: ICLR (2019)
25. Roddenberry, T.M., Glaze, N., Segarra, S.: Principled simplicial neural networks for trajectory prediction. In: ICML, pp. 9020–9029 (2021)

26. Schaub, M.T., Benson, A.R., Horn, P., Lippner, G., Jadbabaie, A.: Random walks on simplicial complexes and the normalized Hodge 1-laplacian. *SIAM Rev.* **62**(2), 353–391 (2020)
27. Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., Eliassi-Rad, T.: Collective classification in network data. *AI Mag.* **29**(3), 93–93 (2008)
28. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: ICLR (2018)
29. Yan, Z., Ma, T., Gao, L., Tang, Z., Chen, C.: Link prediction with persistent homology: an interactive view (2021)
30. Zhang, M., Chen, Y.: Link prediction based on graph neural networks (2018)



GraphNILM: A Graph Neural Network for Energy Disaggregation

Rui Shang¹(✉), Siji Chen¹, Zhiqian Chen², and Chang-Tien Lu¹

¹ Virginia Polytechnic Institute and State University Blacksburg, Blacksburg, VA 24061-0002, USA

{srui,sijic}@vt.edu, zchen@cse.msstate.edu

² Mississippi State University, 75 B.S. Hood Drive, Mississippi, MS 39762, USA
ctlu@vt.edu

Abstract. Non-Intrusive Load Monitoring (NILM) remains a critical issue in both commercial and residential energy management, with a key challenge being the requirement for individual appliance-specific deep learning models. These models often disregard the interconnected nature of loads and usage patterns, stemming from diverse user behavior. To address this, we introduce GraphNILM, an innovative end-to-end model that leverages graph neural networks to deliver appliance-level energy usage analysis for an entire home. In its initial phase, GraphNILM employs Gaussian random variables to depict the graph edges, later enhancing prediction accuracy by substituting these edges with observations of appliance interrelationships, stripping the individual load energy from the aggregated main energy all at one time, resulting in reduced memory usage, especially with more than three loads involved, thus presenting a time and space-efficient solution for real-world implementation. Comprehensive testing on popular NILM datasets confirms that our model outperforms existing benchmarks in both accuracy and memory consumption, suggesting its considerable promise for future deployment in edge devices.

Keywords: NILM · Energy Disaggregation · Graph Neural Network

1 Introduction

Energy conservation is a crucial research area in today's scientific world. Household and commercial electrical use accounts for approximately 60% of global energy consumption [7]. Real-time monitoring of power consumption is a useful approach for assisting homes, utilities, appliance manufacturers, and policymakers in making more informed decisions. However, obtaining individual appliance-level load data in real-time typically requires the installation of a sensor per load, which can be costly and impractical for older houses or office buildings. As a result, non-intrusive load monitoring (NILM) technology has gained popularity due to its low installation and maintenance costs, as well as its respect for privacy.

By gathering data from the main power measurements and computing the projected individual power consumption without additional measuring equipment, NILM provides a cost-effective solution for disaggregating energy consumption. Several surveys [1] have demonstrated the business case for NILM, revealing that energy savings outweigh installation costs. Moreover, research has shown that providing active energy data feedback to customers through NILM can reduce energy use by 5–20% [24]. However, NILM is inherently challenging due to the various load combinations in a given place and consumers' complex consumption preferences. Addressing this problem requires the development of innovative and efficient models that can accurately disaggregate energy consumption at the appliance level, which is the focus of this study.

Figure 1 depicts the NILM application scenario utilizing state-of-the-art disaggregation techniques versus our proposed GraphNILM method. In most houses or office buildings, the number of routinely used devices exceeds four. As the number of appliances increases, the amount of resources needed to estimate the instantaneous load power rises. GraphNILM, in contrast, utilizes roughly the same amount of memory size to achieve comparable results, which seems more reasonable to be deployed on edge devices in the houses or office buildings.

There are three main challenges in the NILM field. **(1) A low rate of sampling.** The sampling rate in the NILM field, which is typically 1Hz for common datasets, is significantly lower than the working frequency of the loads; and thus, the sampled data cannot be fully restored according to the Nyquist-Shannon sampling theorem. Adoption of classical algorithms, like hidden Markov models and their variants [17,22], yields restricted results under specific conditions, making widespread adoption challenging. **(2) Homogeneous data with restricted characteristics.** The majority of available data consists solely of aggregated power readings in a timely order, which makes it difficult for domain experts to quantify the dedicated load power numbers from readings only. Numerous studies therefore focus on the classification problem [8] by Convolutional Neural Network (CNN) and the Recurrent Neural Network (RNN) [5], rather than quantitative analysis, but even good classification results are of limited utility to consumers. Non-stable power consumption and the complex combinations of loads makes disaggregation a challenging problem. **(3) More memory resources required as more loads are introduced** In the NILM commercial sector, a regression model with instantaneous individual load power output appears more desirable than a categorization model. In recent years, non-linear regression models in NILM have emerged with respectable performance, but the vast majority of deep learning models have a high memory footprint and

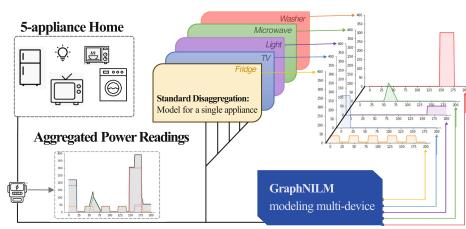


Fig. 1. NILM general explanations and comparisons of our proposed method with common state-of-art methods.

results, which seems more reasonable to be deployed on edge devices in the houses or office buildings.

increased computational complexity [23]. To improve performance, the majority of regression models [13, 26] repeat the same structure with different parameters for different loads. A residence with ten loads will necessitate the concurrent operation of ten times the proposed model, indicating the high cost of business implementations on edge devices. This research aims to develop an end-to-end model with reduced memory consumption and cutting-edge performance for future business deployments on edge devices. To solve the challenges highlighted in the NILM field, this paper designs the GraphNILM model and provides the following significant contributions:

- **Formulating a novel end-to-end framework for energy disaggregation.** The paper presents GraphNILM, a model that uses a modified convolutional neural network for initial disaggregation and a graph neural network for refinement, enabling simultaneous disaggregated power readings. GraphNILM efficiently extracts power features from low-rate sequences, fine-tuning the results using load relationships before producing the final output.
- **Constructing an effective algorithm to characterize load relations.** This paper categorizes relationships between distinct loads as synchronous and asynchronous. Supplementing our approach, we introduce a new algorithm to calculate the synchronous relations between the aggregated power readings and individual load by correlations, and asynchronous relations between loads by dynamic matching.
- **Designing a new structure for memory reduction.** In response to single-load targeted models, we construct a weighted graph for GraphNILM, transforming loads into nodes using pre-established relationships. This allows simultaneous power disaggregation, reducing memory and computational requirements by avoiding separate individual load trainings. We are the first to use dynamic time wrapping relationship structure in the NILM field.
- **Conducting extensive experimental performance evaluations.** The proposed GraphNILM network has been evaluated utilizing data from standard NILM datasets: REDD [16] and UK-DALE [14]. It often surpasses competing methods across different metrics, utilizing a fraction of memory compared to benchmarks. We also discuss the practical benefits of integrating GraphNILM into edge devices.

2 Related Work

The NILM field was pioneered by Hart [10] about three decades ago. In order to solve the NILM problem with low rate sampling data, Hidden Markov based Model (HMM, FHMM, etc.) [17] and its variants [22] were adopted in the early stage. Such methods were categorized as event-based methods, which usually contain three procedures: edge detection, feature extraction , and classification [9, 18], and were broadened to handle NILM disaggregations problems in other fields [6]. With deep learning flourishing in most domains in recent years, deep neural networks and convolution neural networks have lowered the obstacles in the NILM field for researchers to extract power features without the help of

domain experts [13]. Deep learning has lighted up a new direction for solving the NILM problem [19]. Long-short-term memory network (LSTM) [20] extracted dominate appliance usage from the aggregate power signals, which are collected at a low sampling rate. The widely-used Seq2Point model [26], showed great improvement on regression tasks in the NILM domain and received challenges all the time since its debut, but still need trainings per load introduced [4]. The introduction of graph signal processing (GSP) to the NILM field is a novel concept. Stankovic's research group [2, 11, 27] tracked this technique by segmenting aggregated energy sequences to do classification task in NILM. Similar work has been conducted by Bing and other groups [25], with all of them utilizing graph neural networks to perform load identification tasks which extracted information is insufficient to meet individual and commercial energy planning requirements.

Our aim is to create an easily implementable real-world model that can produce instantaneous disaggregated load energy with reduced memory consumption, while maintaining accuracy equivalent to the state-of-the-art techniques. Realized the ignorance of relations among loads in regression tasks and the difficulty of transforming homogeneity data to adapt to the multi-variate inputs needs for graph signal processing, we decide to design a new integrated framework for energy disaggregation by utilizing the advantages of the deep convolution neural network and the graph neural network. As a result, the proposed model can not only fine-tune the disaggregated power results, but also reduce total computations by incorporating the graph design.

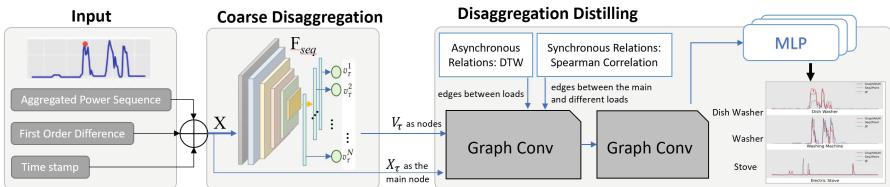


Fig. 2. GraphNILM Total Structure.

3 Proposed Model

3.1 Problem Setup: Disaggregation

Given the aggregated power meter readings, x_t , we wish to disaggregate the immediate power contribution of each individual load. The disaggregation of $x_t \in \mathbb{R}$ at time t is formulated as:

$$x_t = \sum_{n=1}^N y_t^n + \text{noise}, \quad (1)$$

N is the number of loads which are to be monitored and y_t^n stands for the power of the n -th load at time t . The noise and unmonitored loads in the given dataset have been generalized to the noise. Our purpose is to get the individual load power together at the same time, as $Y_t = [y_t^1, y_t^2, \dots, y_t^N]^T$ based on the measurement of the aggregated power x_t from our proposed model.

3.2 GraphNILM

The proposed GraphNILM model consists of two major components: coarse disaggregation and disaggregation distillation, see Fig. 2. It utilizes convolutional neural networks to extract the representations of power from the aggregated power sequence. The output of the coarse disaggregation component represents the power characteristics of various loads and will be used as the coarse disaggregation results for the disaggregation distilling's inputs. GCN components in our disaggregation distillation section will use these inputs along with the constructed relations as the graph edges to disaggregate the main power to the individual load power in the given house.

(1) Coarse Disaggregation

The coarse disaggregation part is for extracting power characteristics. The input $x_{t-W+1:t}$ is a length W time series representing the aggregated main power within W time stamps. Then we intentionally add the first order difference between two consecutive aggregated main power readings which brings approximately half sigma better performance in the later experiment. F_{seq} represents the convolutional layers in the proposed architecture, and it outputs

$$V_\tau = F_{seq}(x_{t-W+1:t}) = [v_\tau^1, v_\tau^2, \dots, v_\tau^N]^T, \quad (2)$$

where $\tau = t - \frac{W-1}{2}$ representing the middle point of the window W time series, and V_τ stands for the extracted power characteristics of the input sequence $x_{t-W+1:t}$. The intuition behind the midpoint selection is based on the assumption that the model can learn the information of aggregated power before and after the midpoint [21]. These results will be further used as the nodes in the graph structure in the following Disaggregation Distilling part.

(2) Disaggregation Distilling

The introduction of GraphNILM's second component, graph structure, is a novel concept in the NILM field, as the accessible data in popular datasets include no relational information. A graph \mathcal{G} usually consists of nodes set \mathcal{V} and adjacency matrix A and is represented as $\mathcal{G} = \{\mathcal{V}, A\}$. $v_i \in \mathcal{V}$ denotes for the i -th node in \mathcal{V} , which is v_τ^i from equation (2). The adjacency matrix defines the edges $a_{ij} \in A$ and their weights in the graph. These weighted edges are mapping our expectations to utilize relations among loads in the design of our proposed model. Some loads have simultaneous direct relationships, while others may have

asynchronous relations. Weighted edges in a graph can appropriately describe such relationships when they can be quantified.

To leverage the relational strengths of the graph model, we map both the aggregated power and the coarse disaggregated characteristics - obtained from the coarse disaggregation stage - onto the nodes in \mathcal{G} , signifying that $x_\tau, V_\tau \subseteq \mathcal{V}$. To avoid over-fitting, we add x_τ , the mid-point of the aggregated power sequence, as the central node in the graph.

Finding meaningful edges between nodes, i.e., interpreting the relationships between loads in NILM, is the core principle in our graph construction. According to our observations, there are primarily two sorts of relationships in given houses: synchronous and asynchronous. For example, in a given house, the owner prefers to watch television with her food prepared. Before turning on the television, she toasts a slice of bread and then boils some water in the kettle. These events occur sequentially and have strong relationships from an asynchronous perspective: the individual power reading peak for one load occurs close to the power reading peaks for other loads. On the other hand, it is straightforward to conclude that the aggregated power x_τ closely connected with each load at the same time, e.g. the aggregated power would rise at the same time she turns on a new load, a typical synchronous relationship. Though for each house, owners' habits may vary, the major loads for the house are still similar, which makes our pre-trained model transferrable.

For *synchronous* relations, spearman correlations can be used in this model since it measures the strength and direction of monotonic association between two variables. Thus we use it to denote weight edges from the aggregated main power meter to the disaggregated individual load. For *asynchronous* relationships, dynamic time wrapping (DTW) is used to determine the correlations between each load. In both of the typical datasets with which we explored, missing values at different time for different loads hampered our ability to obtain relationships through simple correlation techniques. Therefore, using DTW is a good choice for asynchronous relations. For the calculation of the distance between two load sequences, we define the k -th load with p samples as s_o^k and the l -th load with q samples as s_o^l : $s_o^k = [y_1^k, \dots, y_p^k]$ and $s_o^l = [y_1^l, \dots, y_q^l]$. However, since the power ranges of each load are different and DTW accumulates the absolute distance, meaning two asynchronous well-correlated loads with small power may get smaller results than two asynchronous uncorrelated loads with large power. Therefore, we must normalize the load sequence in order to have meaningful DTW results:

$$s^k = \frac{s_o^k}{E(s_o^{kON})}, \quad (3)$$

where $E(s_o^{kON})$ is the mean of the k -th load's active power when the load turned ON. Then, using DTW algorithm [11], we will obtain the final DTW distance as $\mathcal{D}(s^k, s^l)$. Apply the same rule to all loads and we will get DTW standard distances between loads. Next, we translate standard distances, ranging from 0

to ∞ to range 0 to 1, to ensure the asynchronous and synchronous relations lie in the same range, by

$$r_d = e^{-\mathcal{D}(s^k, s^l)/\alpha}, \quad (4)$$

where α is a scale factor chosen manually based on the average distances. The relations between the load and the aggregated main power, r_s , and the relations between the load and the other load, r_d , may thus be applied to the weighted edges in A in our graph \mathcal{G} . The entire process of how the adjacency matrix A is derived from loads and main power is shown in the Algorithm 1. The proposed method utilizes two weeks of known load data $S = [s^1, \dots, s^N]$ concatenated from Eq. 3 in the given house to perform our algorithm. The standard distance between the k -the load and l -th load is the element positioned at the k -the row/column l -th column/row in A . By resampling the aggregated power sequence X to X^k , which has aligned samples with s^k in the given time, spearman correlation r_s^k can be performed simultaneously. Then the r_s^k could be placed in A 's k -th row/column $N + 1$ -th column/row. With two relations being calculated and put into the appropriate locations in A , the building of the adjacency matrix for the designed weighted graph is completed.

By mapping our design into standard GCN layer [15], our proposed method completes the initial distilling. With the repeated GCN layer nodes fully connected to the MLP layer at the output, the GraphNILM will return N results representing the disaggregated power readings for N separate loads.

Algorithm 1: Adjacency matrix from DTW and correlation

```

input : X, S
output: The adjacency matrix A
1 Start  $\alpha = 1000$ ,  $A = [1]_{N+1,N+1}$ 
2 while not all edges,  $r_s, r_d$ , in A have been computed do
3    $\mathcal{D}(0, 0) \leftarrow 0$                                  $\triangleright$  Initialize the start point
4    $s^k, s^l \sim S$                                      $\triangleright$  Sample  $s^k, s^l$  from S
5    $r_d = \exp(-\mathcal{D}(s^k, s^l)/\alpha)$                  $\triangleright$  Equation (4)
6    $X^k, X^l$                                           $\triangleright$  Resample X to match  $s^k, s^l$ 
7    $r_s^k, r_s^l$                                       $\triangleright$   $X^k, s^k$  and  $X^l, s^l$  to perform correlation
8    $A(k, l) \leftarrow r_d, A(l, k) \leftarrow r_d$ 
9    $A(N + 1, k) \leftarrow r_s^k, A(k, N + 1) \leftarrow r_s^k$ 
10   $A(N + 1, l) \leftarrow r_s^l, A(l, N + 1) \leftarrow r_s^l$ 
11 end

```

4 Experiment

This study involves the examination of two mainstream open-access datasets: REDD [16], UK-DALE [14]. All datasets have labeled appliance-level power consumption along with whole-house power consumption. We also use NILMTK [3] for data prepossessing and comparing results among benchmarking algorithms. All these algorithms are implemented in Python3 and run on NVIDIA QUADRO P5000 GPU. The model is implemented using Pytorch and Pyg. To ensure the consistency of our results, each experiment was performed 20 times with a fixed

seed. The Adam optimizer was employed, with a learning rate 0.005 and the batch size is 1024. The training and testing split is 0.8 vs 0.2. The loss function applied in the proposed model is L2 loss. Standard normalization is used on both input and output with data from each appliance normalized separately. Any negative values post-denormalization are set to 0. An early stop was implemented after 49 non-improved validation losses. The reason for setting the threshold at 49 for monitoring validation loss is due to fluctuations observed in the validation loss. Initially, the model fits into the mean value of each appliance, causing the loss to decrease. However, once the mean is fitted, the model begins the disaggregation process, thereby inducing an increase in validation loss. The validation loss subsequently decreases once the disaggregation pattern is discerned. In addition to Seq2Point and GraphNILM model, Seq2MultiPoint model whose structure is largely similar to that of GraphNILM except the GCN layers is also tested for ablation study. GraphNILM* is for investigating the proposed model without first-order difference. This paper also evaluates the classical FHMM method for comparison.

4.1 Dataset

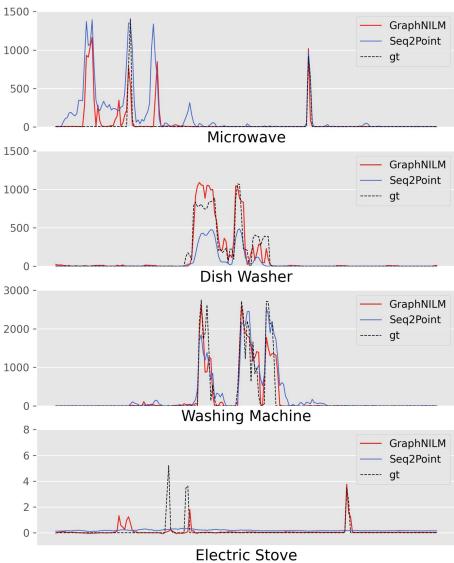


Fig. 3. REDD House 1 case study.

2014-02-01 to 2014-02-14 for the completeness of data during this period; the testing data is from 2014-02-15 to 2014-02-28. DTW relations are calculated from 2014-02-01 3 am to 2014-02-07 3am.

4.2 Metrics

For evaluating the performance, MAE and NEP metrics were chosen since they are the most encountered metrics to assess the disaggregated energy [12]. Mean

The REDD dataset contains 6 residential houses with 17 uniquely labeled appliances in Boston area from April 2011 to Jun 2011. Though it recorded the power consumption at very low rate, extending up to fifty seconds in some cases, we still choose the training data from 2011-04-20 to 2011-04-30 and test from 2011-05-01 to 2011-05-03 for classic dataset results comparison. **The UK-DALE dataset** contains 5 residential houses with 62 load-level unique labels in Southern England from November 2012 to January 2015. The experiment elected to utilize data from house 1 due to the superior number of appliances and data points collected every six seconds for each appliance therein. The training data is from

Absolute Error (MAE): MAE measures how accurately the disaggregated energy is compared to the true energy consumption. Normalized Error in Assigned Power (NEP) is an accuracy measures across different appliances.

$$\text{NEP} = \sqrt{\sum_{i=1}^N \sum_{t=1}^T (\hat{y}_t^i - y_t^i) / \sum_{i=1}^N \sum_{t=1}^T (y_t^i)}. \quad (5)$$

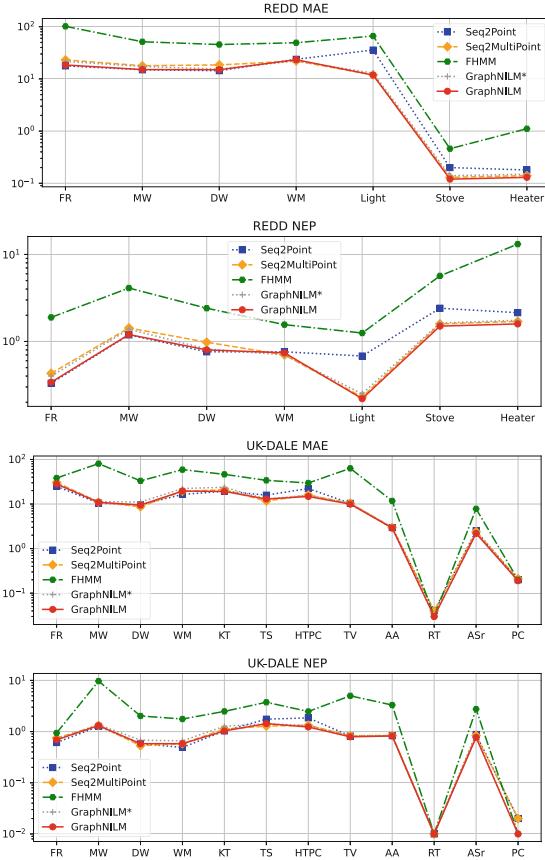


Fig. 4. Comparison w.r.t. MAE and NEP (log y-axis). (left to right) Fridge, Microwave oven, Dish washer, Washing machine, Kettle, Toaster, Audio amplifier, Router, Active subwoofer, Computer.

learning models. The proposed model has 4 best BM model has 3 best performance appliances. Not surprisingly, Seq2Point model shows slightly better MAE and NEP on disaggregating fridge and microwave, whose pattern could be more easily to learn from separate single model. GraphNILM performs closely to Seq2Point models on these three loads. With load

When the requirement is to compare performance across different appliances, the NEP provides a more effective measurement framework. The larger the MAE and NEP values are, the more error is produced by the model compared to the ground truth.

4.3 Results

Figure 3 shows the disaggregation results for 4 devices in REDD, revealing how GraphNILM captures the patterns for different appliances as Seq2Point model does.

REDD. Follow the methodology discussed earlier, seven appliances in REDD dataset were chosen with the results in Fig. 4. Overall the proposed model produces comparable results compared to BM model when we only use 3.9% of memory compared to BM deep learning models. For classic method FHMM which uses least of memory resources, its performance is much worse than all the deep performance appliances while

learning models. The proposed model has 4 best

BM model has 3 best performance appliances. Not surprisingly, Seq2Point model shows slightly better MAE and NEP on disaggregating fridge and microwave,

whose pattern could be more easily to learn from separate single model. GraphNILM performs closely to Seq2Point models on these three loads. With load

relationship taken into consideration, washing machine, light, stove and heater results from the proposed model outperform results from the BM models. To understand the impact of the DTW and GCN layers in GraphNILM, it is compared to Seq2MultiPoint: GraphNILM is better in 6 out of 7 appliances in MAE and NEP, which means the GraphNILM solution is constantly providing extra information needed to disaggregate energy consumption. The first order difference introduced in the proposed design also contributes to the overall better performance by comparing GraphNILM and GraphNILM*.

UK-DALE. The same methodology is adopted here and 12 appliances are chosen in UK-DALE dataset. The result is shown in Fig. 4. In this experiment, the proposed method achieves the same level of performance compared to BM model while only uses 2.6% of memory. Still, Seq2Point produces the best results for fidge and microwave while GraphNILM better disaggregates all other appliances except the washing machine in this dataset. One reason for the better disaggregation in GraphNILM should be the strong asynchronous relationship among loads observed by DTW: the owner of the house usually uses kettle and toaster in a timely order, and computer, audio amplifier, router, active subwoofer are always functioning in nearby time slot. Therefore, these loads with strong relationships converted to graph edge weights in GraphNILM seem to help it outperform other models. When comparing Seq2MultiPoint to the GraphNILM where the only difference is the DTW and GCN layers, the proposed model performs better on 11 out 12 devices in both MAE and NEP, which stresses the importance of the DTW and GCN in the proposed design. With both datasets' results, the proposed model shows a good and reliable performance in general for solving NILM problems. The popular SOA solution Seq2Point trains a dedicated model for each chosen appliance separately hence the total memory size increases linearly along with the number of appliances increases. However in GraphNILM since the number of parameter increase is only nodes and edges in the distilling part, the memory increase is much less significant compared to Seq2Point. Table 1 shows the memory usage for Seq2Point and GraphNILM at window size is 99. The number of parameter in Seq2Point is calculated using NILMTK provided model. To disaggregate one device, Seq2Point requires 3.6 million parameters while the proposed GraphNILM model uses 832 thousands parameters. In a modern home, at least 5 appliances are presented to be disaggregate. In this way more than 80% of the energy consumption could be explained. GraphNILM model only requires 5.2% parameters compared to Seq2Point to disaggregate 5 devices. Besides, in UK-DALE experiment, the runtime for GraphNILM is 98.32 s while Seq2Point requires 254.74 s for

Table 1. Memory usage of Seq2Point and GraphNILM

Load amounts	1	4	8	12
Seq2Point	3.6 MB	14.4 MB	28.8 MB	43.2 MB
GraphNILM	832 KB	832 KB	832 KB	832 KB

12 loads training. Therefore, in terms of memory saving, efficiency and transfer implementation, GraphNILM shows competitive advantage.

Transferability. This paper also did a quick study on the transferability of our model by using our trained model from UK-DALE House 1 to predict UK-DALE House 5. The chosen houses have similar amounts and categories of loads, which is more like the office building usecase. Table 2 shows the proposed model is at least one sigma better than the Seq2MultiPoint model, stressing the DTW and GCN importance again in the proposed design.

Table 2. UK-DALE House 1 Model transferred to House 5 under metric 3*MAE

Model	FR	MW	DW	WM	KT	TS	HTPC	TV	AA	RT	ASr	PC	Overall
GraphNILM	43.15	30.04	22.73	57.20	16.61	5.64	63.17	24.20	23.61	6.03	4.03	13.07	309.53 ± 13.99
Seq2MultiPoint	44.37	29.83	20.87	66.56	18.68	5.33	70.22	20.84	25.10	6.03	2.75	13.08	323.72 ± 13.99

5 Conclusion

GraphNILM outperforms the benchmarks in terms of both the total memory saving, runtime efficiency and the overall MAE performance. Especially for loads with evident relationships, such as the TV, toaster, and kettle groups, the proposed method produces nearly all better results than the current state-of-art method. Even for an independent working device like a fridge or washer, GraphNILM achieves comparable satisfactory results based on MAE and NEP. Given the proposed framework only consumes up to the reciprocal of the total load amount of the memory size in the benchmark, the computational cost of a house with typical loads is drastically reduced. Therefore, extensive experiments conducted on REDD and UK-DALE demonstrate the extraordinary competitiveness of less memory usage and better performance provided by GraphNILM. The deployment of the NILM technique in edge devices for commercial use seems to be around the corner.

References

1. Armel, C., Gupta, A., Shrimali, G., Albert, A.: Is disaggregation the holy grail of energy efficiency? the case of electricity. *Energy Policy* **52**, 213–234 (2013)
2. Batic, D., Tanoni, G., Stankovic, L., Stankovic, V., Principi, E.: Improving knowledge distillation for non-intrusive load monitoring through explainability guided learning. In: 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (ICASSP 2023), pp. 1–5 (2023)
3. Batra, N., et al.: Nilmtk: an open source toolkit for non-intrusive load monitoring (2014)
4. Çimen, H., Çetinkaya, N., Vasquez, J.C., Guerrero, J.M.: A microgrid energy management system based on non-intrusive load monitoring via multitask learning. *IEEE Trans. Smart Grid* **12**(2), 977–987 (2020)

5. De Baets, L., Ruyssinck, J., Develder, C., Dhaene, T., Deschrijver, D.: Appliance classification using VI trajectories and convolutional neural networks. *Energy Build.* **158**, 32–36 (2018). <https://doi.org/10.1016/j.enbuild.2017.09.087>
6. Dong, H., Wang, B., Lu, C.T.: Deep sparse coding based recursive disaggregation model for water conservation. In: Twenty-Third International Joint Conference on Artificial Intelligence (2013)
7. Faustine, A., Mvungi, N.H., Kaijage, S.F., Kisangiri, M.: A survey on non-intrusive load monitoring methodologies and techniques for energy disaggregation problem. arXiv preprint [arXiv:1703.00785](https://arxiv.org/abs/1703.00785) (2017)
8. Ferraz, F.C., Monteiro, R.V.A., Teixeira, R.F.S., Bretas, A.S.: A siamese CNN + KNN-based classification framework for non-intrusive load monitoring. *J. Control Automat. Electric. Syst.*
9. Ghosh, S., Chatterjee, A., Chatterjee, D.: Extraction of statistical features for type-2 fuzzy NILM with IoT enabled control in a smart home. *Exp. Syst. Appl.* **212**, 118750 (2023)
10. Hart, G.W.: Prototype nonintrusive appliance load monitor. Tech. Rep. Progress Report 2, MIT Energy Laboratory (1985)
11. He, K., Stankovic, V., Stankovic, L.: Building a graph signal processing model using dynamic time warping for load disaggregation. *Sensors* **20**, 6628 (2020)
12. Huber, P., Calatroni, A., Rumsch, A., Paice, A.: Review on deep neural networks applied to low-frequency NILM (2021)
13. Kelly, J., Knottenbelt, W.: Neural nilm: deep neural networks applied to energy disaggregation. In: Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments, pp. 55–64 (2015)
14. Kelly, J., Knottenbelt, W.J.: The UK-DALE: a dataset recording UK domestic appliance-level electricity demand and whole-house demand. arXiv preprint [arXiv:1404.0284](https://arxiv.org/abs/1404.0284) (2014)
15. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks (2016)
16. Kolter, J., Johnson, M.: Redd: a public data set for energy disaggregation research. *Artif. Intell.* **25** (2011)
17. Kolter, J., Jaakkola, T.: Approximate inference in additive factorial HMMs with application to energy disaggregation. In: International Conference on Artificial Intelligence and Statistics, pp. 1472–1482 (2012)
18. Liu, B., Zhang, J., Luan, W.: Load oscillation pattern detection for nilm based on scale space decomposition. In: 2023 8th Asia Conference on Power and Electrical Engineering (ACPEE), pp. 1590–1595 (2023)
19. Liu, Y., Xu, Q., Yang, Y., Zhang, W.: Detection of electric bicycle indoor charging for electrical safety: a NILM Approach. *IEEE Trans. Smart Grid* **14**(5), 3862–3875 (2023)
20. Mauch, L., Yang, B.: A New Approach for Supervised Power Disaggregation by Using a Deep Recurrent LSTM Network, pp. 63–67 (2015)
21. Oord, A.v.d., et al.: Wavenet: a generative model for raw audio (2016)
22. Parson, O., Ghosh, S., Weal, M., Rogers, A.: Non-intrusive load monitoring using prior models of general appliance types (2012)
23. Sykiotis, S., et al.: Performance-aware NILM model optimization for edge deployment. *IEEE Trans. Green Commun. Netw.* **7**(3), 1434–1446 (2023). <https://doi.org/10.1109/TGCN.2023.3244278>
24. Vine, D., Buys, L., Morris, P.: The effectiveness of energy feedback for conservation and peak demand: a literature review. *Open J. Energy Efficiency* **2**, 2013 (2013)

25. Zhang, B., Zhao, S., Shi, Q., Zhang, R.: Low-rate non-intrusive appliance load monitoring based on graph signal processing. In: 2019 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC) (2019)
26. Zhang, C., Zhong, M., Wang, Z., Goddard, N., Sutton, C.: Sequence-to-point learning with neural networks for nonintrusive load monitoring (2016)
27. Zhao, B., Stankovic, L., Stankovic, V.: Blind non-intrusive appliance load monitoring using graph-based signal processing. In: 2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP) (2015)



A Contraction Tree SAT Encoding for Computing Twin-Width

Yinon Horev¹, Shiraz Shay¹, Sarel Cohen¹, Tobias Friedrich², Davis Issac², Lior Kamma¹, Aikaterini Niklanovits², and Kirill Simonov^{2(✉)}

¹ The Academic College of Tel Aviv-Yaffo, Tel Aviv-Yaf, Israel
`{yinonho,shirazsh,sarelco,liorkm}@mta.ac.il`

² Hasso Plattner Institute, University of Potsdam, Potsdam, Germany
`{tobias.friedrich,davis.issac,aikaterini.niklanovits,kirill.simonov}@hpi.de`

Abstract. Twin-width is a structural width parameter and matrix invariant introduced by Bonnet et al. [FOCS 2020], that has been gaining attention due to its various fields of applications. In this paper, inspired by the SAT approach of Schidler and Szeider [ALENEX 2022], we provide a new SAT encoding for computing twin-width. The encoding aims to encode the contraction sequence as a binary tree. The asymptotic size of the formula under our encoding is smaller than in the state-of-the-art relative encoding of Schidler and Szeider. We also conduct an experimental study, comparing the performance of the new encoding and the relative encoding.

Keywords: Twin-width · SAT encoding

1 Introduction

Twin-width is a graph and matrix invariant recently introduced by Bonnet et al. [4, 5, 7], inspired by a width invariant defined on permutations by Guillemot and Marx [9]. Since its inception, twin-width received tremendous interest in the scientific community. From the algorithmic perspective, the benefits of twin-width are twofold. First, many diverse graph families are known to have bounded twin-width, for example graphs of bounded treewidth or clique-width, graphs excluding a fixed minor, planar graphs, posets of bounded width (in particular, unit interval graphs) [7]. Second, many NP-hard problems are solvable in polynomial time on graphs of bounded twin-width.

The latter property is formalized by Bonnet et al. [7] as follows: Given an n -vertex graph G , a witness that its twin-width is at most d , and a first-order sentence ϕ , it can be decided whether ϕ holds on G in $f(d, \phi)n$ time, where f is some computable function. It is worth noting, that this result does not give directly algorithms with practical running times since f is an extremely fast-growing function; this is a common drawback of algorithmic meta-theorems. However, several important NP-hard problems that are expressible by a first-order sentence

of bounded size, are also known to be solvable on graphs of bounded twin-width directly via dynamic programming with efficient running times; in particular, k -CLIQUE, k -DOMINATING SET, and k -ONES SAT for bounded k [5, 8].

Given the above, the natural question is, whether the twin-width of a graph can be computed exactly. That is, given a graph G , can we find the smallest d such that d -contraction sequence exists? Unfortunately, in the general case this turns out to be intractable: even deciding whether a graph has twin-width 4 is NP-complete [2]. While for many graph classes twin-width is bounded, very few results are known for computing (or approximating) twin-width even when the given graph comes from a special graph class. For example, Král and Lamaison [11] showed that planar graphs have twin-width at most 8; however it is wide open whether we can compute the twin-width of a *given* planar graph (since it may also be smaller than 8). This motivates turning our attention to heuristic methods of computing twin-width. The high demand for such results is also illustrated by the 2023 edition of the PACE challenge¹, which focuses exclusively on computing twin-width. For the purpose of this work we only consider algorithms that yield a provably optimal contraction sequence; however, the running time is not necessarily bounded by a polynomial in n in general.

This line of research was pioneered by Schidler and Szeider [14], who devised a SAT encoding for computing twin-width. By supplying that encoding to a SAT solver, they were able to identify the exact value of twin-width for a variety of named graphs. In fact, they presented two different SAT encodings called *absolute* and *relative*. Interestingly, in all their tests the relative encoding vastly outperforms the absolute encoding, despite the fact that the formula in the relative encoding is larger: $\mathcal{O}(n^4)$ clauses versus $\mathcal{O}(n^3)$ clauses for the absolute encoding for an n -vertex graph. To the best of our knowledge, no other SAT encodings for computing twin-width were studied so far.

Our Contribution. In this work we propose an alternative SAT encoding for computing twin-width, which is conceptually different from the encodings of Schidler and Szeider [14]. We prove formally the correctness of the encoding, and argue that the size of the formula in the encoding is only $\mathcal{O}(n^3)$ clauses, which is asymptotically smaller than $\Omega(n^4)$ in the relative encoding of [14]. We also supply an implementation of the encoding, and conduct empirical tests comparing the performance of our encoding to that of the state-of-the-art relative encoding. The results show that there are cases where the new encoding allows to compute twin-width much faster, although the converse happens as well. We highlight that our main contribution is presenting a new SAT-encoding based on a binary contraction tree that is significantly different than the encoding presented so far. In the context of our theoretical analysis, we prove the correctness of our

¹ PACE stands for Parameterized Algorithms and Computational Experiments; the challenge is dedicated to bringing the gap between theoretical and practical parameterized algorithms. The official website of the challenge is <https://pacechallenge.org/2023/>.

SAT-encoding and analyze the big-O size of the formula, while the experimental part is mainly to empirically validate the correctness and feasibility of the encoding.

Related Work. Here we list some of the known results on twin-width. Graphs of twin-width 0 are exactly cographs, and can be recognized in poly-time [7]. Later it was shown that graphs of twin-width 1 can also be recognized in poly-time [6]. Jacob and Pilipczuk [10] show, among other results, that twin-width of a graph is at most $3 \cdot 2^{\text{tw}-1}$, where tw is the treewidth of the graph; it is also known that twin-width can be exponential in treewidth [3]. Balabán and Hliněný [1] show that twin-width is linear in the poset width, which implies that twin-width of unit interval graphs is at most two, and can be computed in poly-time. Twin-width of planar graphs is at most 8 [11], and can be as high as 7 [12].

2 Preliminaries

All graphs mentioned in this paper are simple, undirected and finite and we use standard graph-theoretic notations. In particular, given a graph G , we denote by $V(G)$ and $E(G)$ its set of vertices (or nodes) and edges respectively. Moreover, given a vertex set $S \subseteq V(G)$ we denote by $G[S]$ and $G - S$, the graph induced by the vertices of S and the graph induced by the vertices $V(G) - S$ respectively. When referring to the *open neighborhood* of a vertex $v \in V(G)$, i.e. the set of neighbors of v without v , we write $N_G(v)$, while we omit G when the graph we refer to is clear from the context. Similarly, we denote by $N_G[v]$ the *closed neighborhood* of v , i.e. $N_G(v) \cup v$. Two distinct vertices u, v are called *false twins* if $N(u) = N(v)$ and *true twins* if $N[u] = N[v]$. Given a pair $u, v \in V(G)$ we characterize the process of deleting those vertices and creating a new one with neighborhood $N(u) \cup N(v)$ as *contraction of u, v* . The graph that occurs from G after the contraction of u, v is denoted by $G/u, v$.

We now proceed in defining twin-width, following the definition of Bonnet et al. in [7]: A graph $G = (V, B, R)$ is a *trigraph* if B and R are two disjoint sets of edges on V (referred to as black and red respectively). Note that an ordinary graph is a trigraph where $B = E(G)$ and $R = \emptyset$. A trigraph (V, B, R) such that (V, R) has maximum degree d is called a d -trigraph. The neighborhood of a vertex v on a trigraph is all of its adjacent vertices, regardless of the color of the edge that connects them. Given a trigraph $G = (V, B, R)$ and a pair of distinct vertices $u, v \in V(G)$ we define the trigraph $G' = G/u, v$ such that, for the neighbors of the vertex w that occur from the contraction the following holds: A vertex $x \in N(w)$ is connected to w through a black edge if and only if it was connected through a black edge to both u, v in G . Otherwise (if at least one was already connected through a red edge or if x was not adjacent to both of the contracted vertices), the edge connecting x to w in G' is red.

A sequence of d -contractions for a trigraph is a sequence of d -trigraphs G_0, G_1, \dots, G_{n-1} , where $G_0 = G$, $|V(G_{n-1})| = 1$ and G_i for $i \geq 1$ is obtained

from G_{i-1} by a contraction (see Fig. 1 for an example). The twin-width of a graph is the smallest d for which a d -sequence exists and is denoted by $tw(G)$. Such a sequence is called a *d -contraction sequence*.

Since the main result of this paper is based on encoding a contraction sequence as a binary tree, we also define what the Conjunctive Normal Form (CNF) of a formula is. A literal is a (propositional) variable or the negation of it, and we call a clause a disjunction of literals. A formula ϕ is in Conjunctive Normal Form (CNF) if it is the conjunction of clauses.

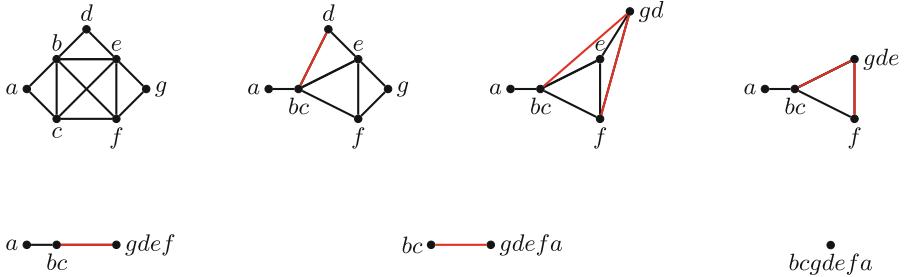


Fig. 1. A 2-contraction sequence of a graph.

3 Binary SAT Encoding

To obtain our binary SAT encoding of the d -contraction sequence problem we express all restrictions into propositional logic and then convert them to CNF. Throughout the description of our encoding we refer to $V(G)$ as vertices, and to $V(T)$ as nodes.

Let $G = (V, E)$ be the input graph and consider an initialization of E , denoting by $\neg edge_{i,j}$ the non-existing edges between vertices i, j , by $edge_{i,j}$ the existing ones, and by $\neg red_{i,j}$ the absence of red edges on the input graph.

$$\left(\bigwedge_{\substack{i, j \in [n], j > i \\ ij \notin E}} \neg edge_{i,j} \right) \wedge \left(\bigwedge_{\substack{i, j \in [n], j > i \\ ij \in E}} edge_{i,j} \right) \wedge \left(\bigwedge_{i, j \in [n], j > i} \neg red_{i,j} \right)$$

Observe that a contraction sequence can be represented by a rooted binary tree, where the leaves correspond to the vertices of G , internal nodes to subsequent contractions, and the root corresponds to the final contraction. Since twin-width is always at most d when the number of vertices does not exceed $d + 1$, we can avoid encoding the final $d + 1$ contractions in the sequence. In terms of the tree representation, this corresponds to removing the nodes of the

tree representing the final $d + 1$ contractions, which results in a binary forest with at most $d + 2$ trees. We refer to the binary forest of the sought-after optimal contraction sequence as T . We note that $|V(T)| \leq 2n - d - 2$ since the whole contraction tree contains $2n - 1$ nodes as a binary tree with $n = |V(G)|$ leaves, and the nodes corresponding to the final $d + 1$ contractions are not considered.

To encode T we define variables $lc_{i,j}$ and $rc_{i,j}$ for each pair of vertices, which are true if and only if j is the left child of i and j is the right child of i respectively. Using these variables we ensure that T is binary by encoding the following:

- Each node has at most one parent.

$$\bigwedge_{i \in [2n-d-3]} \left\langle \sum_{j=\max(i+1, n+1)}^{2n-d-2} (lc_{j,i} + rc_{j,i}) \leq 1 \right\rangle$$

Note that the final vertex is not included in this encoding.

- Each parent node has exactly one left and one right child.

$$\bigwedge_{j \in [n+1, 2n-d-2]} \left\langle \sum_{i \in [j]} rc_{j,i} = 1 \right\rangle \quad \bigwedge_{j \in [n+1, 2n-d-2]} \left\langle \sum_{i \in [j]} lc_{j,i} = 1 \right\rangle$$

Lemma 1. *The encoding above produces a binary forest on $2n - d - 2$ nodes.*

Proof. Assume that there is a node v_i that has at least two parents v_x, v_y , where $x, y > i$. Then the clause $\bigwedge_{i \in [2n-d-3]} \left\langle \sum_{j=\max(i+1, n+1)}^{2n-d-2} (lc_{j,i} + rc_{j,i}) \leq 1 \right\rangle$ corresponding to v_i is false since $\sum_{j=\max(i+1, n+1)}^{2n-d-2} (lc_{j,i} + rc_{j,i})$ includes $(lc_{x,i} + rc_{x,i}) + (lc_{y,i} + rc_{y,i})$ which is equal to 2, leading to contradiction. Note that we construct those clauses only considering values greater than n , since all the vertices of G correspond to leaf nodes.

Regarding the fact that each parent has exactly one left and one right child, we do not consider the leaves and hence we start from $n + 1$. Here it suffices to note that for each possible parent node we check all the possible right and left children to ensure that it has exactly one of each.

We proceed with encoding the already contracted vertices through the variable $vanish_{p,i}$. In particular, we define this variable to be true if i is contracted to any vertex with number at most p . Intuitively, this variable is true if and only if at the time when the node p is formed through some contraction, the node i doesn't exist any more. To implement the semantics we encode the following:

- Each leaf node is vanished at the moment its parent node is formed and hence, for each $i \in [n]$

$$vanish_{n+1,i} \iff lc_{n+1,i} \vee rc_{n+1,i}$$

- Iteratively defined, an internal node i is already vanished when p is formed, either if it is one of its children or if it has been vanished in some previous

contraction, due to it being a child of some other contracted node. Hence, for each $p \in [n+2, 2n-d-2]$, $i \in [p-1]$

$$\text{vanish}_{p,i} \iff lc_{p,i} \vee rc_{p,i} \vee \text{vanish}_{p-1,i}$$

- Lastly, in order to ensure that no node is considered to be “contracted to itself” at the time it is formed, we set for each $p \in [n+1, 2n-d-2]$

$$\neg \text{vanish}_{p,p}$$

We now encode the edges of G between the children of some node and the other nodes. In particular, the left adjacency of i, p , denoted $la_{i,p}$, means that there is an edge between the vertex represented by the left child of p and the one represented by i . Similarly the right adjacency of i, p is defined, and denoted by $ra_{i,p}$. In the list encoding these adjacencies below, the first argument refers to the child of p while the second denotes whether or not an edge (red or black) to vertex c exists. The minimum value is placed first when choosing the edge, to be consistent with how we initially encoded $E(G)$. For each $p \in [n+1, 2n-d-2]$, $c \in [p-1]$, $i \in [p-1]$ such that $i \neq c$, and similarly for their negation

$$\begin{array}{ll} lc_{p,c} \wedge \text{edge}_{\min(i,c), \max(i,c)} \Rightarrow la_{i,p}, & rc_{p,c} \wedge \text{edge}_{\min(i,c), \max(i,c)} \Rightarrow ra_{i,p} \\ lc_{p,c} \wedge \text{red}_{\min(i,c), \max(i,c)} \Rightarrow lr_{i,p}, & rc_{p,c} \wedge \text{red}_{\min(i,c), \max(i,c)} \Rightarrow rr_{i,p} \\ lc_{p,c} \wedge \neg \text{edge}_{\min(i,c), \max(i,c)} \Rightarrow \neg la_{i,p}, & rc_{p,c} \wedge \neg \text{edge}_{\min(i,c), \max(i,c)} \Rightarrow \neg ra_{i,p} \\ lc_{p,c} \wedge \neg \text{red}_{\min(i,c), \max(i,c)} \Rightarrow \neg lr_{i,p}, & rc_{p,c} \wedge \neg \text{red}_{\min(i,c), \max(i,c)} \Rightarrow \neg rr_{i,p} \end{array}$$

Using the right and left adjacencies of each node, we are able to encode the edges created from the contractions. In particular, an edge connecting node i that occurs from some contraction to another node j exists, if any of the children of i (the ones that get contracted in order to create i) is adjacent in G to j , and j still exists at the moment i is created. Moreover, this edge is red if exactly one of those children is adjacent to j . Formally, this is encoded as follows:

For each $i \in [n+1, 2n-d-2]$, $j \in [i-1]$

$$\begin{aligned} \text{edge}_{j,i} &\iff (la_{j,i} \vee ra_{j,i}) \wedge \neg \text{vanish}_{i,j} \\ \text{red}_{j,i} &\iff \text{edge}_{j,i} \wedge (lr_{j,i} \vee rr_{j,i} \vee (la_{j,i} \oplus ra_{j,i})) \end{aligned}$$

The encoding of the edges occurring from the contraction we described above $\text{edge}_{j,i} \iff (la_{j,i} \vee ra_{j,i}) \wedge \neg \text{vanish}_{i,j}$ is converted to CNF as follows:

$$\begin{aligned} (\neg \text{edge}_{j,i} \vee la_{j,i} \vee ra_{j,i}) \wedge (\neg \text{vanish}_{i,j} \vee \neg \text{edge}_{j,i}) \wedge (\neg la_{j,i} \vee \text{edge}_{j,i} \vee \text{vanish}_{i,j}) \\ \wedge (\neg ra_{j,i} \vee \text{edge}_{j,i} \vee \text{vanish}_{i,j}) \end{aligned}$$

Similarly for the encoding of the red edges, $\text{red}_{j,i} \iff \text{edge}_{j,i} \wedge (lr_{j,i} \vee rr_{j,i} \vee (la_{j,i} \oplus ra_{j,i}))$ we have:

$$\text{red}_{j,i} \iff \text{edge}_{j,i} \wedge (lr_{j,i} \vee rr_{j,i} \vee ((la_{j,i} \vee ra_{j,i}) \wedge (\neg la_{j,i} \vee \neg ra_{j,i})))$$

$$\begin{aligned}
red_{j,i} \iff & (edge_{j,i}) \wedge (lr_{j,i} \vee rr_{j,i} \vee la_{j,i} \vee ra_{j,i}) \\
& \wedge (lr_{j,i} \vee rr_{j,i} \vee \neg la_{j,i} \vee \neg ra_{j,i}) \\
\Rightarrow & (\neg red_{j,i} \vee edge_{j,i}) \wedge (\neg red_{j,i} \vee lr_{j,i} \vee rr_{j,i} \vee la_{j,i} \vee ra_{j,i}) \\
& \wedge (\neg red_{j,i} \vee lr_{j,i} \vee rr_{j,i} \vee \neg la_{j,i} \vee \neg ra_{j,i}) \\
\Leftarrow & (red_{j,i} \vee \neg edge_{j,i} \vee \neg lr_{j,i}) \wedge (red_{j,i} \vee \neg edge_{j,i} \vee \neg rr_{j,i}) \\
& \wedge (red_{j,i} \vee \neg edge_{j,i} \vee \neg la_{j,i} \vee ra_{j,i}) \wedge (red_{j,i} \vee \neg edge_{j,i} \vee \neg ra_{j,i} \vee la_{j,i})
\end{aligned}$$

Now, in order to encode the red un-vanished edges at the moment of a contraction, to be able to restrict the maximum degree we introduce a new variable $reduv_{i,j,k}$. This is created at the moment node i is formed, and a red edge between the nodes j and k exists (and the nodes j and k of course still have not vanished). For each $i \in [n+1, 2n-d-2]$, for each $j \in [i]$, for each $k \in [j+1, i]$,

$$reduv_{i,j,k} \iff \neg vanish_{i,j} \wedge \neg vanish_{i,k} \wedge red_{j,k}$$

This is expressed in CNF as:

$$\begin{aligned}
(\neg vanish_{i,j} \vee \neg reduv_{i,j,k}) \wedge (\neg vanish_{i,k} \vee \neg reduv_{i,j,k}) \wedge (red_{j,k} \vee \neg reduv_{i,j,k}) \\
\wedge (vanish_{i,j} \vee vanish_{i,k} \vee \neg red_{j,k} \vee reduv_{i,j,k})
\end{aligned}$$

Lastly we need to ensure that the maximum red degree at any point is at most d which is expressed as

$$\bigwedge_{\substack{i \in [n+1, 2n-d-2], \\ j \in [i]}} \left\langle \sum_{k \in [i], j \neq k} reduv_{i,\min(j,k),\max(j,k)} \leq d \right\rangle$$

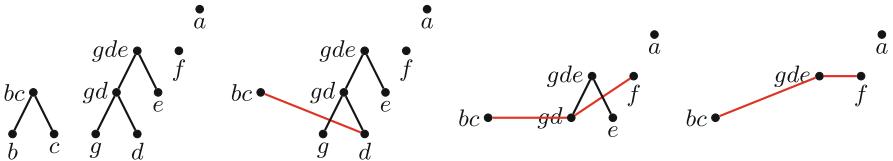


Fig. 2. The contraction forest and the sequence up to $2n - d - 2$ for $d = 2$ that occurs through the binary encoding for the 2-contraction sequence of Fig. 1. Observe that for $d = 2$ we stop once 4 vertices remain since after the next contraction a graph on 3 vertices occurs which can have at most 2 red degree.

Theorem 1. *Given a graph G of order n and an integer d , we construct in polynomial time a CNF formula which is satisfiable if and only if $tw(G) \leq d$, and has $\mathcal{O}(n^3)$ clauses.*

Proof. We first prove that given an assignment of the variables that satisfies the SAT formula corresponding to a graph G , we are able to build a d -contraction sequence for G . First, we create the contraction tree following the parent-child relations occurring by the assignment values for the variables $lc_{i,j}$, $rc_{j,i}$. Then, we construct a contraction sequence based on the children of $n_1, \dots, 2n - d - 2$. By the construction of our formula we ensure that all edges of G are encoded. Also, for every contraction node i , the assignment denoting whether an edge between its children and other nodes exists, and the red edges created through the contraction of a pair of nodes, are counted, as the clauses equivalent to $edge_{j,i} \iff (la_{j,i} \vee ra_{j,i}) \wedge \neg vanish_{i,j}$ are satisfied. Lastly, if during this contraction sequence the red degree becomes greater than d , the clauses equivalent to $\bigwedge_{i \in [n+1, 2n-d-2], j \in [i]} \sum_{k \in [i], j \neq k} reduv_{i, \min(j,k), \max(j,k)} \leq d$ are not satisfied, leading to a contradiction. Hence, the sequence occurring from contracting the child nodes of $n + 1, \dots, 2n - d - 2$ is indeed a d -contraction sequence of G .

Given a d -contraction sequence of G we construct an assignment that satisfies the formula corresponding to G as follows: For the i^{th} -contraction pair, with the contracted vertices being x and y , where $i \leq 2n - d - 2$ we assign 1 to $lc_{x,n+i}, rc_{y,n+i}$ and 0 for the other values of i . Hence, we have “built” our contraction tree. We also initialize $edge_{i,j}$ to 1 for the adjacent pairs of vertices in G , and $red_{i,j}$ to 0 for all pairs of vertices, to encode G . Similarly for right and left adjacencies of the nodes of T . We then follow the changes each contraction causes on G , and assign truth values to the respective variables. For example, for each variable $vanish_{p,i}$ if the vertex i has been contracted when it is p ’s turn to participate in some contraction, we assign 1 to it. Similarly, we assign 1 to $reduv_{i,j,k}$ for the red edges that survive each contraction. Notice that because a d -contraction sequence of G is given, due to the construction of our formula, the assignment created so far also satisfies the clauses of $\bigwedge_{i \in [n+1, 2n-d-2], j \in [i]} \sum_{k \in [i], j \neq k} reduv_{i, \min(j,k), \max(j,k)} \leq d$, satisfying the CNF corresponding to G .

We now analyze the size of the CNF formula created from a graph G and an integer d , by counting the clauses occurring from each information encoding. For initializing $E(G)$, we use one variable for each edge, one for each non-edge and one to denote the absence of red edges, hence $\mathcal{O}(n^2)$ in total. To encode the contraction tree we use, for the “one-parent property”, for each $i \in [2n - d - 3]$ at least $2(n - d - 2)$ variables, meaning $\mathcal{O}(n^2)$, and for the “binary property”, for each $j \in [n + 1, 2n - d - 2]$, j variables for each child, hence $\mathcal{O}(n^2)$ in total. When converted to CNF (the same holds for bounding the red degree by d , by [13]) we get $\mathcal{O}(n^2)$ clauses. For $vanish_{p,i}$ we get two literals for each leaf, at most $2n - d - 2$ for each of the internal nodes $2n - d - 2$ for the “self variables”, meaning $\mathcal{O}(n^2)$ in total. To encode right and left adjacencies (edges between the vertices represented by the children of a node and vertices represented by other nodes) we have $\mathcal{O}(n^2)$ variables to choose which child to refer to, and $\mathcal{O}(n)$, meaning $\mathcal{O}(n^3)$ in total.

Table 1. Results for the PACE dataset, random graphs, and Paley graphs. The numbers under “binary” and “relative” are the running times of the encodings, in seconds.

name	n	m	tww	binary	relative	name	n	m	tww	binary	relative
EX_001	19	64	6	12.86	9.23	ER_0.1	30	44	3	304.23	37.23
EX_002	20	69	6	4.27	1.15	ER_0.2	30	100	6	100.81	933.68
EX_003	25	97	6	48.64	85.84	ER_0.3	30	120	7	587.99	703.2
EX_004	25	181	7	107.33	68.72	ER_0.4	—	—	—	—	—
EX_006	28	131	7	1390.23	480.01	ER_0.5	30	236	9	375.13	319.4
EX_007	28	205	7	178.24	181.24	ER_0.6	—	—	—	—	—
EX_008	28	210	10	26.83	8.68	ER_0.7	30	293	8	409.17	—
EX_009	28	228	7	98.99	68.77	ER_0.8	30	342	6	502.77	—
EX_010	28	235	6	735.61	438.79	ER_0.9	30	381	4	—	1418.76
EX_011	29	174	8	—	1077.78	P_013	13	39	6	0.25	0.1
EX_012	29	180	8	—	1163.65	P_017	17	68	8	0.83	0.25
EX_013	30	155	8	—	1498.19	P_029	29	203	14	14.27	7.52
EX_014	30	175	8	1026.46	873.93	P_037	37	333	18	85.19	35.83
EX_015	30	178	8	414.99	858.01	P_041	41	410	20	175.01	58.7
EX_016	30	195	8	—	1527.49	P_053	53	689	26	660.1	352.25
EX_017	30	207	8	—	409.64						
EX_018	31	52	3	648.12	175.37						
EX_019	32	90	5	1578.88	—						
EX_031	48	80	3	974.02	1206.32						
EX_034	51	240	4	484	16.66						
EX_035	52	53	2	253.07	135.8						

When encoding the existence of an edge as a relation of right and left adjacencies and the variable *vanish*, we get 4 clauses for each edge variable, and 4 for the existence of each red edge, so $\mathcal{O}(n^2)$ clauses in total. Similarly 4 clauses are created for each variable *reduv* that represents the existence of red edges at the moment of a contraction, making them $\mathcal{O}(n^3)$ in total. Lastly, for bounding the maximum red degree by d we need at most $\mathcal{O}(n^2)$ variables, which also produce $\mathcal{O}(n^2)$ clauses. Hence, the size of the formula produced by our encoding is $\mathcal{O}(n^3)$.

4 Experiments

We implemented the binary SAT encoding presented above and run it on several datasets. For the implementation, we used Python 3.10.12 with PySAT 3.1.0, and Cadical as the specific SAT solver. The tests were run on a PC with AMD Ryzen 7 6800HS CPU, 16 GB RAM, and Ubuntu 22.04, using only a single

Table 2. Comparison between binary encoding and *absolute* encoding on the tiny dataset of PACE.

name	n	m	tww	time, absolute	time, binary	factor
tiny001.gr	10	9	1	1.48	0.09	x16
tiny002.gr	10	10	2	3.7	0.13	x29
tiny005.gr	25	40	3	>10 h	21.35	>x2000
tiny007.gr	14	13	2	3131.77	1.16	x2689
tiny008.gr	10	15	4	65.80	0.07	x937
tiny009.gr	7	7	1	0.01	0.03	x0.33

thread. We also include an implementation of the relative encoding of [14], and compare the performance of our binary SAT encoding with the relative encoding. To make a fair comparison, both encodings were run using exactly the same settings and auxiliary code. We generally do not compare with the absolute encoding of [14] (except for Table 2), as it is hopelessly inefficient compared to both relative encoding and our binary encoding. We also implemented modular decomposition as the standard preprocessing step for computing twin-width; for detailed description of the preprocessing see [14]. The implementations and the testing data are provided in the supplementary material.

Next, we describe the datasets that were used for testing, and provide tables that compare performance of the binary encoding and the relative encoding. In all tests the computed twin-width value is the same for both encodings (as both are shown to output the optimal value of twin-width), so we only compare the time used by each encoding. In the results that we list, time is always measured in seconds. All tests were run with a time limit of 30 min (1800 s) and a memory limit of 10 gigabytes. Entries marked with “-” are used to denote that the solver exceeded time and/or memory limit on the corresponding instance.

Random Graphs. We first construct Erdős–Rényi graphs $G(n, p)$ where $n = 30$, and p ranges from 0.1 to 0.9 with an increment of 0.1. The graph $G(n, p)$ contains n vertices, and each edge is created with probability p . The results for Erdős–Rényi graphs are listed in Table 1 in the “ER” rows. It is interesting to note that while the relative encoding performs slightly better in the uniform setting ($p = 0.5$), and considerably better when the graph is very sparse ($p = 0.1$) or very dense ($p = 0.9$), the binary encoding vastly outperforms the relative encoding in the intermediate cases ($p \in \{0.2, 0.3, 0.7, 0.8\}$).

Moreover, the performance of both solvers is worse for larger p compared to smaller p ; this implies that neither encoding is exploiting to the fullest the property that the twin-width of a graph is always equal to the twin-width of its complement. That is, computing the twin-width of $G(n, p)$ should be equally hard as computing the twin-width of $G(n, 1 - p)$, since in the complement of $G(n, p)$ each edge exists independently with probability $1 - p$.

PACE 2023 Challenge. We next compare the encodings on several instances from the PACE challenge. We first compare the binary encoding and the absolute encoding of [14] on the “tiny” sample set of the challenge². The results are shown in Table 2, and highlight that the performance of the absolute encoding is much worse than that of the binary encoding, despite the same asymptotic size of the encoding. For this reason, we do not include absolute encoding in the other tests.

Table 1 then shows the result of comparison between the binary and the relative encoding on the regular instances of the challenge, see rows starting with “EX”. While the original dataset contains 200 graphs³, we only list the instances where at least one of the two solvers was able to compute the twin-width under the time and memory limit of 30 min and 10GB. While the relative encoding is generally faster on this dataset, the binary encoding is still able to perform considerably better on several instances. This further shows that the binary encoding is conceptually different from the relative, and may be used to deal with the instances where the relative encoding is not efficient.

Paley Graphs. Finally, following the experiments of [14], we test the encodings on Paley graphs. We construct Paley graphs for several prime numbers; for a prime p with $p \equiv 1 \pmod{4}$, Paley graph is a $\frac{p-1}{2}$ -regular graph on p vertices. The vertices of the Paley graph are associated with the elements of the unique finite field of order p , and the edge between x and y appears if and only if $x - y$ is a square in the field; in the case $p \equiv 1 \pmod{4}$, $x - y$ is a square if and only if $y - x$ is a square so the edges are symmetric. Last part of Table 1 lists the results. While the relative encoding is generally faster, it is interesting to observe that the memory usage of the binary encoding is lower on the Paley graphs. This may be attributed to the smaller size of the encoding, although this effect does not show on the other instances.

5 Conclusion

In this work, we introduced a novel SAT encoding for computing twin-width of a graph. Theoretically, we have shown that the encoding is sound, and that the size of the encoding is smaller than that of the state-of-the-art relative encoding of [14]. Further, we conducted experiments on several datasets, comparing the performance of the binary encoding, and the relative and absolute encoding of [14]. Similar to relative encoding, binary encoding vastly outperforms the absolute encoding. Comparing the binary and relative encoding, the results suggest that neither encoding dominates the other, while in many cases the binary encoding is much more efficient than the relative encoding (although the converse also happens). Therefore, the introduction of the binary encoding indeed improves the ability to compute twin-width in practice. This also motivates further work

² <https://pacechallenge.org/2023/tiny-set.pdf>.

³ All instances are available at <https://pacechallenge.org/2023/>.

on comparing different twin-width encodings empirically, identifying families of instances where twin-width can be computed efficiently with either encoding.

References

1. Balabán, J., Hliněný, P.: Twin-width is linear in the Poset width. In: Golovach, P.A., Zehavi, M. (eds.) 16th International Symposium on Parameterized and Exact Computation (IPEC 2021). Leibniz International Proceedings in Informatics (LIPIcs), vol. 214, pp. 6:1–6:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl (2021). <https://doi.org/10.4230/LIPIcs.IPEC.2021.6>
2. Bergé, P., Bonnet, É., Déprés, H.: Deciding twin-width at most 4 is np-complete. In: Bojanczyk, M., Merelli, E., Woodruff, D.P. (eds.) 49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, 4–8 July 2022, Paris. LIPIcs, vol. 229, pp. 18:1–18:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2022). <https://doi.org/10.4230/LIPIcs.ICALP.2022.18>
3. Bonnet, E., Déprés, H.: Twin-width can be exponential in treewidth. *J. Comb. Theory Ser. B* **161**(C), 1–14 (2023). <https://doi.org/10.1016/j.jctb.2023.01.003>
4. Bonnet, É., Geniet, C., Kim, E.J., Thomassé, S., Watrigant, R.: Twin-width II: small classes. In: Marx, D. (ed.) Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, 10–13 January 2021, pp. 1977–1996. SIAM (2021). <https://doi.org/10.1137/1.9781611976465.118>
5. Bonnet, É., Geniet, C., Kim, E.J., Thomassé, S., Watrigant, R.: Twin-width III: max independent set, min dominating set, and coloring. In: Bansal, N., Merelli, E., Worrell, J. (eds.) 48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, 12–16 July 2021, Glasgow (Virtual Conference). LIPIcs, vol. 198, pp. 35:1–35:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2021). <https://doi.org/10.4230/LIPIcs.ICALP.2021.35>
6. Bonnet, E., Kim, E.J., Reinald, A., Thomassé, S., Watrigant, R.: Twin-width and polynomial kernels. In: Golovach, P.A., Zehavi, M. (eds.) 16th International Symposium on Parameterized and Exact Computation (IPEC 2021). Leibniz International Proceedings in Informatics (LIPIcs), vol. 214, pp. 10:1–10:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl (2021). <https://doi.org/10.4230/LIPIcs.IPEC.2021.10>
7. Bonnet, É., Kim, E.J., Thomassé, S., Watrigant, R.: Twin-width I: tractable FO model checking. *J. ACM* **69**(1), 3:1–3:46 (2022). <https://doi.org/10.1145/3486655>
8. Ganian, R., Pokrývka, F., Schidler, A., Simonov, K., Szeider, S.: Weighted model counting with twin-width. In: Meel, K.S., Strichman, O. (eds.) 25th International Conference on Theory and Applications of Satisfiability Testing, SAT 2022, 2–5 August 2022, Haifa. LIPIcs, vol. 236, pp. 15:1–15:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2022). <https://doi.org/10.4230/LIPIcs.SAT.2022.15>
9. Guillemot, S., Marx, D.: Finding small patterns in permutations in linear time. In: Chekuri, C. (ed.) Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, 5–7 January 2014, pp. 82–101. SIAM (2014). <https://doi.org/10.1137/1.9781611973402.7>
10. Jacob, H., Pilipczuk, M.: Bounding twin-width for bounded-treewidth graphs, planar graphs, and bipartite graphs. In: Bekos, M.A., Kaufmann, M. (eds.) Graph-Theoretic Concepts in Computer Science. WG 2022. LNCS, vol. 13453, pp. 287–299. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-15914-5_21

11. Král, D., Lamaison, A.: Planar graph with twin-width seven. arXiv preprint [arXiv:2209.11537](https://arxiv.org/abs/2209.11537) (2022)
12. Král', D., Lamaison, A.: Planar graph with twin-width seven. Eur. J. Combinator. 103749 (2023). <https://doi.org/10.1016/j.ejc.2023.103749>
13. Marques-Silva, J., Lynce, I.: Towards robust CNF encodings of cardinality constraints. In: Bessiere, C. (ed.) Principles and Practice of Constraint Programming. CP 2007. LNCS, vol. 4741, pp. 483–497. Springer, Cham (2007). https://doi.org/10.1007/978-3-540-74970-7_35
14. Schidler, A., Szeider, S.: A SAT approach to twin-width. In: Phillips, C.A., Speckmann, B. (eds.) Proceedings of the Symposium on Algorithm Engineering and Experiments, ALENEX 2022, Alexandria, 9–10 January 2022, pp. 67–77. SIAM (2022). <https://doi.org/10.1137/1.9781611977042.6>

Author Index

A

Ariouat, Hanane 223
Azad, Ariful 144

B

Basak, Debolena 260
Bedychaj, Andrzej 118
Birke, Robert 235
Borovica-Gajic, Renata 68

C

Castillo, Joel 418
Chen, Chen 352
Chen, Lin 93
Chen, Lydia Y. 235
Chen, Ming-Syan 3
Chen, Pin-Yu 235
Chen, Siji 431
Chen, Ting-An 3
Chen, Wenguang 15
Chen, Yu-Hsiu 55
Chen, Yuzhou 418
Chen, Zhi 171
Chen, Zhiqian 431
Chenin, Eric 223
Choi, Jae Won 418
Cohen, Sarel 444

D

Desarkar, Maunendra Sankar 260
Dong, Yushun 352

F

Frías, José 418
Friedrich, Tobias 444
Funiak, Stano 171

G

Gao, Chongyang 301
Gao, Min 352

Gao, Wei 210
Gel, Yulia 418
Ghiță, Alexandru 43
Guo, Longkun 327
Guo, Ye 286

H

Han, Yoonsang 158
Hasegawa, Tai 376
He, Fang 339
Hon, Wing-Kai 248
Hong, Chi 235
Horev, Yinon 444
Huang, Chenghan 301
Huang, Jay 248
Hung, Hui-Ju 339
Huynh, Du Q. 405

I

Ionescu, Radu Tudor 43
Issac, Davis 444

J

Jabbour, Florian 223
Jung, Kyomin 184

K

Kamma, Lior 444
Karunasekera, Shanika 68
Kashima, Hisashi 28
Kharlamov, Evgeny 390
Kim, Jeongho 197

L

Lakmal, Dimuthu 68
Le, Binh Minh 131
Lee, Che-Rung 248
Lee, Vincent CS 273
Lee, Wang-Chien 339
Lei, Zhen 339

Li, Chi-Chang 248

Li, Jianqiang 81

Li, Juan 81

Li, Jundong 352

Li, Yang 286

Liao, Kewen 327

Liao, Zhibin 327

Lin, Xiaofeng 28

Liu, Hanbing 286

Liu, Jiajun 171

Liu, Rui 273

Liu, Wei 405

Liu, Xin 376

Lu, Chang-Tien 431

Lu, Xiaotian 28

Luo, Ziye 105

M

Ma, Jing 352

Ma, Weicheng 301

Ma, Yue 390

Ma, Zerui 81

Majeske, Nicholas 144

Moon, Gordon Euhyun 158

Murata, Tsuyoshi 376

N

Niklanovits, Aikaterini 444

P

Park, Kyeongman 184

Perera, Kushani 68

Phua, Yin Jun 376

Pignal, Marc 223

Prifti, Edi 223

Q

Qu, Zhijie 81

R

Reynolds, Mark 405

S

Samizadeh, Mina 314

Shang, Rui 431

Shay, Shiraz 444

Shen, Chih-Ya 339

Shuai, Hong-Han 55

Simonov, Kirill 444

Sklab, Youcef 223

Śmieja, Marek 118

Srijith, P. K. 260

Sun, Qiang 405

T

Tabor, Jacek 118

Tam, Zhi Rui 55

Tang, Buzhou 105

Tong, Guangmo 314

V

Vignes Lebbe, Régine 223

Vosoughi, Soroush 301

W

Wang, Hongwei 364

Wang, Jingge 286

Wang, Lili 301

Wang, Sen 171

Wen, Hong 364

Wen, Mi 364

Woo, Simon S. 131, 197

Wu, Yi 364

X

Xiong, Bo 390

Xiong, Ying 105

Xu, Xuwei 171

Xue, Yunsheng 364

Y

Yang, Hui 390

Yang, Nakyeong 184

Yao, Ruiye 301

Ye, Yan-Ting 3

Yoon, Bokyeong 158

Yuan, Man-Jie 210

Yun, Sukwon 376

Z

Zhang, Guoxi 28

Zhang, Xianren 352

Zhang, Xiaobo 93

Zhang, Xuan 286

Zhang, Yi 171

Zhang, Yunchao 327

Zhang, Zhehao 93

- Zhao, Peng 314
Zhao, Ying 15
Zheng, Hailong 93
Zhou, Dongzhuoran 390
Zhou, Yuxin 93
- Zhou, Ziheng 15
Zhu, Fengda 273
Zou, Zheng 210
Zucker, Jean-Daniel 223
Zuo, Haojia 15