

Wenjie Zhang  
Lei Zou  
Zakaria Maamar  
Lu Chen (Eds.)

LNCS 13080

# Web Information Systems Engineering – WISE 2021

22nd International Conference  
on Web Information Systems Engineering, WISE 2021  
Melbourne, VIC, Australia, October 26–29, 2021  
Proceedings, Part I

1  
Part I

 Springer

## Founding Editors

Gerhard Goos

*Karlsruhe Institute of Technology, Karlsruhe, Germany*

Juris Hartmanis

*Cornell University, Ithaca, NY, USA*


## Editorial Board Members

Elisa Bertino

*Purdue University, West Lafayette, IN, USA*

Wen Gao


*Peking University, Beijing, China*

Bernhard Steffen 

*TU Dortmund University, Dortmund, Germany*

Gerhard Woeginger 

*RWTH Aachen, Aachen, Germany*

Moti Yung 

*Columbia University, New York, NY, USA*

More information about this subseries at <http://www.springer.com/series/7409>

Wenjie Zhang · Lei Zou ·  
Zakaria Maamar · Lu Chen (Eds.)

# Web Information Systems Engineering – WISE 2021


22nd International Conference  
on Web Information Systems Engineering, WISE 2021  
Melbourne, VIC, Australia, October 26–29, 2021  
Proceedings, Part I

*Editors*

Wenjie Zhang  
School of Computer Science  
and Engineering  
The University of New South Wales  
Sydney, NSW, Australia

Zakaria Maamar  
Zayed University  
Dubai, United Arab Emirates

Lei Zou  
Peking University  
Beijing, China

Lu Chen   
Swinburne University of Technology  
Melbourne, VIC, Australia

ISSN 0302-9743                      ISSN 1611-3349 (electronic)  
Lecture Notes in Computer Science  
ISBN 978-3-030-90887-4              ISBN 978-3-030-90888-1 (eBook)  
<https://doi.org/10.1007/978-3-030-90888-1>

LNCS Sublibrary: SL3 – Information Systems and Applications, incl. Internet/Web, and HCI

© Springer Nature Switzerland AG 2021

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

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

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

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

# Preface

Welcome to the proceedings of the 22nd International Conference on Web Information Systems Engineering (WISE 2021), held in Melbourne, Australia, during October 26–29, 2021. The series of WISE conferences aims to provide an international forum for researchers, professionals, and industrial practitioners to share their knowledge in the rapidly growing area of web technologies, methodologies, and applications. The first WISE event took place in Hong Kong, China (2000). Then the trip continued to Kyoto, Japan (2001); Singapore (2002); Rome, Italy (2003); Brisbane, Australia (2004); New York, USA (2005); Wuhan, China (2006); Nancy, France (2007); Auckland, New Zealand (2008); Poznan, Poland (2009); Hong Kong, China (2010); Sydney, Australia (2011); Paphos, Cyprus (2012); Nanjing, China (2013); Thessaloniki, Greece (2014); Miami, USA (2015); Shanghai, China (2016); Puschino, Russia (2017); Dubai, UAE (2018); Hong Kong, China (2019); Amsterdam and Leiden, The Netherlands (2020); and this year, WISE 2021 was held in Melbourne, Australia.

A total of 229 research papers were submitted to the conference for consideration, and each paper was reviewed by at least three reviewers. Finally, 55 submissions were selected as regular papers (an acceptance rate of 24% approximately), plus 29 as short papers. The research papers cover the areas of blockchain, social networks, graph neural networks, graph query, crowdsourcing, knowledge graph and entity linking, spatial temporal data analysis, service computing, cloud computing, text mining, recommender systems, database systems, workflow, deep learning, data mining, and applications. In addition to regular and short papers, the WISE 2021 program also featured tutorial and demo sessions.

We would like to sincerely thank our keynote speakers:

- Munindar P. Singh, North Carolina State University, USA
- Jie Lu, University of Technology Sydney, Australia
- James B. D. Joshi, University of Pittsburgh, USA
- Xiaokui Xiao, National University of Singapore, Singapore

In addition, special thanks are due to the members of the international Program Committee and the external reviewers for a rigorous and robust reviewing process. We are also grateful to Springer and the International WISE Society for supporting this conference. The WISE Organizing Committee is also grateful to the demo organizers for their great efforts in helping promote web information system research to broader domains.

We expect that the ideas that have emerged in WISE 2021 will result in the development of further innovations for the benefit of scientific, industrial, and social communities.

October 2021

Wenjie Zhang  
Lei Zou  
Zakaria Maamar  
Lu Chen

# Organization

## General Co-chairs

Xiaofang Zhou	Hong Kong University of Science and Technology, Hong Kong
Yannis Manolopoulos	Aristotle University of Thessaloniki, Greece

## Program Co-chairs

Wenjie Zhang	University of New South Wales, Australia
Lei Zou	Peking University, China
Zakaria Maamar	Zayed University, Dubai, United Arab Emirates

## Publication Chair

Lu Chen	Swinburne University of Technology, Australia
---------	---

## Publicity Co-chairs

Xiaohui Tao	University of Southern Queensland, Australia
Georgios Kambourakis	University of the Aegean, Greece
Manik Sharma	DAV University, India
Xin Wang	Tianjin University, China

## Diversity and Inclusion Chair

Wenny Rahayu	La Trobe University, Australia
--------------	--------------------------------

## PhD School Chair

Shazia Sadiq	University of Queensland, Australia
--------------	-------------------------------------

## Demo Co-chairs

Weiguo Zheng	Fudan University, China
Dong Wen	University of Technology Sydney, Australia

## Tutorial and Workshop Chair

Guandong Xu	University of Technology Sydney, Australia
-------------	--

## **Industry Relationship Chair**

Jian Yang Macquarie University, Australia

## **Finance Chair**

Sudha Subramani Victoria University, Australia

## **Website Chair**

Yong-Feng Ge La Trobe University, Australia

## **Senior Program Committee**

Yanchun Zhang Victoria University, Australia  
Qing Li Hong Kong Polytechnic University, Hong Kong  
Xiaohua Jia City University of Hong Kong, Hong Kong  
Elisa Bertino Purdue University, USA  
Athman Bouguettaya University of Sydney, Australia

## **WISE Steering Committee Representatives**

Yanchun Zhang Victoria University, Australia  
Qing Li Hong Kong Polytechnic University, Hong Kong

## **Program Committee**

Marco Aiello University of Stuttgart, Germany  
Mohammed Eunus Ali Bangladesh University of Engineering and Technology,  
Germany  
Toshiyuki Amagasa University of Tsukuba, Japan  
Bernd Amann Sorbonne Université - LIP6, France  
Chutiporn Anutariya Asian Institute of Technology, Thailand  
Boualem Benatallah University of New South Wales, Australia  
Djamal Benslimane Université Claude Bernard Lyon 1, France  
Devis Bianchini University of Brescia, Italy  
Mohamed Reda Bouadjenek Deakin University, Australia  
Athman Bouguettaya University of Sydney, Australia  
Bin Cao Zhejiang University of Technology, China  
Jinli Cao La Trobe University, Australia  
Xin Cao University of New South Wales, Australia  
Barbara Catania University of Genoa, Italy  
Richard Chbeir Université de Pau et des Pays de l'Adour - LIUPPA,  
France  
Cindy Chen University of Massachusetts Lowell, USA  
Lu Chen Zhejiang University, China



Lu Chen	Swinburne University of Technology, Australia
Xiaojun Chen	College
Xiaoshuang Chen	University of New South Wales, Australia
Dickson K. W. Chiu	University of Hong Kong, Hong Kong
Theodoros Chondrogiannis	University of Konstanz, Germany
Dario Colazzo	Université Paris-Dauphine - LAMSADE, France
Alexandra Cristea	Durham University, UK
Hai Dong	RMIT University, Australia
Schahram Dustdar	Vienna University of Technology, Austria
Abdelaziz Elfazziki	University of Marrakech, Morocco
Nora Faci	Université Claude Bernard Lyon 1, France
Zhang Fan	Peking University, China
Yixiang Fang	Chinese University of Hong Kong, Shenzhen, China
Xiaoming Fu	University of Goettingen, Germany
Yunjun Gao	Zhejiang University, China
Dimitrios Georgakopoulos	Swinburne University of Technology, Australia
Azadeh Ghari Neiat	Deakin University, Australia
Xiangyang Gou	Peking University, China
Daniela Grigori	Université Paris-Dauphine - LAMSADE, France
Tobias Grubenmann	University of Bonn, Germany
Viswanath Gunturi	IIT Ropar, India
Armin Haller	Australian National University, Australia
Kongzhang Hao	University of New South Wales, Australia
Yu Hao	University of New South Wales, Australia
Tanzima Hashem	Bangladesh University of Engineering and Technology, Bangladesh
Md Rafiul Hassan	King Fahd University of Petroleum and Minerals, Saudi Arabia
Yizhang He	University of New South Wales, Australia
Lin Hu	Peking University, China
Chenji Huang	University of New South Wales, Australia
Hao Huang	Wuhan University, China
Xin Huang	Hong Kong Baptist University, Hong Kong
Yilun Huang	University of Technology Sydney, Australia
Zhisheng Huang	Vrije Universiteit Amsterdam, The Netherlands
Zi Huang	University of Queensland, Australia
Dawei Jiang	Zhejiang University, China
Jiawei Jiang	ETH Zurich, Switzerland
Jyun-Yu Jiang	University of California, Los Angeles, USA
Lili Jiang	Umeå University, Sweden
Peiquan Jin	University of Science and Technology of China, China
Eleanna Kafeza	Athens University of Economics and Business, Greece
Georgios Kambourakis	University of the Aegean, Greece
Verena Kantere	University of Ottawa, Canada
Georgia Kapitsaki	University of Cyprus, Cyprus
Panagiotis Karras	Aarhus University, Denmark

Kyoung-Sook Kim	National Institute of Advanced Industrial Science and Technology, Japan
Hong Va Leong	Hong Kong Polytechnic University, Hong Kong
Binghao Li	University of New South Wales, Australia
Hui Li	Xiamen University, China
Jianxin Li	Deakin University, Australia
Youhuan Li	Hunan University, China
Xiang Lian	Kent State University, USA
Kewen Liao	Australian Catholic University, Australia
Dan Lin	University of Missouri, USA
Qingyuan Linghu	University of New South Wales, Australia
Sebastian Link	University of Auckland, New Zealand
An Liu	Soochow University, China
Boge Liu	University of New South Wales, Australia
Guanfeng Liu	Macquarie University, Australia
Cheng Long	Nanyang Technological University, China
Hua Lu	Roskilde University, Denmark
Siqiang Luo	Nanyang Technological University, China
Jianming Lv	South China University of Technology, China
Fenglong Ma	Pennsylvania State University, USA
Jiangang Ma	Federation University Australia, Australia
Zakaria Maamar	Zayed University, UAE
Murali Mani	University of Michigan–Flint, USA
Yannis Manolopoulos	Open University of Cyprus, Cyprus
Yuren Mao	University of New South Wales, Australia
Xiaoye Miao	Zhejiang University, China
Sajib Mistry	Curtin University, Australia
Natwar Modani	Adobe Research, India
Amira Mouakher	Corvinus University of Budapest, Hungary
Tsz Nam-Chan	Hong Kong Baptist University, Hong Kong
Mitsunori Ogihara	University of Miami, USA
Mourad Oussalah	University of Oulu, Finland
M. Tamer Ozsu	University of Waterloo, Canada
George Pallis	University of Cyprus, Cyprus
Yue Pang	Peking University, China
George Papastefanatos	Athena Research Center, Greece
Peng Peng	Hunan University, China
Zhiyong Peng	Wuhan University, China
Francesco Piccialli	University of Naples Federico II, Italy
Dimitris Plexousakis	Institute of Computer Science - FORTH, Greece
Nicoleta Preda	Université de Versailles, France
Tieyun Qian	Wuhan University, China
Lu Qin	University of Technology Sydney, Australia
Yu-Xuan Qiu	University of Technology Sydney, Australia
Jarogniew Rykowski	Poznan University of Economics and Business, Poland
Dimitris Sacharidis	Université Libre de Bruxelles, Belgium

Shazia Sadiq	University of Queensland, Australia
Heiko Schuldt	University of Basel, Switzerland
Mohamed Sellami	Telecom SudParis, France
Caihua Shan	University of Hong Kong, Hong Kong
Yingxia Shao	Beijing University of Posts and Telecommunications, China
Wei Shen	Nankai University, China
Yain-Whar Si	University of Macau, Macau
Shaoxu Song	Tsinghua University, China
Kostas Stefanidis	Tampere University, Finland
Xunbin Su	Peking University, China
Qingqiang Sun	University of New South Wales, Australia
Stefan Tai	TU Berlin, Germany
Bo Tang	Southern University of Science and Technology, China
Chaogang Tang	China University of Mining and Technology, China
Xiaohui Tao	University of Southern Queensland, Australia
Dimitri Theodoratos	New Jersey Institute of Technology, USA
Leong Hou U	University of Macau, Macau
Athena Vakali	Aristotle University of Thessaloniki, Greece
Dirk Van Gucht	Indiana University Bloomington, USA
De Wang	Georgia Institute of Technology, USA
Hanchen Wang	University of Technology Sydney, Australia
Hongzhi Wang	Harbin Institute of Technology, China
Hua Wang	Victoria University, Australia
Junhu Wang	Griffith University, Australia
Kai Wang	University of New South Wales, Australia
Lizhen Wang	Yunnan University, China
Xin Wang	Tianjin University, China
Shiting Wen	Zhejiang University, China
Dingming Wu	Shenzhen University, China
Adam Wójtowicz	Poznań University of Economics and Business, Poland
Xiaokui Xiao	National University of Singapore, Singapore
Jianliang Xu	Hong Kong Baptist University, Hong Kong
Yuanyuan Xu	Nankai University, China
Jeffrey Xu-Yu	Chinese University of Hong Kong, Hong Kong
Bingcong Xue	Peking University, China
Hayato Yamana	Waseda University, Japan
Lei Yang	Peking University, China
Peilun Yang	University of Technology Sydney, Australia
Zhengyi Yang	University of New South Wales, Australia
Lina Yao	University of New South Wales, Australia
Xun Yi	RMIT University, Australia
Hongzhi Yin	University of Queensland, Australia
Man Lung Yiu	Hong Kong Polytechnic University, Hong Kong
Jianming Yong	University of Southern Queensland, Australia
Sira Yongchareon	Auckland University of Technology, New Zealand

Detian Zhang	Jiangnan University, China
Ji Zhang	University of Southern Queensland, Australia
Jiujing Zhang	Guangzhou University, China
Wenjie Zhang	University of New South Wales, Australia
Yanchun Zhang	Victoria University, Australia
Ying Zhang	University of Technology Sydney, Australia
Lei Zhao	Soochow University, China
Kai Zheng	University of Electronic Science and Technology of China
Xiangmin Zhou	RMIT University, Australia
Yuqi Zhou	Peking University, China
Yi Zhuang	Zhejiang Gongshang University, China
Lei Zou	Peking University, China

## Additional Reviewers

Abeyssekara, Prabath	Lan, Michael
Abusafia, Amani	Li, Chunbo
Akram, Junaid	Li, Huan
Alharbi, Ahmed	Li, Meng
Allani, Sabri	Li, Xinghao
Alnazer, Ebaa	Liao, Ningyi
Aryal, Sunil	Liao, Zhibin
Aytimur, Mehmet	Liesaputra, Veronica
Bahutair, Mohammed	Liu, Boge
Bornholdt, Johann	Lumbantoruan, Rosni
Chaki, Dipankar	Maaradji, Abderrahmane
Chan, Harry Kai-Ho	Maheshwari, Ayush
Chen, Dong	Mahmood, Md Tareq
Chen, Xiaocong	Maroulis, Stavros
Chen, Xiaoshuang	Mountantonakis, Michalis
Du, Jing	Nicewarner, Tyler
Efthymiou, Vasilis	Papadakos, Panagiotis
Fang, Uno	Papoutsoglou, Maria
Georgievski, Ilche	Paschalides, Demetris
Haghshenas, Kawsar	Qiu, Yu-Xuan
Han, Keqi	Qu, Liang
Hotz, Manuel	Rashid, Syed Mukit
Huang, Guanjie	Salman, Muhammad
Islam, Fariha Tabassum	Sarwar, Kinza
Islam, Khandker Aftarul	Sassi, Salma
Joan, Yin	Setz, Brian
Kassawat, Firas	Sha, Alyssa
Kelarev, Andrei	Shahzaad, Babar
Kunchala, Jyothi	Shao, Yachao

Sikdar, Sagor  
Song, Xiangyu  
Stamatopoulos, Vasileios  
Taoufik, Yeferny  
Tripto, Nafis Itiza  
Wang, Haixin  
Wang, Hanchen  
Wang, Jiaqi  
Wu, Xiaoying  
Wu, Yingpei  
Yan, Qian

Yang, Peilun  
Ye, Zesheng  
Yin, Hui  
Yu, Yuanhang  
Zeginis, Chrysostomos  
Zhang, Boyu  
Zhang, Han  
Zhang, Junhua  
Zhang, Yunxiao  
Ziaur, Rahman

# Contents – Part I

## BlockChain and Crowdsourcing

Crowdsourcing Software Vulnerability Discovery: Models, Dimensions, and Directions . . . . .	3
<i>Mortada Al-Banna, Boualem Benatallah, Moshe C. Barukh, Elisa Bertino, and Salil Kanhere</i>	
Expertise-Aware Crowdsourcing Taxonomy Enrichment . . . . .	14
<i>Yuquan Wang, Yanpeng Wang, Yiming Mao, Jifan Yu, Kaisheng Zeng, Lei Hou, Juanzi Li, and Jie Tang</i>	
Transaction Confirmation Time Estimation in the Bitcoin Blockchain . . . . .	30
<i>Limeng Zhang, Rui Zhou, Qing Liu, Jiajie Xu, and Chengfei Liu</i>	
Automatic Malicious Worker Detection in Crowdsourced Paraphrases . . . . .	46
<i>Mohammad-Ali Yaghoub-Zadeh-Fard and Boualem Benatallah</i>	
A Blockchain-Based Approach for Trust Management in Collaborative Business Processes . . . . .	59
<i>Ada Bagozi, Devis Bianchini, Valeria De Antonellis, Massimiliano Garda, and Michele Melchiori</i>	

## Database System and Workflow

Exploiting Unblocking Checkpoint for Fault-Tolerance in Pregel-Like Systems . . . . .	71
<i>Yi Yang, Zhenhua Yang, and Chen Xu</i>	
A Low-Latency Metadata Service for Geo-Distributed File Systems . . . . .	87
<i>Chuangwei Lin, Bowen Liu, Wei Zhou, Yueyue Xu, Xuyun Zhang, and Wanchun Dou</i>	
XTuning: Expert Database Tuning System Based on Reinforcement Learning . . . . .	101
<i>Yanfeng Chai, Jiake Ge, Yunpeng Chai, Xin Wang, and BoXuan Zhao</i>	
CELA: An Accurate Learned Cardinality Estimator with Strong Generalization Ability and Dimensional Adaptability . . . . .	111
<i>Weiqing Zhou, Siyu Zhan, Lei Guo, and Bo Dai</i>	

Cost-Based Lightweight Storage Automatic Decision for In-Database Machine Learning . . . . . 119  
*Shuangshuang Cui, Hongzhi Wang, Haiyao Gu, and Yuntian Xie*

**Data Mining and Applications**

NP-PROV: Neural Processes with Position-Relevant-Only Variances . . . . . 129  
*Xuesong Wang, Lina Yao, Xianzhi Wang, Feiping Nie, and Boualem Benatallah*

A Minority Class Boosted Framework for Adaptive Access Control Decision-Making . . . . . 143  
*Mingshan You, Jiao Yin, Hua Wang, Jinli Cao, and Yuan Miao*

Recognizing Hand Gesture in Still Infrared Images by CapsNet . . . . . 158  
*Hongwang Xiao, Yun Yang, Ke Yu, Jiao Tian, Xinyi Cai, Ying Zhao, Kai Zhang, Na Guo, and Jinjun Chen*

Vertical Federated Principal Component Analysis on Feature-Wise Distributed Data . . . . . 173  
*Yiu-ming Cheung, Jian Lou, and Feng Yu*

Anchoring-and-Adjustment to Improve the Quality of Significant Features . . . 189  
*Eunkyung Park, Raymond K. Wong, Junbum Kwon, and Victor W. Chu*

Data Mining Based Artificial Intelligent Technique for Identifying Abnormalities from Brain Signal Data . . . . . 198  
*Md. Nurul Ahad Tawhid, Siuly Siuly, Kate Wang, and Hua Wang*

Where Should I Go? A Deep Learning Approach to Personalize Type-Based Facet Ranking for POI Suggestion . . . . . 207  
*Esraa Ali, Annalina Caputo, Séamus Lawless, and Owen Conlan*

Modeling Without Sharing Privacy: Federated Neural Machine Translation . . . . . 216  
*Jianzong Wang, Zhangcheng Huang, Lingwei Kong, Denghao Li, and Jing Xiao*

**Knowledge Graph and Entity Linking**

Encoding the Meaning Triangle (Object, Entity, and Concept) as the Semantic Foundation for Entity Alignment . . . . . 227  
*Kaisheng Zeng, Chengjiang Li, Yan Qi, Xin Lv, Lei Hou, Guozheng Peng, Juanzi Li, and Ling Feng*

Incorporating Network Structure with Node Information for Semi-supervised Anomaly Detection on Attributed Graphs . . . . .	242
<i>Bofeng Chen, Jingdong Li, Xingjian Lu, Chaofeng Sha, Xiaoling Wang, and Ji Zhang</i>	
OntoSP: Ontology-Based Semantic-Aware Partitioning on RDF Graphs . . . . .	258
<i>Sizhuo Li, Weixue Chen, Baozhu Liu, Pengkai Liu, Xin Wang, and Yuan-Fang Li</i>	
Optimal Subgraph Matching Queries over Distributed Knowledge Graphs Based on Partial Evaluation . . . . .	274
<i>Jiao Xing, Baozhu Liu, Jianxin Li, Farhana Murtaza Choudhury, and Xin Wang</i>	
Enhancing both Local and Global Entity Linking Models with Attention . . . . .	290
<i>Jinliang Li, Haoyu Liu, Yulong Zhang, Li Zhang, Qiang Yang, Jianfeng Qu, and Zhixu Li</i>	
HyperJOIE: Two-View Hyperbolic Knowledge Graph Embedding with Entities and Concepts Jointly . . . . .	305
<i>Jing Dong, Binbin Gu, Jianfeng Qu, An Liu, Lei Zhao, Zhigang Chen, and Zhixu Li</i>	
IOPE: Interactive Ontology Population and Enrichment Guided by Ontological Constraints . . . . .	321
<i>Shadi Baghernezhad-Tabasi, Loïc Druette, Fabrice Jouanot, Celine Meurger, and Marie-Christine Rousset</i>	
<b>Graph Neural Network</b>	
Controversy Detection: A Text and Graph Neural Network Based Approach . . . . .	339
<i>Samy Benslimane, Jérôme Azé, Sandra Bringay, Maximilien Servajean, and Caroline Mollevi</i>	
GMGCN: Gated Memory Graph Convolutional Network for Passenger Demand Prediction . . . . .	355
<i>Tianyuan Bi, Kai Han, and Cheng Shen</i>	
Event Detection in Social Media via Graph Neural Network. . . . .	370
<i>Wang Gao, Yuan Fang, Lin Li, and Xiaohui Tao</i>	
Knowledge-Guided Fraud Detection Using Semi-supervised Graph Neural Network . . . . .	385
<i>Yizhuo Rao, Xiaoguang Ren, Chengyuan Duan, Xianya Mi, Jiajun Cheng, Yu Chen, Hongliang You, Qiang Gao, Zhixian Zeng, and Xiao Wei</i>	



MSSF-GCN: Multi-scale Structural and Semantic Information Fusion Graph Convolutional Network for Controversy Detection. . . . .	394
<i>Haiyang Wang, Xin Song, Bin Zhou, Ye Wang, Liqun Gao, and Yan Jia</i>	
A Syntax-Aware Encoder for Authorship Attribution. . . . .	403
<i>Jianbo Liu, Zhiqiang Hu, Jiasheng Zhang, Roy Ka-Wei Lee, and Jie Shao</i>	
Graph Attentive Leaping Connection Network for Chinese Short Text Semantic Classification . . . . .	412
<i>Jingdan Zhu</i>	
<b>Graph Query</b>	
Graph Ordering: Towards the Optimal by Learning. . . . .	423
<i>Kangfei Zhao, Yu Rong, Jeffrey Xu Yu, Wenbing Huang, Junzhou Huang, and Hao Zhang</i>	
Fast Approximate All Pairwise CoSimRanks via Random Projection . . . . .	438
<i>Renchi Yang and Xiaokui Xiao</i>	
Critical Nodes Identification in Large Networks: An Inclination-Based Model. . . . .	453
<i>Chen Chen, Xijuan Liu, Shuangyan Xu, Mengqi Zhang, Xiaoyang Wang, and Xuemin Lin</i>	
LPMA - An Efficient Data Structure for Dynamic Graph on GPUs. . . . .	469
<i>Fan Zhang, Lei Zou, and Yanpeng Yu</i>	
Updating Maximal $(\Delta, \gamma)$ -Cliques of a Temporal Network Efficiently. . . . .	485
<i>Suman Banerjee and Bithika Pal</i>	
<b>Social Network</b>	
Web of Students: Class-Level Friendship Network Discovery from Educational Big Data. . . . .	497
<i>Teng Guo, Tao Tang, Dongyu Zhang, Jianxin Li, and Feng Xia</i>	
Event Cube for Suicidal Event Analysis: A Case Study . . . . .	512
<i>Qing Li, Zhihan Yan, Jun Li, Zhenguo Yang, Zehang Lin, Hong Va Leong, Lei Chen, and Nancy Xiaonan Yu</i>	
Cross-modal Attention Network with Orthogonal Latent Memory for Rumor Detection . . . . .	527
<i>Zekai Wu, Jiaxin Chen, Zhenguo Yang, Haoran Xie, Fu Lee Wang, and Wenyin Liu</i>	

OMT: An Operate-Based Approach for Modelling Multi-topic Influence Diffusion in Online Social Networks . . . . .	542
<i>Chenting Jiang, Weihua Li, Shiqing Wu, and Quan Bai</i>	
Modeling User Profiles Through Multiple Types of User Interaction Behaviors . . . . .	557
<i>Yimin Lv, Xinzhou Dong, Beihong Jin, and Wei Zhuo</i>	
HACK: A Hierarchical Model for Fake News Detection . . . . .	565
<i>Yanqi Li, Ke Ji, Kun Ma, Zhenxiang Chen, Jun Wu, Yidong Li, and Guandong Xu</i>	
<b>Spatial and Temporal Data Analysis</b>	
An Efficient Approach for Spatial Trajectory Anonymization . . . . .	575
<i>Yuetian Wang, Wen Hua, Fengmei Jin, Jing Qiu, and Xiaofang Zhou</i>	
Developing a Deep Learning Based Approach for Anomalies Detection from EEG Data . . . . .	591
<i>Ashik Mostafa Alvi, Siuly Siuly, and Hua Wang</i>	
Dynamic Transit Flow Graph Prediction in Spatial-Temporal Network. . . . .	603
<i>Liyang Jiang, Yongxuan Lai, Quan Chen, Wenhua Zeng, Fan Yang, Fan Yi, and Qisheng Liao</i>	
Disatra: A Real-Time Distributed Abstract Trajectory Clustering . . . . .	619
<i>Liang Chen, Pingfu Chao, Junhua Fang, Wei Chen, Jiajie Xu, and Lei Zhao</i>	
Extra-Budget Aware Task Assignment in Spatial Crowdsourcing . . . . .	636
<i>Shuhan Wan, Detian Zhang, An Liu, and Junhua Fang</i>	
Expert Recommendations with Temporal Dynamics of User Interest in CQA . . . . .	645
<i>Xiaoqi Lv, Ke Ji, Zhenxiang Chen, Kun Ma, Jun Wu, Yidong Li, and Guandong Xu</i>	
<b>Author Index</b> . . . . .	653

## Contents – Part II

### Deep Learning (1)

Efficient Feature Interactions Learning with Gated Attention Transformer . . .	3
<i>Chao Long, Yanmin Zhu, Haobing Liu, and Jiadi Yu</i>	
Interactive Pose Attention Network for Human Pose Transfer . . . . .	18
<i>Di Luo, Guipeng Zhang, Zhenguo Yang, Minzheng Yuan, Tao Tao, Liangliang Xu, Qing Li, and Wenyin Liu</i>	
Exploiting Intra and Inter-field Feature Interaction with Self-Attentive Network for CTR Prediction. . . . .	34
<i>Shenghao Zheng, Xuefeng Xian, Yongjing Hao, Victor S. Sheng, Zhiming Cui, and Pengpeng Zhao</i>	
AMBD: Attention Based Multi-Block Deep Learning Model for Warehouse Dwell Time Prediction. . . . .	50
<i>Xingyi Lv, Wei Zhao, Jiali Mao, Ye Guo, and Aoying Zhou</i>	
Performance Evaluation of Pre-trained Models in Sarcasm Detection Task . . .	67
<i>Haiyang Wang, Xin Song, Bin Zhou, Ye Wang, Liqun Gao, and Yan Jia</i>	

### Deep Learning (2)

News Popularity Prediction with Local-Global Long-Short-Term Embedding. . . . .	79
<i>Shuai Fan, Chen Lin, Hui Li, and Quan Zou</i>	
An Efficient Method for Indoor Layout Estimation with FPN . . . . .	94
<i>Aopeng Wang, Shiting Wen, Yunjun Gao, Qing Li, Ke Deng, and Chaoyi Pang</i>	
Lightweight Network Traffic Classification Model Based on Knowledge Distillation . . . . .	107
<i>Yanhui Wu and Meng Zhang</i>	
RAU: An Interpretable Automatic Infection Diagnosis of COVID-19 Pneumonia with Residual Attention U-Net . . . . .	122
<i>Xiacong Chen, Lina Yao, and Yu Zhang</i>	

Comparison the Performance of Classification Methods for Diagnosis of Heart Disease and Chronic Conditions . . . . .	137
<i>Jiarui Si, Haohan Zou, Chuanyi Huang, Huan Feng, Honglin Liu, Guangyu Li, Shuaijun Hu, Hong Zhang, and Xing Wang</i>	

### Recommender Systems (1)

Capturing Multi-granularity Interests with Capsule Attentive Network for Sequential Recommendation . . . . .	147
<i>Zihan Song, Jiahao Yuan, Xiaoling Wang, and Wendi Ji</i>	

Multi-Task Learning with Personalized Transformer for Review Recommendation . . . . .	162
<i>Haiming Wang, Wei Liu, and Jian Yin</i>	

ADQ-GNN: Next POI Recommendation by Fusing GNN and Area Division with Quadtree . . . . .	177
<i>Yu Wang, An Liu, Junhua Fang, Jianfeng Qu, and Lei Zhao</i>	

MGSAN: A Multi-granularity Self-attention Network for Next POI Recommendation . . . . .	193
<i>Yepeng Li, Xuefeng Xian, Pengpeng Zhao, Yanchi Liu, and Victor S. Sheng</i>	

HRFA: Don't Ignore Strangers with Different Views . . . . .	209
<i>Senhui Zhang, Wendi Ji, Jiahao Yuan, and Xiaoling Wang</i>	

### Recommender Systems (2)

MULTIPLE: Multi-level User Preference Learning for List Recommendation . . . . .	221
<i>Beibei Li, Beihong Jin, Xinzhou Dong, and Wei Zhuo</i>	

Deep News Recommendation with Contextual User Profiling and Multifaceted Article Representation . . . . .	237
<i>Dai Hoang Tran, Salma Hamad, Munazza Zaib, Abdulwahab Aljubairy, Quan Z. Sheng, Wei Emma Zhang, Nguyen H. Tran, and Nguyen Lu Dang Khoa</i>	

Intent-Aware Visualization Recommendation for Tabular Data . . . . .	252
<i>Atsuki Maruta and Makoto P. Kato</i>	

Existence Conditions for Hidden Feedback Loops in Online Recommender Systems . . . . .	267
<i>Anton Khritankov and Anton Pilkevich</i>	

Retrieval-Based Factorization Machines for CTR Prediction . . . . . 275  
*Xu Wang, Yuancai Huang, Xiaokai Zhao, Weinan Zhao, Yu Tang,  
and Yitao Duan*

**Text Mining (1)**

Adversarial Training for a Hybrid Approach to Aspect-Based  
Sentiment Analysis . . . . . 291  
*Ron Hochstenbach, Flavius FrasinCAR, and Maria Mihaela Trușcă*

A Dual Reinforcement Network for Classical and Modern Chinese Text  
Style Transfer. . . . . 306  
*Minzhang Xu, Min Peng, and Fang Liu*

Representation Learning for Short Text Clustering . . . . . 321  
*Hui Yin, Xiangyu Song, Shuiqiao Yang, Guangyan Huang,  
and Jianxin Li*

*ReAct: A Review Comment Dataset for Actionability (and more).* . . . . . 336  
*Gautam Choudhary, Natwar Modani, and Nitish Maurya*

**Text Mining (2)**

Document-Level Relation Extraction with Entity Enhancement  
and Context Refinement. . . . . 347  
*Meng Zou, Qiang Yang, Jianfeng Qu, Zhixu Li, An Liu, Lei Zhao,  
and Zhigang Chen*

TDM-CFC: Towards Document-Level Multi-label Citation  
Function Classification. . . . . 363  
*Yang Zhang, Yufei Wang, Quan Z. Sheng, Adnan Mahmood,  
Wei Emma Zhang, and Rongying Zhao*

NOCOL - Nonnegative Orthogonal Constraint Outlier Learning . . . . . 377  
*Thirunavukarasu Balasubramaniam, Wathsala Anupama Mohotti,  
Richi Nayak, and Chau Yuen*

Semantic Parsing with Syntax Graph of Logical Forms . . . . . 386  
*Chen Chang*

Case Study of Few-Shot Learning in Text Recognition Models. . . . . 394  
*Jianzong Wang, Shijing Si, Zhenhou Hong, Xiaoyang Qu, Xinghua Zhu,  
and Jing Xiao*

## Service Computing and Cloud Computing (1)

Detecting Document Versions and Their Ordering in a Collection . . . . .	405
<i>Natwar Modani, Anurag Maurya, Gaurav Verma, Inderjeet Nair, Vaidehi Patil, and Anirudh Kanfade</i>	
Focus on Misinformation: Improving Medical Experts' Efficiency of Misinformation Detection . . . . .	420
<i>Aleksandra Nabożny, Bartłomiej Balcerzak, Mikołaj Morzy, and Adam Wierzbicki</i>	
Sensory Monitoring of Physiological Functions Using IoT Based on a Model in Petri Nets . . . . .	435
<i>Kristián Fodor and Zoltán Balogh</i>	

## Service Computing and Cloud Computing (2)

A Multi-perspective Model of Smart Products for Designing Web-Based Services on the Production Chain . . . . .	447
<i>Ada Bagozi, Devis Bianchini, and Anisa Rula</i>	
A Multi-view Learning Approach for the Autonomic Management of Big Services . . . . .	463
<i>Fedia Ghedass and Faouzi Ben Charrada</i>	
Towards a Deep Learning-Driven Service Discovery Framework for the Social Internet of Things: A Context-Aware Approach . . . . .	480
<i>Abdulwahab Aljubairy, Ahoud Alhazmi, Wei Emma Zhang, Quan Z. Sheng, and Dai Hoang Tran</i>	

## Tutorial and Demo

Graph Data Mining in Recommender Systems . . . . .	491
<i>Hongxu Chen, Yicong Li, and Haoran Yang</i>	
Emerging Applications in Healthcare and Their Implications to Academia and Practice . . . . .	497
<i>Raj Gururajan, Xiaohui Tao, Yuefeng Li, Xujuan Zhou, Soman Elangovan, Srinivas Kondalsamy Chennakesavan, and Revathi Venkataraman</i>	
GPUGraphX: A GPU-Aided Distributed Graph Processing System . . . . .	501
<i>Qi Li, Kai Zou, Deyu Kong, Huhao Guan, and Xike Xie</i>	

<b>SQL2Cypher: Automated Data and Query Migration from RDBMS to GDBMS</b> . . . . .	<b>510</b>
<i>Shunyang Li, Zhengyi Yang, Xianhang Zhang, Wenjie Zhang, and Xuemin Lin</i>	
<b>MOBA Game Analysis System Based on Neural Networks</b> . . . . .	<b>518</b>
<i>Kangwei Li, Mengwei Li, Jia Tian, Xiaobo Cao, Tiezheng Nie, Yue Kou, and Derong Shen</i>	
<b>FedAggs: Optimizing Aggregate Queries Evaluation in Federated RDF Systems</b> . . . . .	<b>527</b>
<i>Ningchao Ge, Peng Peng, Zheng Qin, and Mingdao Li</i>	
<b>JUST-Studio: A Platform for Spatio-Temporal Data Map Designing and Application Building</b> . . . . .	<b>536</b>
<i>Yuan Sui, Ruiyuan Li, Xu Wang, Jun Liu, and Juncheng Tang</i>	
<b>Author Index</b> . . . . .	<b>547</b>

# **BlockChain and Crowdsourcing**





# Crowdsourcing Software Vulnerability Discovery: Models, Dimensions, and Directions

Mortada Al-Banna<sup>1</sup>(✉), Boualem Benatallah<sup>1</sup>, Moshe C. Barukh<sup>1</sup>, Elisa Bertino<sup>2</sup>,  
and Salil Kanhere<sup>1</sup>

<sup>1</sup> UNSW, Sydney, Australia

m.al-banna@unsw.edu.au

<sup>2</sup> Purdue University, West Lafayette, USA

**Abstract.** Software systems prove indispensable amongst a variety of fields. With our increasing reliance on them coupled with their heightened complexity, the demand for protection increases as well. In this article, we explore how crowdsourcing could be used for vulnerability discovery. We examine the models of crowdsourcing that has been applied in vulnerability discovery, identify dimensions of this crowdsourced task, and discuss applicable concerns and future research directions.

**Keywords:** Crowdsourcing · Security · Vulnerability discovery · Bug-bounty

## 1 Introduction

A vulnerability is a security flaw that arises from system design, implementation, or maintenance (e.g., SQL injection vulnerability, and memory corruption vulnerability). By exploiting these vulnerabilities, malicious parties could gain unauthorized access to protected resources. Software systems are becoming increasingly complex with modern-day development distributed across multiple heterogeneous, autonomous, and evolving cloud services. Furthermore, the reliance on third-party software (e.g., cloud, open-APIs, and external libraries) make it difficult for in-house IT experts to deal with inherent risks of using external software. To overcome vulnerability issues, organizations rely on several methods (e.g., automated tools, penetration testing firms, and crowdsourced vulnerability discovery). In this paper, we specifically examine how vulnerability management can be improved with crowdsourcing.

Crowdsourcing harnesses the wisdom of large communities working independently to solve problems, much as open source does for software development. Several crowdsourcing platforms have emerged like Crowdfunder, Innocentive, TopCoder, Kaggle, and uTest. Some platforms use crowdsourcing for tasks that require skilled workers (e.g., Web design, testing, Web development tasks, and R&D challenges).

In a crowdsourced vulnerability discovery program (also known as a bug bounty), software providers (who could be the task requester here) submit vulnerability discovery tasks to a community of security professionals (henceforth referred to as SecPros).

SecPros play the role of the attackers and compete to discover and report vulnerabilities. Software providers then evaluate the outcomes and may reward SecPros that submit valid vulnerabilities. Rewards can be monetary, gifts, reputation badges, etc. Crowdsourcing vulnerability discovery could be considered economical compared to hiring full-time SecPros or relying on penetration testing firms, giving its higher volume of output in a shorter time [1]. It has been observed that malicious parties are more agile in exploiting vulnerabilities than software providers are in repairing them<sup>1</sup>. It thus makes sense to harness the power of a crowd of SecPros to discover vulnerabilities as quickly as possible following what Linus' Law stated, "given enough eyeballs, all bugs are shallow" [2]. Additionally, diversity in expertise of SecPros is also highly desired. Empirical studies have confirmed that SecPros with different backgrounds, skills and interests tend to discover different vulnerabilities [3].

Previous research has investigated crowdsourcing models and dimensions in general areas like software engineering [4]. Crowdsourcing vulnerability discovery differ from general crowdsourcing in the following (i) tasks are open-ended, as the requester does not know how many vulnerabilities will be discovered (ii) although it is competition based crowdsourcing but there may be multiple winners; (iii) the crowd participants need to possess very specific and specialized skills compared to participants in general crowd tasks; (iv) tasks (and data) are by nature very sensitive and require special agreements regarding privacy, intellectual property and data protection; (v) some commonly used quality assurance techniques, such as redundancy or voting, do not make sense, as information about vulnerabilities are sensitive and remains so until the vulnerability is remediated; and (vi) rewards are different since there might be different tiers for rewards tailored according to the severity of the vulnerability (e.g., high severity vulnerabilities pay higher reward). Previous research has found that the lack of knowledge to manage the task is one of the pre-adoption concerns that organizations have in regard to crowdsourcing vulnerability discovery [5]. This article provides an understanding of crowdsourced vulnerability discovery. We identify crowdsourcing models and key dimensions to characterize vulnerability discovery tasks. We also discuss future research directions, that we believe will help in the advancement of the field.

## 2 Models for Crowdsourcing Vulnerability Discovery

The concept of crowdsourcing vulnerability discovery is relatively new. NetScape in 1995 launched the first official program<sup>2</sup>. Since then, several big companies (e.g., Google, Facebook, and Microsoft) have adopted the same concept and launched their own programs. Moreover, several targeted crowdsourcing platforms for vulnerability discovery have been established such as BugCrowd (bugcrowd.com), Cobalt (cobalt.io), and Synack (synack.com). Additionally, cyber security competitions are regularly held (Pwn2Own, Pwnium, and Pwn0rama). In the following, we examine the models that involve responsible disclosure of vulnerabilities. We do not consider unregulated markets for vulnerabilities (e.g., Black, or Grey markets).

<sup>1</sup> <https://www.cyber.gov.au/acsc/view-all-content/reports-and-statistics/acsc-annual-cyber-threat-report-july-2019-june-2020>.

<sup>2</sup> [www.theregister.co.uk/2016/02/22/bug\\_bounty\\_feature](http://www.theregister.co.uk/2016/02/22/bug_bounty_feature).

## 2.1 Direct Vulnerability Discovery

In this model, the organization (software vendor or provider) makes an open call, describes the scope and rules of the task. SecPros can then perform vulnerability testing and submit a report describing the vulnerabilities they have discovered. The organization then verifies the legitimacy of the submitted vulnerability reports and determines the proper reward if any. In this model, there is no time frame for accepting submissions (i.e., open duration), unless the organization decides to terminate the program or migrate to another model. In this model, organization is fully responsible for managing the crowdsourced vulnerability discovery program.

An example for direct vulnerability discovery programs is Mozilla bug bounty program<sup>3</sup>. They provide monetary rewards ranging from 500 USD for medium vulnerabilities to more than 10,000 USD for exceptional vulnerabilities. Mozilla also provides non-monetary incentives such as expressing gratitude on the hall of fame. Mozilla SecPros submit vulnerability reports directly to Mozilla and the Bounty Committee assesses the legitimacy of the submission, determines the severity of the vulnerability and its eligibility for reward. From the date of its initiation (in 2004) Mozilla bug bounty program has paid out over 1.6 million dollars.

## 2.2 Platform Managed Vulnerability Discovery

This model is gaining increasing popularity. In this model, the platform providers (e.g., Bugcrowd, Cobalt, Synack, Hackerone and Bounty Factory) are responsible for managing the community of SecPros to perform the task: providing reputation policies to motivate SecPros (e.g., through gamification, badges), providing SecPros selection strategies and communicating with them, providing guidance to both the organization and SecPros, and managing rewards. Some platforms provide the choice of launching a program for a certain period of time (e.g., before launching a new software product), or keep the program running for an unspecified period.

An example of this model is Bugcrowd, which is a platform that was founded in 2012 and currently maintains a community of around 22,000 SecPros. Bugcrowd ranks SecPros according to their performance, based on indicators such as number of discovered vulnerabilities, quality of submitted reports and impact of discovered vulnerabilities. The platform provides support for handling vulnerability report submissions (e.g., filtering ineligible reports, such as duplicate and out-of-scope vulnerabilities reports). Bugcrowd also offers organizations the option to either run an on-going program (e.g., Heroku program), or a time-boxed program (e.g., Aruba Networks program), and to define the criteria of selecting the SecPros to be invited to participate.

## 2.3 Cyber Security Contests

In this model, software providers, platforms or security concerned organizations submit tasks for the discovery of vulnerabilities in a contest which is usually bound over a short period of time and a certain location. SecPros submit discovered vulnerabilities to

---

<sup>3</sup> [www.mozilla.org/en-US/security/bug-bounty](http://www.mozilla.org/en-US/security/bug-bounty).

the organizers of the contest who will be responsible to assess submissions for validity and to determine the winner of the contest (e.g., according to severity of discovered vulnerabilities).

Pwn2Own is one of the most famous cyber security contests. It usually runs annually in conjunction with the CanSecWest security conference. SecPros eligible to participate must be registered and available on-site at the time of the contest. SecPros need to exploit a series of undisclosed vulnerabilities to compromise a system (e.g., computer or mobile phone). SecPros, participating in the contest, get quite the media coverage and high rewards (e.g., total prizes for Pwn2Own2017 were around 800,000 dollars).

**Table 1.** Dimensions for characterizing crowdsourcing vulnerability discovery

Dimension	Brief Description	Possibilities
<b>Crowd Size</b>	Number of SecPros to participate in the task	Small; Medium; Large
<b>Incentives</b>	Motivations for SecPros to engage in the task	Monetary; fame and glory; passion
<b>Duration of the Task</b>	The amount of time the task would be running for	time-boxed; open; hybrid
<b>Task Context</b>	The information provided to SecPros to perform the task	None; little; medium; extensive
<b>Task Management</b>	The party responsible for managing the task (e.g., communicating with the crowd, and payments)	Organizations; platforms; shared between organizations and platforms.
<b>Selection of SecPros</b>	The selection of SecPros to participate in the task	SecPros self-select; organizations select according to certain criterion (e.g., credentials, experience, and reputation).
<b>Information protection</b>	The protection against exposure of sensitive information (e.g., customer information, PII), or Intellectual property (e.g., source code, unreleased software)	Manual methods; encrypting or masking; proxy access; staging environment; privacy policies; terms and conditions, limit the access of SecPros to public information; vetting SecPros; splitting the task to small modules
<b>Legal Terms</b>	The form of legal contract between the SecPros and the organization	No explicit legal terms relying on Anti-hacking laws, explicitly dictating weaving of legal pursuit when SecPros commit to terms and conditions, authorizing SecPros to discover vulnerabilities bonded by platforms term and conditions and the organization terms and conditions

### 3 Dimensions of Crowdsourcing Vulnerability Discovery

To compare, contrast and appreciate the underlying design of each of the models, we reviewed literature and the current implementations of crowdsourcing vulnerability discovery and identified eight dimensions that characterize these models (as shown in Table 1). We build upon other models that employ the use of crowdsourcing more generally [4, 6]. The identified dimensions can be used to describe not only the current models for crowdsourcing vulnerability discovery but also possible variations. Examples for each model are illustrated in Table 2.

### 3.1 Crowd Size

Greater number of participating SecPros means a higher probability that vulnerabilities will be discovered and reported [2, 3]. The burden to manage the crowd and verify the legitimacy of submitted vulnerability reports also increases [7]. In the direct model, the potential number of SecPros participating is large since the task is open for everyone. While for the platform managed model, the potential number is medium to small, since the organization controls the number of SecPros invited to participate and is limited by the size of the community maintained by the platform. On the other hand, in the contest model the potential number of SecPros are small since it is restricted by the capacity of the venue where the cyber security contest is held.

### 3.2 Incentives

It has been observed that the crowd is motivated by money, fame and glory, and passion [6, 8, 9]. In the direct vulnerability discovery model, a combination of money, fame and glory and passion is relied upon. SecPros, in addition to acquiring cash rewards, usually are mentioned in the hall of fame or gratitude page (e.g., Facebook white hat program). On the other hand, for the platform managed model, money and passion play an important factor in motivating SecPros. Additionally, platforms tend to rely on gamifications and ranking SecPros according to their contributions to increase the level of engagement (e.g., Hackerone leader board). For the cyber security contest model, the most important incentives are money, fame and glory since such contests often attract large attention from the media compared to other models.

### 3.3 Duration of the Task

In the direct model, the task duration is not time-boxed and hence SecPros may submit vulnerability reports at any time (ideally as soon as a vulnerability is discovered before other SecPros). In the platform managed model, the organization has the choice of running a time-boxed or open-ended program. Whereas, in the contest model the task is time-boxed which may indicate that it is the fastest form, but sometimes SecPros lean toward vulnerability hoarding (i.e., discovering vulnerabilities beforehand and not disclosing them until the time of the contest). Google stated that one of the reasons for terminating the Pwnium contest was to “remove the incentives for bug hoarding”<sup>4</sup>. On the other hand, it has been discovered that running the task for longer time give SecPros more time to discover vulnerabilities and potentially increase the number of vulnerabilities submitted [3].

### 3.4 Task Context

To perform vulnerability discovery tasks, SecPros may require information to help them on their quest (e.g., test dataset, instructions how to bypass a security mechanism, or source code of the software). Information about the software and development environment has been proven to be important to help SecPros to perform the task [9].

<sup>4</sup> [blog.chromium.org/2015/02/pwnium-v-never-ending-pwnium.html](http://blog.chromium.org/2015/02/pwnium-v-never-ending-pwnium.html).

In the direct model, the organization shares publicly available information. In the platform managed model, the organization is more comfortable sharing information (e.g., credentials to bypass web proxy, test accounts, access to staging environment), especially if the vulnerability discovery program is private (i.e., could be visible only to invited SecPros). In the contest model, little context (e.g., the software for testing and general rules) is provided in the beginning (as the task information is available publicly), but further information could be supplied on-demand to SecPros at the contest venue (e.g., access to the system through the network, entry points that can be used to simulate the attack). Task Management.

### 3.5 Task Management

Managing tasks includes managing vulnerability description reports (e.g., identifying out of scope reports, identifying duplicates, storing reports securely), communicating with SecPros (e.g., requesting clarification, requesting proof of concept), managing rewards (e.g., payments to SecPros, managing the hall of fame and gratitude pages). In the direct model, the organization is responsible for managing the task. In the platform managed model, the task is managed by the platform. In the contest model, the task could be managed by the organization, the platform or combination of both.

### 3.6 Selecting Security Professionals

Different vulnerability discovery tasks require that SecPros have different sets of skills, knowledge, and expertise (e.g., Web application vulnerabilities require knowledge about the software itself, networking protocols, Web frameworks, and the types of Web vulnerabilities). Hence, it is important to select qualified SecPros. Previous research identified the indicators to the expertise of SecPros participating in the crowdsourced task of vulnerability discovery [10]. Also, it has been discovered that selecting SecPros with high skill diversity help discover more diverse vulnerability types [8].

In the direct model, the SecPros self-select themselves, although organizations may sometimes impose entry conditions (e.g., Mozilla bug bounty guidelines require that no employee of the Mozilla foundation or affiliated service provider is eligible to participate). Whereas, in the platform managed model, the platform maintains SecPros' profiles including details provided by the SecPros themselves (e.g., certification, public professional profiles) along with their ongoing platform measured performance (e.g., number of vulnerabilities discovered, ranking against other SecPros). Organizations may invite SecPros within the community according to certain criterion (e.g., allow only the top 100 ranked SecPros, allow only CISSP certified SecPros). In the contest model, although SecPros self-select themselves to participate in the task, entry conditions may be imposed (e.g., traveling to the contest location).

### 3.7 Information Protection

Major privacy concerns arise when software providers are required to share data that is needed for vulnerability discovery. Even when providing anonymized (or synthetic) data, it is still possible that sensitive data be leaked by mistake or inferred via aggregation and correlation analysis. Other privacy threats may arise when the system has a vulnerability that would grant access to private information (e.g., a software glitch in the bicycle-sharing program Citi-Bike leaked customer details by accident, which led to a breach of privacy policies). Vulnerability description reports need to be protected as well, since their disclosure before discovered vulnerabilities are repaired could be damaging to organizations.

In the direct model, organizations rely on special controls to protect sensitive data from malicious attackers (e.g., automated tools to monitor access and report to internal staff for response and encrypting or masking sensitive data). In the platform managed model, since tasks are released to members of a closed community, all members of the community must consent to the terms and conditions imposed by the platform before participating in tasks. Moreover, since the number of invited SecPros is known, the organizations can provide proxy access to the task environment and monitor all the network traffic when SecPros are accessing the system in order to block any suspicious activity. In addition, a staging environment could be used for vulnerability testing (e.g., Movember bug bounty program run with Bugcrowd allowed SecPros to access a staging environment for vulnerability discovery). Similarly, in the contest model a prearranged environment is setup and SecPros agree to the terms and conditions before participating in tasks.

Intellectual Property leakage could result in the loss of competitive advantage, which may in turn result in financial losses (e.g., leakage of IP cost Sony millions of dollars in the latest Sony security breach incident). An organization may share sensitive intellectual property (e.g., source code for review, or access to an unreleased application) with SecPros. To protect IP, organizations in the direct model, limits access to public and restricted information to avoid potential risks, albeit this comes at a cost of reduced capabilities of the SecPros in performing vulnerability discovery. On the other hand, in the platform managed model the platform may choose to only disclose to SecPros with verified identities and backgrounds. In addition, SecPros are bound to sign nondisclosure agreements (NDAs). Similarly, in the contest model, all SecPros need to register to participate in the contest (i.e., verifying their identities) and sign NDAs before participation. Another possible solution to mitigate IP leakage is to split the task into micro tasks (e.g., modules of the system to be tested for vulnerabilities). By having access to only small chunks of code or modules of the software, SecPros will not be able to deduce the overall functionality of the system. In a similar context, LaToza et al. suggested a methodology to decompose programming work into micro tasks and found that the benefit of decomposing the task outweigh the overhead [11].

**Table 2.** Examples of crowdsourcing vulnerability discovery models

Dimension	Direct (Mozilla)	Platform Managed (Bugcrowd-Movember)	Contest (Pwn2Own)
<b>Crowd Size</b>	Large	Medium	Small
<b>Incentives</b>	Money; Passion; Fame; Glory	Money; Passion	Money; Fame; Glory
<b>Duration of the task</b>	Open	Time Boxed	Time Boxed
<b>Task Management</b>	Organizations	Platform	Organizations and Platform
<b>Selecting SecPros</b>	SecPros Self-select	The organization Select from a Closed Community	SecPros Self-select
<b>Information protection</b>	Rely on Internal Staff and Methods and Sharing only Public data	Staging Environment	Staging Environment
<b>Legal Terms</b>	No legal Authorization and only binding SecPros with Anti-hacking laws	Platforms T&C and Organization T&C weaving legal liability	Authorizing the SecPros Directly

### 3.8 Legal Terms

Relying on the crowd to discover vulnerabilities is governed by legal terms that represents a contract between SecPros and organizations. Missing detailed legal terms would cause inconvenience for both parties and it especially puts the SecPros at greater risks if the task is governed by the anti-hacking laws only [12]. In the direct model, organizations explicitly state legal terms as a binding contract (e.g., Microsoft have detailed legal terms and conditions stating what are the SecPros authorized to do, the obligations of the SecPro and what are the conditions in which Microsoft would take legal actions). In the platform managed programs, the platform has legal binding terms in addition to the legal terms the organization has (e.g., Hackerone has their own legal terms and conditions and the organization running the program can add their own legal terms). In the contest model, the organization and the SecPro will be under more specific and direct legal contract (e.g., SecPros to sign agreements to be able to participate).

## 4 Discussion and Future Research Directions

Reflecting on the above analysis, we identify a range of possible future research directions:

### 4.1 Improving the Quality of Vulnerability Tasks Descriptions and Reports

Ensuring the clarity and completeness of the task description is important to allow SecPros to perform the task properly [13], platforms may employ machine learning



models to assess task description quality regarding clarity, specificity and completeness. This may involve textual analytics and machine learning algorithms that gauge the effectiveness of crowd task postings. Additionally, the verification and repair of vulnerabilities would be delayed if poorly written reports are submitted, as it would take more time to identify and understand the problems. This is exemplified in an incident where a SecPro exploited a vulnerability in Facebook to prove its legitimacy after being rejected due to quality issues<sup>5</sup>.

It would also be interesting to investigate natural language processing (NLP) techniques to review the submitted vulnerability reports, and to possibly provide warnings or suggestions to improve the quality of the report prior to submission. One aspect worth investigating is whether the vulnerability described in the submitted report is within the scope of the task and notifying the SecPro before the report is submitted. This will minimize the number of invalid reports the organization may receive. Another interesting approach would be to rely on the crowd to provide feedback and enhance quality, or even to verify the legitimacy of the vulnerability reports. Allowing SecPros to assess and comment on the reports could help improve the quality of the vulnerability reports. In a similar context, Top-Coder relies on the crowd to provide code review for submitted software<sup>6</sup>. In the same context but by different approach, Su and Pan proposed a platform where administrators can collaborate on vulnerability report verification [14].

## 4.2 Protecting Against Intellectual Property Leakage

It is also important to investigate task decomposition techniques in order to ensure that SecPros will not have full access to the source code, or full view of the functionality of the software. However, an important challenge is to identify what types of vulnerabilities could be discovered in isolation and which types require complete view of the source code. As an example, buffer overflow and over-bound memory reads could be considered as vulnerabilities that could be discovered in isolation while business logic vulnerabilities may not be that straightforward. Investigating the different types of vulnerabilities, and how their discovery could be facilitated is an important research direction.

An additional interesting research direction would be to investigate what type of vulnerability discovery model would be most suitable for the type of software being tested. As an example, the direct model would be more suitable for open-source software since the IP leakage concerns would be minimized.

## 4.3 Crowdsourcing Vulnerability Discovery Quality Analytics

Although a wealth of information about SecPros and organizations are available (e.g., profiles, achievement portfolios, work history), most remains largely unharnessed.

A possible research direction is to model and capture the SecPros' behavioral patterns as a summary in terms of both aggregated work metrics (e.g., reputation scores, average time spent on tasks, number of unaccepted vulnerability reports) and semantically

<sup>5</sup> [techcrunch.com/2013/08/18/security-researcher-hacks-mark-zuckerbergs-wall-to-prove-his-exploit-works](http://techcrunch.com/2013/08/18/security-researcher-hacks-mark-zuckerbergs-wall-to-prove-his-exploit-works).

<sup>6</sup> [www.topcoder.com](http://www.topcoder.com).

meaningful behavior categories (e.g., honest, deceiver or colluding). Recent empirical research has discovered various categories of crowd workers (e.g., ineligible, fast deceivers, smart deceivers, gold star preys) [15].

In order to predict the quality of the vulnerability report, it is important to investigate non-intrusive and statistical analysis techniques over the SecPros' activity history for given task types. A key issue that is not considered in existing techniques is uncertainty related to predicting performance of new SecPros (i.e., cold start case). Relying on indicators extracted from various sources (e.g., activities in SecPro communities like BlackHat<sup>7</sup>, contribution to open-source projects in GitHub), it would be possible to "predict" the quality of submitted vulnerability discovery reports and detect the malicious behavior of the SecPros.

Using dynamic access control based on the allocation of "roles" is an interesting direction to investigate. For instance, a SecPro who is not known to a vulnerability discovery platform is allocated to the "untrusted" role. Their trust level can increase as the platform collects more information about them (e.g., task performance or reputation in external platforms). Moreover, the trust level would also increase as the SecPros improve their reputation score (e.g., discovering new vulnerabilities and disclosing them responsibly, or acquiring some achievement on other platforms). On the other hand, if SecPros did not adhere to the guidelines or best practices, then their trust would decrease.

Another interesting research direction is to investigate how to combine crowdsourcing with machine automation to improve vulnerability discovery. Votipka et al. characterized the process of vulnerability discovery [9]. A possible direction is investigate which phases of the process will benefit from automation and which phases will be benefit from human computation, as well investigation of hybrid (i.e., combing human and machine computation [16]). As an example, tasks like information gathering and attack surface detection can take advantage from automation. This can save time and effort for SecPros who can focus on other tasks like vulnerability recognition.

## 5 Conclusion

There are several models for crowdsourcing vulnerability discovery and these models are characterized by different dimensions. Understanding the models and dimensions for crowdsourcing vulnerability discovery helps practitioners make informed decisions when adopting this approach. It also paves the way for researchers to conduct a deeper investigation of the area if crowdsourcing is to have the same kind of impact in vulnerability discovery that it has in other fields.

## References

1. Finifter, M., Akhawe, D., Wagner, D.: An empirical study of vulnerability rewards programs. In: Proceedings of the 22Nd USENIX Conference on Security, pp. 273–288 (2013)
2. Maillart, T., Zhao, M., Grossklags, J., Chuang, J.: Given enough eyeballs, all bugs are shallow? Revisiting Eric Raymond with bug bounty programs. *J. Cybersecur.* **3**, 81–90 (2017)

<sup>7</sup> [www.blackhat.com](http://www.blackhat.com).

3. Zhao, M., Grossklags, J., Liu, P.: An empirical study of web vulnerability discovery ecosystems. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security - CCS 2015, pp. 1105–1117 (2015)
4. LaToza, T., van der Hoek, A.: Crowdsourcing in software engineering: models, motivations, and challenges. *IEEE Softw.* **33**(1), 74–80 (2016)
5. Al-Banna, M., Benatallah, B., Schlagwein, D., Bertino, E., Barukh, M.: Friendly hackers to the rescue: how organizations perceive crowdsourced vulnerability discovery. In: Pacific Asia Conference on Information Systems (PACIS) (2018)
6. Malone, T.W., Laubacher, R., Dellarocas, C.: The collective intelligence genome. *IEEE Eng. Manag. Rev.* **38**(3), 38 (2010)
7. Laszka, A., Zhao, M., Grossklags, J.: Banishing Misaligned Incentives for Validating Reports in Bug-Bounty Platforms, pp. 161–178. Springer, Cham (2016)
8. Zhao, M., Grossklags, J., Chen, K.: An exploratory study of white hat behaviors in a web vulnerability disclosure program. In: Proceedings of the 2014 ACM Workshop on Security Information Workers - SIW 2014, pp. 51–58 (2014)
9. Votipka, D., Stevens, R., Redmiles, E., Hu, J., Mazurek, M.: Hackers vs. testers: a comparison of software vulnerability discovery processes. In: 2018 IEEE Symposium on Security and Privacy (SP), pp. 374–391 (2018)
10. Al-Banna, M., Benatallah, B., Barukh, M.C.: Software security professionals: expertise indicators. In: 2016 IEEE 2nd International Conference on Collaboration and Internet Computing (CIC), pp. 139–148 (2016)
11. LaToza, T.D., Ben Towne, W., Adriano, C.M., van der Hoek, A.: Microtask programming. In: Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology - UIST 2014, pp. 43–54 (2014)
12. Gamero-Garrido, A., Savage, S., Levchenko, K., Snoeren, A.C.: Quantifying the pressure of legal risks on third-party vulnerability research. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security - CCS 2017, pp. 1501–1513 (2017)
13. Zhao, M., Laszka, A., Grossklags, J.: Devising effective policies for bug-bounty platforms and security vulnerability discovery. *J. Inf. Policy* **7**, 372 (2017)
14. Su, H.-J., Pan, J.-Y.: Crowdsourcing platform for collaboration management in vulnerability verification. In: 2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS), pp. 1–4 (2016)
15. Gadiraju, U., Kawase, R., Dietze, S.: Understanding malicious behavior in crowdsourcing platforms: the case of online surveys. In: 33rd Annual ACM Conference on Human Factors in Computing Systems, pp. 1631–1640 (2015)
16. Krivosheev, E., Casati, F., Baez, M., Benatallah, B.: Combining crowd and machines for multi-predicate item screening. *Proc. ACM Hum.-Comput. Interact.* **2**(CSCW), 1–18 (2018)



# Expertise-Aware Crowdsourcing Taxonomy Enrichment

Yuquan Wang<sup>1</sup>(✉), Yanpeng Wang<sup>2</sup>, Yiming Mao<sup>3</sup>, Jifan Yu<sup>1</sup>, Kaisheng Zeng<sup>1</sup>,  
Lei Hou<sup>1</sup>, Juanzi Li<sup>1</sup>, and Jie Tang<sup>1</sup>

<sup>1</sup> Department of Computer Science and Technology, BNRist, Tsinghua University,  
Beijing 100084, China

{wangyuqu19,yujf18,zks19}@mails.tsinghua.edu.cn,  
{houlei,lijuanzi,jietang}@tsinghua.edu.cn

<sup>2</sup> School of Computer Science and Engineering, Beihang University,  
Beijing 100191, China

wyp1998911@icloud.com

<sup>3</sup> Global Innovation Exchange (GIX) Institute, Tsinghua University,  
Beijing 100084, China

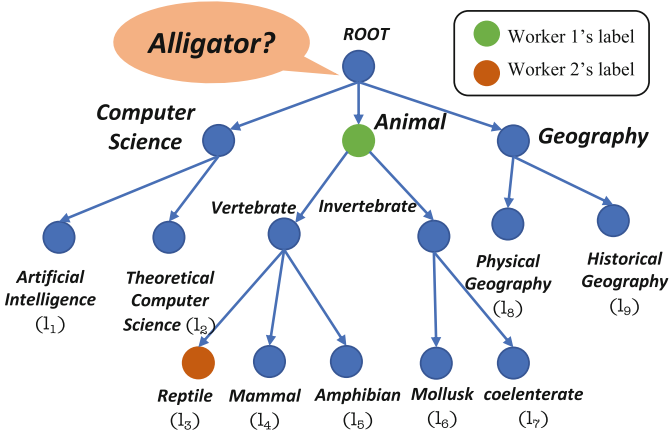
mym19@mails.tsinghua.edu.cn

**Abstract.** Taxonomy is an indispensable part of Knowledge Base. Taxonomy Enrichment is the task that new instances are classified to possibly **thousands of concepts** of an existing taxonomy. Recent works considered to enrich a taxonomy by crowdsourcing [17, 24] to promote the quality of new instance classification. However, the **broad coverage of knowledge** induced by a large taxonomy makes it a skill-sensitive knowledge-intensive crowdsourcing (KI-C) task. Naively applying existing skill estimation methods [23, 25, 36] for quality control requires quizzes on each concept of the taxonomy, which is prohibitive for large scale Taxonomy Enrichment. In this work, we propose a unified crowdsourcing framework to mitigate both challenges. It leverages the skill locality of workers with a Graph Gaussian Process model. Our quality control method automatically matches instances with a worker's expertise and gives hints for later workers to reduce their time to search over the taxonomy. In our experiments with 314 workers, our method outperforms baseline BTKS [9] by 6.6% in terms of accuracy on a taxonomy of 10 levels and 632 nodes.

**Keywords:** Crowdsourcing · Quality control · Taxonomy construction · Taxonomy enrichment · Knowledge base

## 1 Introduction

Taxonomy, providing hierarchical relations (*instanceOf* and *subClassOf*) among instances and classes [26], is the core in knowledge base and has wide applications in question answering [1], machine translation [11] and information extraction [10]. *Taxonomy Enrichment* [13, 22, 24] is an important step where an existing taxonomy is extended by adding new instances as illustrated in Fig. 1. Despite



**Fig. 1.** An example of Crowdsourcing Taxonomy Enrichment. Workers are asked to label where *alligator* belongs to in this taxonomy. Worker 1 labels it to *Animal* and worker 2 who is more skilled at distinguishing animals labels it to *Reptile*.

recent advancement in automatic Taxonomy Enrichment, maintaining high quality taxonomies still requires massive human annotation [13, 22, 28], which is especially challenging for large-scale taxonomies in practice.

Crowdsourcing provides a cheaper (but noisier) alternative to expert labeling for Taxonomy Enrichment [24]. For example, Fig. 1 shows a possible crowdsourcing case when labeling *alligator* to a concept of a large taxonomy whose ground truth concept is *Reptile*. A quick google shows that it is a crocodile-like animal, but it is still difficult for inexperienced workers to tell whether it belongs to *Reptile* or *Amphibian*. Nevertheless, a single worker does not need to know every knowledge covered by the taxonomy. One can label *alligator* to *Animal*, which could guide us to find workers good at distinguishing animals to label further.

However, Crowdsourcing Taxonomy Enrichment faces two challenges. First, a large-scale taxonomy may cover broad disciplines of fine-grained knowledge, e.g., YAGO [26] contains more than 350 000 classes spanning from Biology to Software and Music, making it prohibitive to **identify workers' expertise** with test questions beforehand on all concepts as in previous works [23, 27]. The lack of proper skill model limits their quality control methods of both task assignment [23] and answer aggregation [9] to small scale taxonomies.

Second, **searching answers over the entire taxonomy** with thousands of concepts takes workers redundant time and effort. Existing works for large taxonomies divide the concept search process into binary [24] or multiple [17] choice questions to lower task autonomy and avoid presenting thousands of choices to workers at once. Choices can be generated by automatic methods [22, 24, 28]. For example, the machine may guess *alligator* belongs to *Animal/Invertebrate/Mollusk* and generates a question with 3 choices for workers, i.e., “Does *alligator* belong to *Animal/Invertebrate/Mollusk* or ‘*Animal/Invertebrate* but not *Mollusk*’ or ‘*Animal* but not *Invertebrate*’?”. However, since the initial guess is wrong, annotations in this scheme only give the information that *alligator*

does NOT belong to the branch of *Invertebrate*. Further questions have to make guesses among leaf concepts of all other branches of *Animal*. Moreover, limited task autonomy may lower the quality and quantity of worker participation due to their decreased intrinsic motivation [16, 38], especially in knowledge-intensive crowdsourcing (KI-C) tasks that require diverse knowledge skills [38].

In this work, we propose a new crowdsourcing framework for Taxonomy Enrichment to mitigate these challenges. First, it performs *skill estimation* with a Graph Gaussian Process model to identify workers’ expertise, i.e., estimates confusion matrices and confidence of them. These are then used in *answer aggregation* in each round, guiding online *task assignment* to find proper tasks for workers. Second, instead of searching the entire taxonomy or constraining to only multiple choice questions, we propose subtree recommendation to present workers with a small fragment the taxonomy to increase the efficiency of annotation. For example, recommending subtree of *Animal* for worker 2 after worker 1’s label in Fig. 1 and let her navigate freely from there with high task autonomy.

We conduct both synthetic and real-world experiments to empirically verify our claims. In the real-world experiment of a 10-level taxonomy of 632 nodes, our framework outperforms baseline methods by 6.6% in terms of accuracy and reduces average annotation time by 7.1%. The code can be found at this URL<sup>1</sup>. Our contribution includes:

a) We propose an expertise-aware crowdsourcing framework for Taxonomy Enrichment. Our model can estimate workers’ skills on thousands of concepts, even those without gold answers using skill locality assumptions.

b) We propose an answer aggregation method based on Graph Gaussian Process and a task assignment method with subtree recommendation, enabling workers to directly label instances to massive concepts with high task autonomy.

c) We conduct both simulated and real-world experiments to evaluate the proposed method. Our crowdsourcing framework is used in a large scale dataset MOOCCube [35] with over 510k instances enriched to taxonomies.

## 2 Problem Formulation

In this section, we formally define Taxonomy Enrichment and highlight the key problems in quality control for Crowdsourcing Taxonomy Enrichment (CTE).

**Definition 1.** *Taxonomy*  $\mathcal{T} = (\mathcal{I}, \mathcal{C}, \mathcal{R})$  where  $\mathcal{I}$ ,  $\mathcal{C}$  and  $\mathcal{R}$  are set of instances and concepts and *isA* relations respectively.  $\mathcal{R} = \mathcal{R}_i \cup \mathcal{R}_s$  consists of *instanceOf* and *subClassOf* triples respectively.  $\mathcal{R}_i = \{(i, isA, c) \mid i \in \mathcal{I}, c \in \mathcal{C}\}$  collects all *instanceOf* triples and  $\mathcal{R}_s = \{(c, isA, c') \mid c, c' \in \mathcal{C}\}$  collects all *subClassOf* triples.

Taxonomy Enrichment is a specific kind of taxonomy construction where  $\mathcal{C}$  and  $\mathcal{R}_s$  are fixed and new instances are added into the proper level of the taxonomy, typically leaf concepts.

<sup>1</sup> <https://github.com/THU-KEG/ECTE>.

**Definition 2.** Given  $\mathcal{T} = (\mathcal{I}, \mathcal{C}, \mathcal{R})$ , **Taxonomy Enrichment** is to convert  $(\mathcal{I}', \mathcal{T})$  into an enlarged taxonomy  $\mathcal{T}' = (\mathcal{I}' \cup \mathcal{I}, \mathcal{C}, \mathcal{R}')$ , where  $\mathcal{I}' \cap \mathcal{I} = \emptyset$ ,  $\mathcal{R}' = \mathcal{R}'_i \cup \mathcal{R}$  and  $\mathcal{R}'_i$  is the set of new instanceOf relations for instances in  $\mathcal{I}'$ .

Normally, the taxonomy  $\mathcal{T}$  forms a tree with  $\mathcal{C}$  and  $\mathcal{R}_s$  as its nodes and edges. In addition, the children of every concept  $c \in \mathcal{C}$  represent a partition of  $c$ , i.e., they are disjoint concepts and their union is exactly  $c$ . We use  $\mathcal{L} = \text{leaf}(\mathcal{T}) \subset \mathcal{C}$  to denote leaf concepts of  $\mathcal{T}$  and use  $l$  to denote a leaf concept.

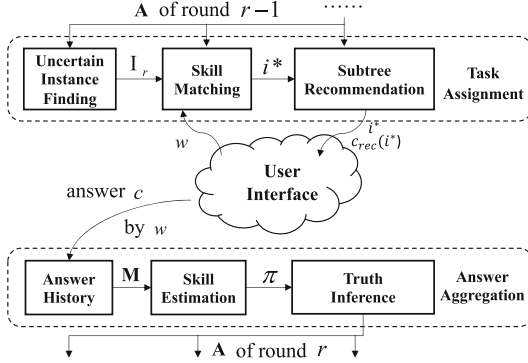
When  $\mathcal{I}'$  is large, the workload of Taxonomy Enrichment becomes prohibitive for a small group of experts. Crowdsourcing provides a way to utilize the collective knowledge. In Taxonomy Enrichment, crowdsourcing is a process where a *requester* gives  $\mathcal{I}' \cup \mathcal{I}_g$ ,  $\mathcal{C}$  and  $\mathcal{R}_s$  to a set of *workers*  $\mathcal{W}$ , who return their belief of *instanceOf* relations between  $\mathcal{I}' \cup \mathcal{I}_g$  and  $\mathcal{C}$ , where  $\mathcal{I}_g \subset \mathcal{I}$  is a set of instances called *gold instances* aiming to monitor workers' performance. Given an instance  $i \in \mathcal{I}' \cup \mathcal{I}_g$ , a worker is encouraged to assign it to the leaf concepts, i.e.,  $(i, \text{instanceOf}, l)$ , and non-leaf annotations are also acceptable.

Given  $\mathcal{I}'$ ,  $\mathcal{T}$  and  $\mathcal{W}$ , the core problem for **Crowdsourcing Taxonomy Enrichment** consists of *task assignment* and *answer aggregation*. In practice, the budget for crowdsourcing is usually limited. When a worker is ready to label, **task assignment** returns an instance  $i^* \in \mathcal{I}' \cup \mathcal{I}_g$ . After annotation, workers may give conflicting answer concepts on same instances. **Answer aggregation** takes answers of all workers and returns the probability  $P(i \in l)$  for every  $i \in \mathcal{I}' \cup \mathcal{I}_g$  and every  $l \in \mathcal{L}$ , where  $i \in l$  denotes relation  $(i, \text{instanceOf}, l)$ . To address the challenge that Taxonomy Enrichment requires identifying skills of a worker among large amounts of concepts, our **skill estimation** method estimates a worker's skill, i.e., the confusion matrix [5], from her previous annotations with locality assumptions introduced in Sect. 3.3. Considering that finding the best concept out of  $|\mathcal{C}|$  concepts is time-consuming and labor-intensive, we propose **subtree recommendation**, i.e., along with the assigned task  $i^*$ , we recommend a concept  $c_{\text{rec}(i^*)}$  for workers to start searching, as opposed to always traversing  $\mathcal{T}$  from root.

*Example 1.* In the taxonomy shown in Fig. 1,  $\mathcal{C}$  consists of 15 concepts including the virtual node *ROOT*, and  $\mathcal{L}$  is the subset with 9 leaf concepts. Before crowdsourcing,  $\mathcal{T}$  has three instances  $\mathcal{I} = \{\text{dog}, \text{word2vec}, \text{frog}\}$  which belong to *Mammal*, *ArtificialIntelligence* and *Amphibian* respectively. We select two of them as gold instances  $\mathcal{I}_g = \{\text{word2vec}, \text{frog}\}$ . We have three workers  $\mathcal{W} = \{w_1, w_2, w_3\}$  and two unknown instances  $\mathcal{I}' = \{\text{alligator}, \text{rock}\}$ . An example task is to find out which leaf concept every instance of  $\mathcal{I}'$  belongs to from  $\mathcal{L}$  by asking the crowd under a limited budget, e.g., 10 answers at most.

### 3 Our Method

In this section, we propose a crowdsourcing framework shown in Fig. 2. In every round, our framework performs task assignment and answer aggregation. Our crowdsourcing framework allows workers to join or quit at any time and label



**Fig. 2.** Round  $r$ 's workflow of our crowdsourcing framework.

any amount of instances. We always assign a task to a worker upon request. For notation simplicity, we use  $\mathcal{I}$  to denote  $\mathcal{I}' \cup \mathcal{I}_g$  in the following sections because we do not distinguish  $\mathcal{I}'$  and  $\mathcal{I}_g$  when assigning instances.

In round  $r$ , **task assignment** takes *aggregation matrix*  $\mathbf{A}$  of round  $r - 1$  (or prior belief on answers in the first round) and selects  $\gamma|\mathcal{I}|$  most uncertain instances for some  $\gamma \in (0, 1)$  as instances  $\mathcal{I}_r$  to be labeled, where aggregation matrix  $\mathbf{A} \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{L}|}$  represents aggregated answers.  $\mathbf{A}_{il}$  is the probability of  $i \in \mathcal{I}, \forall i \in \mathcal{I}, \forall l \in \mathcal{L}$ . When a worker  $w$  asks for an instance, it returns a recommended instance  $i^*$  matching her skill and a concept  $c_{rec}(i^*)$  to start searching. We call the procedure of recommending a starting concept *subtree recommendation*. Workers receive recommended instances and starting concepts until all instances in  $\mathcal{I}_r$  are annotated.

After all instances of  $\mathcal{I}_r$  are annotated by workers, we perform **answer aggregation**, which consists of *skill estimation* to discover workers' expertise within  $\mathcal{T}$  and *truth inference* to infer answers of each instance. Answers of workers are formally defined as an *answer tensor*  $\mathbf{M} \in \{0, 1\}^{|\mathcal{W}| \times |\mathcal{I}| \times |\mathcal{C}|}$  where  $\mathbf{M}_{ic}^w = 1$  if and only if worker  $w$  chooses  $c$  for instance  $i$  and  $\mathbf{M}_{ic}^w = 0$  otherwise. A *confusion matrix*<sup>2</sup>  $\boldsymbol{\pi}^w \in \mathbb{R}^{|\mathcal{L}| \times |\mathcal{C}|}$  of worker  $w$  depicts  $w$ 's skill on each leaf concept  $l$  as follows. For every concept  $c$ ,  $\boldsymbol{\pi}_{lc}^w$  represents the probability of worker  $w$  labels an instance of ground truth leaf concept  $l$  to concept  $c$ . Note that our confusion matrix has more columns than rows because  $l$  is a leaf concept but  $c$  may not be a leaf concept. Skill estimation turns  $\mathbf{M}$  into  $\boldsymbol{\pi}$  and truth inference turns  $\mathbf{M}$  and  $\boldsymbol{\pi}$  into  $\mathbf{A}$ . After budget runs out, our framework returns  $\mathbf{A}$ . The requester may choose  $\arg \max_{l \in \mathcal{L}} \mathbf{A}_{il}$  as answer for every instance  $i \in \mathcal{I}$ .

*Example 2.* Consider two rounds of annotation of Example 1. Suppose  $\mathcal{I}_1 = \{\text{alligator}, \text{frog}, \text{rock}\}$  and  $\mathcal{I}_2 = \{\text{word2vec}, \text{frog}, \text{alligator}\}$ . In round 1,  $w_1$  asks the task assignment component for an instance to label and gets *alligator*. Then

<sup>2</sup> We use a matrix (or tensor) with several subscripts or superscripts to denote the corresponding slice. For example,  $\boldsymbol{\alpha}$  denotes the matrix of all workers' skill score,  $\boldsymbol{\alpha}^w$  for worker  $w$ 's skill score vector and  $\boldsymbol{\alpha}_l^w$  for worker  $w$ 's skill score on leaf  $l$ .



$w_1$  submits *Animal* back, i.e.,  $\mathbf{M}^{w_1}_{alligator,Animal} = 1$ . After all of  $\mathcal{I}_1$  are labeled, answer aggregation sets  $\mathbf{A}_{alligator,l}$  higher for each leaf  $l$  of *Animal*'s descendants.

Suppose  $w_2$  labels *frog* correctly to *Amphibian* in round 1, our estimation of her skills near *Amphibian* goes up. In round 2, when  $w_2$  asks for an instance, the framework assigns *alligator*, which is labeled correctly to *Reptile* as in Fig. 1.

### 3.1 Uncertain Instances Finding and Skill Matching

We find  $\gamma|\mathcal{I}|$  instances  $\mathcal{I}_r$  for some  $\gamma \in (0, 1)$  for round  $r$ . At the beginning of each round  $r$ , we calculate the entropy of every instance  $i$ 's aggregated posterior distribution  $\mathcal{H}(\mathbf{A}_i)$  and find top  $\gamma|\mathcal{I}|$  instances of  $\mathcal{I}$  as  $\mathcal{I}_r$ .

To assign each instance to the most suitable worker, when worker  $w$  posts a task request, we assign the instance  $i^* \in \mathcal{I}_r$  whose  $\mathbf{A}_{i^*}$  matches skill score vector  $\boldsymbol{\alpha}^w$  best to  $w$ . i.e.

$$i^* = \arg \max_{i \in \mathcal{I}_r} \mathbf{A}_i^\top \mathbb{E}_{\boldsymbol{\alpha}^w \sim P(\boldsymbol{\alpha}|\mathbf{M})} \boldsymbol{\alpha}^w \quad (1)$$

The calculation of  $\boldsymbol{\alpha}^w$  is introduced in Sect. 3.3. It represents our approximation of worker  $w$ 's skill on leaf concepts. In practice, we approximate the expectation of Eq. 1 by averaging  $N$  samples of  $\boldsymbol{\alpha}^w$ 's by Eq. 3.

### 3.2 Subtree Recommendation

For huge taxonomies, searching for the most suitable concept could be extremely time-consuming. In our framework, a worker can select an internal node if she is not sure which leaf concept an instance belongs to. This rough information can help our task assignment algorithm to find other suitable workers to further label it. We leverage previous workers' labels to provide hints for later workers as well as to save their time finding the most suitable leaf concept.

We choose a hyperparameter threshold  $\tau \in (0, 1)$ . For any instance  $i$ , we call a node  $c$   $\tau$ -*recommendable* if  $P(i \in c|\mathbf{M}) > \tau$  and none of  $c$ 's children  $c' \in children(c)$  has  $P(i \in c'|\mathbf{M}) > \tau$ . Given  $\mathbf{A}$ , we calculate  $P(i \in c|\mathbf{M})$  by summing  $P(i \in l|\mathbf{M}) = \mathbf{A}_{il}$  for all  $c$ 's descendant leaf concepts  $l$ , i.e.,

$$P(i \in c|\mathbf{M}) = \sum_{l \in leaf(c)} \mathbf{A}_{il} \quad (2)$$

where  $leaf(c)$  denotes all descendant leaf concepts of  $c$ . This procedure guarantees that if  $P(i \in c|\mathbf{M})$  is estimated correctly, with probability  $\tau$ , the worker would save time. If there are multiple  $\tau$ -recommendable nodes for  $i$ , we randomly choose one as the recommended concept  $c_{rec(i)}$ .

In the user interface, when recommending a node  $c$ , we unfold every node  $p$  of the shortest path from *ROOT* to  $c$ , i.e., display  $p$ 's children for a worker. Workers can traverse the tree by clicking a node to unfold its subtree.

*Example 3.* Consider a perfectly balanced binary tree with root  $c_{root} = root(\mathcal{T})$  of height more than 2. Let  $c_1$  and  $c_2$  be the left child and right child of  $c_{root}$ .

Suppose we set uniform prior distribution to all leaf concepts for an instance  $i$  and set  $\tau$  to be 0.6, Then initially  $i$  has probability 0.5 of being in  $c_1$  and 0.5 of being in  $c_2$ . Neither of them is over  $\tau$ . Therefore  $c_{root}$  is 0.6-recommendable and it is recommended and we set  $c_{rec(i)}$  to be  $c_{root}$ .

If some workers have labeled  $i$  to some leaf concepts in the right branch, after answer aggregation, the framework may get aggregation matrix  $\mathbf{A}$  such that  $P(i \in c_2 \mid \mathbf{M}) = \sum_{l \in \text{leaf}(c_2)} \mathbf{A}_{il} > 0.6$ . In this case  $c_{root}$  is no longer 0.6-recommendable because a child of it has probability over 0.6. We can recursively prove that there must be a 0.6-recommendable node  $c$  in the subtree of  $c_2$ , which will be recommended in the next time a worker labels  $i$ .

### 3.3 Skill Estimation

Initially, we have no information of workers' skills and we want our framework to learn about workers as they label. The broad coverage of knowledge in Taxonomy Enrichment requires the algorithm to output skill estimation on  $|\mathcal{L}|$  concepts. This is particularly challenging when  $|\mathcal{L}|$  is large. What's more, workers normally label dozens of instances in practice in a medium-size crowdsourcing task [30] but  $|\mathcal{L}|$  can be as large as thousands. Fortunately, the skills of workers tend to have locality. For example, if a worker is good at labeling concepts in data structure, she is more likely to be good at concepts of computer algorithms compared with average workers. We capture accuracy locality in Assumption 1:

**Assumption 1 Accuracy Locality:** *workers have similar accuracy on concepts that are near to each other on taxonomy  $\mathcal{T}$ .*

Moreover, labeling biases tend to have locality. Consider a worker labeling instance *alligator*, whose correct concept is *Reptile*. If she labels incorrectly, she is more likely to label it to a concept near *Reptile*, for instance *Amphibian*, than some concepts far away from them, for instance *WordEmbedding* in *Computer-Science* branch [8]. We capture bias locality in Assumption 2:

**Assumption 2 Bias Locality:** *for instance  $i$ , workers are more likely to choose a concept near  $i$ 's ground truth concept even if they choose an incorrect concept.*

To estimate  $\pi^w$ , we first estimate  $w$ 's accuracy<sup>3</sup> on every  $l \in \mathcal{L}$  utilizing accuracy locality assumption. Next, we generate  $\pi^w$  by bias locality assumption.

We use *skill score*  $\alpha \in \mathbb{R}^{|\mathcal{W}||\mathcal{L}|}$  to model workers' skills on each leaf concept  $l \in \mathcal{L}$ . We use Gaussian Process defined on  $\mathcal{T}$ 's graph structure to model  $\alpha$ . Specifically,  $\alpha^w \sim \mathcal{GP}(\cdot \mid \mathbf{0}, \mathbf{K}^w)$  over  $\mathcal{T}$  for every worker  $w$ , where  $\mathbf{K}^w = \{k(c_{j_1}, c_{j_2})\}_{j_1, j_2=1}^{|\mathcal{L}|}$  can be any valid kernel matrix over  $\mathcal{T}$  with  $|\mathcal{L}|$  nodes. Without further mention, we use radial basis function kernel  $k(c_1, c_2) = \exp(-\eta \cdot d(c_1, c_2)^z)$  in the rest of this paper where  $\eta$  and  $z$  are hyperparameters controlling the smoothness of  $\alpha^w$  and  $d(\cdot, \cdot)$  is the graph distance on taxonomy  $\mathcal{T}$ , i.e., number of edges between nodes.

<sup>3</sup> We used a transformed accuracy called skill score defined below.

$\alpha_l^w$  captures how likely  $w$  chooses  $l$  or its nearby concepts if  $l$  is ground truth. If  $w$  has labeled gold instances of leaf concept  $l$ , we can observe her gold accuracy  $a_l^w$ . Then  $\alpha_l^w$  is defined as a deterministic monotonic transform  $t$  of  $a_l^w$ , i.e.,  $\alpha_l^w = t(a_l^w)$ . We use  $t(a_l^w) = \zeta a_l^w + \psi$  in our experiments where  $\zeta$  and  $\psi$  are scalar values encoding the prior belief of variance and mean of skill score that can be tuned according to datasets. Compared with MLE based methods, Bayesian methods like ours with an appropriately chosen prior may avoid overfitting [37] and perform better, especially when most workers ask for a small amount of tasks, as is often the case in real systems [30].

Suppose  $w$  has labeled gold instances on  $K$  leaf concepts  $\widetilde{\mathcal{L}}^w = \{l_k\}_{k=1}^K$  and the gold accuracy is  $\{a_{l_k}^w\}_{k=1}^K$ . To estimate  $w$ 's skill score on all leaf concepts with Gaussian Process, it follows that

$$\alpha^w | \widetilde{\alpha}^w, \widetilde{\mathcal{L}}^w, \mathcal{L}^w \sim \mathcal{N}(\boldsymbol{\mu}^w, \boldsymbol{\Sigma}^w) \quad (3)$$

$$\begin{aligned} \boldsymbol{\mu}^w &= \mathbf{K}(\mathcal{L}^w, \widetilde{\mathcal{L}}^w)(\mathbf{K}(\widetilde{\mathcal{L}}^w, \widetilde{\mathcal{L}}^w) + \sigma^2 I)^{-1} \widetilde{\alpha}^w \\ \boldsymbol{\Sigma}^w &= \mathbf{K}(\mathcal{L}^w, \mathcal{L}^w) + \sigma^2 I \\ &\quad - \mathbf{K}(\mathcal{L}^w, \widetilde{\mathcal{L}}^w)(\mathbf{K}(\widetilde{\mathcal{L}}^w, \widetilde{\mathcal{L}}^w) + \sigma^2 I)^{-1} \mathbf{K}(\widetilde{\mathcal{L}}^w, \mathcal{L}^w) \end{aligned} \quad (4)$$

$\sigma$  is set to a small value as skill observation noise and  $\widetilde{\alpha}^w = t(\widetilde{a}^w) = [t(a_l^w)]_{l \in \widetilde{\mathcal{L}}^w}$ .

*Example 4.* Consider Taxonomy Enrichment problem of Example 1. Suppose worker  $w_2$  has labeled gold instances *word2vec* wrongly to *TheoreticalComputerScience* ( $l_2$ ) and *frog* correctly to *Amphibian* ( $l_5$ ). We choose  $\zeta = 1$  and  $\psi = 0$  for illustration purpose. In this case  $\widetilde{\alpha}^{w_2}$  is a 2-element vector with  $\widetilde{\alpha}_{l_1}^{w_2} = t(0/1) = 0$  and  $\widetilde{\alpha}_{l_5}^{w_2} = t(1/1) = 1$ . Let  $d$  be edge distance of graph. We have  $d(l_1, l_5) = 5$  and  $d(l_3, l_5) = 2$ . We use  $d$  to get  $\mathbf{K}$  and Eq. 4 to get  $\boldsymbol{\mu}^{w_2}$  and  $\boldsymbol{\Sigma}^{w_2}$ . Now we can draw  $\alpha^{w_2}$  samples according to Eq. 3, which are 9-element vectors representing  $w_2$ 's skills on all leaves of  $\mathcal{T}$ .

Finally we use  $\alpha^w$  to generate confusion matrix  $\boldsymbol{\pi}^w$ , which is widely used to model worker skills in crowdsourcing frameworks [5, 32, 33].

$$\pi_{lc}^w(\alpha) = \frac{\exp(-\lambda s(\alpha_l^w) \cdot d(l, c))}{\sum_{c' \in \mathcal{C}} \exp(-\lambda s(\alpha_l^w) \cdot d(l, c'))} \quad (5)$$

where  $\lambda$  is a tunable hyperparameter and  $s(\cdot)$  is the softplus function.  $\pi_{lc}^w$  is the probability of worker  $w$  choosing  $c$  for an instance whose ground truth is  $l$ .

### 3.4 Truth Inference

Workers may give conflicting answer concepts on the same instance. Unskilled workers may provide answers with more noise than skilled workers. Our truth inference is a natural application of the proposed worker skill estimation model. Truth inference procedure uses  $\boldsymbol{\pi}$  and  $\mathbf{M}$  to derive the posterior distribution of every instance's leaf concepts, represented by aggregation matrix  $\mathbf{A}$ . By Bayesian theorem, aggregation matrix  $\mathbf{A}_{il}$  can be calculated by

$$\mathbf{A}_{il} = P(i \in l | \mathbf{M}) = \frac{P(\mathbf{M} | i \in l)P(i \in l)}{\sum_{l' \in \mathcal{L}} P(\mathbf{M} | i \in l')P(i \in l')} \quad (6)$$

The prior distribution  $P(i \in l)$  can be set to uniform distribution or the distribution produced by automatic methods [22, 28]. In practice, we use Monte Carlo sampling method to calculate  $P(\mathbf{M} | i \in l)$ . After getting aggregation matrix  $\mathbf{A}$ , we may choose  $l^* = \arg \max_{l \in \mathcal{L}} \mathbf{A}_{il}$  as  $i$ 's answer. We summarize our method in Algorithm 1.

---

**Algorithm 1.** The Whole Crowdsourcing Process
 

---

**Input:**  $\mathcal{T}, \mathcal{I}, \mathcal{W}$ 
**Parameter:**  $\gamma, \zeta, \phi, \eta, \lambda, \tau, \sigma, z, N$ 
**Output:** aggregation matrix  $\mathbf{A}$ 

```

1:  $(\mathbf{M}, \boldsymbol{\alpha}, r, \mathcal{I}_r) \leftarrow (\mathbf{0}^{|\mathcal{W}| \cdot |\mathcal{I}| \cdot |\mathcal{C}|}, \mathbf{0}^{|\mathcal{W}| \cdot |\mathcal{L}|}, 0, \emptyset)$ 
2:  $\mathbf{A} \leftarrow$  Prior distribution
3: while budget does not run out do
4:   if  $\mathcal{I}_r$  is not empty then
5:     Get next worker  $w$ 
6:     Give  $w$  the instance  $i^* \in \mathcal{I}_r$  matching  $\boldsymbol{\alpha}$  best by (1)
7:     Return  $i^*$  and  $c_{rec}(i^*)$  to  $w$ 
8:     Get  $w$ 's answer  $c$ 
9:      $\mathbf{M}_{i^*c}^w \leftarrow 1$ 
10:  else
11:    Calculate each worker's gold accuracy  $\tilde{a}$  from  $\mathbf{M}$ 
12:    Calculate  $\boldsymbol{\mu}$  and  $\boldsymbol{\Sigma}$  by (4)
13:    Sample  $N$   $\boldsymbol{\alpha}$ s from  $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ 
14:    Calculate  $N$   $\boldsymbol{\pi}$ s from  $N$   $\boldsymbol{\alpha}$ s by (5)
15:    Calculate and set  $\mathbf{A}$  from  $n$   $\boldsymbol{\pi}$ s and  $\mathbf{M}$  by (6)
16:    Calculate posterior entropy from  $\mathbf{A}$ 
17:     $\mathcal{I}_r \leftarrow \gamma|\mathcal{I}|$  instances of  $\mathcal{I}$  with largest entropy
18:     $\boldsymbol{\alpha} \leftarrow \boldsymbol{\mu}$ 
19:     $r \leftarrow r + 1$ 
20:  end if
21: end while
22: return  $\mathbf{A}$ 

```

---

## 4 Experiments

### 4.1 Baseline Methods

There are various quality control methods for both answer aggregation and task assignment in the literature [5, 8, 12, 14] but not all of them are suitable for Taxonomy Enrichment due to its broad knowledge coverage and enormous concepts. We focus on comparing with BTSK [9]. Specifically, we consider the following quality control methods.

For task assignment: **UA:** uniform assignment (UA) algorithm assigns all instances in  $\mathcal{I}$  in each round  $r$ . For any instance  $i$ , it can be labeled  $r$  times only

when other instances have been labeled  $r - 1$  times. **KBIS**: Knowledge Based Iterative Scheduling (KBIS) is the task assignment algorithm used in BTSK [9].

For answer aggregation: **MV**: Majority voting (MV) returns the answer chosen by most workers. If some answers have the same votes, it returns one of them at random. **WMV**: Weighted majority voting (WMV) weights votes of workers by their accuracy on gold answers. **MWK+**: An enhanced variant of Majority Voting with External Knowledge (MWK) proposed in [9], balancing specific labels with highly incorrect risk and less specific ones with insufficient knowledge. **ProFull**: A expectation-maximization probabilistic method proposed in [9]. It differs with DS [5] by taking hypernym relationship into account.

## 4.2 Evaluation Metrics

Let  $a(i) = \arg \max_{l \in \text{leaf}(\mathcal{T})} \mathbf{A}_{il}$  be the inferred leaf concept for instance  $i$ . The **accuracy** on  $\mathcal{I}$  is defined as

$$\text{acc}(\mathbf{A}, \mathcal{I}) = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \mathbb{I}[a(i) = \text{gold}(i)] \quad (7)$$

where  $\text{gold}(i)$  is the ground truth leaf concept of  $i$  and  $\mathbb{I}$  is indicator function.

## 4.3 Simulated Experiment

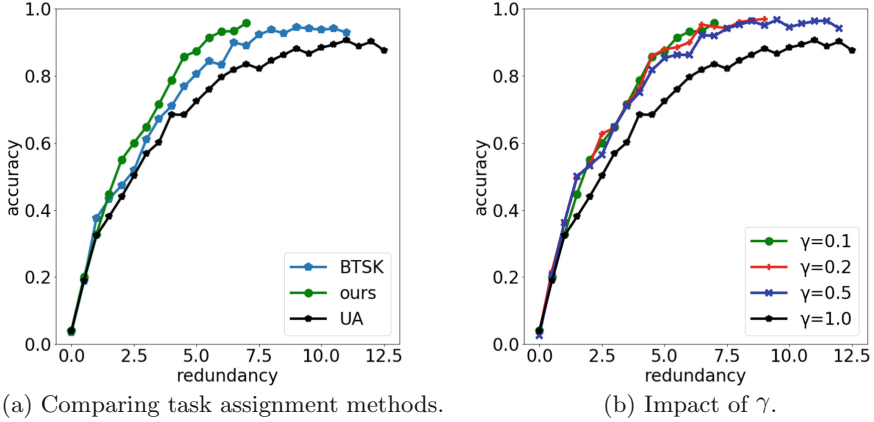
**Experiment Settings.** We generate a synthetic dataset by generating a perfectly balanced binary tree of height 10 to represent the synthetic taxonomy. 1000 synthetic instances are sampled uniformly from its leaves.

We generate 20 synthetic workers with the following procedure. First, we sample workers’ skill scores from the prior distribution of the Gaussian Process model. Then we transform workers’ skill scores to confusion matrices by Eq. 5 and sample answers from them. The hyper-parameters are set empirically such that every synthetic worker’s average accuracy is about 30%, which is similar to real workers’ accuracy on real dataset.

Simulation is conducted as follows: Both our task assignment and baseline algorithm KBIS organize instances by rounds. For either of them, within each round, the algorithm submits 100 instances to be labeled. Synthetic workers are chosen uniform randomly to label these instances. Answers are sampled from the corresponding confusion matrix of workers. Each instance is labeled at most 15 times in the whole simulation process. When all instances left within a round have been labeled by a worker, the worker is skipped and gets no instance to label. All gold instances are labeled once before non-gold ones are labeled.

**Empirical Effectiveness.** We first fix task assignment algorithm and compare our truth inference algorithm with baseline methods. Table 1 shows aggregated answer accuracy changing with label redundancy, which refers to how many times a label is repeatedly labeled. Our method converges faster regardless of task assignment algorithms in the first few rounds. We next fix answer aggregation

and compare our task assignment algorithm with BTSK and UA. Percentage of gold instances in both BTSK and our task assignment are set to 0.1. Ours, BTSK and UA collected 7361, 11195 and 12986 answers before termination. Figure 3a shows that our task assignment algorithm uses less workload and achieves better accuracy compared to BTSK and UA.



**Fig. 3.** Accuracy performance in simulated experiments.

**Table 1.** Accuracy of synthetic experiments. “OTA” and “OTI” are short for “Our Task Assignment” and “Our Truth Inference” respectively.  $r$  is for label redundancy.

$\gamma$	$r$	0.1					0.2				
		MV	WMV	MWK+	ProFull	OTI	MV	WMV	MWK+	ProFull	OTI
KBIS	3	0.309	0.374	0.521	0.442	<u>0.611</u>	0.323	0.387	0.532	0.462	<u>0.587</u>
	5	0.551	0.590	0.738	0.714	<u>0.806</u>	0.557	0.606	0.765	0.745	<u>0.801</u>
	Final	0.936	0.948	<u>0.983</u>	0.959	0.930	0.915	0.936	<u>0.965</u>	0.954	0.934
OTA	3	0.379	0.436	0.614	0.508	<u>0.648</u>	0.386	0.457	0.619	0.509	<u>0.646</u>
	5	0.589	0.641	0.839	0.786	<u>0.874</u>	0.612	0.650	0.834	0.790	<u>0.879</u>
	Final	0.920	0.927	<u>0.963</u>	0.943	0.958	0.947	0.942	<u>0.985</u>	0.969	0.970
$\gamma$	$r$	0.5					1				
		MV	WMV	MWK+	ProFull	OTI	MV	WMV	MWK+	ProFull	OTI
KBIS	3	0.331	0.395	0.580	0.490	<u>0.616</u>	0.330	0.354	<u>0.525</u>	0.418	0.516
	5	0.570	0.624	0.771	0.729	<u>0.822</u>	0.514	0.568	0.681	0.641	<u>0.734</u>
	Final	0.932	0.945	<u>0.974</u>	0.965	0.940	0.911	0.921	<u>0.961</u>	0.947	0.938
OTA	3	0.365	0.436	0.607	0.474	<u>0.647</u>	0.347	0.391	<u>0.551</u>	0.431	0.523
	5	0.608	0.665	0.810	0.762	<u>0.853</u>	0.527	0.565	0.703	0.640	<u>0.742</u>
	Final	0.942	0.953	<u>0.982</u>	0.978	0.941	0.912	0.914	<u>0.962</u>	0.943	0.902

**Impact of Problems Per Round.** Figure 3b shows impact of  $\gamma$  on our algorithm, the percentage of problems delivered in each round. As  $\gamma$  increases, the accuracy converges slower because more problems that are not recommended by task assignment algorithm of small  $\gamma$  is recommended to workers. At the same time, skill estimation is performed less often with large  $\gamma$ . With less  $\gamma$ , however, the task assignment process may terminate before every instance gets max permitted number labels. This can happen when the most ambiguous instances are those with largest entropy, which are labeled by all workers or have reached maximum permitted number of labels. Figure 3b shows that with properly chosen  $\gamma$ , this early termination does not damage accuracy and thus saves requester’s budget. In our experiments,  $\gamma = 0.1$  and  $\gamma = 0.2$  cases converge almost equivalently fast when label redundancy is between 2 and 7.5. When  $\gamma$  increases to 0.5, the convergence speed begins to slow down. Accuracy details are shown in Table 1.

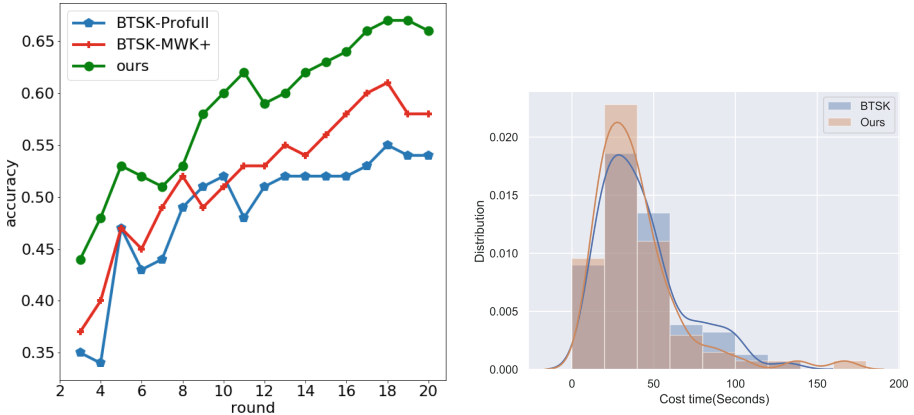
#### 4.4 Real-World Experiment

**Dataset.** We first extract a concept taxonomy tree from Knowledge Graph of K12 Education<sup>4</sup> and Chinese primary science curriculum standards with 606 nodes with *Science* as the root node. Then we get instances from the science category of Baidu Baike as data. In total, we collect 196,492 instances and pick 269 in categories of *GeoScience* and *LifeSciences* for labeling. A human annotator majoring in earth science manually labeled them as gold instances.

**Experiment Settings.** We run 4 experiments on the real-world dataset. The former 2 experiments are used to tune parameters for BTSK and our algorithm. We report the performance of the latter 2 experiments. In each experiment, 314 workers are informed, most of whom are students in the university. In each experiment, there are 100 instances to be labeled. Each instance can be labeled at most 15 times. The average tree depth of our taxonomy is 5. In each round of every experiment, at most<sup>5</sup> 50 instances ( $\gamma = 0.5$ ) are assigned to workers according to the task assignment algorithm. In all experiments, workers are paid proportional to the number of correct edges matched with gold concepts. 36 and 44 workers participated the experiments of BTSK and our algorithm. Worker’s average accuracy is 34.44% and 35.21%. For all real-world experiments, we set hyper-parameters  $\zeta, \phi, \eta, \lambda, \tau, \sigma$  and  $z$  to 3.0, 1.0, 0.25, 0.3, 0.39, 0.01 and 2.

<sup>4</sup> <http://edukg.cn>.

<sup>5</sup> After 15 rounds, there may be less than 50 available instances that have not been answered 15 times. Every instance can be annotated by a worker at most once.



(a) Accuracy of our proposed crowdsourcing framework and BTSK.

(b) Frequency distribution histogram for average annotation time of each worker.

**Fig. 4.** Real world experiments.

**Effectiveness of Quality Control.** Figure 4a shows accuracy performance of our crowdsourcing framework and BTSK [9]. Average of accuracy of our crowdsourcing framework over round 3 to 20 is 58.6%, while BTSK with MWK+ has average accuracy 52.0% and BTSK with ProFull is 48.7%.

**Effectiveness of Subtree Recommendation.** With subtree recommendation, we reduce average annotation time by 7.1% (39.77 s vs 42.79 s) compared with BTSK. In Fig. 4b, workers having average annotation time between 20 and 40 s per instance increased from 37.2% to 45.6% while that between 40 and 60 s decreased from 26.9% to 22.1%. These suggest that our subtree recommendation algorithm can save workers’ time of traversing the tree and clicking nodes.

## 5 Related Works

We review two lines of related works: taxonomy construction and quality control in crowdsourcing.

**Taxonomy Construction and Enrichment.** Prior works on construction of taxonomies mainly utilize term embedding [21] and self-supervised learning techniques to achieve stronger precision [22, 28]. To leverage human intelligence, CASCADE [4] and DELUGE [3] introduced crowdsourcing for taxonomy construction but relatively in small scale. Large-scale categorization like Taxonomy Enrichment is often conducted by asking workers binary option questions such as whether an image belongs to a category [7, 31], or multiple-choice questions such as whether an instance belongs to one class or another [17, 24]. Yan and Huang [34] considered labels with different cost at different levels of a taxonomy. However, Recent research suggests increasing task autonomy can promote worker participation by incenting workers’ intrinsic motivation [16, 38].



Our crowdsourcing framework enables a skilled worker to label an instance to the finest granularity while remaining a user-friendly interface by presenting only a small part of whole taxonomy. Our answer aggregation also goes beyond majority voting in previous works [7, 17, 24] with existing answers and worker skills considered.

**Quality Control in Crowdsourcing.** A main line of research of answer aggregation utilizes confusion matrices with EM algorithm [5, 6, 12] or variations of majority voting [18, 32, 39], leveraging worker similarities [15, 32, 33] and task similarities [29, 37] with either point estimator [6] or Bayesian estimators [14, 15, 19, 32]. Online task assignment algorithms utilize both worker skill estimation [25, 36] and task uncertainty, with entropy [2] and other uncertainty scores [20] to maximize performance under limited budgets. Inspired by observations of Duan et al. [8] that task assignment can benefit from properly constructed taxonomies, we explicitly model confusion matrix with an existing taxonomy.

## 6 Conclusion and Future Work

We conduct a new investigation on crowdsourcing for Taxonomy Enrichment. We formally define Crowdsourcing Taxonomy Enrichment and propose a crowdsourcing framework with skill estimation, task assignment, answer aggregation and subtree recommendation with Accuracy Locality and Bias Locality. We examine the effectiveness of the proposed quality control method in both real-world and synthetic experiments. Promising future work directions include enriching existing taxonomies with our method to produce large taxonomy datasets, as well as designing proper incentive mechanisms to reward expertised workers.

**Acknowledgment.** This work is supported by the Key-Area Research and Development Program of Guangdong Province (2019B010153002), the NSFC Key Project (U1736204), a grant from the Institute for Guo Qiang, Tsinghua University (2019GQB0003) and a grant from Beijing Academy of Artificial Intelligence.

## References

1. Abacha, A.B., Zweigenbaum, P.: MEANS: a medical question-answering system combining NLP techniques and semantic web technologies. *Inf. Process. Manag.* **51**(5), 570–594 (2015)
2. Boim, R., Greenshpan, O., Milo, T., Novgorodov, S., Polyzotis, N., Tan, W.C.: Asking the right questions in crowd data sourcing. In: *ICDE 2012*. IEEE (2012)
3. Bragg, J., Weld, D.S., et al.: Crowdsourcing multi-label classification for taxonomy creation. In: *First AAAI Conference on Human Computation and Crowdsourcing* (2013)
4. Chilton, L.B., Little, G., Edge, D., Weld, D.S., Landay, J.A.: Cascade: crowdsourcing taxonomy creation. In: *CHI 2013*, pp. 1999–2008. ACM (2013)
5. Dawid, A.P., Skene, A.M.: Maximum likelihood estimation of observer error-rates using the EM algorithm. *J. Roy. Stat. Soc.: Ser. C (Appl. Stat.)* **28**(1), 20–28 (1979)

6. Demartini, G., Difallah, D.E., Cudré-Mauroux, P.: ZenCrowd: leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In: WWW 2012 (2012)
7. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: CVPR 2009. IEEE (2009)
8. Duan, X., Tajima, K.: Improving multiclass classification in crowdsourcing by using hierarchical schemes. In: WWW 2019. ACM (2019)
9. Han, T., Sun, H., Song, Y., Wang, Z., Liu, X.: Budgeted task scheduling for crowd-sourced knowledge acquisition. In: CIKM 2017. ACM (2017)
10. Han, X., Yu, P., Liu, Z., Sun, M., Li, P.: Hierarchical relation extraction with coarse-to-fine grained attention. In: EMNLP 2018 Proceedings, pp. 2236–2245 (2018)
11. Hovy, E.: Combining and standardizing large-scale, practical ontologies for machine translation and other uses. In: LREC 1998 (1998)
12. Jing, Z., Wu, X.: Multi-label inference for crowdsourcing. In: KDD 2018. ACM (2018)
13. Jurgens, D., Pilehvar, M.T.: SemEval-2016 task 14: semantic taxonomy enrichment. In: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), pp. 1092–1102. ACL, San Diego, June 2016
14. Karger, D.R., Oh, S., Shah, D.: Budget-optimal task allocation for reliable crowdsourcing systems. *Oper. Res.* **62**(1), 1–24 (2014)
15. Kim, H.C., Ghahramani, Z.: Bayesian classifier combination. In: Artificial Intelligence and Statistics, pp. 619–627 (2012)
16. Lee, C., Chan, C., Ho, S., Choy, K., Ip, W.: Explore the feasibility of adopting crowdsourcing for innovative problem solving. *Ind. Manag. Data Syst.* **115**, 803–832 (2015)
17. Li, Y., Wu, X., Jin, Y., Li, J., Li, G.: Efficient algorithms for crowd-aided categorization. *Proc. VLDB Endow.* **13**(8), 1221–1233 (2020)
18. Littlestone, N., Warmuth, M.K., et al.: The weighted majority algorithm. University of California, Santa Cruz, Computer Research Laboratory (1989)
19. Liu, Q., Peng, J., Ihler, A.T.: Variational inference for crowdsourcing. In: Advances in Neural Information Processing Systems, pp. 692–700 (2012)
20. Liu, X., Lu, M., Ooi, B.C., Shen, Y., Wu, S., Zhang, M.: CDAS: a crowdsourcing data analytics system. *Proc. VLDB Endow.* **5**(10), 1–12 (2012)
21. Luu, A.T., Tay, Y., Hui, S.C., Ng, S.K.: Learning term embeddings for taxonomic relation identification using dynamic weighting neural network. In: EMNLP 2016 (2016)
22. Mao, Y., et al.: Octet: Online catalog taxonomy enrichment with self-supervision. In: KDD 2020 (2020)
23. Mavridis, P., Gross-Amblard, D., Miklós, Z.: Using hierarchical skills for optimized task assignment in knowledge-intensive crowdsourcing. In: WWW 2016 (2016)
24. Meng, R., Tong, Y., Chen, L., Cao, C.C.: CrowdTC: Crowdsourced taxonomy construction. In: ICDM 2015, pp. 913–918, November 2015
25. Rangi, A., Franceschetti, M.: Multi-armed bandit algorithms for crowdsourcing systems with online estimation of workers’ ability. In: AAMAS 2018 (2018)
26. Rebele, T., Suchanek, F.M., Hoffart, J., Biega, J., Kuzey, E., Weikum, G.: YAGO: a multilingual knowledge base from Wikipedia, WordNet, and GeoNames. In: ISWC 2016 (2016)
27. Roy, S.B., Lykourantzou, I., Thirumuruganathan, S., Amer-Yahia, S., Das, G.: Task assignment optimization in knowledge-intensive crowdsourcing. In: VLDB 2015 (2015)

28. Shen, J., Shen, Z., Xiong, C., Wang, C., Wang, K., Han, J.: TaxoExpan: Self-supervised taxonomy expansion with position-enhanced graph neural network. In: WWW 2020 (2020)
29. Tao, F., Jiang, L., Li, C.: Label similarity-based weighted soft majority voting and pairing for crowdsourcing. *Knowl. Inf. Syst.* **62**, 2521–2538 (2020). <https://doi.org/10.1007/s10115-020-01475-y>
30. Van Horn, G., Branson, S., Loarie, S., Belongie, S., Perona, P.: Lean multiclass crowdsourcing. In: CVPR 2018 (2018)
31. Van Horn, G., et al.: The iNaturalist species classification and detection dataset. In: CVPR 2018 (2018)
32. Venanzi, M., Guiver, J., Kazai, G., Kohli, P., Shokouhi, M.: Community-based Bayesian aggregation models for crowdsourcing. In: WWW 2014, pp. 155–164. ACM, New York (2014)
33. Whitehill, J., Wu, T.F., Bergsma, J., Movellan, J.R., Ruvolo, P.L.: Whose vote should count more: optimal integration of labels from labelers of unknown expertise. In: *Advances in Neural Information Processing Systems*, pp. 2035–2043 (2009)
34. Yan, Y., Huang, S.J.: Cost-effective active learning for hierarchical multi-label classification. In: IJCAI, pp. 2962–2968 (2018)
35. Yu, J., et al.: MOOCCube: a large-scale data repository for NLP applications in MOOCs. In: ACL 2020 (2020)
36. Zhang, H., Ma, Y., Sugiyama, M.: Bandit-based task assignment for heterogeneous crowdsourcing. *Neural Comput.* **27**(11), 2447–2475 (2015)
37. Zhang, J., Sheng, V.S., Wu, J., Wu, X.: Multi-class ground truth inference in crowdsourcing with clustering. In: ICDE 2015, vol. 28 (2015)
38. Zhang, X., Chen, M., Ji, G.: Factors influencing the crowd participation in knowledge-intensive crowdsourcing. In: ICCSE 2019. ACM, New York (2019)
39. Zheng, Y., Wang, J., Li, G., Cheng, R., Feng, J.: QASCA: a quality-aware task assignment system for crowdsourcing applications. In: SIGMOD 2015 (2015)



# Transaction Confirmation Time Estimation in the Bitcoin Blockchain

Limeng Zhang<sup>1,2</sup>, Rui Zhou<sup>1(✉)</sup>, Qing Liu<sup>2</sup>, Jiajie Xu<sup>3</sup>, and Chengfei Liu<sup>1</sup>

<sup>1</sup> Swinburne University of Technology, Melbourne, Australia

{limengzhang,rzhou,cliu}@swin.edu.au

<sup>2</sup> Data61, CSIRO, Hobart, Australia

Q.Liu@data61.csiro.au

<sup>3</sup> Soochow University, Suzhou, China

xujj@suda.edu.cn

**Abstract.** As Bitcoin is universally recognized as the most popular cryptocurrency, more and more Bitcoin transactions are expected to be populated to the Bitcoin blockchain system. However, transactions cannot be confirmed altogether into the next block due to the limited block capacity. One of the most demanding requirements for users to use Bitcoin is to estimate the confirmation time of a newly submitted transaction. In this paper, we propose two approaches for estimating the confirmation time for a single transaction. The first approach DcyMean makes the estimation based on the historical confirmation time of transactions included in the blockchain. The second approach CTEN is built on neural networks to estimate based on a variety of factors including the transaction itself, block states and mempool states. Finally, we conduct experiments on real Bitcoin blockchain datasets to demonstrate the effectiveness and efficiency of our proposed approaches. Each of our approaches can finish training and estimation within one block interval, demonstrating that our approaches can process real-time cases.

**Keywords:** Transaction confirmation time estimation · Bitcoin · Blockchain

## 1 Introduction

Nowadays, cryptocurrency has become a buzzword in both industry and academia [26]. Blockchain works as one of the popular systems to manage cryptocurrency transactions. Compared with traditional ledger systems, blockchain is decentralized, immutable and transparent. Bitcoin, one of the most popular cryptocurrencies, has achieved huge success with its capital market reaching 600

---

Supported by Data61, Australian Research Council Discover (Grant No. DP170104747, No. DP180100212 and No. DP200103700) and National Natural Science Foundation of China (Grant No. 61872258).

© Springer Nature Switzerland AG 2021

W. Zhang et al. (Eds.): WISE 2021, LNCS 13080, pp. 30–45, 2021.

[https://doi.org/10.1007/978-3-030-90888-1\\_3](https://doi.org/10.1007/978-3-030-90888-1_3)

billion dollars in 2021 as shown in [coindesk](https://www.coindesk.com/price/bitcoin)<sup>1</sup>. Meanwhile, PayPal accountholders in the United States have been able to buy, hold, and sell cryptocurrencies directly through PayPal. As a result, more Bitcoin transactions are expected to be populated into the Bitcoin blockchain system. The bulk of new transactions, however, cannot be included altogether in the next block due to the restricted capacity of a block. After a transaction is submitted, the transaction is then verified and put into a miner’s mempool where the transaction competes with other unconfirmed transactions to be included/confirmed into future blocks. As a result, a submitted transaction may suffer from confirmation delay as the number of submitted transactions grows. Concerned about this, it becomes vital to help a user to understand (if possible) how long it may take for a transaction to be confirmed in the Bitcoin blockchain.

Existing research has shed some light on transaction confirmation time estimation in the Bitcoin blockchain [3, 12, 14, 19–22, 45]. However, they are not practical enough due to the following four types of drawbacks: (1) Estimation is not tailored for an individual transaction. The existing works [19, 20, 45] evaluate the average confirmation time of two classes of transactions [19, 20] or of all the unconfirmed transactions [45], instead of the confirmation time of a single transaction. (2) Some works [12, 21] only predict whether a transaction can be confirmed in the next block, i.e. treat the problem as a binary classification problem. They could not provide accurate confirmation time, and thus may be insufficient in practice. (3) Some assumptions are not realistic. The works [3, 14, 22] assume that mempool receives transactions falling into each feerate class at a specific fixed speed. However, the speed of inflow transactions in a feerate class often varies, such as sensitive to peak/off-peak trades, government policies, finance news, etc. In addition, they assume that every block contains a fixed number of transactions. This is often not the case. (4) Transaction features and block features are not sufficiently utilized. To the best of our knowledge, only the works [12, 21] consider transaction metadata, such as transaction weight, number of inputs, number of outputs, etc. No prior work has considered block features which may give some hints on future block sizes and future block generation speeds that are likely to affect when a new transaction can be confirmed.

To address these limitations, we propose two transaction confirmation time estimation approaches, DcyMean and CTEN. DcyMean is designed based on decayed mean of historical transaction confirmation time in the blockchain. It estimates by aggregating the confirmation time of historical transactions in different blocks with feerates falling into the same feerate interval as the given transaction. A decay parameter is used to lower down the importance of earlier blocks. We also propose an advanced approach CTEN (short for Confirmation Time Estimation Network) to make the estimation based on neural networks. It works by predicting the characteristics of future blocks and simulating the mining behavior of miners. Specifically, it considers three types of information: the transaction itself, block states in the blockchain and mempool competition states. Historical block sequence is used to predict the properties of future blocks.

---

<sup>1</sup> <https://www.coindesk.com/price/bitcoin>.

CTEN also derives the mining behavior from the confirmation of transactions in the mempool at different block heights. Both block state sequence and mempool state sequence are learnt using Long Short-Term Memory (LSTM) or Attention mechanisms, which have demonstrated outstanding performance in time series data processing [13, 24, 33, 42].

Our contributions in this paper are summarized as follows:

- We propose a feerate-based confirmation time estimation approach DcyMean, which infers the confirmation time for a transaction by aggregating the confirmation time of historical transactions in different blocks with feerates falling into the same feerate interval as the given transaction.
- We propose an advanced approach CTEN, which works on leveraging the power of neural networks for time series data processing techniques to learn future block characteristics and simulate miners’ mining policy to help with transaction confirmation time estimation.
- Extensive experiments on real Bitcoin blockchain datasets have been conducted to demonstrate the effectiveness and efficiency of our proposed approaches.

## 2 Background and Related Work

In this section, we first introduce some recent attempts on managing blockchain systems, and then introduce the transaction confirmation process in Bitcoin and related works on transaction confirmation time estimation in the Bitcoin Blockchain.

### 2.1 Blockchain Management

Recently, extensive research has been made on the technologies underpinning the deployment of blockchain systems in different aspects, such as scalability, query evaluation, security and privacy, applications and etc. Scalability focuses on increasing transaction confirmation throughout in blockchain systems through, e.g., consensus protocols [4, 15], concurrency control [7, 10, 31] and data management techniques [5, 8, 9, 27, 28, 30, 32, 34, 39]. Query evaluation tackles the problem of querying data stored on blockchains [36, 38, 43, 46, 47]. Another focus of the community is system maintainability and data privacy [40, 41]. There are also works incorporating the blockchain technology into different fields, such as crowd-sourcing [16, 17, 23], disease control [29] and classification [44].

### 2.2 Transaction Confirmation in the Bitcoin Blockchain

In the Bitcoin blockchain system, transactions record the transferring of Bitcoin currencies between participants. Transaction are confirmed when they are included in a block of the blockchain. As a block’s capacity is restricted, unconfirmed transactions implicitly compete to get confirmed. Transaction fee can be

set to prioritize transaction processing. *Transaction feerate* is a term used to describe the fee density of a transaction, which is calculated by dividing the transaction fee by the transaction size. Once transactions are submitted to the blockchain system, they are broadcast across the nodes in the system. If they meet the transactions' validity criteria [1], Bitcoin nodes will add them to their mempools (memory pools), where transactions wait until they can be included (mined) into a block. Miners are the nodes who need to conduct this verification process with their computing power. They play a critical part in the transaction confirmation process, ensuring that transactions are included on the blockchain. In the confirmation, they first select some unconfirmed transactions in the mempool and construct a candidate block. Then they compete to figure out the computational solution (referred to proof-of-work) for the next block. Finally, the miner who first works out this computational problem is awarded, which consists of a fixed Bitcoin reward and transaction fees included in the block. The competition process among miners is called *mining*. Once a new block is linked to the blockchain, miners update their local copy of blockchain data and start mining the next block. The confirmation of transactions in the new block is complete and these transactions will be removed from the miner's mempool.

### 2.3 Related Works on Transaction Confirmation Time Estimation

Some works have been done on transaction confirmation time estimation in the Bitcoin blockchain. In [12, 21], the authors treated the confirmation time estimation problem as a binary classification problem to predict whether a transaction can be confirmed in the next block. They use supervised learning models like SVM, random forest, AdaBoost, and etc. The authors in works [3, 19, 20] introduced different queueing models to estimate the confirmation time of transactions with different feerates. Feerates are categorized into several classes in these works, and the works conduct the estimation for each transaction class. To be specific, it assumes that transactions in different feerate classes arrive in the queue according to an independent Poisson process with a specific fixed rate. To fill in each block, a fixed number of transactions (batch) are removed from the queue. Transactions with greater fees are placed in the front part of the queue, allowing them to be processed earlier than those with lower fees. The number of blocks required for the confirmation of a transaction can be thus calculated by its position in the queue. Works in [19, 20] adopt the  $M/G^B/1$  queueing model with batch  $B$  [6]. Balsamo et al. [3] considered a queueing model  $M/M^B/1$  with batch  $B$  [25], and the works [14, 22] model the confirmation process as Cramér-Lundberg process. In addition, Zhao et al. [45] established a type of non-exhaustive queueing model with a limited batch service and a possible zero-transaction service to study the average confirmation time. It assumes that the number of transactions in the system at a given observation point form a Markov chain, the elapsed time of block generation follows a certain density function in stochastic processes, and transaction arrivals follow a Poisson distribution. Finally, based on Little's law [37], the confirmation time is estimated.

**Table 1.** Notations

Notation	Explanation
$tx$	A transaction
$\hat{tx}$	The submitted transaction to be estimated
$r$	The feerate interval that the feerate of $\hat{tx}$ falls in
$w(tx)$	The weight of the transaction $tx$
$g(tx)$	The confirmation time of the transaction $tx$ , i.e. the gap between The submission and confirmation timestamps of $tx$
$\hat{g}$	The estimated confirmation time for transaction $\hat{tx}$
$dcy$	A decay parameter reflecting the fading effect in DcyMean
$h$	A block height
$h_{cur}$	The height of the current (most recently generated) block
$T_h$	The confirmed transactions in the block with height $h$
$T_h^r$	The confirmed transactions with feerate interval $r$ in $T_h$
$Mem_h$	The unconfirmed transactions in the mempool at block height $h$
$b_h$	The block state of the block with height $h$ in CTEN
$mem_h$	The mempool state of the block with height $h$ in CTEN

### 3 Problem Definition

Given a newly submitted transaction  $\hat{tx}$ , the studied problem is to predict the confirmation time  $\hat{g}(\hat{tx})$  for  $\hat{tx}$ . We consider that three types of information are useful to help with the prediction/estimation: (1) the transaction itself, whose features are denoted as *FeaInfo*; (2) historical confirmed transactions stored in blocks, denoted as *ChainInfo*; (3) unconfirmed transactions constituting the mempool, denoted as *MemInfo*. We model estimating transaction confirmation time as looking for a function  $\mathcal{F}$  taking the above three types of information as input and giving the estimation as output.

$$\hat{g} = \mathcal{F}(FeaInfo, ChainInfo, MemInfo)$$

The rationale to select the above three types of features is as follows: (1) *FeaInfo* contains the unique features of a particular transaction, e.g., feerate, weight, validation complexity, submitted time, etc., which could differentiate one transaction from other transactions; (2) *ChainInfo* has historical transaction confirmations, which may give some hints on the new transaction estimation. Meanwhile, the block sequence may contain block size and block generation trends, implicitly reflecting the volume and speed of confirming multiple transactions as a whole; (3) *MemInfo* has all the unconfirmed transaction, i.e. the competitiveness of the submitted transaction can be evaluated against all the waiting transactions. As a result, we propose to learn an estimation function from these three types of information. We list some frequently used notations in Table 1.



## 4 Methods for Confirmation Time Estimation

In this section, we present the confirmation time estimation approaches.

### 4.1 Decayed Mean (DcyMean)

The first approach we propose is based on decayed mean of historical transactions (named DcyMean), which estimates the confirmation time for a new transaction, given a small feerate interval that the new transaction falls in. It infers the confirmation time from historical transactions that fall into the same feerate interval. More importantly, DcyMean considers that confirmation time of transactions confirmed in recent blocks are more helpful than those confirmed in earlier blocks when computes the estimation. To aggregate the effect of records from different blocks, it introduces a decay parameter  $dcy \in [0, 1]$ . A smaller  $dcy$  means DcyMean relies more on the results from recent blocks than earlier blocks. When a new block is mined, the impact of confirmation time of transactions in prior blocks will fade by a factor of  $dcy$ . When  $dcy = 0$ , it means only the current block (most recently generated block) is used to compute the estimation. When  $dcy = 1$ , it means all the historical blocks have equivalent importance. The estimated confirmation time is given in Equation (1):

$$\hat{g} = \frac{\sum_{h \leq h_{cur}} \sum_{tx \in T_h^r} g(tx) * dcy^{h_{cur}-h}}{\sum_{h \leq h_{cur}} \sum_{tx \in T_h^r} dcy^{h_{cur}-h}} \quad (1)$$

where  $h$  denotes a historical block height,  $h_{cur}$  denotes the current block height,  $r$  denotes the feerate interval  $\hat{t}_x$  falls in and  $T_h^r$  denotes the transactions confirmed in block  $h$  within the feerate interval  $r$ .

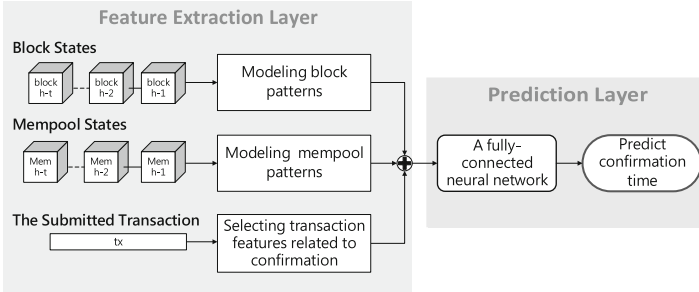
By collecting historical confirmation time related to a given small feerate interval in the blockchain, DcyMean obtains the aggregated confirmation time of that feerate interval to predict for a new transaction. In fact, a transaction's confirmation is involved in a more complicated block mining process. which may be affected by not only the transaction itself, but also the hidden mining policies, blockchain states, and possibly what other unconfirmed transactions are. In Sect. 4.2, we propose a more powerful neural network based approach to solve the transaction confirmation time estimation problem.

### 4.2 Confirmation Time Estimation Network (CTEN)

The second approach we propose is based on neural networks to estimate the confirmation time for a new transaction (named CTEN). Before introducing CTEN, we summarize three groups of features that may influence the confirmation time:

- **Transaction Features**, which describes the submitted transaction.
- **Block States**, which reflects the characteristics of the mined blocks including block size, block generation speed, etc.

- **Mempool States**, which records the distribution of fees of unconfirmed transactions in the mempool, implicitly modelling the competition among unconfirmed transactions.



**Fig. 1.** The framework of CTEN.

These three groups of features correspond to the three types of information fed to the estimation function  $\mathcal{F}$  in Sect. 3. Although transaction features are already available in the submitted transaction, future block states and mempool states are not known. However, such features are desirable, because if we had known how many transactions would be contained in future blocks, how fast future blocks would be generated, how competitive the submitted transaction would be in future mempools, we would increase the chance to predict the confirmation time more accurately. Consequently, in CTEN, our main idea is to predict future block states and mempool states from historical state sequences by utilizing sequence learning models. Finally, we combine the three groups of features to do the estimation.

**Estimation Framework.** The estimation framework can be divided into two layers, one *feature extraction layer* to extract patterns from block states, mempool states and transaction features of the submitted transaction, and one *prediction layer* to analyze the relationship between transaction confirmation time and the extracted features. Figure 1 shows the framework.

- **Feature Extraction Layer.** It includes three parts. Other than modelling the submitted transaction itself, the feature extraction layer also predicts the future characteristics of block states and model the miners' mining behavior from mempool competition states of the unconfirmed transactions.
  1. *Transaction Features* describe the details of the submitted transaction waiting for confirmation. We select the features that we believe may affect a transaction's validation and confirmation. In addition, we add an additional feature to represent its current competition position in the mempool for modelling the mining priority.

- *transaction feerate* Generally, transactions with higher feerates are likely to be more competitive than those with lower feerates.
  - *number of inputs* and *number of outputs* Miners need to verify the validity of the satoshis claimed in transaction inputs by tracking historical transactions which transferred satoshis to those addresses.
  - *transaction weight* It is a parameter measuring the transaction size. It is related to the blockchain protocol Segwit, and a transaction containing Segwit inputs has additional script data. In order to represent the size of transactions under different protocols, transaction virtual size (weight) is used in our work.
  - *transaction first-seen time* The first-seen time refers to the time that a transaction is first observed. It is difficult to find out the exact submission time of a historical transaction, so its publicly recorded first-seen time is treated as its submission time in our work.
  - *time interval to last block* It is the interval between *transaction first-seen time* and the timestamp of the last generated block. This feature is added to address the disparity in transaction confirmation time in the scenario when two transactions are submitted at the same block height but at different timestamps, since the later one may experience a shorter confirmation delay due to its proximity to the timestamp of the next block's confirmation time.
  - *position in the current mempool* It aggregates the weight of unconfirmed transactions in the mempool with higher feerates than the submitted transaction. Strictly speaking, it is a feature built from both the transaction itself and the mempool.
2. *Block States* are the predicted features of future blocks, which are expected to encode future block size and generation speed, which can affect a transaction's confirmation time. Historical block states are learned as a sequence to predict future block states.
- *block weight* and *the number of confirmed transactions* They represent the size of a block. We evaluate from two aspects, the overall weight of transactions in a block and the number of transactions in this block.
  - *difficulty* It reflects the difficulty of the mathematical problem in mining a block. In the Bitcoin system, *difficulty* is tuned to control that blocks can be generated on an average 10-minute basis.
  - *block interval* It is constructed based on the time gap between a block and its precedent block. It assesses the pace with which blocks are generated.
  - *average feerate in the block* This is the average feerate of the transactions in one block. We aim to capture the feerate trend in continuous blocks by implementing this feature.
  - *feerate distribution in the block* The feerate distribution is extracted by aggregating the weight of the transactions confirmed in one block within each feerate interval.
3. *Mempool States* Transactions in the mempool compete to be included in the next block. We model transaction competition in the mempool at a

certain block height  $h$  by computing the overall weight of transactions in the specific feerate interval  $r$ .

$$mem_h^r = \sum_{tx \in Mem_h^r} w(tx) \quad (2)$$

In this way, before each block is generated, we have a mempool, whose feature is modelled as a concatenated weight-histograms of different feerate intervals. Similar to block states, historical mempool states are learned to predict future mempool states.

In CTEN, block states and mempool states are learnt by sequence models. We implement two alternatives, LSTM and Attention. In both models, we use  $x_i$  to represent the input feature. To be specific, for block states,  $x_i$  is block characteristics  $b_h$ ; for mempool states,  $x_i$  is mempool vector  $mem_h$ .  $t$  is the length of the sequence.

**Approach 1: LSTM** [18] aggregates information on a token-by-token basis in sequential order. Compared to the standard RNN, LSTM addresses the vanishing gradient problem by incorporating three gating functions into state dynamics. At each time step, LSTM maintains a hidden vector  $h$  and a memory vector  $c$  responsible for controlling state updates and outputs.

$$\begin{aligned} i_t &= \sigma(W^i x_t + M^i n_{t-1}) \\ f_t &= \sigma(W^f x_t + M^f n_{t-1}) \\ o_t &= \sigma(W^o x_t + M^o n_{t-1}) \\ \tilde{c}_t &= \tanh(W^c x_t + M^c n_{t-1}) \\ c_t &= i_t \odot \tilde{c}_t + f_t \odot c_{t-1} \\ n_t &= o_t \odot \tanh(c_t) \end{aligned} \quad (3)$$

where  $[W^i, W^f, W^o, W^c, M^i, M^f, M^o, M^c]$  are weight matrices,  $x_t$  is the vector input at the time step  $t$ ,  $n_t$  is the current exposed hidden state,  $c_t$  is the memory cell state, and  $\odot$  is element-wise multiplication.

**Approach 2: Attention** is another popular strategy for dealing with sequential data. It attempts to capture the relationships between different positions of a single sequence in order to generate a representation for the sequence. Unlike LSTM which returns the final hidden state as the extracted feature, the attention mechanism returns a new representation based on the importance in different positions in a sequence. In CTEN, the additive attention architecture [2] is used to replace the LSTM module as an alternate feature extraction module:

$$\begin{aligned} n_{t,t'} &= \tanh(W^t x_t + W^x x_{t'}) \\ e_{t,t'} &= \sigma(W^a n_{t,t'}) \\ a_t &= \text{softmax}(e_t) \\ l_t &= \sum_{t'} a_{t,t'} x_{t'} \end{aligned} \quad (4)$$

where  $[W^t, W^x, W^a]$  are weight matrices,  $x_t$  is the vector input (block states  $b_h$  or mempool states  $mem_h$  in this work) at time step  $t$ ,  $n_t$  is the current exposed hidden state.

We also study the performance of self attention [35] and weighted attention [11]. Results are reported in the experiment section.

- **Prediction Layer** is designed to generate estimation based on the concatenated features from transaction features, block states and mempool states, which is fed to a fully-connected neural network for confirmation time estimation.

## 5 Experiments

We conduct experiments on real-world blockchain data to evaluate the performance of our proposed approaches DcyMean and CTEN. Experiment settings are detailed in Sect. 5.1. The performance of the approaches is reported in Sect. 5.2. In addition, we also demonstrate the efficiency of our approaches.

### 5.1 Experiment Settings

**Datasets and Data Processing.** We constructed two datasets by randomly selecting two different block intervals (S1: block range 621057–621281 and S2: block range 622057–622281) via Blockchain Explorer<sup>2</sup>. Each dataset has 225 blocks, the first 180 blocks are used for training (about 400,000 transaction instances) and the last 45 blocks for testing. At each block height, transactions whose first-seen time fall in between the confirmation time of the current block and the next block are considered as training instances (for the first 180 blocks) or testing instances (for the last 45 blocks).

Considering that transaction feerate in the Bitcoin blockchain system is a continuous value, we discretize it into different small intervals. In DcyMean and the mempool states in CTEN, transaction feerates are divided into 1001 intervals, with 1 being the first and increased by 1 in every step. Feerates more than 1000 are put to in the last interval 1001. Our interval division is based on a statistical study of all the transactions confirmed in the block range 621001–621500 (about 1.18 million), which shows that 99.98% transactions are with feerates under 1000.

**Evaluation Metrics.** In order to evaluate the accuracy on the confirmation time, we employ two measures, Mean Squared Error (MSE) and Mean Absolute Error (MAE), to report the performance. MSE and MAE are defined as follows, where the estimated result is  $\hat{y}_i$  and the ground truth is  $y_i$ .

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (5)$$

<sup>2</sup> <https://www.blockchain.com/explorer>.

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (6)$$

**Compared Methods.** We compare the performance of our proposed DcyMean and CTEN variants.

- **DcyMean:** *dcy* is set to 0.96.
- **CTEN\_Lstm** employs LSTM to extract patterns in block state sequence and mempool states sequence.
- **CTEN\_Wht**, **CTEN\_Self** and **CTEN\_Adv** correspond to use different attention techniques in the CTEN approach for extracting features from block and mempool sequences: weighted attention [11], self attention [35] and additive attention [2].
- **Adv\_Tx**, **Adv\_MemTx** and **Adv\_BloTx** They correspond to different feature combinations in CTEN\_Adv for ablation study.
  - **Adv\_Tx:** Transaction feature only
  - **Adv\_MemTx:** Mempool states and transaction features
  - **Adv\_BloTx:** Block states and transaction features

**Implementation Details.** The hidden units in sequence processing is set to 64 and the prediction layer is made up of a fully-connected three-layer neural network with hidden units 64, 8 and 1, respectively. The length of sequence is set to 3. When training models, parameters are optimized by stochastic gradient descent (SGD) with the Adam optimizer, and the objective function is set to the standard mean squared error with batch size set to 1000. All the methods are implemented in the TensorFlow framework, and all the experiments are run on one NVIDIA P100 12 GB PCIe GPU.

## 5.2 Result Analysis

In this part, we evaluate the performance of confirmation time estimation methods, DcyMean and CTEN.

**Evaluation of DcyMean and CTEN.** The estimation performance is presented in Table 2. According to Table 2, Adv\_Tx outperforms DcyMean on both datasets, which reveals the potential of incorporating transaction features and employing neural networks. Meanwhile, all the CTEN models are super to DcyMean, which reinforces this conclusion. Among all the CTEN models, additive attention CTEN\_Adv performs the best.

**Table 2.** Evaluation of methods under MSE and MAE

Model		MSE		MAE (e7)	
		S1	S2	S1	S2
CTEN	CTEN_Lstm	2532.4	2513.7	2.4	3.7
	CTEN_Wht	2611.7	2445.6	2.6	3.2
	CTEN_Self	2637.7	2491.8	3.0	3.8
	CTEN_Adv	<b>2070.5</b>	<b>2179.1</b>	<b>1.7</b>	<b>3.1</b>
Adv_Tx		2481.3	4331.4	3.2	7.4
DcyMean		3197.4	8023.1	4.8	18.8

**Impact of Different Features in CTEN\_Adv.** We test four different feature compositions (Adv\_Tx, Adv\_MemTx, Adv\_BloTx and CTEN\_Adv) to study the importance of the features. The performance is shown in Table 3. CTEN\_Adv which combines three different features, stands out among these four variants. Meanwhile, after comparing Adv\_Tx and Adv\_MemTx, we can see that adding mempool states to transaction features can improve accuracy significantly. Although the accuracy of Adv\_BloTx on dataset S1 is lower than that of Adv\_Tx, the accuracy is further improved when mempool states are added, which turns into CTEN\_Adv. In conclusion, transaction confirmation time estimation can benefit from both block states and mempool states. Moreover, mempool states contribute more significantly, which sheds light on the significance of mempool competition state research.

**Table 3.** Impact of different features in CTEN

Model	MSE		MAE (e7)	
	S1	S2	S1	S2
CTEN_Adv	<b>2070.5</b>	<b>2179.1</b>	<b>1.7</b>	<b>3.1</b>
Adv_MemTx	2168.5	2330.1	2.1	3.2
Adv_BloTx	3819.2	2904.3	5.7	5.2
Adv_Tx	2481.3	4331.4	3.2	7.4

**Time Efficiency of CTEN.** We conduct another set of experiments to study the training time of the models under 100 training epochs, with 8G Memory and 1000 instances as a batch. The training time of *DcyMean* can be ignored since the results can be computed directly by updating the estimation result at the last time step with the newly mined block. From Table 4, we can find that all the attention variants in CTEN can finish training within 10 min (roughly one block interval), which means that our approaches can serve for real-time Bitcoin

blockchain data. Moreover, compared to CTEN\_Lstm, the training time of the attention variants can reduce by almost 50%.

**Table 4.** Training time of estimation models (seconds)

	Adv_Tx	CTEN_Lstm	CTEN_Wht	CTEN_Self	CTEN_Adv
S1	137	653	357	317	383
S2	145	648	370	323	401

## 6 Conclusion

In this work, we propose two approaches on estimating the confirmation time of a submitted transaction in the Bitcoin blockchain system. The first approach DcyMean computes the estimation based on decayed mean of the confirmation time of historical transactions. The second approach CTEN works on learning the relationship between transaction confirmation time and a variety of factors including block states, mempool states, and the transaction itself. Our experimental results show that CTEN outperforms DcyMean in tackling this problem on the real datasets. Meanwhile, we demonstrate that the competition in mempool is not neglectable on estimating transaction confirmation time.

## References

1. Antonopoulos, A.M.: *Mastering Bitcoin: Programming the Open Blockchain*. O'Reilly Media, Inc. (2017)
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint [arXiv:1409.0473](https://arxiv.org/abs/1409.0473) (2014)
3. Balsamo, S., Marin, A., Mitrani, I., Rebagliati, N.: Prediction of the consolidation delay in blockchain-based applications. In: *Proceedings of the ACM/SPEC International Conference on Performance Engineering*, pp. 81–92 (2021)
4. Buchnik, Y., Friedman, R.: FireLedger: a high throughput blockchain consensus protocol. *Proc. VLDB Endow.* **13**(9), 1525–1539 (2020)
5. Bui, H.T., Hussain, O.K., Saberi, M., Hussain, F.: Assessing the authenticity of subjective information in the blockchain: a survey and open issues. *World Wide Web* **24**(2), 483–509 (2021)
6. Chaudhry, M., Templeton, J.: The queuing system M/GB/1 and its ramifications. *Eur. J. Oper. Res.* **6**, 57–61 (1981)
7. Chen, Z., et al.: SChain: a scalable consortium blockchain exploiting intra-and inter-block concurrency. *Proc. VLDB Endow.* **14**(12), 2799–2802 (2021)
8. Dang, H., Dinh, T.T.A., Loghin, D., Chang, E.C., Lin, Q., Ooi, B.C.: Towards scaling blockchain systems via sharding. In: *Proceedings of the 2019 International Conference on Management of Data*, pp. 123–140 (2019)



9. El-Hindi, M., Heyden, M., Binnig, C., Ramamurthy, R., Arasu, A., Kossmann, D.: BlockchainDB-towards a shared database on blockchains. In: Proceedings of the 2019 International Conference on Management of Data, pp. 1905–1908 (2019)
10. Fang, M., Zhang, Z., Jin, C., Zhou, A.: High-performance smart contracts concurrent execution for permissioned blockchain using SGX. In: 2021 IEEE 37th International Conference on Data Engineering (ICDE), pp. 1907–1912. IEEE (2021)
11. Felbo, B., Mislove, A., Søgaard, A., Rahwan, I., Lehmann, S.: Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and Sarcasm. arXiv preprint [arXiv:1708.00524](https://arxiv.org/abs/1708.00524) (2017)
12. Fiz, B., Hommes, S., State, R.: Confirmation delay prediction of transactions in the bitcoin network. In: Park, J.J., Loia, V., Yi, G., Sung, Y. (eds.) CUTE/CSA -2017. LNEE, vol. 474, pp. 534–539. Springer, Singapore (2018). [https://doi.org/10.1007/978-981-10-7605-3\\_88](https://doi.org/10.1007/978-981-10-7605-3_88)
13. Fu, R., Zhang, Z., Li, L.: Using LSTM and GRU neural network methods for traffic flow prediction. In: 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC), pp. 324–328. IEEE (2016)
14. Gundlach, R., Gijsbers, M., Koops, D., Resing, J.: Predicting confirmation times of bitcoin transactions. ACM SIGMETRICS Perform. Eval. Rev. **48**(4), 16–19 (2021)
15. Gupta, S., Rahnema, S., Hellings, J., Sadoghi, M.: ResilientDB: global scale resilient blockchain fabric. Proc. VLDB Endow. **13**(6), 868–883 (2020)
16. Han, S., Xu, Z., Zeng, Y., Chen, L.: Fluid: a blockchain based framework for crowdsourcing. In: Proceedings of the 2019 International Conference on Management of Data, pp. 1921–1924 (2019)
17. Hao, K., Xin, J., Wang, Z., Wang, G.: Outsourced data integrity verification based on blockchain in untrusted environment. World Wide Web **23**(4), 2215–2238 (2020)
18. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
19. Kasahara, S., Kawahara, J.: Effect of bitcoin fee on transaction-confirmation process. J. Ind. Manag. Optimiz. **15**(1), 365 (2019)
20. Kawase, Y., Kasahara, S.: Priority queueing analysis of transaction-confirmation time for bitcoin. J. Ind. Manag. Optimiz. **16**(3), 1077 (2020)
21. Ko, K., Jeong, T., Maharjan, S., Lee, C., Hong, J.W.-K.: Prediction of bitcoin transactions included in the next block. In: Zheng, Z., Dai, H.-N., Tang, M., Chen, X. (eds.) BlockSys 2019. CCIS, vol. 1156, pp. 591–597. Springer, Singapore (2020). [https://doi.org/10.1007/978-981-15-2777-7\\_48](https://doi.org/10.1007/978-981-15-2777-7_48)
22. Koops, D.: Predicting the confirmation time of bitcoin transactions. arXiv preprint [arXiv:1809.10596](https://arxiv.org/abs/1809.10596) (2018)
23. Ma, Y., Sun, Y., Lei, Y., Qin, N., Lu, J.: A survey of blockchain technology on security, privacy, and trust in crowdsourcing services. World Wide Web **23**(1), 393–419 (2020)
24. McNally, S., Roche, J., Caton, S.: Predicting the price of bitcoin using machine learning. In: 2018 26th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), pp. 339–343. IEEE (2018)
25. Miller, D.R.: Computation of steady-state probabilities for M/M/1 priority queues. Oper. Res. **29**(5), 945–958 (1981)
26. Mukhopadhyay, U., Skjellum, A., Hambolu, O., Oakley, J., Yu, L., Brooks, R.: A brief survey of cryptocurrency systems. In: 2016 14th annual conference on privacy, security and trust (PST), pp. 745–752. IEEE (2016)
27. Nathan, S., Govindarajan, C., Saraf, A., Sethi, M., Jayachandran, P.: Blockchain meets database: design and implementation of a blockchain relational database. arXiv preprint [arXiv:1903.01919](https://arxiv.org/abs/1903.01919) (2019)

28. Peng, Y., Du, M., Li, F., Cheng, R., Song, D.: FalconDB: blockchain-based collaborative database. In: Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, pp. 637–652 (2020)
29. Peng, Z., Xu, C., Wang, H., Huang, J., Xu, J., Chu, X.: P2B-trace: privacy-preserving blockchain-based contact tracing to combat pandemics. In: Proceedings of the 2021 International Conference on Management of Data, pp. 2389–2393 (2021)
30. Qi, X., Zhang, Z., Jin, C., Zhou, A.: BFT-store: storage partition for permissioned blockchain via erasure coding. In: 2020 IEEE 36th International Conference on Data Engineering (ICDE), pp. 1926–1929. IEEE (2020)
31. Ruan, P., Lohin, D., Ta, Q.T., Zhang, M., Chen, G., Ooi, B.C.: A transactional perspective on execute-order-validate blockchains. In: Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, pp. 543–557 (2020)
32. Sharma, A., Schuhknecht, F.M., Agrawal, D., Dittrich, J.: Blurring the lines between blockchains and database systems: the case of hyperledger fabric. In: Proceedings of the 2019 International Conference on Management of Data, pp. 105–122 (2019)
33. Srivastava, N., Mansimov, E., Salakhudinov, R.: Unsupervised learning of video representations using LSTMs. In: International Conference on Machine Learning, pp. 843–852 (2015)
34. Tao, Y., Li, B., Jiang, J., Ng, H.C., Wang, C., Li, B.: On sharding open blockchains with smart contracts. In: 2020 IEEE 36th International Conference on Data Engineering (ICDE), pp. 1357–1368. IEEE (2020)
35. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
36. Wang, H., Xu, C., Zhang, C., Xu, J.: vChain: a blockchain system ensuring query integrity. In: Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, pp. 2693–2696 (2020)
37. Wolff, R.W., Yao, Y.C.: Little’s law when the average waiting time is infinite. *Queueing Syst.* **76**(3), 267–281 (2014)
38. Xu, C., Zhang, C., Xu, J.: vChain: enabling verifiable Boolean range queries over blockchain databases. In: Proceedings of the 2019 International Conference on Management of Data, pp. 141–158 (2019)
39. Xu, C., Zhang, C., Xu, J., Pei, J.: SlimChain: scaling blockchain transactions through off-chain storage and parallel processing. *Proc. VLDB Endow.* **14**(11), 2314–2326 (2021)
40. Xu, Z., Chen, L.: DIV: resolving the dynamic issues of zero-knowledge set membership proof in the blockchain. In: Proceedings of the 2021 International Conference on Management of Data, pp. 2036–2048 (2021)
41. Yan, Y., et al.: Confidentiality support over financial grade consortium blockchain. In: Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, pp. 2227–2240 (2020)
42. Yin, J., Tang, M., Cao, J., Wang, H.: Apply transfer learning to cybersecurity: predicting exploitability of vulnerabilities by description. *Knowl.-Based Syst.* **210**, 106529 (2020)
43. Zhang, C., Xu, C., Xu, J., Tang, Y., Choi, B.: GEM $\wedge$  2-tree: a gas-efficient structure for authenticated range queries in blockchain. In: 2019 IEEE 35th International Conference on Data Engineering (ICDE), pp. 842–853. IEEE (2019)
44. Zhang, Z., et al.: Refiner: a reliable incentive-driven federated learning system powered by blockchain. *Proc. VLDB Endow.* **14**(12), 2659–2662 (2021)

45. Zhao, W., Jin, S., Yue, W.: Analysis of the average confirmation time of transactions in a blockchain system. In: Phung-Duc, T., Kasahara, S., Wittevrongel, S. (eds.) QTNA 2019. LNCS, vol. 11688, pp. 379–388. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-27181-7\\_23](https://doi.org/10.1007/978-3-030-27181-7_23)
46. Zhu, Y., Zhang, Z., Jin, C., Zhou, A., Qin, G., Yang, Y.: Towards rich Qery blockchain database. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, CIKM 2020, pp. 3497–3500 (2020)
47. Zhu, Y., Zhang, Z., Jin, C., Zhou, A., Yan, Y.: SEBDB: semantics empowered blockchain database. In: 2019 IEEE 35th International Conference on Data Engineering (ICDE), pp. 1820–1831. IEEE (2019)



# Automatic Malicious Worker Detection in Crowdsourced Paraphrases

Mohammad-Ali Yaghoub-Zadeh-Fard<sup>(✉)</sup> and Boualem Benatallah

University of New South Wales, Sydney, Australia  
{m.yaghoubzadehfard,b.benatallah}@cse.unsw.edu.au

**Abstract.** Automatic paraphrasing has numerous applications such as query reformulation in query-answering systems, dialog systems, and machine translation. Recently, due to the limitations of automatic paraphrasing systems, crowdsourcing has been growing in popularity for obtaining corpora of paraphrases. Since crowdsourced paraphrases are generated by crowd workers with varied skills and motivations, such paraphrases may suffer from various quality issues (e.g., semantic and linguistic errors). Such quality issues stem from the lack of quality assessment methods for open-ended text in crowdsourcing platforms (e.g., Mechanical Turk). Thus unqualified workers and spammers may generate erroneous paraphrases. Crowdsourced paraphrases must be verified to remove erroneous paraphrases, imposing an extra cost. Detecting such erroneous paraphrases requires an understanding of how malicious workers generate paraphrases. In this paper, we provide a taxonomy of cheating behaviors in crowdsourced paraphrasing. We also propose an effective approach to detect cheating behaviors.

**Keywords:** Crowdsourcing · Paraphrasing · Quality control

## 1 Introduction

Paraphrasing is the task of rephrasing a given sentence into a new variation which conveys the same meaning as the original sentence. Paraphrasing has numerous applications in natural language processing systems such as evaluation of machine translation systems, query reformulation in question answering, information retrieval, and dialog systems [15, 27]. It has also applications in building automatic plagiarism detection, text summarization and natural language generation systems [18].

Existing approaches for paraphrasing involve either automated (e.g., paraphrasing systems) or crowdsourcing techniques [18, 31]. Automated paraphrasing is potentially cost-free. However, even state-of-art techniques fall short in producing semantically correct but sufficiently diverse paraphrases [12]. In crowdsourced paraphrasing, crowd workers generate one or more paraphrases for a given sentence (usually up to three paraphrases) [31]. However, it has been shown that a considerable percentage of crowdsourced paraphrases (up to 40%) may be

of unacceptable quality [7]. Crowdsourced paraphrases suffer from various quality issues such as misspellings and grammatical errors. A crowdsourced paraphrase may not also be an exact paraphrase of the given sentence (conveying semantically different meaning). In addition, malicious crowd workers may intentionally generate erroneous paraphrases [7, 31]. Thus crowdsourced paraphrases must be checked for quality [31]. However, crowdsourcing platforms lack in-built facilities for assessing such open-ended inputs. This makes submission of unqualified paraphrases easier.

The common practice for quality assessment of crowdsourced paraphrases is to design another crowdsourcing task (called validation task) in which workers validate crowdsourced paraphrases [31]. However, such an approach is costly since malicious workers are paid for the erroneous paraphrases they generate, not to mention the cost imposed by the validation task (either by experts or by crowd workers). Automated approaches for quality assessment of crowdsourced paraphrases are also limited to removing misspelled paraphrases and discarding submissions from workers with low/high task completion time [31]. An alternative approach is to discard low quality workers automatically during crowdsourcing before allowing them to submit paraphrases. Consequently, malicious workers can be detected in real time without letting them submit erroneous paraphrases (and get paid). Such an approach must have at least two characteristics: (i) it must be domain-independent to be used in various crowdsourcing tasks; (ii) it must be of high precision so that valid paraphrases generated are not rejected accidentally.

In this paper, we investigate how malicious crowd workers (also called cheaters) generate erroneous paraphrases and categorize various types of cheating behaviors (e.g., using foreign languages, adding/removing characters) in crowdsourced paraphrases. Moreover, based on the identified cheating behaviors, we propose a set features to be used in training machine learning models for detecting malicious workers. More specifically, our contributions are two-folded:

- By manual inspections on the *ParaQuality*<sup>1</sup> [31] dataset which contains 6000 paraphrases in 40 domains, we identify a taxonomy of common cheating behaviors in crowdsourced paraphrases (e.g., adding/removing random words to the sentence to generate a paraphrase). Accordingly, we discuss the characteristics of each category of cheating behaviors and how they can be detected automatically.
- Based on the identified characteristics of cheating behaviors, we identify various features from the literature to be used in automatic detection of malicious workers [17, 31]. We also discuss the shortcomings of existing features, namely calculating semantic similarity between multiple paraphrases and tracking how a worker edits the given sentence to generate passphrases. Thus we propose two new features to overcome the identified shortcomings: (i) *inter-paraphrase semantic similarity* and (ii) *worker’s editing patterns* (e.g., which part of sentence is rephrased to generate the paraphrase). Our experiments

---

<sup>1</sup> <https://github.com/mysilver/ParaQuality>.

indicate that the proposed method is an effective approach to detect cheating behaviors in crowdsourced paraphrases.

## 2 Cheating Behaviors

To characterize cheating behaviors in crowdsourced paraphrases, we investigated paraphrases labeled *Cheating* in the *ParaQuality* dataset [31]. This dataset includes 6000 annotated paraphrases (i.e., *Correct*, *Cheating*, *Linguistic Errors*) from 40 domains in which crowd workers provided three paraphrases for a given sentence (totally 40 sentences with 2000 triple paraphrases). Based on manual inspections, we recognized 5 primary categories of cheating behaviors as listed below.

**Character-Level Edits.** One of the common cheating behaviors is to add/remove one or a few characters to/from the given sentence (e.g., “*what is the taxi fare from home to airport*”), potentially resulting in spelling errors in the generated paraphrase (e.g., “*what is the taxi fare from home to airport*sdsdsd**”). Sincere workers may also occasionally have typos in their paraphrases (“*from*” → “*form*”), but such typos may differ from those generated by malicious workers (“*from*” → “*frommm*”). In particular, adding random characters may create nonsense words (also called gibberish words e.g., “*sdsds*”) in the paraphrase. In addition, such erroneous paraphrases and the corresponding sentences have a low *edit distance*<sup>2</sup> since only a few characters are added/removed from the given sentence. However, our manual inspections indicates that a sincere worker may also create a correct paraphrase with low edit distance (e.g., “*what is the taxi rate from home to airport*”), which is not an act of cheating.

**Word-Level Edits.** Similar to the previous type of cheating behavior, cheaters may also add/remove words to/from the given sentence to generate a paraphrase (e.g., “*what is the taxi ~~rate~~ from home to airport*”). Having low edit-distance with the given sentence is the main characteristic of such a paraphrase. Moreover, removed/added words may result in grammatical errors (e.g., “*what the taxi fare from home to airport*”). Needless to say, simply having grammatical errors does not mean a worker is cheating. This type of cheating may also add new entities not mentioned in the given sentence (e.g., “*what is the taxi rate from home to airport for 10 people*”).

**Random Sentences.** Another cheating behavior is to rewrite the given sentence (e.g., “*create a public playlist named NewPlaylist*”) substantially in a very random way (e.g., “*public saw many NewPlaylist this year*”). Detecting such paraphrases requires sophisticated approaches to compute the semantic similarity between two sentences to detect if they are conveying the same meaning or not. The reason stems from the fact that existing solutions fail in detecting semantic divergence when sentences share many words but convey different

---

<sup>2</sup> Edit distance between two strings is the minimum number of operations required to transform one string into the other.

meanings [10]. However, it is easier to detect such case of cheating when a crowd worker provides more than one paraphrase for a given sentences. Take the following paraphrases generated by a worker as example for the sentence “*Create a public playlist named NewPlaylist*”:

- That song hits the public NewPlaylist this year
- Public really loved that NewPlaylist played on the event
- Public saw many NewPlaylist this year

Since these are paraphrases for the same sentence, we can also consider the inter semantic similarity between paraphrases. Considering such similarities (as supplementary measures) can assist in detecting semantically incorrect paraphrases as discussed more in Sect. 3.

**Answering to Canonical Utterance.** When a given sentence is a question (e.g., “*what is the taxi fare from home to airport?*”) or a request for information (e.g., “*I want to know the taxi fare from home to airport*”), malicious workers may start giving responses to the given sentence instead of paraphrasing (e.g., “*the taxi fare is \$10*”). However, such type of errors may also be a result of misunderstanding the task by the crowd worker [31]. Our analysis indicates that this type of cheating may result in having new named entities in the paraphrases which do not exist in the given sentence (e.g., “\$10” in the above-mentioned paraphrase). Thus any automated approach must check if the entities in the paraphrase and given sentence match. Comparing the speech act<sup>3</sup> (e.g., answering, requesting) of a given sentence and its paraphrase can be useful in detecting these cases. However, it has been shown that existing tools for detecting speech acts do not perform well in domain independent settings yet [31].

**Foreign Language.** Cheaters occasionally use their own native languages instead of paraphrasing in the given language. A sincere worker may also generate paraphrases in other languages as a result of misunderstanding the task [31]. However, our investigations indicates that malicious workers provides sentences in other languages which do not express the same meaning as the given sentence. Detecting such cases of cheating seems easy by using Language identification tools (e.g., DetectLanguage<sup>4</sup>). However, these tools may fail in language detection for sentences with misspellings [31]. That is to say, misspellings substantially impact the performance of language detection tools, and consequently the language is detected incorrectly for misspelled sentences. Given that the majority of words in a sentence in another language must considered misspellings by spell checkers in the expected language, we can condition the use of language detection tools for the cases when the sentence has many spelling errors [31].

### 3 Cheating Detection

Deep-learning based approaches gain popularity because they eliminate the burden of manual feature engineering. However using such techniques require a large

<sup>3</sup> The function of a sentence in communication (e.g., such as answering, requesting, greetings).

<sup>4</sup> <https://ws.detectlanguage.com/>.

**Table 1.** Summary of feature library

Category	Feature library
<i>Semantic similarity</i>	Semantic textual similarity method proposed in [10]; Word Mover’s Distance [16] between word embeddings of sentence and word embeddings of its paraphrase; cosine similarities and euclidean distances between vectors of expression and paraphrase generated by Sentence-BERT [25], Universal Sentence Encoder [4], and Concatenated Power Mean Embeddings [26]
<i>Editing patterns</i>	Edit distance between the sentence and its paraphrase; N-gram overlap, exclusive longest common prefix n-gram overlap, and Sumo [6]; Gaussian, Parabolic, and Trigonometric functions proposed in [14]; Paraphrase In N-gram Changes (PINC) [5]; Bilingual Evaluation Understudy (BLEU) [22]; Google’s BLEU (GLEU) [30]; NIST’s <sup>a</sup> ngram score function [8]; Character n-gram F-score (CHRF) [23]; and the length of the longest common subsequence
<i>Language correctness</i>	A function to detect if the paraphrase is written in English [31]; count of gibberish words in the paraphrase <sup>b</sup> ; count of spelling and grammatical errors in the paraphrase; language-model score based on the probability distribution of ngrams in the given paraphrase <sup>c</sup> ; functions to detect if the given paraphrase is a question, an answer or an imperative sentence [31]; a function to detect weather the tenses/pronouns/entities of the sentence and paraphrase match; a function to detect if the paraphrase can be parsed as a proper English sentence by delph-in <sup>d</sup>
<i>General</i>	Difference between count of characters in expression and paraphrase; difference between count of words in expression and paraphrase; time spent by the worker for paraphrasing the given sentence

<sup>a</sup> <https://www.nist.gov/>.<sup>b</sup> <https://github.com/rrenaud/Gibberish-Detector>.<sup>c</sup> <https://kite.com/python/docs/nltk.probability.LaplaceProbDist>.<sup>d</sup> <https://github.com/delph-in/pydelphin>.

amount of high quality annotated data sets [11, 32]. Because of lack of such data sets, we thus manually identified a set of features as discussed in this section.

### 3.1 Feature Engineering

Detecting a cheating behavior requires a deep understanding of how paraphrases are generated by workers. As such, based on the identified types of cheating and their characteristics, we identified high-level categories of features required for detecting cheating behaviors, including for *semantic similarity metrics*, *edit patterns*, *language correctness*, and a few *general features* as listed in Table 1.



Paraphrase the following sentence:

Search for a restaurant near the university

Paraphrase 1 (required)

Paraphrase 2 (required)

Paraphrase 3 (required)

**Fig. 1.** Crowdsourced Paraphrasing in figure-eight

Asking for triple paraphrases is a common practice in crowdsourced paraphrasing [13,31]. Figure 1 shows a sample paraphrasing task designed for getting three paraphrases for a given sentence in figure-eight<sup>5</sup>. Given a sentence and three paraphrases, we propose two novel sets of features: *Inter-Paraphrase Semantic Similarity* and *Edit Features*.

**Inter-paraphrase Semantic Similarity (IPSS).** Measuring semantic similarity between two units of text has numerous applications and has been given much attention by researchers [4,10]. Yet, existing approaches may fail in correctly understanding the extend of semantic similarity between two sentences. Examples includes assigning higher similarity values for a pair of sentences which share large number words regardless of the semantics, and having a low semantic similarity between two equivalent paraphrases without sharing any word [29]. To mitigate the impact of such issues, we propose a new metric called Inter-Paraphrase Semantic Similarity (IPSS). IPSS depends on semantic similarity of a paraphrase not only with the given sentence but also with the other two paraphrases submitted by the same worker for the same sentence. Our intuition is that the triple paraphrases and the given sentence must semantically convey the same meaning, and thus inter-paraphrase semantic similarity should be also high (as well as that of between a sentence and each paraphrase). Moreover, inter-paraphrase similarity scores must be close to each other since they are equivalent paraphrases (indicating low variance). As such, the model can detect if a malicious worker is generating random paraphrases (expressing different meanings).

$$PS = \{sim(u, m) | \forall u \in \{s\} \cup P, \forall m \in P, m \neq u\}$$

$$IPSS(PS) = \frac{mean(PS)}{variance(PS) + \epsilon} \quad (1)$$

where  $P$  represents a set of paraphrases  $\{p_1, p_2, \dots, p_n\}$  for the given sentence  $s$ , and  $sim(\cdot)$  is a function to measure the semantic similarity between two pieces

<sup>5</sup> <https://www.figure-eight.com>.

of text. In our implementation, we used three different similarity functions which have been reported to outperform existing solutions in the semantic textual similarity task: (1) the cosine similarity between embeddings generated by Universal Sentence Encoder [4], (2) the cosine similarity between embeddings generated by Sentence BERT [25], and (3) the similarity model proposed in [10].

**Worker’s Editing Patterns.** As mentioned earlier, a common type cheaters’ editing habits is to create paraphrases by inserting/removing text in/from a particular position of a given sentence (or the previous paraphrase). On the other hand, sincere workers usually create diverse paraphrases by editing the given sentence extensively. Thus, locating the positions (in scale of  $[0,1]$  with 0 showing the beginning and 1 showing the end of paraphrase) of the main edited parts in the paraphrases can give an insight into how a worker generates paraphrases. For a given pair of sentences  $(s1, s2)$ , we define *edit position (EP)* as follow:

$$EP(p1, p2) = \begin{cases} 1 - \frac{p1.index(lcs(p1,p2))}{length(p1)+\epsilon}, & \text{if } lcs(p1, p2) \neq \phi \\ 0.5, & \text{otherwise} \end{cases}$$

where *lcs* shows Longest Common Subsequence between the given sentences. Intuitively, it can be an act of cheating if a worker keeps editing the same part of a sentence to generate a paraphrase based on the previous paraphrase (or the sentence) without much rephrasing. Thereby, such paraphrases should have similar *edit position* values. The variance of edit positions is lower for such paraphrases. Based on that, we define a new feature for training machine learning models called Edit Position Score (EPS) as follow:

$$PS = \{EP(p_i, p_j) | \forall i, j \in \{1, 2, \dots, n\}, i = j + 1\} \cup \{EP(s, p_1)\}$$

$$EPS(PS) = variance(PS) \quad (2)$$

EPS only considers the position of longest common sub-string between sentences. Given that a sincere worker provides diverse paraphrases, the paraphrases do have not have many common ngrams. As such, we calculate the ngram distance (NGD) between a sentence (*s*) and all of corresponding paraphrases (*P*) generated by a worker:

$$NgSet = \bigcup_{p \in P \cup \{s\}} \bigcup_{i=1}^5 ngrams(p, n)$$

$$likelihood(ngram) = \frac{\sum_{p \in P \cup \{s\}} count(ngram, p)}{|P| + 1}$$

$$NGD(s, P) = \frac{1}{|NgSet|} \sum_{ngram \in NgSet} 1 - likelihood(ngram) \quad (3)$$

where *ngrams(.)* extracts ngrams in a given sentence, and *count(.)* represents the count of a given ngram occurrences in the given sentence.

### 3.2 Malicious Worker Detection

It has been shown in [31] that Random Forest classifier [1] outperformed other classifiers in terms of recall and F1 score, Support Vector Machine based classifier (SVM) also gained highest precision. As such, we propose a model based on these algorithms. Using the aforementioned features and the Random Forest classifier, we estimate if a paraphrase ( $p$ ) for a given sentence ( $s$ ) indicates a cheating ( $ch$ ) behavior:  $p(ch|p, s)$ . In other words, the probability of a paraphrase generated by a cheater is the proportion the trees which classified the paraphrase as cheating in the ensemble. However, a malicious worker may have also shown cheating behaviors in their previously submitted paraphrases. The problem can be reformulated as the probability of a worker cheating based on his/her provided paraphrases (in particular we only consider the last three paraphrases submitted by the worker):  $p_{worker}(ch|s, p1, p2, p3)$ . To estimate if a worker cheats, we used Support Vector Regression (SVR) conditioned on probabilities of all submitted paraphrases showing cheating behaviors:

$$p_{worker}(ch|s, p1, p2, p3) = p_{worker}(ch|p(ch|p1, s), p(ch|p2, s), p(ch|p3, s)) \quad (4)$$

## 4 Experiment and Results

Similar to [31], we used 10-fold cross validation on the *ParaQuality* dataset without sharing paraphrases between domains in the test and train folds. In our settings, the test and training sets did not share paraphrases from the same domain, to evaluate the proposed method in a domain-independent manner.

**Table 2.** Automatic malicious worker detection

Model (features)	Precision	Recall	F1
Worker Modeling (All Features + IPSS + Editing Patterns)	0.873	<b>0.585</b>	<b>0.701</b>
Paraphrase Modeling (All Features + IPSS + Editing Patterns)	0.848	0.555	0.671
Worker Modeling (IPSS + Editing Patterns)	0.719	0.586	0.646
Worker Modeling (IPSS)	0.737	0.518	0.609
SVM [31]	<b>0.878</b>	0.223	0.356
K-Nearest Neighbor [31]	0.871	0.248	0.386
Random Forest [31]	0.843	0.546	0.663
Maximum Entropy [31]	0.756	0.440	0.557
Decision Tree [31]	0.632	0.566	0.597
Naive Bayes [31]	0.473	0.426	0.449

## 4.1 Evaluation

Table 2 provides the performance of the proposed method in comparison to prior work. The proposed method to estimate the worker’s cheating probability (Worker Modeling) has a promising performance in comparison with the methods proposed in [31]. It surpasses existing approaches by higher recall and F1 scores.

In *ParaQuality* dataset, 18% of paraphrases are labeled as cheating paraphrases [31]. Figure 2 shows the cheating rate<sup>6</sup> for each domain (totally 40 domains). As shown in this figure, the proposed method works well in different domains which makes it appropriate for being used in crowdsourcing platforms for paraphrasing in any domain. Overall, if we automatically reject triple paraphrases which are detected as cheating, the cheating rate drops to 8.2%. However, if the worker is also prevented to continue paraphrasing when they are suspected as cheaters for the first time, the cheating rate will drop to 4.4%. This also indicates that submitted paraphrases can be verified automatically in real time, and if the submitted paraphrases are detected as cheating cases, the worker can be banned automatically from the paraphrasing task (potentially saving money and improving the quality of collected dataset).

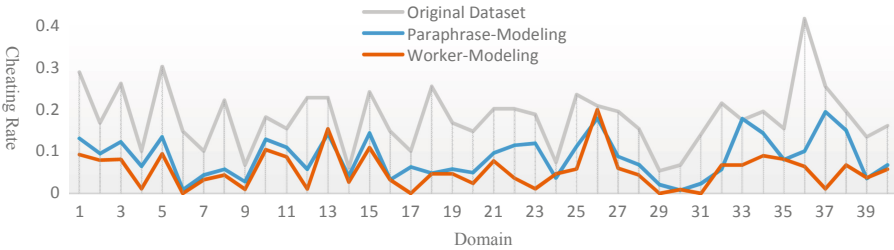


Fig. 2. Cheating rate across different domains

## 4.2 Error Analysis

While the proposed approach outperforms the prior work, it still fails in a few cases. The related work indicates that existing language tools frequently fail in detecting linguistic errors (e.g., grammatical and spelling errors), and thus affects the proposed method as well [31]. Moreover, our investigations indicate that the key to detecting invalid paraphrases is measuring the semantic similarity. Using the state of art approaches to measure semantic similarity (i.e., Universal Sentence Encoder), the proposed feature (IPSS) improved the model performance over the prior work. However, we still require more accurate approaches to measure the similarity between a given sentence and its paraphrases.

In addition, using the features of *worker’s editing patterns* may also result in rejecting valid paraphrases with low edit distance to the given sentence (e.g.,

<sup>6</sup> Cheating rate is the percentage of invalid paraphrases generated by cheaters.

“request a taxi to take me home” and “get a taxi to take me home”). Such paraphrases are valid paraphrases; however, they might not be proper paraphrases since the main reason for paraphrasing is obtaining diverse paraphrases (ideally with new wording and structure). Thus, rejecting such paraphrases may also be ideal in many applications [13, 24].

Our evaluation also indicates that in 78.4% of times that the proposed model incorrectly detected a cheating behavior, the paraphrases were not proper paraphrases (suffer from other types of quality issues such as misspelling, grammatical errors). Thus, in most cases, the proposed method only bans cheaters and low quality workers from continuing the job (less likely to remove high quality workers).

## 5 Related Work

To the best of our knowledge, that our work is the first to identify and categorize cheating behaviors of crowdsource workers generating paraphrases. Nevertheless, our work is related to the areas of (i) quality issues of crowdsourced paraphrases; (ii) quality control in crowdsourced natural language datasets; and (iii) semantic similarity.

**Quality Issues of Crowdsourced Paraphrases.** Crowdsourced paraphrases have been shown to suffer from various quality issues [3, 31]. For instance, crowdsourced paraphrases usually include paraphrases which are *not semantically equivalent* with a given sentence (e.g., “request a taxi from home to the airport”). Examples includes missing entity values (e.g., “get a taxi from home”), using wrong entity values (e.g., “get a taxi from office to the airport) [28]. Linguistic errors (misspellings and grammatical errors) have also reported to be one of most common mistakes in paraphrasing [9, 20]. Examples include verb errors, preposition errors, vocabulary errors (improper word substitution), and incorrect singular/plural nouns are just a few types of linguistic errors occur in crowdsourced paraphrases. In this paper, we also concentrated on an important source of having erroneous paraphrases generated by malicious workers. Specifically, we investigated how malicious workers generate invalid paraphrases and how they can be detected automatically.

**Quality Control.** Quality of a submitted task (e.g., paraphrases) can be assessed after or before data acquisition. *Post-hoc* approaches assess the quality of submitted tasks when all tasks are collected. *Pre-hoc* approaches, on the other hand, can prevent submission of low quality tasks during crowdsourcing.

The most prevalent *post-hoc* approach for verifying crowdsourced paraphrases is launching a verification task [20]. In crowdsourced paraphrases, other prevalent approaches include discarding submissions from workers with low/high task completion time [17], and removing paraphrases with spelling errors [28]. Machine-learning based approaches have also been explored in plagiarism detection systems to assess the quality of paraphrases [2]. While such approaches are desirable to denoise collected paraphrases, their main drawback is having to pay for both the erroneous paraphrases and the verification task.

*Pre-hoc* methods, on the other hand, rely on real time approaches to assess the quality of submissions during crowdsourcing [21]. Such methods can be used to detect malicious workers or to generate automatic feedback to assist crowd workers in generating high quality paraphrases. Precog [21], as an example of such approaches which is designed for crowdsourced product reviews, automatically generate feedback for multi-paragraph text. This paper aims for paving the way for building automatic *pre-hoc* approaches, and detecting malicious workers before they are paid. However, the proposed method can also be used for building *post-hoc* methods to automatically omit faulty paraphrases.

**Semantic Similarity.** Measuring similarity between two sentences has a wide range of applications in Natural Language Processing (NLP) systems. Examples of such application include plagiarism detection, question-answering systems, paraphrase detection [10, 19, 33]. Recent advances in sentence embedding (e.g., Universal Sentence Encoder [4]) can be exploited to detect if a given paraphrase conveys the same meaning. However, existing approaches may fail in correctly understanding the extent of semantic similarity between two sentences. Examples includes assigning higher similarity values for a pair of sentences which share many words even regardless of the semantics, and having a low semantic similarity between two exact paraphrases without sharing any word [29]. To reduce the impact of such issues, in this paper, we introduced a new feature called IPSS. IPSS has built on the fact that a worker’s paraphrases for the same sentence must semantically express the same meaning.

## 6 Conclusion

In this paper, we employed a data-driven approach to investigate various cheating behaviors and their characteristics in crowdsourced paraphrases. Moreover, we identified various features from literature based on the characteristics of each type of cheating behaviors. Then we proposed two new features to overcome shortcomings in the literature (e.g., multi-paraphrase similarity, user edit habits) and propose an domain-independent method for modeling malicious workers. As a future work, we will be working on building a software service which can be integrated with existing crowdsourcing platforms, together with many other exciting opportunities as extensions to this work (e.g., automatic quality issue detection, and automatic feedback generation to assist workers while they are paraphrasing).

**Acknowledgment.** This research was supported fully by the Australian Government through the Australian Research Council’s Discovery Projects funding scheme (project DP1601104515).

## References

1. Breiman, L.: Random forests. UC Berkeley TR567 (1999)
2. Burrows, S., Potthast, M., Stein, B.: Paraphrase acquisition via crowdsourcing and machine learning. *ACM Trans. Intell. Syst. Technol. (TIST)* **4**(3), 43 (2013)
3. Campagna, G., Ramesh, R., Xu, S., Fischer, M., Lam, M.S.: Almond: the architecture of an open, crowdsourced, privacy-preserving, programmable virtual assistant. In: *WWW 2017*, pp. 341–350 (2017)
4. Cer, D., et al.: Universal sentence encoder (2018)
5. Chen, D.L., Dolan, W.B.: Collecting highly parallel data for paraphrase evaluation. In: *ACL*, pp. 190–200 (2011)
6. Cordeiro, J., Dias, G., Brazdil, P.: A metric for paraphrase detection. In: *ICCGI 2007*, p. 7. *IEEE* (2007)
7. Daniel, F., Kucherbaev, P., Cappiello, C., Benatallah, B., Allahbakhsh, M.: Quality control in crowdsourcing: a survey of quality attributes, assessment techniques, and assurance actions. *ACM Comput. Surv. (CSUR)* **51**(1), 7 (2018)
8. Doddington, G.: Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In: *Proceedings of the Second International Conference on Human Language Technology Research, HLT 2002*, pp. 138–145 (2002)
9. Enokibori, Y., Takahashi, K., Mase, K.: Statistical response method and learning data acquisition using gamified crowdsourcing for a non-task-oriented dialogue agent. In: *ICAART 2014*, vol. 8946, p. 119. Springer (2015)
10. Fakouri-Kapourchali, R., Yaghoub-Zadeh-Fard, M.-A., Khalili, M.: Semantic textual similarity as a service. In: Beheshti, A., Hashmi, M., Dong, H., Zhang, W.E. (eds.) *ASSRI 2015/2017. LNBP*, vol. 234, pp. 203–215. Springer, Cham (2018). [https://doi.org/10.1007/978-3-319-76587-7\\_14](https://doi.org/10.1007/978-3-319-76587-7_14)
11. Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y.: *Deep Learning*, vol. 1. MIT Press Cambridge (2016)
12. Gupta, A., Agarwal, A., Singh, P., Rai, P.: A deep generative framework for paraphrase generation. *AAAI* (2018)
13. Jiang, Y., Kummerfeld, J.K., Laseck, W.S.: Understanding task design trade-offs in crowdsourced paraphrase collection. *ACL* (2017)
14. Joao, C., Gaël, D., Pavel, B.: New functions for unsupervised asymmetrical paraphrase detection. *J. Softw.* **2**(4), 12–23 (2007)
15. Kriz, R., Miltsakaki, E., Apidianaki, M., Callison-Burch, C.: Simplification using paraphrases and context-based lexical substitution. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 207–217 (2018)
16. Kusner, M., Sun, Y., Kolkin, N., Weinberger, K.: From word embeddings to document distances. In: *International Conference on Machine Learning*, pp. 957–966 (2015)
17. Ma, X., Neeraj, T., Naaman, M.: A computational approach to perceived trustworthiness of Airbnb host profiles (2017). <https://aaai.org/ocs/index.php/ICWSM/ICWSM17/paper/view/15630>
18. Madnani, N., Dorr, B.J.: Generating phrasal and sentential paraphrases: a survey of data-driven methods. *Comput. Linguist*
19. Mannarswamy, S., Chidambaram, S.: GEMINIO: finding duplicates in a question haystack. In: Phung, D., Tseng, V.S., Webb, G.I., Ho, B., Ganji, M., Rashidi, L. (eds.) *PAKDD 2018. LNCS (LNAI)*, vol. 10938, pp. 104–114. Springer, Cham (2018). [https://doi.org/10.1007/978-3-319-93037-4\\_9](https://doi.org/10.1007/978-3-319-93037-4_9)

20. Negri, M., Mehdad, Y., Marchetti, A., Giampiccolo, D., Bentivogli, L.: Chinese whispers: cooperative paraphrase acquisition. In: LREC, pp. 2659–2665 (2012)
21. Nilforoshan, H., Wang, J., Wu, E.: Precog: Improving crowdsourced data quality before acquisition. CoRR (2017)
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 on ACL, pp. 311–318. ACL (2002)
23. Popović, M.: CHRf deconstructed: beta parameters and n-gram weights. In: Proceedings of the First Conference on Machine Translation, pp. 499–504. Association for Computational Linguistics (2016)
24. Ravichander, A., Manzini, T., Grabmair, M., Neubig, G., Francis, J., Nyberg, E.: How would you say it? Eliciting lexically diverse dialogue for supervised semantic parsing. In: The 18th SIGdial, pp. 374–383 (2017)
25. Reimers, N., Gurevych, I.: Sentence-BERT: Sentence embeddings using Siamese BERT-networks. arXiv preprint [arXiv:1908.10084](https://arxiv.org/abs/1908.10084) (2019)
26. Rücklé, A., Eger, S., Peyrard, M., Gurevych, I.: Concatenated  $p$ -mean word embeddings as universal cross-lingual sentence representations. CoRR (2018)
27. Su, Y., Hassan Awadallah, A., Khabsa, M., Pantel, P., Gamon, M.: Building natural language interfaces to web APIs. In: CIKM 2017 (2017)
28. Wang, W.Y., Bohus, D., Kamar, E., Horvitz, E.: Crowdsourcing the acquisition of natural language corpora: methods and observations. In: SLT, pp. 73–78. IEEE (2012)
29. Wang, Z., Mi, H., Ittycheriah, A.: Sentence similarity learning by lexical decomposition and composition. arXiv preprint [arXiv:1602.07019](https://arxiv.org/abs/1602.07019) (2016)
30. Wu, Y., et al.: Google’s neural machine translation system: Bridging the gap between human and machine translation. CoRR (2016)
31. Yaghoubzadeh, M., Benatallah, B., Chai Barush, M., Zamanirad, S.: A study of incorrect paraphrases in crowdsourced user utterances. In: NAACL 2019 (2019)
32. Yang, J., Drake, T., Damianou, A., Maarek, Y.: Leveraging crowdsourcing data for deep active learning an application: learning intents in Alexa. In: IW3C2, pp. 23–32 (2018)
33. Yang, Y., et al.: Learning semantic textual similarity from conversations. In: Proceedings of The Third Workshop on Representation Learning for NLP, pp. 164–174. Association for Computational Linguistics (2018)





# A Blockchain-Based Approach for Trust Management in Collaborative Business Processes

Ada Bagozi<sup>(✉)</sup>, Devis Bianchini, Valeria De Antonellis, Massimiliano Garda, and Michele Melchiori

Department of Information Engineering, University of Brescia, Via Branze 38, 25123 Brescia, Italy

{a.bagozi,devis.bianchini,valeria.deantonellis,  
m.garda001,michele.melchiori}@unibs.it

**Abstract.** Blockchain is becoming a powerful technology for re-engineering collaborative business processes implemented on Web-based distributed systems, spanning across enterprises. On the blockchain, cross-organisation Web services orchestrated to form collaborative business processes can be transparently deployed as smart contracts. However, proper methods and tools are required to guide the process designer for exploiting the blockchain technology. To preserve data and business logics ownership and to ensure performance/cost tradeoff, only data and process activities requiring transparency and trust among the distributed process actors should be stored as transactions on the blockchain and deployed as smart contracts. In this paper, we propose a methodology and a tool that rely on methodological steps to support blockchain-based trust management in Web-based collaborative business processes originally designed according to a centralised BPM strategy. The methodology and the tool are grounded on a set of criteria, properly enforced with metrics, to identify trust-demanding elements to be considered for their deployment on the blockchain. The approach has been validated on a real case study of food quality certification in the biological domain.

**Keywords:** Blockchain · Smart contract · Collaborative processes · BPM

## 1 Introduction

Collaborative processes implemented on Web-based distributed systems are widely used to model cooperation between (potentially untrustworthy) organisations, that cooperate in order to increase their value. In recent years, researchers proposed the adoption of blockchain and smart contracts for implementing collaborative business processes going beyond a centralised Business Process Management (BPM) perspective [8, 9]. Blockchain and BPM have been jointly investigated in several real case scenarios, such as supply chain management,

logistics, manufacturing and material industry [7]. Existing approaches investigated mainly how smart contracts can be generated from BPMN models, further proposing cost optimisation strategies [4]. In general, they are platform-dependent, referring to specific blockchain technologies. Recently, the need for proper methods and tools, to guide the process designer for exploiting the blockchain technology, is emerging. To preserve data and business logics ownership and to ensure performance/cost tradeoff, only data and process activities requiring transparency and trust among the distributed process actors should be stored as transactions on the blockchain and deployed as smart contracts. In [1] we defined such elements as trust-demanding objects and trust-demanding activities, respectively. In this paper, we propose a methodology and a tool to support blockchain-based trust management in Web-based collaborative business processes originally designed according to a centralised BPM strategy. The methodology and the tool are grounded on a set of criteria, properly enforced with metrics, to identify trust-demanding elements to be considered for their deployment on the blockchain. This paper further extends our previous research [1] with the introduction of quantitative metrics in the methodological steps and the Web-based tool that supports the process re-engineering.

The paper is organised as follows: Sect. 2 discusses the cutting-edge features of the approach with respect to related work; Sect. 3 describes a real world case study for food quality certification in the biological domain; Sect. 4 presents the methodology; Sect. 5 describes implementation and validation issues; finally, Sect. 6 closes the paper and sketches future research directions.

## 2 Related Work

The exploitation of blockchain technology in BPM lifecycle has been fruitfully investigated in recent work. In particular, Model-Driven Engineering solutions [5, 7] have been proven to be effective for modelling blockchain-based collaborative business processes. In [2] models at various levels of abstraction have been conceived to produce smart contracts code for implementing, either totally or partially, the collaborative business process. The Caterpillar approach [8] introduces an abstraction layer over the Ethereum blockchain, recording states of each process instance on the blockchain and deploying the control flow logic as smart contracts. The Lorikeet approach [9] provides an extension of BPMN to represent asset registries as list of information recorded by a trusted authority. The work in [6] focuses on the importance of conceptual modelling to demonstrate how business artifacts leverage the data-centric nature of blockchain. In the latter contributions, proper policies to select which data should be stored on-chain are also considered. Distinction between off-chain and on-chain elements is extensively investigated in [3], where on-chain and off-chain design patterns are described.

Following the lesson learned in [3], we propose a methodology that guides, with the help of proper criteria, the identification of collaborative business process elements to be moved on-chain and elements to be left off-chain. The criteria

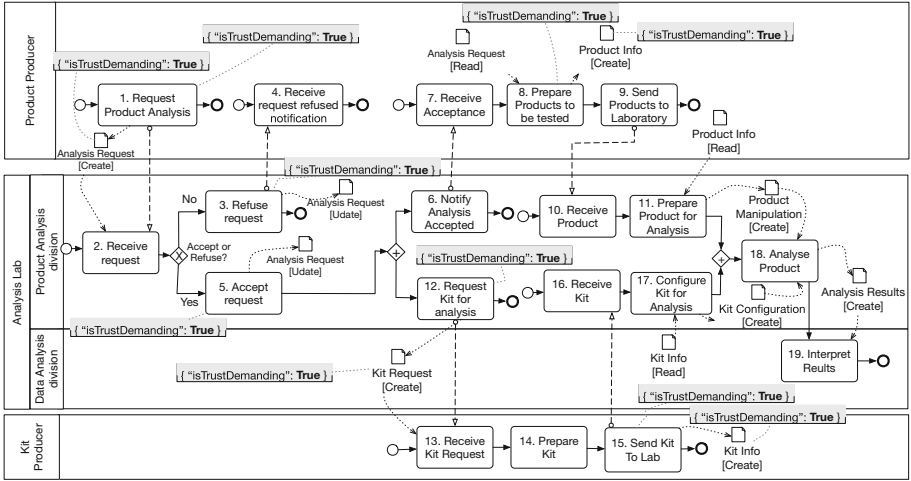


Fig. 1. BPMN diagram of the food quality certification process.

are based on the notion of trust-demanding objects and activities as defined in [1] and are enforced through quantitative metrics. With respect to [3], the proposed methodology produces a blockchain-based implementation of the business process. The methodology has been implemented in a Web-based tool, which like Caterpillar [8] guides the process designer from business process design to its deployment on-chain. Furthermore, with respect to the other approaches, the methodology supports the generation of Abstract Smart Contracts that are independent of any specific blockchain technology.

### 3 Motivating Example

Figure 1 reports the BPMN diagram representing the orchestration of Web-based services in a real case study of food quality certification. In the considered process, a product producer issues a request to an analysis laboratory (organised into a Product Analysis and a Data Analysis division) to obtain a quality certification in order to trade the product, in compliance with current regulations. The product to be tested is prepared by product producer and sent to the Product Analysis division, which prepares the product for the analysis. In parallel, the kit producer prepares the kit and sends it to the Product Analysis division, which configures the kit. Once both the product and the kit have been prepared, the Product Analysis division analyses the manipulated product using the prepared kit. Finally, the analysis results are sent to the Data Analysis division to be analysed.

In this scenario, analysis results and certificate could be potentially sources of trust problems between process actors and must not be repudiable. Similarly, the analysis procedure should be transparently shared among all involved

participants. Moreover, finding a central authority that ensures trust among participants may be difficult if participants change over time (e.g., if the product producer decides to rely on a different analysis lab). To cope with the former issues, blockchain technology comes to the rescue. However, when deploying the business process on-chain, there is the need of ensuring performance and cost tradeoff. According to these considerations, the methodology proposed in this paper aims at supporting the identification of candidate data to be stored as transactions on the blockchain and activities to be deployed as smart contracts.

**Trust-Demanding Objects and Activities.** In the BPMN diagram of Fig. 1, annotations are used to highlight elements to be stored as transactions on the blockchain, and deployed as smart contracts. In [1] we defined such elements as *trust-demanding objects* and *trust-demanding activities*, respectively. A trust-demanding object is created/updated/deleted by a participant  $p_i$  and read as input of another activity associated with  $p_j \neq p_i$ , where  $p_i$  and  $p_j$  belong to different pools (e.g., **Product Info** data object is created by the *Product Producer* and read by the *Product Analysis* division). Conversely, participants corresponding to different lanes within the same pool are conceived as trusted actors, since they represent distinct divisions within the same organisation. Trust-demanding activities create/update/delete trust-demanding objects (e.g., **Product Info** data object is generated by Activity 8) and thus the logic behind such manipulations should be transparently shared among potentially untrusting participants.

## 4 Methodology

The proposed methodology is conceived as a set of (possibly iterative) steps and starts from an AS-IS collaborative process represented in BPMN and supports the process designer for preparing the implementation of the TO-BE blockchain-based process. We remark that the first three steps are independent of any specific blockchain technology.

**1) Candidate on-chain objects and activities identification.** The input of this step is the BPMN diagram representing the AS-IS collaborative process. Herein, the identification criteria exposed in Sect. 3 are used to automatically highlight candidate objects and activities to be deployed on-chain.

**2) On-chain objects and activities selection.** The process designer manually selects and annotates objects and activities to be deployed on-chain with the support of proper metrics providing a quantitative feedback regarding her selection strategy (i.e., either to foster trustworthiness or costs containment). Indeed, identification does not automatically entail selection. The process designer may decide to keep off-chain candidate elements, based on her knowledge of the process (e.g., two participants that, albeit modelled with different pools in the BPMN, belong to the same consortium).

**3) Abstract Smart Contracts generation.** Once on-chain objects and activities have been selected, a set of Abstract Smart Contract (ASC) descriptors is generated, to provide high level description, independent of the blockchain

technology adopted, of: (a) each activity selected for on-chain migration, because its business logic must be stored on the blockchain as shared code; (b) each selected on-chain object, in which case the ASC includes as functions the CRUD actions performed on the object when stored as transaction on the blockchain.

**4) Concrete Smart Contracts deployment.** Starting from the set of ASC descriptors generated in the previous step, the developer implements the set of Concrete Smart Contracts (CSCs) on a specific blockchain platform. Developers' skills are clearly distinguished from the ones of process designers, who are in charge of supervising the blockchain-based re-engineering of the collaborative process.

#### 4.1 Metrics to Support On-Chain Elements Selection

**Deployment Metric.** The first metric aims at supporting the process designer during the selection of activities (resp., data objects) to be deployed on-chain. On the one hand, the metric can be defined in order to suggest to deploy on-chain as much activities as possible. We refer to this strategy as *trust-oriented*, that may be typical of collaborative business processes that are characterised by a low degree of trustworthiness between process participants. On the other hand, the metric can pursue the improvement of the performance/cost ratio. We refer to this strategy as *performance/cost-oriented*, since it gives more importance to the containment of costs and performance. Focusing on the set of activities  $A$  in the AS-IS process, we denote with  $A_{on}^c \subseteq A$  the set of candidate on-chain activities, with  $A_{off}^c = A \setminus A_{on}^c$  the set of candidate off-chain activities, with  $A_{on}^s \subseteq A$  the set of activities that have been selected by the designer for their migration on-chain and with  $A_{off}^s = A \setminus A_{on}^s$  the set of not selected activities:

$$m_A = \alpha \cdot \frac{|A_{on}^c \cap A_{on}^s|}{|A_{on}^c|} + \beta \cdot \left(1 - \frac{|A_{on}^s \cap A_{off}^c|}{|A_{on}^s|}\right) \in [0, 1] \quad (1)$$

where  $\alpha$  and  $\beta$  weights balance the impact of the terms in Eq. (1) ( $\alpha + \beta = 1$ , with  $\alpha = \beta = \frac{1}{2}$  both strategies are equally considered). In particular,  $|A_{on}^c \cap A_{on}^s|/|A_{on}^c|$  is maximised when  $A_{on}^c \equiv A_{on}^s$  (trust-oriented strategy). On the other hand,  $|A_{on}^s \cap A_{off}^c|/|A_{on}^s|$  is minimised when candidate off-chain activities are not selected to be deployed on chain (performance/cost-oriented strategy). The deployment metric  $m_O$  for data objects is defined in a similar way.

**Context Switch Metric.** The deployment metric  $m_A$  does not consider the cost of a deployment strategy in terms of context switches between on-chain and off-chain activities. Indeed, switches in the process execution between on-chain and off-chain parts of the process typically affect negatively its execution cost and performances (e.g., to deploy and execute smart contracts functions). To limit the negative effects of context switch, a proper metric is proposed, aimed at quantifying the average number of switching from an activity executed off-chain to an activity executed on-chain, and vice versa. Let  $p_{\langle a_i, a_j \rangle}$  be a path in the BPMN process, i.e. a sequence of activities from  $a_i$  to  $a_j$  (where  $a_i, a_j \in A$ ).

The following metric  $\mu p_{\langle a_i, a_j \rangle}$  is used to count the average number of context switches along  $p_{\langle a_i, a_j \rangle}$ :

$$\mu p_{\langle a_i, a_j \rangle} = \frac{\sum_{l=i}^{j-1} (isSel(a_l) - isSel(a_{l+1}))^2}{|p_{\langle a_i, a_j \rangle}| - 1} \in [0, 1] \quad (2)$$

where  $a_l$  and  $a_{l+1}$  represent two consecutive activities along  $p_{\langle a_i, a_j \rangle}$  and  $isSel(a_l) \in \{0, 1\}$  represents a boolean function that checks whether  $a_l$  is selected to be on-chain or not. Considering that there may be multiple paths from  $a_i$  to  $a_j$ , an overall *context switch* metric  $\Phi_{\langle a_i, a_j \rangle}$  is proposed:

$$\Phi_{\langle a_i, a_j \rangle} = \frac{\sum_{k=1}^{|P_{\langle a_i, a_j \rangle}|} \mu p_{\langle a_i, a_j \rangle}^k}{|P_{\langle a_i, a_j \rangle}|} \in [0, 1] \quad (3)$$

where  $P_{\langle a_i, a_j \rangle}$  is the set of all the possible paths from  $a_i$  to  $a_j$ . If there is a loop from  $a_i$  to  $a_j$ , only one iteration is considered in Eq. (3).

**Metrics-Driven Selection of On-Chain/Off-Chain Activities.** The knowledge provided by the metrics is leveraged by the process designer to correct/revise her design policy. In fact, Eq. (1) does not consider context switch. Thus, to limit the latter, the designer has to move off-chain activities that are currently on-chain (performance/cost-oriented strategy) or vice versa (trust-oriented strategy). In this respect, if the strategy adopted is trust-oriented, the activities lying on the path between two candidate on-chain activities can be in turn selected to be moved on-chain, to reduce the context switch metric value. Otherwise, if the strategy adopted is performance/cost-oriented, the designer may choose to move off-chain several candidate activities initially conceived to be on-chain.

## 4.2 Abstract Smart Contracts Generation

Once the sets of selected on-chain objects and activities have been chosen, a set of Abstract Smart Contract (ASC) descriptors can be generated. ASC descriptors are independent of the adopted blockchain technology and are used by the developer, after selecting a target blockchain technology, to automatically generate the skeleton of Concrete Smart Contracts. According to the definition given in [1], an ASC  $asc$  is modelled as a tuple containing: (i) the name  $n_{asc}$  of the ASC; (ii) the set of state variables  $VAR_{asc}$  of the ASC (i.e., primitive data types or objects) on which contract functions operate; (iii) the set of participants  $P_{asc}$  registered on the blockchain allowed to interact with the ASC; (iv) the set  $F_{asc}$  of the signatures of the functions of the ASC.

*Example 1.* If the trust-demanding activity 8 in Fig. 1 is selected to be deployed on-chain, the following ASC  $asc^w$  is generated:

$$\begin{aligned} n_{asc^w} &= \text{“PrepareProductsToBeTestedSC”} \\ VAR_{asc^w} &= \{AnalysisRequest, ProductInfo\} \\ P_{asc^w} &= \{ProductProducer, BPMNengine\} \\ F_{asc^w} &= \{prepareProducts([AnalysisRequest]) : [AnalysisRequest]\} \end{aligned}$$

## 5 Implementation and Validation Issues

**Tool for Blockchain-Based Process Re-engineering.** The Web-based tool<sup>1</sup> conceived to support process designers has been implemented on top of the architecture presented in [1]. Through the dashboard (Fig. 2), the process designer invokes the routine apt to automatically highlight candidate on-chain objects (indicated by letter “T” in the symbol) and activities (marked with a red border). In the process of Fig. 1, the tool identifies 6 activities and 4 data objects as candidates to be deployed on-chain. Focusing on activities, the process designer selected for on-chain deployment 7 activities, but one of them has not been identified as on-chain candidate (i.e., *Analyse Product*), leading to a  $m_A = 92.5\%$ . In this case, the designer’s choice affects mainly the performance/cost-oriented strategy (the selected activity is not candidate to be on-chain). To fulfil the performance/cost-oriented strategy, the tool suggests to deselect the “Analyse Product” activity (**Tip 1**). Regarding the context switch metric  $\Phi_{\langle a_i, a_j \rangle}$ , it is represented as a heatmap wherein red sections represent paths where the context is switched for every activity. For instance, **Tip 3** states that from activity 1 to activity 6, the context has to be changed each step (activities 1 and 5 are selected to be on-chain).

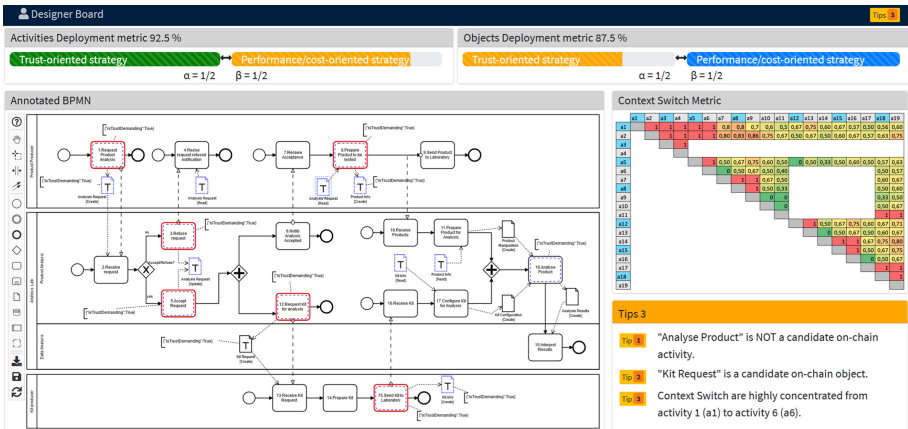


Fig. 2. Dashboard of the tool for blockchain-based business process re-engineering.

**Preliminary Cost Analysis.** We demonstrated that the proposed metrics-based selection of elements to be migrated on-chain enables mitigation of deployment and execution costs. Based on the process in Fig. 1, we considered three different configurations on the Ethereum permissionless blockchain: (1) deployment of all the activities and data objects on-chain; (2) deployment of all the

<sup>1</sup> Screencast of the tool is available at: <https://tinyurl.com/wise-screencast>.

activities and of on-chain candidate data objects only; (3) deployment of on-chain candidate activities and candidate data objects. Practically, we generated CSC skeletons in Solidity language starting from the related ASC descriptors, and calculated both smart contracts deployment and functions execution costs. As expected, configuration (1) yields the highest costs ( $\approx 2.5$  times the ones of (3)), which indirectly affect the average confirmation time for transaction on the blockchain (please refer to [1] for further details).

## 6 Concluding Remarks and Future Research

In this paper, a methodology to support blockchain-based re-engineering of collaborative business processes has been proposed, originally designed according to a centralised BPM strategy. Furthermore, a Web-based tool implementing methodological steps has been developed as well. The methodology and the tool are grounded on a set of criteria, properly enforced with metrics, to identify on-chain elements to be considered for their deployment on the blockchain. Future efforts will consider further usability experiments and the validation of the Web-based tool in specific application domains, namely energy distribution on smart grids and food quality certification. Furthermore, the set of criteria used to distinguish between on-chain and off-chain elements will be enriched, for example based on other experiences like the ones described in [3], in order to make more accurate the performance/costs tradeoff obtained by applying our methodology.

## References

1. Bagozi, A., Bianchini, D., De Antonellis, V., Garda, M., Melchiori, M.: A three-layered approach for designing smart contracts in collaborative processes. In: Proceedings of the 27th International Conference on Cooperative Information Systems, CoopIS 2019, Rhodes, Greece, pp. 440–457 (2019)
2. Di Ciccio, C., et al.: Blockchain support for collaborative business processes. *Informatik Spektrum* **42**(3), 182–190 (2019)
3. Eberhardt, J., Tai, S.: On or off the blockchain? Insights on off-chaining computation and data. In: De Paoli, F., Schulte, S., Broch Johnsen, E. (eds.) ESOCC 2017. LNCS, vol. 10465, pp. 3–15. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-67262-5\\_1](https://doi.org/10.1007/978-3-319-67262-5_1)
4. Garcia-Banuelos, L., Ponomarev, A., Dumas, M., Weber, I.: Optimized execution of business processes on blockchain. In: Proceedings of the 15th International Conference on Business Process Management (BPM) (2017)
5. Garcia-Garcia, J., Sanchez-Gomez, N., Lizcano, D., Escalona, M., Wojdyski, T.: Using blockchain to improve collaborative business process management: systematic literature review. *IEEE Access* **8**, 142312–142336 (2020)
6. Hull, R., Batra, V., Chee, Y., Deutsch, A., Health, F., Vianu, V.: Towards a shared ledger business collaboration language based on data-aware processes. In: Proceedings of the International conference on Service Oriented Computing (ICSOC 2016), Banff, AB, Canada, pp. 18–36 (2016)
7. Mendling, J., et al.: Blockchains for business process management - challenges and opportunities. *ACM Trans. Manag. Inf. Syst.* **9**, 1–16 (2018)



8. López-Pintado, O., García-Bañuelos, L., Dumas, M., Weber, I., Ponomarev, A.: Caterpillar: a business process execution engine on the Ethereum blockchain. *Softw.: Pract. Exp.* **49**(7), 1162–1193 (2019)
9. Tran, A.B., Lu, Q., Weber, I.: Lorikeet: a model-driven engineering tool for blockchain-based business process execution and asset management. In: *BPM (Dissertation/Demos/Industry)*, Sydney, Australia, pp. 56–60 (2018)

# **Database System and Workflow**



# Exploiting Unblocking Checkpoint for Fault-Tolerance in Pregel-Like Systems

Yi Yang<sup>1</sup>, Zhenhua Yang<sup>1</sup>, and Chen Xu<sup>1,2</sup>(✉)

<sup>1</sup> East China Normal University, Shanghai, China

{yiyang, zhyang}@stu.ecnu.edu.cn, cxu@dase.ecnu.edu.cn

<sup>2</sup> Science and Technology on Parallel and Distributed Processing Laboratory (PDL),  
Changsha, China

**Abstract.** With the explosive growth of graph size, a series of Pregel-like systems have emerged. Typically, these systems employ checkpointing and rollback mechanisms to achieve fault-tolerance in either blocking or unblocking manner. The blocking checkpointing pauses the iterative processing while checkpointing, whereas the unblocking checkpointing writes the checkpoints in parallel with the iterative processing. The unblocking checkpointing decreases the checkpointing overhead, but incurs resource contention due to checkpointing concurrently. Hence, it may prolong the time on execution and checkpointing. In this work, we propose a *queuing strategy* to alleviate the contention. This strategy employs a checkpoint queue to store all the pending checkpoints, which allows to concurrently write a certain number of checkpoints at most from the queue following a First-In-First-Out (FIFO) policy. To further utilize the characteristics of checkpoint in Pregel-like systems, we define checkpoint staleness and checkpoint tardiness, and then propose *staleness/tardiness-aware skipping* policy to replace the FIFO policy. Extensive experiments verified that the queuing strategy with the skipping policy outperforms blocking and unblocking checkpointing in Pregel-like systems.

**Keywords:** Graph processing · Fault tolerance · Checkpoint

## 1 Introduction

Graphs are widely employed in various application areas including social network analysis and online recommendation. With the explosive growth of graph size, the big data represented by graphs might exceed the capacity of a single machine in terms of computation and storage, etc. To effectively process large graph data, Pregel [10] as well as many Pregel-like systems Giraph [1] and Sedge [17] typically scale out in a distributed way by increasing the number of compute nodes. However, failure is common in distributed environment especially with a large number of computational nodes [6]. Meanwhile, graph processing often costs a long execution time, because they require iterative computations. Hence, it is important for Pregel-like systems to effectively handle failures.

A typical fault tolerance mechanism proposed in [12] employs a reactive approach to tolerate failure. This approach does not perform any operations during normal execution. Once failure happens, this approach reloads and repartitions the input data to recover the lost vertices as well as edges. Then, it recovers the value of vertices by utilizing the replicas of vertices and a user-defined compensation function. This approach sacrifices the accuracy of computed result [13], even though it achieves fault-tolerance. To keep the result accuracy, most of Pregel-like systems adopt a proactive checkpoint-based approach to tolerate failure [16]. This approach requires the systems to periodically write checkpoints into stable storage during normal execution. Once failure happens, the systems recover from the latest checkpoint. Usually, these Pregel-like systems (e.g., Giraph and Hama [2]) write checkpoints in a blocking manner, which saves the state of the system while pausing computation. Clearly, it incurs an additional overhead on execution time. Different from the blocking checkpointing, the unblocking checkpointing proposed in the literature writes the checkpoints in parallel with the computation [5]. However, its adoption in Pregel-like systems remains unexplored. While applying the unblocking checkpointing to these systems, there may be a resource contention problem due to concurrently checkpointing. Further, the contention brings a negative impact on the time of overall execution and checkpointing. In particular, resource contention may intensify the delay on the execution when multiple checkpoints are issued simultaneously.

The goal of this work is to alleviate the resource contention incurred by writing multiple unblocking checkpoints concurrently, so as to reduce the total execution time. Instead of issuing checkpoints without a limitation, we propose a *queuing strategy* to limit the number of concurrent unblocking checkpoints. Our queuing strategy inserts all the pending checkpoints into a checkpoint queue, and employs a First-In-First-Out (FIFO) policy by default. At a certain time, this strategy limits the number of checkpoints taken from the queue, so as to ensure that the system writes up to  $k$  checkpoints concurrently. Our experimental results show that the unblocking checkpointing with queuing strategy decreases the total execution time by 48.1% and 41.5% compared to the blocking and unblocking checkpointing in the failure-free cases, respectively.

Moreover, in Pregel-like systems, the checkpoint in a later superstep is more useful for failure recovery than the one in an earlier superstep, since recovering from a later superstep enables the system to recompute less supersteps once failure happens. In addition, we find that the checkpoint with a shorter writing time is more helpful for failure recovery than the one with a longer writing time, as the one with a shorter writing time becomes available sooner. In Pregel-like systems, this writing time can be estimated by the checkpoint size. Clearly, the FIFO policy is oblivious to the characteristics of checkpointing, which may result in more recovery time when failure happens. Instead of FIFO policy, we first define checkpoint staleness and checkpoint tardiness, and then propose *staleness/tardiness-aware skipping* policy, in order to reduce the recovery time in case of failure. Our experiments show that the staleness/tardiness-aware skipping policy saves 52.3% of recovery time as against the FIFO policy.

In the rest of this paper, we introduce the background of Pregel-like systems in Sect. 2. Then, we make the following contributions:

- We propose *queuing strategy* in Sect. 3 to alleviate the resource contention incurred by multiple concurrent unblocking checkpointing, so as to reduce the overall execution time.
- We propose *staleness/tardiness-aware skipping* policy in Sect. 4 to replace the FIFO policy in queuing strategy, in order to reduce the recovery time.
- We implement a prototype system on Giraph and our experimental studies in Sect. 5 demonstrate that the queuing strategy with staleness/tardiness-aware skipping policy significantly improves the performance of the existing unblocking checkpointing and blocking checkpointing.

In addition, we review the related work in Sect. 6 and conclude our work in Sect. 7.

## 2 Background of Pregel-Like Systems

This section introduces the workflow of Pregel-like systems, and illustrates two typical checkpointing approaches, i.e., blocking and unblocking checkpointing.

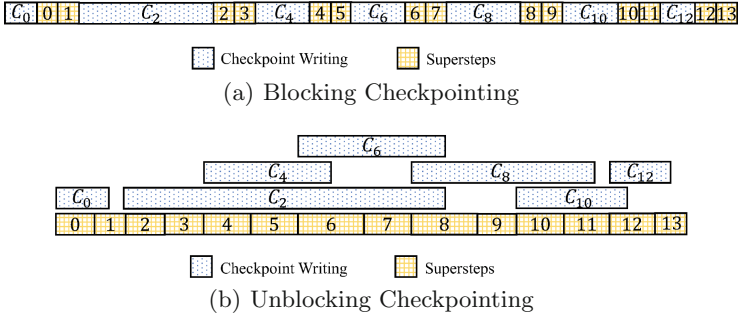
### 2.1 Execution Workflow

Pregel-like systems (e.g., Giraph, Seraph) adopt a vertex-centric programming model for iterative graph processing. The communication among the vertices in these systems is usually achieved by the message passing technique. Generally, message passing is implemented via synchronized execution [11]. The execution includes a sequence of iterations, called *supersteps*. Each superstep consists of three phases, including computation, communication and synchronization.

In the computation phase, the state of each vertex is either active or inactive. These active vertices handle the messages received from the last superstep and execute the same user-defined function to update the value of vertices in parallel. In the communication phase, each vertex sends messages to their neighbor vertices. The neighbor vertices temporarily store the messages and will handle them in the next superstep. In addition, the vertices update their states accordingly. In the synchronization phase, the vertices that complete the communication phase in advance have to wait for other vertices. Only when all vertices complete the communication phase, they will enter the next superstep. The three phases are executed iteratively till all vertices are inactive.

### 2.2 Checkpointing

**Blocking Checkpointing.** Many Pregel-like systems achieve fault tolerance in the way of blocking checkpointing. For example, in Giraph, users specify a checkpoint interval and the system writes checkpoints to a distributed file system by pausing the superstep where the checkpoint is needed. This checkpoint



**Fig. 1.** The process of checkpointing

consists of all vertex states, edges and messages. Once failure occurs, the system reads the latest checkpoint for rollback. In this work, we employ  $C_i$  to indicate the checkpoint at superstep  $i$ . Figure 1(a) shows the blocking checkpointing that writes a checkpoint every two supersteps. At superstep 2, the system writes the checkpoint  $C_2$  and then starts the execution of this superstep. If failure happens at superstep 3, the system rolls back to superstep 2 and reloads  $C_2$ . Clearly, blocking checkpointing incurs a significant overhead cost on execution time.

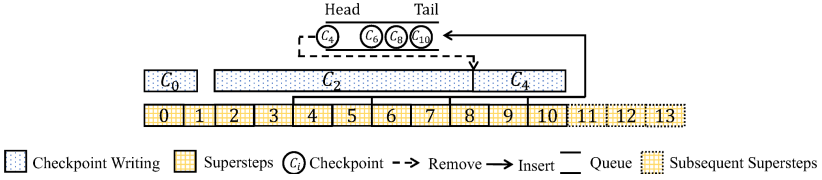
**Unblocking Checkpointing.** Different from the blocking checkpointing, some studies in high performance computing propose the unblocking checkpointing, which writes the checkpoints in parallel with the computation [5]. By adopting the unblocking checkpointing, the Pregel-like systems decrease the overhead of blocking checkpointing. As shown in Fig. 1(b), the system writes  $C_2$  along with the execution of supersteps 2, 3 and 4, which avoids the execution time overhead incurred by writing  $C_2$ .

### 3 Queuing Strategy

In this section, we discuss the resource contention in unblocking checkpointing, and then propose a queuing strategy to alleviate the resource contention.

#### 3.1 Resource Contention

Unblocking checkpointing may incur resource contention, although it decreases the execution time compared to blocking checkpointing. The resource contention prolongs the time of superstep execution and checkpointing, since checkpointing is in parallel to the execution of supersteps. As shown in Fig. 1(b), the unblocking checkpointing requires more time to write  $C_6$  and execute superstep 6, compared to the blocking checkpointing in Fig. 1(a). Moreover, the system may write multiple checkpoints concurrently if it takes more time to write a single checkpoint than to execute a single superstep. Writing multiple checkpoints concurrently intensifies the resource contention, which further increases the time of superstep execution and checkpointing.



**Fig. 2.** Checkpoint queue

The overall execution time of unblocking checkpointing with resource contention may still be shorter than the one of blocking checkpointing. However, the delay of checkpointing has a negative impact on the execution time in case of failures. In addition, since the resource contention prolongs checkpointing, the checkpoint is more likely to be unavailable when failure happens, leading to longer recovery time.

### 3.2 Checkpoint Queuing

Instead of issuing the unblocking checkpoints without a limitation, we propose a queuing strategy to limit the number of concurrent checkpoints as a user-defined parameter  $k$ , so as to balance the trade-off between maximizing resource usage and alleviating resource contention [18]. The optimal choice of  $k$  is determined by the system configuration. In our queuing strategy, all the pending checkpoints are inserted into a checkpoint queue. When the number of checkpoints  $n$  that the system is writing is less than  $k$  allowed by our strategy, our queuing strategy follows a First-In-First-Out (FIFO) policy and takes the first  $k - n$  checkpoints in the queue to write concurrently.

Figure 2 depicts how the queuing strategy works during execution when  $k = 1$ . In Fig. 2, the queuing strategy inserts the pending checkpoints  $C_4$ ,  $C_6$  and  $C_8$  into the queue while writing checkpoint  $C_2$ . Once the system writes the checkpoint  $C_2$  completely, as shown in Fig. 2, the system takes the  $C_4$  in the queue and starts writing this checkpoint. Similarly, the queuing strategy inserts the checkpoint  $C_{10}$  into the queue when the system reaches superstep 10.

Algorithm 1 illustrates the implementation details of the queuing strategy. Before executing the first superstep, the system registers a handler for the event which is triggered when a checkpoint completes (line 2). During the execution of supersteps, when the superstep  $i$  meets the checkpoint interval  $\tau$ , the system inserts the checkpoint  $C_i$  into the queue  $Q$  and calls the function CHECKPOINTING (line 4 to line 5). This function gets the number of checkpoints being written, i.e.,  $n$  (line 11). When  $n < k$ , the queuing strategy removes the first  $k - n$  checkpoints in the queue and temporarily saves these checkpoints in list  $L$  (line 12 to 14). Finally, the system writes the checkpoints stored in  $L$  (line 15). Moreover, to avoid the circumstance that there exists checkpoints in the queue  $Q$  while less than  $k$  checkpoints are being written, the system triggers the forementioned event and calls the function CHECKPOINTING when it writes a checkpoint completely.

---

**Algorithm 1.** Queuing Strategy

---

```

1: initial queue  $Q$ , list  $L$ , counter  $c$ ;
2: register event handler CHECKPOINTING;
3: while there are active vertices do
4:   if superstep  $i \bmod \tau = 0$  then
5:     insert  $C_i$  into  $Q$ ;
6:     CHECKPOINTING( $Q$ );
7:   end if
8: end while
9:
10: function CHECKPOINTING(queue  $Q$ )
11:    $n \leftarrow$  calculate the number of checkpoints being written;
12:   if  $n < k$  then
13:      $Q \leftarrow$  SKIPPING( $Q$ );
14:      $L \leftarrow$  first  $k - n$  checkpoints from  $Q$ ;
15:     write the checkpoints in  $L$ ;
16:   end if
17: end function
18:
19: function SKIPPING(queue  $Q$ )
20:   return  $Q$ ;
21: end function

```

---

## 4 Skipping Policy

In this section, we illustrate the staleness and tardiness of checkpoints in the queue. Then, we design the staleness/tardiness-aware skipping policy to improve the queuing strategy.

### 4.1 Checkpoint Staleness

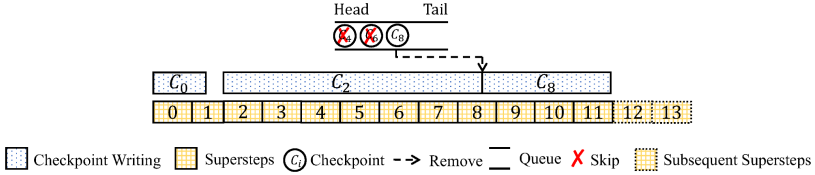
The checkpoints in Pregel-like systems store the vertices, edges as well as messages at a certain superstep into external storage. Interestingly, the checkpoint in a later superstep is more useful for failure recovery than the one in an earlier superstep, since recovering from a later superstep enables the system to recompute less supersteps once failure happens. For example, in Fig. 2, compared to the checkpoint  $C_4$  at superstep 4, the checkpoint  $C_8$  at superstep 8 avoids the recomputation from superstep 4 to superstep 7 once the failure happens. To evaluate this property, we illustrate checkpoint staleness in Definition 1.

**Definition 1.** (*Checkpoint Staleness*). For a checkpoint  $C_i$  at superstep  $i$ , the staleness  $S(C_i)$  of this checkpoint is the reciprocal of  $i$ , i.e.,  $S(C_i) = 1/i$ .

The smaller the value of  $S(C_i)$  is, the later the superstep  $i$  is. Consequently, the smaller staleness value indicates a more useful checkpoint for failure recovery. The queuing strategy in Sect. 3 takes the checkpoints from the head of the queue. Once failure happens, the system has to roll back to an earlier superstep which



requires more recovery time, since these checkpoints at the head of the queue have larger staleness values. Hence, the system should take out  $k$  checkpoints with smaller staleness values, so as to take less time to recover in case of failure. In Example 1, the system may avoid the recomputation from superstep 4 to superstep 7 if it takes out  $C_8$  instead of  $C_4$  and  $C_6$ .



**Fig. 3.** Checkpoint staleness

*Example 1.* Figure 3 illustrates the process of checkpointing considering the checkpoint staleness. For simplicity, we assume  $k = 1$ . Clearly, we have  $S(C_8) < S(C_6) < S(C_4)$ . Hence, after finishing writing  $C_2$ , the system will skip both  $C_4$  and  $C_6$ , and then start writing  $C_8$ . In case of failure happens at superstep 13, the system recomputes from superstep 8 rather than superstep 4.

## 4.2 Checkpoint Tardiness

In addition to the staleness, we observe that the checkpoint with a shorter writing time is more useful for failure recovery than the one with a longer writing time. As shown in Fig. 2 and 3, the checkpoint  $C_4$  has a shorter writing time than  $C_8$ . Clearly, the checkpoint  $C_4$  is more useful than  $C_8$  when failure happens at the superstep 11. In this case,  $C_8$  cannot be used for recovery, since the system has not completed  $C_8$  yet. Instead, the system rolls back to superstep 4, since  $C_4$  has been finished at superstep 10. To evaluate this property, we define checkpoint tardiness in Definition 2.

**Definition 2.** (*Checkpoint Tardiness*). For a checkpoint  $C_i$  at superstep  $i$ , the tardiness  $T(C_i)$  of this checkpoint is denoted by  $T(C_i) = t_i$ , where  $t_i$  is the time consumed on writing  $C_i$ .

The larger the value of  $T(C_i)$  is, the more time to write  $C_i$  the system takes. Hence, it is possible that the checkpoint  $C_i$  is unavailable when failure occurs, so that the system has to roll back to an earlier superstep. Considering the checkpoint tardiness, the system should pick up checkpoints with a smaller tardiness values, in order to spend less time to recover once failure happens. As described in Example 2, the system owns a complete checkpoint earlier if it picks up  $C_4$  rather than  $C_8$ .

*Example 2.* Following Example 1, we take Fig. 4 as an example to illustrate the checkpoint tardiness. Here, we suppose  $T(C_4) = T(C_6) < T(C_8)$ . Different from Example 1, the system should write either  $C_4$  or  $C_6$  when considering the checkpoint tardiness. As shown in Fig. 4, the system picks up checkpoint  $C_4$  and finishes

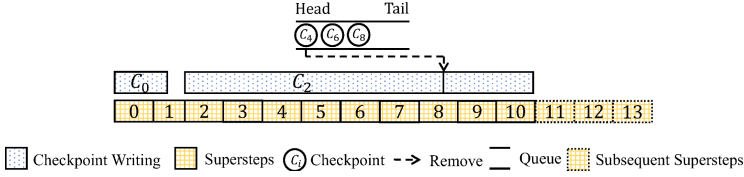


Fig. 4. Checkpoint tardiness

writing this checkpoint at superstep 10. In case of failure happens at superstep 11, the system would recompute from superstep 4, since it has a complete checkpoint  $C_4$ . However, if the system chooses to write  $C_8$ , this checkpoint has not been completed writing yet. Hence, the system has to recompute from superstep 2.

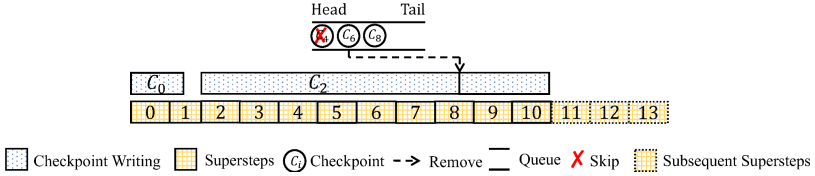
In general, the tardiness value  $T(C_i)$  of the checkpoint  $C_i$  depends on the checkpoint size. The tardiness value increases, along with the increasing of the checkpoint size. Hence, we employ the checkpoint size to indicate the length of  $T(C_i)$ . In Pregel-like systems, the checkpoint consists of the vertices, edges and messages. Hence, we calculate the size of the checkpoint  $C_i$  by summing the size of vertices  $V_i$ , edges  $E_i$  and messages  $M_i$  at superstep  $i$ . Consequently, the tardiness value  $T(C_i)$  is estimated by the checkpoint size, i.e.,  $T(C_i) \approx Size(V_i) + Size(E_i) + Size(M_i)$ .

### 4.3 Staleness/Tardiness-Aware Skipping

Clearly, the queuing strategy with a FIFO policy is oblivious to the checkpoint staleness and tardiness. In particular, it is unnecessary to write the checkpoint at the head of the queue with a large staleness and tardiness value. Alternatively, the system should skip such kind of checkpoints. Here, we propose a staleness/tardiness-aware skipping policy to replace the FIFO policy.

Ideally, the checkpoint with both smaller staleness and tardiness values enables the system to reduce the recovery time. However, it is not always true that checkpoint with a small staleness value also has a small tardiness value. As in Example 2, compared to the checkpoint  $C_4$  with a staleness value of  $1/4$ , the checkpoint  $C_8$  with a staleness value of  $1/8$  has a larger tardiness value. Intuitively, once the system completes a checkpoint, it can be utilized for failure recovery. Hence, we prefer to write a checkpoint with a small tardiness value, even though its staleness value is large. Instead of FIFO policy, our staleness/tardiness-aware skipping policy first sorts the checkpoints in the queue by tardiness value and then applies a secondary sort on the staleness value. After that, it issues writing the first  $k - n$  checkpoints in the queue. Example 3 illustrates how the skipping policy works during execution.

*Example 3.* Following Example 1 and 2, the system should pick up checkpoint  $C_6$  when both checkpoint staleness and tardiness are considered, since  $T(C_4) = T(C_6) < T(C_8)$  and  $S(C_8) < S(C_6) < S(C_4)$ . As shown in Fig. 5, the system



**Fig. 5.** Staleness/tardiness-aware skipping policy

skips  $C_4$  and picks up  $C_6$  to write. Compared to the system picking up  $C_4$  or  $C_8$ , the system picking up  $C_6$  takes the least time to recover when failure happens at superstep 11.

Algorithm 2 describes the implementation details of the skipping policy, which replaces the SKIPPING function in Algorithm 1. The skipping policy estimates the tardiness value  $T(C_i)$  (line 2) and calculates the staleness value  $S(C_i)$  of checkpoint  $C_i$  (line 3). The checkpoint queue is sorted by a composite key with two attributes  $T(C_i)$  and  $S(C_i)$ . The sorting employs  $T(C_i)$  as the primary attribute for comparing checkpoints and  $S(C_i)$  as the secondary attribute for comparing checkpoints with the same primary attribute (line 4). Finally, this policy returns the queue  $Q$  (line 5).

---

#### Algorithm 2. Skipping Policy

---

```

1: function SKIPPING(queue  $Q$ )
2:    $T(C_i) \leftarrow Size(V_i) + Size(E_i) + Size(M_i)$ ;
3:    $S(C_i) \leftarrow 1/i$ ;
4:   sort  $Q$  by a composite key ( $T(C_i)$ ,  $S(C_i)$ );
5:   return  $Q$ ;
6: end function

```

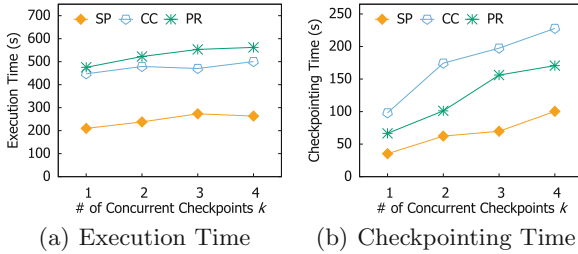
---

## 5 Experimental Studies

We implemented unblocking checkpointing [7, 9] as well as our proposed queuing strategy and skipping policy in Giraph. This section reports the efficiency of our two contributions under failure-free and failure cases, respectively.

### 5.1 Experimental Setting

**Cluster Setup.** In order to run Giraph, we deploy Hadoop 2.5.1 on a seven-node cluster. Each node has  $2 \times 4$ -core Intel Xeon E5606 CPUs, 100GB memory, a 2TB HDD and 1 Gbps Ethernet. In addition, we deploy Zookeeper 3.5.5 on this cluster for Giraph’s master election and barrier synchronization. By default, we limit the number of map tasks executing on each node to one and allocate 76GB memory for each map task. Among these map tasks, one is the master of Giraph and the others are the workers of Giraph.



**Fig. 6.** The impact of # of concurrent checkpoints (i.e.,  $k$ ) on queuing strategy

**Workloads.** We choose the Single Source *Shortest Path*, *Connected Components* and *PageRank* algorithms for our experiments, since they are widely adopted to evaluate the performance of graph processing systems [4, 16, 17]. For simplicity, in the rest of this paper, we refer to the Single Source *Shortest Path*, *Connected Components* and *PageRank* algorithms as *SP*, *CC* and *PR*, respectively. Moreover, we conducted these algorithms over two real-life social network graphs, Twitter<sup>1</sup> and Friendster<sup>2</sup>, of million-scale and billion-scale edges respectively.

**Baselines.** In our experiments, we employ the default blocking checkpointing approach in Giraph and the unblocking checkpointing implemented in Giraph as two baselines. In comparison to these baselines, we evaluate the efficiency of the queuing strategy and staleness/tardiness-aware skipping policy. In the following, to simplify the presentation, we denote the unblocking checkpointing using queuing strategy with FIFO policy as *queuing strategy*. Similarly, we denote the unblocking checkpointing with both the queuing strategy and staleness/tardiness-aware skipping policy as *skipping policy*.

## 5.2 Efficiency of Queuing Strategy

In this group of experiments, we evaluate the impact of the number of concurrent checkpoints (i.e.,  $k$ ) on queuing strategy, and then compare the performance of queuing strategy against blocking and unblocking checkpointing.

**Impact of  $k$ .** To evaluate the impact of  $k$  on the performance of queuing strategy, we run three algorithms on the Twitter dataset. Figure 6 illustrates the execution time of the system and the checkpointing time. Here, we set the checkpoint interval to one and vary the value of  $k$  from one to four. As shown in Fig. 6(a), when the value of  $k$  increases, the execution time of the system using the queuing strategy increases. Moreover, in Fig. 6(b), the checkpointing time follows the trend of execution time. Clearly, the queuing strategy achieves the best performance when the  $k = 1$ . Hence, we set  $k = 1$  in subsequent experiments.

<sup>1</sup> <https://networkrepository.com/soc-twitter.php>.

<sup>2</sup> <https://snap.stanford.edu/data/com-Friendster.html>.

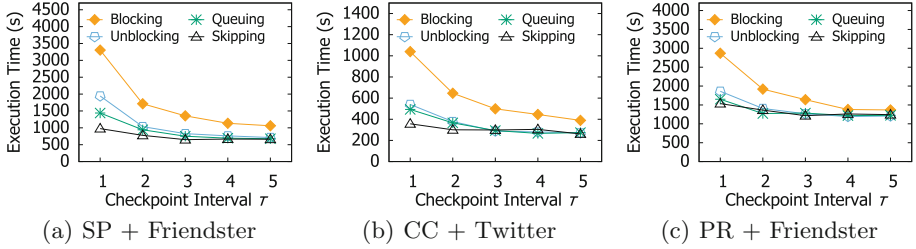


Fig. 7. Execution time without failure

**Failure-free Cases.** The change of the checkpoint interval  $\tau$  has an impact on the number of concurrent checkpointing, and therefore decides the performance of the queuing strategy. To evaluate the performance of queuing strategy, Fig. 7 shows the total execution time in different  $\tau$ . We vary  $\tau$  from one to five. As shown in Fig. 7, the queuing strategy outperforms the blocking and the unblocking checkpointing. For example, in Fig. 7(a), when  $\tau = 1$ , the queuing strategy decreases the overhead by 56.4% on execution time compared to the blocking checkpointing. This is because the blocking checkpointing pauses the execution of supersteps to write the checkpoints, which incurs significant overhead of execution time. Likewise, compared to the unblocking checkpointing, the queuing strategy reduces the overhead by 25.9%, since it alleviates the resource contention by setting  $k = 1$ .

As  $\tau$  increases, benefits of the queuing strategy on execution time decrease as against the blocking checkpointing and unblocking checkpointing. However, the queuing strategy performs at least as well as these two checkpointing approaches. As shown in Fig. 7(b), when  $\tau$  increases from one to five, the overhead reduced by queuing strategy decreases from 52% to 29.5% compared to the blocking checkpointing. The reason is that the increasing of  $\tau$  reduces the number of checkpoints, and the overhead of a blocking checkpoint on execution time is higher than that of an unblocking checkpoint. Compared to the unblocking checkpointing, the execution time of queuing strategy is shorter when  $\tau < 3$ . However, when  $\tau \geq 3$ , the queuing strategy has a similar execution time as the unblocking checkpointing. This is because the system with large  $\tau$  does not write multiple checkpoints concurrently, so that the queuing strategy is ineffective. In other words, as  $\tau$  increases, the queuing strategy eventually degenerates to unblocking checkpointing.

**Failure Cases.** Next, we study the performance of queuing strategy under failure cases. Figure 8 depicts the overall execution time. Here, we set  $\tau = 1$  and vary the failed superstep from 8 to 12. As shown in Fig. 8, no matter in which superstep failure happens, the queuing strategy outperforms the blocking checkpointing and the unblocking checkpointing. As an example in Fig. 8(a), when failure happens at superstep 8, the queuing strategy decreases the overhead by 48.1% as against the blocking checkpointing. Similarly, in comparison to the unblocking checkpointing, the queuing strategy decreases the execution time by 41.5%. In addition, the unblocking checkpointing in Fig. 8(b) fails to recover

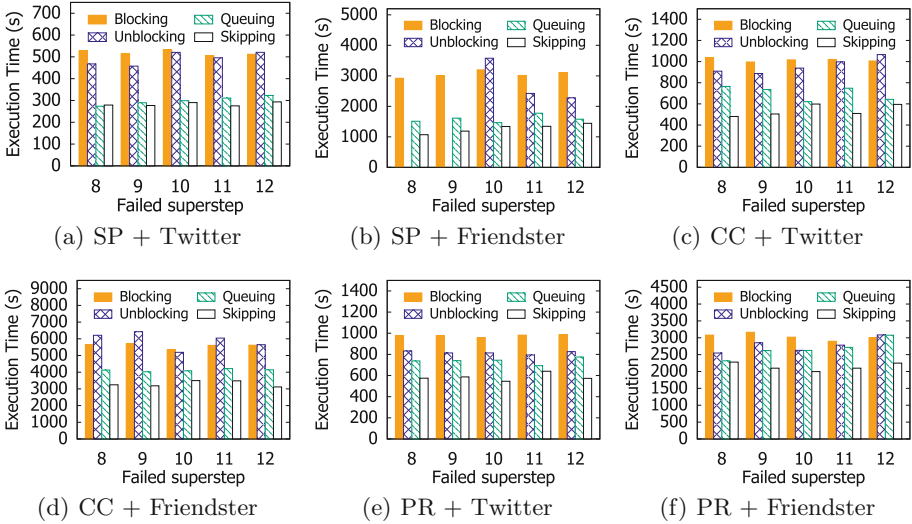


Fig. 8. Execution time with failure ( $\tau = 1$ )

when failure happens at superstep 8 or superstep 9, since the system has not finished writing any checkpoints.

In summary, by alleviating the resource contention, the queuing strategy effectively reduces the overall execution time and the checkpointing time, so as to achieve better performance than the blocking and unblocking checkpointing. Moreover, the queuing strategy has at least the same performance as the blocking and unblocking checkpointing even if the queuing strategy loses effect.

### 5.3 Efficiency of Skipping Policy

In this section, we evaluate the performance of the skipping policy against the other checkpointing approaches under failure-free and failure cases. To further illustrate the efficiency of skipping policy under failure cases, we discuss the recovery time of these checkpointing approaches. Moreover, we also analyze the impact of the checkpoint interval on the performance of these checkpointing approaches under failure cases.

**Failure-free Cases.** As shown in Fig. 7, the skipping policy outperforms other checkpointing approaches. As an example in Fig. 7(a), the skipping policy decreases the execution time by 32% compared to the queuing strategy when  $\tau = 1$ , since it picks up the checkpoints with smaller tardiness values instead of the checkpoints with larger tardiness values, which reduces the time of resource contention.

Similar to queuing strategy, as  $\tau$  increases, the benefits of skipping policy on execution time decreases as against the blocking checkpointing and unblocking checkpointing. Moreover, both of them eventually achieve the same performance. For example, in Fig. 7(b), when  $\tau \leq 3$ , the skipping policy decreases the execution

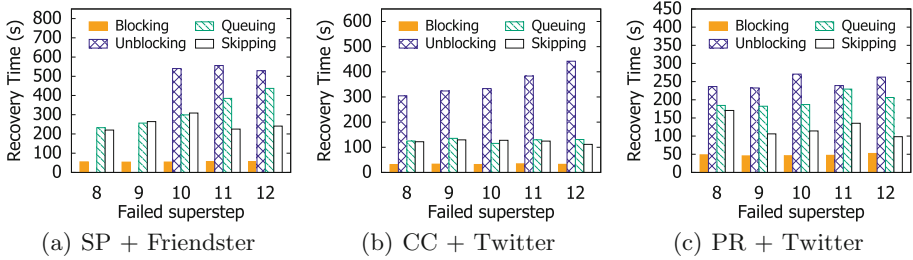


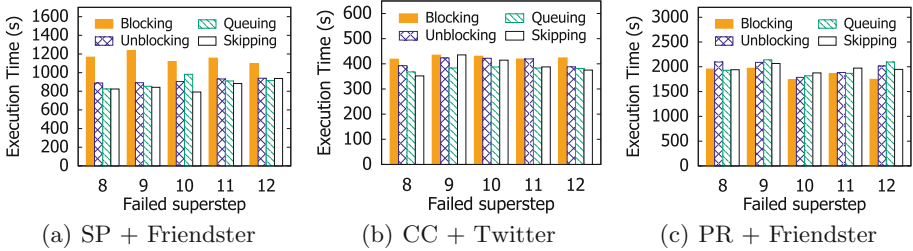
Fig. 9. Recovery time ( $\tau = 1$ )

time by up to 27.6% as against queuing strategy. Then, the execution time of skipping policy is almost the same as the queuing strategy when  $\tau > 3$ . The reason is that both the skipping policy and the queuing strategy eventually degenerate to the unblocking checkpointing.

**Failure Cases.** As shown in Fig. 8, in most failure cases, the skipping policy always outperforms other checkpointing approaches, and achieves the best performance. For example, in Fig. 8(c), when failure happens at superstep 8, the skipping policy decreases the overhead by 37.1% compared to the queuing strategy. The skipping policy allows writing checkpoints with smaller tardiness values which have an impact on the recovery time. Next, in order to further evaluate the performance of these checkpointing approaches, we analyze recovery time of them in failure cases. Moreover, we study the impact of checkpoint interval  $\tau$  on the performance of the skipping policy under failure cases, since the experiments in Fig. 8 only consider  $\tau = 1$ .

**Recovery Time.** Figure 9 provides the recovery time across checkpointing approaches when  $\tau = 1$ . Due to space limitation, we take *SP* on Friendster dataset, *CC* on Twitter dataset and *PR* on Friendster dataset as examples. However, similar results hold in other cases. In addition, Fig. 9(a) does not include the results of unblocking checkpointing, because there are no available checkpoints to recover for unblocking checkpointing. Clearly, the blocking checkpointing achieves the lowest overhead as against other checkpointing approaches, since it rolls back to the latest superstep. However, since the blocking checkpointing incurs an additional execution time overhead, the performance of queuing strategy and skipping policy is still better than that of the blocking checkpointing.

In addition, the queuing strategy obtains a lower recovery overhead than the unblocking checkpointing. For example, in Fig. 9(c), when failure happens at superstep 12, the queuing strategy decreases the overhead by 21.3% as against the unblocking checkpointing. The reason is because the queuing strategy speeds up the writing of checkpoints so as to obtain a complete checkpoint at a later superstep. Besides, the skipping policy recovers faster than the queuing strategy. Compared to the queuing strategy, the skipping policy decreases the overhead by up to 52.3%. The reason is that the skipping policy avoids writing the checkpoints with larger staleness and tardiness values, which allows the system to have an



**Fig. 10.** Execution time with failure ( $\tau = 5$ )

available checkpoint at a later superstep. Therefore, the skipping policy achieves better performance than the unblocking checkpointing and the queuing strategy.

**Checkpoint Interval.** Similarly, Fig. 10 takes  $\tau = 5$  as an example to illustrate the execution time of these checkpointing approaches. Clearly, the skipping policy achieves similar performance to the unblocking and queuing strategy. Moreover, the skipping policy outperforms the blocking checkpointing. As an example in Fig. 10(a), no matter in which superstep failure happens, the execution time of skipping policy is similar to that of the unblocking checkpointing and the queuing strategy. The reason is that the system does not write multiple checkpoints concurrently when  $\tau = 5$ , making the queuing strategy and the skipping policy ineffective. In other words, the skipping policy and queuing strategy degenerates to the unblocking checkpointing when  $\tau = 5$ . In addition, when failure happens at superstep 8, the skipping policy saves up to 29.2% of execution time as against the blocking checkpointing, since the blocking checkpointing pauses the execution of supersteps while writing the checkpoints.

Generally, as a complement to the queuing strategy, the skipping policy effectively avoids writing the checkpoint with a larger staleness value and tardiness value, so as to decrease the execution time and minimize the rollback under failure cases. Moreover, even if the skipping policy loses effect, it is still no worse than the blocking and the unblocking checkpointing as well as the queuing strategy. Consequently, the skipping policy achieves the best performance.

## 6 Related Work

There are many studies of fault tolerance in distributed graph processing systems. The survey in [8] summarizes the techniques as checkpointing and rollback mechanism, and message logging as well as replication.

Many Pregel-like systems such as Giraph [1] and Hama [2] employ checkpointing and rollback mechanism to achieve fault tolerance. These systems usually adopt a blocking manner to write the checkpointing, while our work focus on unblocking checkpointing. To decrease the overhead of blocking checkpointing, the work in [15] proposes unblocking checkpointing for graph processing on Flink [3]. This work focuses on the dataflow systems, whereas our work aims



to address the issues of unblocking checkpointing on the Pregel-like systems. Lightweight checkpointing [16] removes messages from the checkpoint and generates messages from checkpointed vertex states during recovery. Moreover, it avoids to restore the same edges as in the previous checkpoint. Nevertheless, it focus on decreasing the checkpointing overhead by reducing the data volume of checkpoint and supplements our work.

By logging messages, Pregel [10] confines the recovery to the failed workers, so as to accelerate the recovery. This technique is complementary to our proposal. The replication-based technique [14] maintains the replicas of each vertex during normal execution and recovers the lost vertex states from the replica once failure. To reduce the overhead cost on checkpointing, the replication-based technique employs the replicas instead of the checkpoints, whereas our work improves the unblocking checkpointing.

## 7 Conclusions

In this paper, we present a queuing strategy with a FIFO policy to alleviate the resource contention incurred by unblocking checkpointing, so as to reduce the overall execution time and the writing time of checkpoints. Moreover, we exploit the checkpoint characteristics in the Pregel-like system to improve the queuing strategy. To utilize the characteristics, we first define checkpoint staleness and checkpoint tardiness, and then propose staleness/tardiness-aware skipping policy to replace FIFO policy. Our experiments illustrate that the unblocking checkpointing with queuing strategy and staleness/tardiness-aware skipping policy achieves better performance than the blocking and unblocking checkpointing. Presently, we have implemented the queuing strategy and skipping policy by extending Giraph. However, the queuing strategy and skipping policy are also fit for other Pregel-like systems such as Pregel+ and Sedge.

**Acknowledgment.** This work was supported by the National Natural Science Foundation of China (No. 61902128), Shanghai Sailing Program (No. 19YF1414200).

## References

1. Apache giraph. <https://giraph.apache.org/>
2. Apache hama. <https://hama.apache.org/>
3. Carbone, P., et al.: Apache flink<sup>TM</sup>: stream and batch processing in a single engine. *IEEE Data Eng. Bull.* **36**, 28–38 (2015)
4. Cheng, Y., et al.: Which category is better: benchmarking relational and graph database management systems. *Data Sci. Eng.* **4**(4), 309–322 (2019)
5. Coti, C., et al.: Blocking vs. non-blocking coordinated checkpointing for large-scale fault tolerant MPI. In: *SC*, p. 127 (2006)
6. Dean, J., Ghemawat, S.: Mapreduce: simplified data processing on large clusters. In: *OSDI*, pp. 137–150 (2004)
7. Gonzalez, J.E., et al.: Powergraph: distributed graph-parallel computation on natural graphs. In: *OSDI*, pp. 17–30 (2012)

8. Heidari, S., et al.: Scalable graph processing frameworks: a taxonomy and open challenges. *ACM Comput. Surv.* **51**(3), 60:1-60:53 (2018)
9. Low, Y., et al.: Distributed graphlab: a framework for machine learning in the cloud. *PVLDB* **5**(8), 716–727 (2012)
10. Malewicz, G., et al.: Pregel: a system for large-scale graph processing. In: *SIGMOD*, pp. 135–146 (2010)
11. McCune, R.R., et al.: Thinking like a vertex: a survey of vertex-centric frameworks for large-scale distributed graph processing. *ACM Comput. Surv.* **48**(2), 25:1-25:39 (2015)
12. Pundir, M., et al.: Zorro: zero-cost reactive failure recovery in distributed graph processing. In: *SoCC*, pp. 195–208 (2015)
13. Vora, K., et al.: Coral: confined recovery in distributed asynchronous graph processing. In: *ASPLOS*, pp. 223–236 (2017)
14. Wang, P., et al.: Replication-based fault-tolerance for large-scale graph processing. In: *DSN*, pp. 562–573 (2014)
15. Xu, C., et al.: Efficient fault-tolerance for iterative graph processing on distributed dataflow systems. In: *ICDE*, pp. 613–624 (2016)
16. Yan, D., et al.: Lightweight fault tolerance in pregel-like systems. In: *ICPP*, pp. 69:1–69:10 (2019)
17. Yang, S., et al.: Towards effective partition management for large graphs. In: *SIGMOD*, pp. 517–528. *ACM* (2012)
18. Yildirim, E., et al.: Prediction of optimal parallelism level in wide area data transfers. *IEEE Trans. Parallel Distrib. Syst.* **22**(12), 2033–2045 (2011)



# A Low-Latency Metadata Service for Geo-Distributed File Systems

Chuangwei Lin<sup>1</sup>, Bowen Liu<sup>1</sup>, Wei Zhou<sup>1</sup>, Yueyue Xu<sup>1</sup>, Xuyun Zhang<sup>2</sup>(✉),  
and Wanchun Dou<sup>1</sup>(✉)

<sup>1</sup> State Key Laboratory for Novel Software Technology,  
Nanjing University, Nanjing, China  
{lcw, liubw, zw, xuyuey}@smail.nju.edu.cn,  
douwc@nju.edu.cn

<sup>2</sup> Department of Computing, Macquarie University, Sydney, Australia  
xuyun.zhang@mq.edu.au

**Abstract.** Geo-distributed file systems have been widely used by web services. An increasing number of time-critical web applications have been deployed on the cloud across geographical regions. In this circumstance, intolerant service latency will occur when user accesses remote servers. In view of this challenge, We design a metadata service which aims at reducing the service invocation latency. This low-latency metadata service is named LoLaMS. Taking advantage of the latency-aware dynamic subtree partition and migration, LoLaMS is capable to handle more metadata service invocations in the nearby metadata server. On account of that, the expected latency could be satisfied by LoLaMS. We implemented the LoLaMS and deployed it in a real-word cloud environment across different regions. The experimental results show that LoLaMS reduces the network latency effectively while ensuring high metadata consistency.

**Keywords:** Distributed file system · Low latency · Metadata service · Wide area storage

## 1 Introduction

With the development of mobile edge computing, industrial Internet, Internet of vehicles and other technologies, there are more and more applications deployed and used across geographical regions. For users, timely data loading can improve their experience. For some applications, such as accessing high-precision maps in navigation systems, resource loading in VR/AR applications, data access latency is the key to user experience. These application scenarios are latency-sensitive. Users and servers are distributed in different regions far away. However, most previous designs for cloud storage services cannot provide users with low enough access latency.

Distributed file system, as a kind of distributed storage service with strong universality, has been widely used in cloud storage. More than 50% of the operations on a file system are metadata operations [12]. Therefore, low latency metadata access is an important part of a low latency distributed file system.

In most modern distributed file systems, a dedicated metadata service is designed to manage metadata. There are two metadata management mechanisms in general: a centralized mechanism and a decentralized mechanism. [25] The centralized metadata management, including PVFS [13] GFS [6] and HDFS [14] use a single metadata server (MDS) to manage metadata of all files and directories. The decentralized metadata management, including CephFS [17] and MRFS [23] use a group of MDS to manager metadata. In order to manage metadata in geo-distributed storage system, Granary [22] and IPFS [2] use Distributed Hash Table (DHT) to locate metadata.

However, most of the distributed file systems such as GFS [6], HDFS [14], Lustre [3], CephFS [17], IPFS [2] are not optimized for latency caused by geographic distance [15]. In order to reduce the latency of metadata access operation in geo-distributed file systems, this paper proposes a low-latency metadata service for geo-distributed file systems—LoLaMS.

LoLaMS is a distributed metadata service that provides strong consistency (sequential consistency). For a general-purpose file system, strong consistency is more appropriate because it can adopt most kinds of applications. This makes it easier for existing applications to use file systems based on LoLaMS without additional work on consistency issues for developers.

LoLaMS is a wide area networked metadata service which runs in data centers distributed in different geographical regions. In order to reduce access latency, LoLaMS takes advantage of the locality of data access by users. The data access behavior of many applications is characterized by geographic locality. In another word, users in a same area prefer to access similar data. For example, navigation system of vehicle only queries high-precision map of its nearby area rather than other remote areas. In file system, data is organized in a directory tree structure. Relevant data is often placed in the same or adjacent directory. LoLaMS dynamically divides the file system directory tree into subtrees based on the analysis of user’s operation behavior. Each subtree is managed by a metadata server. The subtree is placed in a specific position so that as many future incoming access requests to the subtree as possible with less access latency.

We developed a prototype of LoLaMS and deployed it across 8 regions of the world. We tested the performance of LoLaMS and compared it with HDFS by using a real dataset. The results of the experiment show that in LoLaMS, the latency of 78% write operates is less than 50 ms, which is  $3.36\times$  better than HDFS. The latency of 65.6% read operates is less than 50 ms, which is  $2.66\times$  better than HDFS.

The main contribution of this paper is threefold.

- Developed a prototype framework of geo-distributed file system. This framework enables metadata to be distributed across different MDSs and ensures strong consistency of operation. This framework enables metadata migration to proceed correctly.
- Designed a migration method used in LoLaMS. This method records users’ operation behavior, counts the latency between users and each node. When the latency of the user operation exceeds the pre-set threshold, the method

calculates the subtree to migrate out and the target MDS to migrate to, then migrates the metadata to the target MDS in a reliable manner.

- Developed a prototype of LoLaMS and deployed the prototype of LoLaMS across 8 regions of the world. Tested the performance of LoLaMS and compared it with HDFS by using a real dataset.

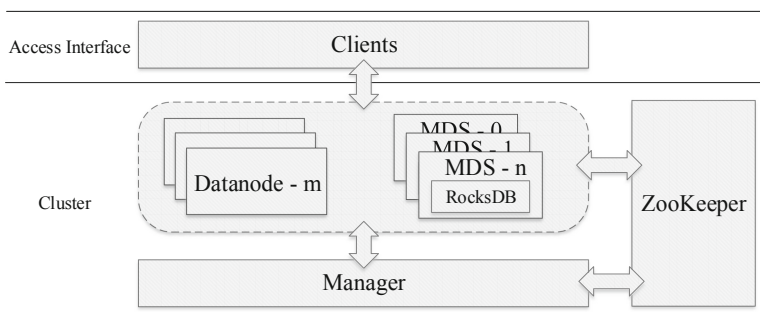
The remainder of this paper is organized as follows. Section 2 introduces the framework of a geo-distributed file system. The details of the proposed low-latency metadata service are introduced in Sect. 3. Section 4 evaluates the proposed metadata service experimentally. The related works are summarized in Sect. 5. Finally, the paper is concluded in Sect. 6.

## 2 The Geo-Distributed File System Framework

LoLaMS runs as a part of a geo-distributed file system. Servers of the system include 3 roles: *Manager*, *MDS* and *Datanode*. *MDS* and *Manager* provide metadata services to *Client*. Figure 1 depicts the framework of the geo-distributed file system.

**Manager.** A cluster of nodes (typically 3 nodes) run *Manager* program and ZooKeeper [7]. *Manager* is used to monitor the health of individual MDSs and deal with MDS failures. At a time, only one Manager node is active, and the remaining Manager nodes are standby nodes. When an active Manager node fails, ZooKeeper re-selects a standby node as the Active node. The active Manager node monitors the performance of individual MDSs. Manager senses and handles the failure when the MDS fails or goes offline. ZooKeeper ensures data consistency between manager nodes.

**Metadata Server.** MDSs are responsible for managing the metadata of the file system. MDS nodes are located in the data center or on the edge server. A client can access any metadata server to start using the LoLaMS. In LoLaMS, there are two types of metadata. The first type of metadata is *INode*, which describes the



**Fig. 1.** The geo-distributed file system framework

directory structure and file information. Each MDS only maintains and persists the inodes within the subtree managed by the MDS. LoLaMS uses RocksDB [5], an embeddable persistent key-value store for fast storage, to persist metadata on the disk. When creating or modifying a file, the client needs to write the inode on the MDS that manages the metadata of the file. When writing an inode, RocksDB first appends the operation to the log file on the disk, and then updates the inode data in the memory. The second type of metadata is *AuthTreeNode*, which describes how the file system directory tree is partitioned and each subtree managed by which MDS. For a path, by querying the *AuthTreeNode*, can know which MDS manages the directory or file. The *AuthTreeNode* is updated only when a migration operation is performed. Replicas of the *AuthTreeNode* are available on all MDSs and managers. The replicas on the manager are the primary replica, and updates to the primary replica of the *AuthTreeNode* are broadcast to all replicas via ZooKeeper, keeping the *AuthTreeNode* data consistent on the system. Dynamic subtree partition [18] divides file system directory tree into multiple subtrees, each of them is managed by different MDS. In LoLaMS, MDSs log client operations and gather latency information between clients and MDSs. If latency higher than the threshold, MDS will use the method described in Sect. 3 to re-partition subtree and migrate metadata.

**Datanode.** Datanodes store file data and perform operations on file data.

**Client.** *Client* provides a command line interface which support common file system operations, such as *mkdir* and *ls*, which enables users to access metadata via file path such as “/aaa/bbb/ccc”.

A client can connect to any known MDS to get a list of current online MDSs. The client measures the latency between itself and each MDS at regular intervals and reports latency information to all MDSs that have already connected to the client.

When a user operates on a path, the client first checks whether it has accessed the path before. If it has, the client directly accesses the MDS corresponding to this path before. If not, the client connects to the MDS nearest to the user and queries which MDS manages the directory or file that path represents.

### 3 A Low-Latency Metadata Service

The concept of latency in this article refers to the time elapsed from when the client sends a request to when the client receives a response. The premise of using the LoLaMS is that users’ access behaviour to metadata follows the principle of locality, including temporal locality and spatial locality. LoLaMS is a distributed service that runs across MDSs. When the MDS finds that the user operation latency exceeds the threshold, LoLaMS selects a directory subtree managed by this MDS and migrates it to the appropriate MDS, so as to make the user operation latency below the threshold as much as possible, and the target MDS is not overloaded. The goal of LoLaMS is to minimize the number of operations whose latency exceeds the threshold.

To formulate the goal of LoLaMS, we define

$$p_{latency} = \frac{\text{number of operations with latency} < T_t}{\text{number of operations}}$$

$T_t$  denotes latency threshold.  $p_{latency}$  denotes the proportion of operations whose latency is less than the latency threshold among all operations.

The goal of LoLaMS can be expressed as

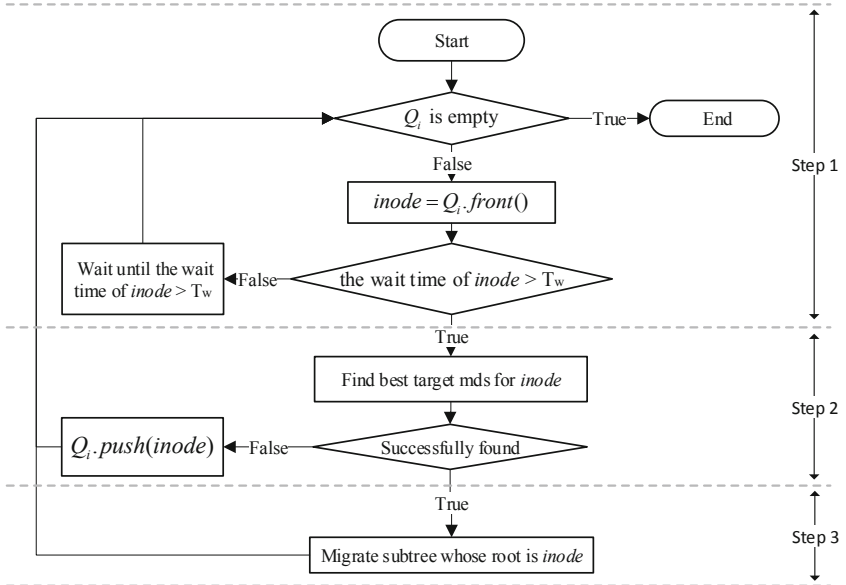
$$\text{Maximize}(p_{latency})$$

The optimal method to achieve this goal is migrating metadata of  $inode_x$  to MDS which is closest to client before the client access  $inode_x$ . However, it is impossible to accurately predict every operation in a real file system.

We have the experience from the storage system that if a user has accessed a path in the recent past, it is more likely that the user will access the path or its adjacent paths in the recent future. If a user has accessed a path frequently in the recent past, there is a high probability that the user will continue to visit the path frequently in the recent future.

LoLaMS takes advantage of the locality of user access to the file system. When timeout access to  $inode_i$  occurs, LoLaMS analyzes the operation behavior of users who visit  $inode_i$  in a period of time to determine the migration of  $inode_i$  and its subdirectories.

Figure 2 shows the flow chart of subtree partition and migration method in LoLaMS which is separated into 3 steps:



**Fig. 2.** The flow chart of subtree partition and migration method in LoLaMS.

**Step 1. Identify Inodes that Need to Be Optimized:** If an operation latency exceeds  $T_t$ , put the inodes that have been accessed in this operation into a queue. Inodes in the queue are taken out for optimization in enqueue order.

**Step 2. Find Best Target MDS for Inode:** Find the best MDS as a migration target for the taken out INode.

**Step 3. Migrate Subtree to Target MDS:** Partition and migrate a subtree from current directory trees managed by  $m_{ds}_i$  whose root is the INode which has been found best target MDS.

**Identify Inodes that Need to Be Optimized.**  $m_{ds}_i$  logs the operation of  $inode_x$  from  $client_c$  and instantly check  $L_c^i$ .  $L_c^i$  denotes network latency between  $client_c$  to  $m_{ds}_i$ . If  $L_c^i > T_t$ , push pair  $\langle inode_x, time_{now} \rangle$  into  $Q_i$ .  $Q_i$  denotes the queue of inodes that are waiting for optimization in  $m_{ds}_i$ .  $time_{now}$  denotes current time. INodes in  $Q_i$  are not optimized immediately when they are pushed into  $Q_i$ . A single timeout log may be accidental. To make better optimization, INodes in  $Q_i$  have to wait for enough time to gather more logs. For each  $inode_x$  in  $Q_i$  which has been waiting for enough time (denoted as  $T_w$ ), try to find the best target MDS for  $inode_x$ .

**Find Best Target MDS for Inode.** Find the best MDS as a migration target for selected INode. If failed, use breadth-first search to find the best target MDS for child nodes of the current selected INode.

Algorithm 1 describes the process to find the best target MDS for  $inode$ . Here,  $m_{ds}_c$  denotes current mds which running this algorithm.  $ep_{inode_t}(m_{ds}_i)$  means if  $inode_t$  is in  $m_{ds}_i$ , the estimate proportion of operations access  $inode_t$  whose latency  $< T_t$ .  $p_{latency}^{expect}$  denotes a threshold  $p_{latency}$  expected to be achieved.

We use  $ep_{inode_x}(m_{ds}_y)$  to measure whether  $m_{ds}_y$  can be a migration target of  $inode_x$ . It means estimate  $p_{latency}$  of  $inode_x$  if  $inode_x \in Set_{m_{ds}_y}$ .  $Set_{m_{ds}_y}$  means the set of INodes managed by  $m_{ds}_y$ .  $ep_{inode_x}(m_{ds}_y)$  reflects if  $inode_x$  is managed by  $m_{ds}_y$ , when clients repeat operations during  $[time_{now} - T_{tr}, time_{now}]$ , the proportion of operations whose latency is less than  $T_t$ . Here,  $T_{tr}$  means the trace-back time of operation logs. In a file system, users usually access files from top to bottom according to the directory hierarchy. For inodes in a subtree that use  $inode_x$  as root, clients normally need to access  $inode_x$  before access other inodes. Therefore,  $ep_{inode_x}(m_{ds}_y)$  is approximate to estimate  $p_{latency}$  of the subtree. If we find a suitable MDS as migrate target for  $inode_x$ , which improved  $p_{latency}$  of  $inode_x$ , then the estimate  $p_{latency}$  of the subtree is equivalently improved. If a suitable MDS cannot be found for  $inode_x$  as the migration target, a breadth-first search will be used to find a suitable migration target for other nodes in the subtree. If found a suitable migration target for  $inode_z$ , the  $inode_z$  will be the root node to divide a new subtree. This subtree will be migrated to the target MDS.



---

**Algorithm 1.** The process to find the best target MDS

---

**Require:**  $inode$ : The INode need to optimize;  $MDS\_list$ : The list of all mds;

**Ensure:**  $state$ : a boolean value represents whether found target mds or not;  $inode_t$ : the root of subtree to be partitioned;  $mds_t$ : the migrate target mds;

```

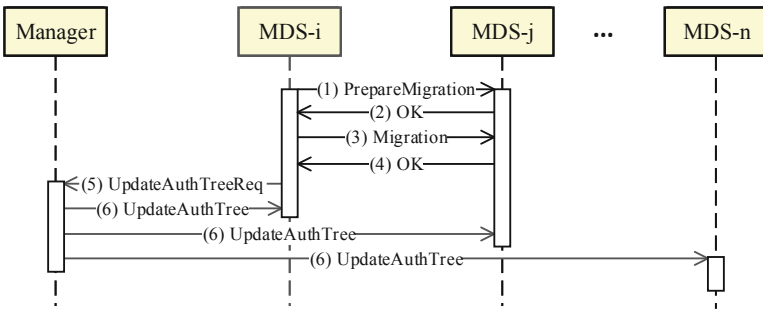
1:  $q.push(inode)$  //  $q$  denotes the queue used for breadth-first search.
2: while  $q$  NOT empty do
3:    $inode_t = q.pop()$ ;
4:   if  $time_{now} - time_{lo}^{inode_i}$  then
5:      $l =$  list of clients accessed  $inode_t$  during  $[time_{now} - T_{tr}, time_{now})$ 
6:      $count_{client_x} =$  the access time from  $client_x$  during  $[time_{now} - T_{tr}, time_{now})$ 
7:      $time_{lo}^{inode_t} = time_{now}$ 
8:     if  $l$  is empty then continue;
9:     for  $mds_i$  in  $MDS\_list$  do
10:       $in\_threshold\_req = 0$ ;
11:      for  $client_j \in l$  do
12:        if  $L_j^i < T_t$  then  $in\_threshold\_req += count_{client_j}$ 
13:       $ep_{inode_t}(mds_i) = in\_threshold\_req / \sum_{client_t \in l} count_{client_t}$ 
14:       $mds_t =$  the mds which has the largest  $ep_{inode_t}$ 
15:      if  $ep_{inode_t}(mds_t) > p_{latency}^{expect}$  then return (True,  $inode_t$ ,  $mds_t$ )
16:      else  $q.push(childnodes\ of\ inode_t)$ 
17: return False

```

---

**Migrate Subtree to Target MDS.** Partition a subtree from current directory trees managed by  $mds_i$  whose root is the INode which has been found best target MDS. Then migrate this subtree to target MDS.

The migration process is depicted in Fig. 3. If an MDS doesn't receive broadcast information due to failure, it will get the latest "UpdateAuthTree" message through zookeeper after it returns to normal. The MDS will fetch all missing updation transactions of *AuthTree* from *Manager*, then redo these transactions. This ensures the consistency of *AuthTree* in the system. Thus, at any given moment, an INode will only be managed by one MDS.



**Fig. 3.** Sequence diagram of migration.

If  $mds_i$  intent to migrate a subtree to  $mds_j$ , the migration process is consists of 6 phases.

1.  $mds_i$  send “PrepareMigration” message to  $mds_j$  to ask if  $mds_j$  has enough storage space.
2.  $mds_j$  send “OK” message to  $mds_i$  to indicate that  $mds_j$  is ready to receive the migration data. If  $mds_i$  does not receive any message from  $mds_j$ ,  $mds_i$  will find another migration target MDS.
3.  $mds_i$  send the metadata of the inodes in the subtree to  $mds_j$  through “Migration” message.
4. After  $mds_j$  receives the “Migration” message, it persists the data and then replies “OK” message to  $mds_i$ .
5.  $mds_i$  send “UpdateAuthTreeReq” to *Manager* to commit the migration. In *Manager*, The change of *AuthTree* is entered into the database as a transaction.
6. The *Manager* send message “UpdateAuthTree” to broadcast the change of *AuthTree*.

## 4 Experiment Evaluation

### 4.1 Experiment Settings

*Experiment Environment.* Table 1 shows the experiment environment. We evaluate LoLaMS using Aliyun’s elastic cloud service. We deploy 9 VMs in 8 different regions. These regions are: cn-beijing (China), cn-shenzhen (China), ap-northeast (Japan), ap-southeast (Australia), us-west (US), ap-south (India), eu-west (London), ap-southeast (Malaysia). Each VM has 2 virtual CPUs, 8 GiB memory and 50 GiB storage. For LoLaMS, a VM in cn-beijing region is used to deploy the Manager, 8 MDSs are deployed in 8 VMs each in a different region. For HDFS, a VM in cn-beijing region is used to deploy the Namenode, 8 Datanodes are deployed in 8 VMs each in a different region.

**Table 1.** Experiment environment.

Item	Description
Number of servers	9
Region of servers	2 in cn-beijing (China), 1 in cn-shenzhen (China), 1 in ap-northeast (Japan), 1 in ap-southeast (Australia), 1 in us-west (US), 1 in ap-south (India), 1 in eu-west (London), 1 in ap-southeast (Malaysia)
CPU	2 vCPU, 3.5GHz
Memory	8 GiB
Storage	50 GiB
Operation System	Ubuntu 18.04 × 64
Network bandwidth	100 Mbps

*Experiment Data.* In the experiment, we use an online shopping platform access log dataset [24] to test the read-write latency and throughput of LoLaMS and HDFS. We take the first 100,000 access records from the dataset for the evaluation. Each user in the dataset is mapped to a specific region. In the 8 regions, we replay the access records of the users mapped to this region in order at the same beginning time.

## 4.2 Experiment Comparison Analysis

*Latency Evaluation.* We measured the latency between these VMs. The round-trip time (RTT) between these regions within the range of 35 and 420 ms. The geographical distance between data centers will inevitably cause latency. In addition, there are other factors that can influence latency, such as network conditions. The average RTT between ap-south-1 and cn-shenzhen is 410 ms while the distance between them is 4280 km. The average RTT between ap-south-1 and us-west-1 is 249 ms while the distance between them is 13545 km. The geographical distance between ap-south-1 and us-west-1 is  $3\times$  that between ap-south-1 and cn-shenzhen, but the RTT between ap-south-1 and us-west-1 is  $0.6\times$  that between ap-south-1 and cn-shenzhen. Therefore, it is not credible to estimate the latency between two nodes according to the geographical location, which can only be used as the lower bound. There are other factors that can affect network latency, such as the number of hops in routing.

Typically, end-user devices (smartphones, tablets, laptops), as well as IoT devices, can reach an edge server with a rather low latency less than 10 ms [4]. The latency between users and clients in the same region usually no more than 50 ms. In the experiment, we set  $T_t$  in LoLaMS to 50 ms.

Table 2 shows the latency performance of LoLaMS and HDFS. The average write latency of LoLaMS is 38.14 ms, which is 16.75% of HDFS. The average read latency of LoLaMS is 35.98% of HDFS. In LoLaMS, the latency of 78% write operates is less than 50 ms, which is  $3.36\times$  better than HDFS. The latency of 65.6% read operates is less than 50 ms, which is  $2.66\times$  better than HDFS.

**Table 2.** Operation latency between LoLaMS and HDFS.

Metadata service	Average latency (ms)	Median latency (ms)	$p_{latency}$
LoLaMS (write)	38.14	5	0.780
HDFS (write)	227.69	153	0.179
LoLaMS (read)	61.17	9	0.656
HDFS (read)	170	148	0.179

Figure 4 shows the cumulative distribution function of operation latency in LoLaMS and HDFS. Figure 5 depicts the proportion of operation latency in different ranges. In LoLaMS, 67.2% of write operations are completed within 10 ms, and 54.3% of reading operations are completed within 10 ms. Compared

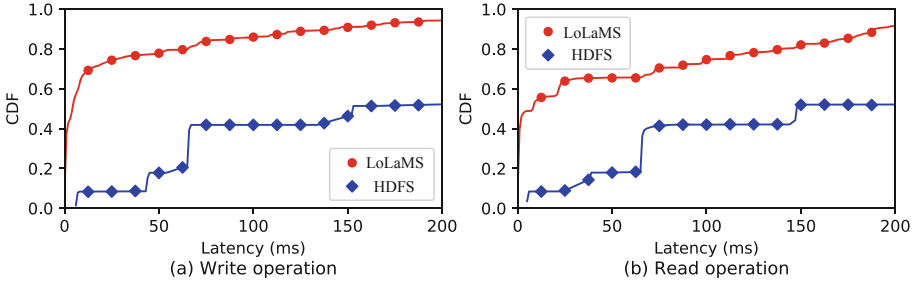


Fig. 4. Cumulative distribution function (CDF) of operation latency.

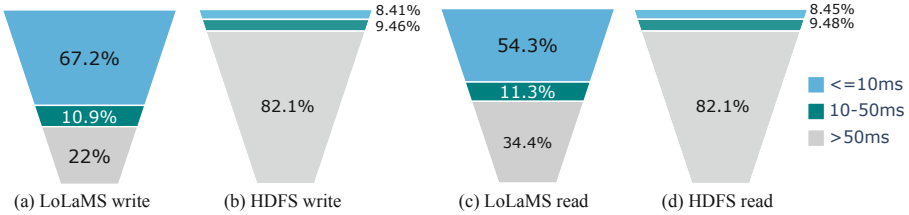


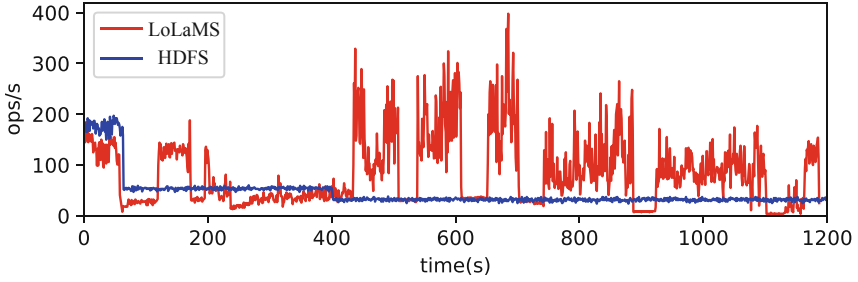
Fig. 5. The proportion of latency in different ranges.

with HDFS, LoLaMS can reduce the latency significantly. Most operations can be completed in a short time, which can make the application run faster and improve user experience.

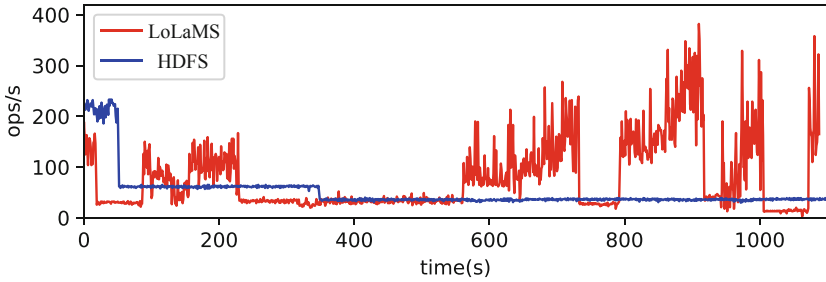
For metadata write operations from clients, both LoLaMS MDS and HDFS Namenode first append the operation to the log file on the disk. Then they update the data in the memory. Finally, they return a confirmation to the client. The latency of an operation comes from two parts: the processing time on the server and the transmission time on the network. In LoLaMS, since a large part of client operations can be completed in nearby MDS, the transmission time is significantly reduced. This makes LoLaMS have a lower latency than HDFS.

In the experiment, the average read operation latency of LoLaMS is higher than the average write operation latency. This is because the experiment replayed the dataset twice. The first replay executed the records as write operations and construct the directory structure. The second replay executed the records as read operations. Replaying the dataset for the second time breaks the temporal locality of the access record. Thus, fewer operations were executed on the nearby MDS during the second replay.

*Throughput Evaluation.* Previous experiments show that HDFS can reach 126100 ops/s when it performs read operations on Namenode [14]. However, in a geo-distributed environment, the bottleneck of system throughput is no longer the single machine performance, but the network performance. Figure 6 and Fig. 7 show the operation per second of write and read operation in LoLaMS and HDFS. LoLaMS reduces the latency significantly, which also improves the



**Fig. 6.** Operation per second (OPS) of write operation.



**Fig. 7.** Operation per second (OPS) of read operation.

throughput of the system. The maximum throughput of HDFS is 197 ops/s for write operations and 233 ops/s for read operations. The maximum throughput of LoLaMS is 389 ops/s for write operations and 382 ops/s for read operations. In terms of maximum read throughput, LoLaMS is 63.9% higher than HDFS. For maximum read throughput, LoLaMS is 97.4% higher than HDFS. This is due to LoLaMS's full use of multiple metadata servers and reasonable division of subtrees. The higher throughput also enables LoLaMS to complete the same number of read and write requests in a shorter time than HDFS.

### 4.3 Evaluation Result

From the experimental results of latency evaluation, LoLaMS performs significantly better than HDFS for the indicator  $p_{latency}$ . This shows that in LoLaMS, more operations can be completed with lower latency. According to the Throughput evaluation results, the Throughput of LoLaMS is higher than HDFS. This shows that LoLaMS can handle more operations in a shorter time. For file systems in geo-distributed scenarios, LoLaMS has advantages in terms of latency and throughput.

## 5 Related Works

Web services have made extensive use of distributed file systems, which can share storage for many users in the same namespace [8]. With the rise of cloud computing and online services, more and more applications with high timeliness are being deployed across borders. The latency induced by geographical distance, on the other hand, is difficult to overcome. In recent years, some researches have been done to address the latency associated with geo-graphical distribution.

Yu et al. [22] develop a distributed storage system called Granary that offers cyber users with a dependable data storage and sharing service. Granary stores file meta-data in a Distributed Hash Table layer and scatters big files using a raw data storage technique. Benet [2] design a peer-to-peer distributed file system InterPlanetary File System (IPFS) that aims to connect all computing devices to a single file system. IPFS is a content-addressed block storage architecture with content-addressed hyper connections that enables high throughput.

The above two studies used the method of storing metadata and file data nearby to reduce latency. In addition, it can also reduce system latency by caching metadata on the server side or on the client side. Yu et al. [23] proposed the Metadata Replication File System(MRFS) which split metadata into non-overlapping portions and kept on MDS, where the creation action is raised, whereas namespace and directory information is preserved in Namespace Servers in this system. Because it serves the majority of requests in local MDS, such a hierarchical architecture not only achieves great scalability but also delivers low latency. Oh et al. [9] present a geo-distributed cloud storage system called Wiera. Wiera provides first-class dynamic support owing to network, workload, and access pattern changes, and can actively handle dynamism at run-time. By externalizing the policy definition, Wiera allows unmodified programs to benefit from flexible data/storage rules. Ren et al. [11] develop SLOG that avoids the tradeoff for workloads which contain physical region locality in data access. In the access mode, each data particle is assigned a master region based on locality. Reading and writing to nearby data can be done quickly without cross-regional communication. Muhammed et al. [16] proposed an approach use erasure coding to significantly reduce costs while successfully mitigating the associated overheads in wide-area latency incurred for preserving consistency.

Existing work mainly focus on storing file data on the nearby server or caching metadata. While our research focus on reducing latency in metadata service without sacrificing consistency. In cloud computing and edge computing, workload migration is often used to improve the quality of experience (QoE) and the quality of service (QoS) [10, 19, 20]. While our idea is to improve QoS by migrating storage workload. In this paper, we propose a geo-distributed metadata service, LoLaMS, which uses the method of partition the directory tree dynamically and migrating subtree to reduce latency while ensuring the consistency of metadata. Virtualization technology has been widely used in cloud computing and edge computing [1, 21]. By combining virtualization technology, distributed file systems based on LoLaMS is expected to provide low-latency storage services for cloud computing and edge computing.

## 6 Conclusion

This paper introduces LoLaMS, a low-latency metadata service for geo-distributed file system. LoLaMS can provide metadata service with lower latency, without sacrificing strong consistency. The design of LoLaMS makes full use of the temporal locality and spatial locality of the user's access to the file system. LoLaMS senses the user's network latency and dynamically divides the file system directory tree into several subtrees according to the user's access behavior. In the running process of LoLaMS, the division of subtree is optimized to make as many user requests as possible be completed on the nearest MDS. The experimental results show that compared with other file system metadata service, LoLaMS can significantly reduce the access latency and improve the throughput of the whole system.

**Acknowledgements.** This work was supported in part by the National Key Research and Development Program of China under Grant No. 2020YFB1707601, the Key Research and Development Program of Jiangsu Province under Grant No. BE2019104. Dr Xuyun Zhang is the recipient of an ARC DECRA (project No. DE210101458) funded by the Australian Government.

## References

1. Alves, M.P., Delicato, F.C., Santos, I.L., Pires, P.F.: LW-CoEdge: a lightweight virtualization model and collaboration process for edge computing. *World Wide Web* **23**(2), 1127–1175 (2020)
2. Benet, J.: IPFS-content addressed, versioned, p2p file system. arXiv preprint [arXiv:1407.3561](https://arxiv.org/abs/1407.3561) (2014)
3. Braam, P.: The lustre storage architecture. CoRR abs/1903.01955 (2019). <http://arxiv.org/abs/1903.01955>
4. Confais, B., Lebre, A., Parrein, B.: Performance analysis of object store systems in a fog and edge computing infrastructure. In: Hameurlain, A., Küng, J., Wagner, R., Akbarinia, R., Pacitti, E. (eds.) *Transactions on Large-Scale Data- and Knowledge-Centered Systems XXXIII*. LNCS, vol. 10430, pp. 40–79. Springer, Heidelberg (2017). [https://doi.org/10.1007/978-3-662-55696-2\\_2](https://doi.org/10.1007/978-3-662-55696-2_2)
5. Facebook: Rocksdb. <http://rocksdb.org/>
6. Ghemawat, S., Gobioff, H., Leung, S.T.: The google file system. In: *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles*, pp. 29–43 (2003)
7. Hunt, P., Konar, M., Junqueira, F.P., Reed, B.: Zookeeper: wait-free coordination for internet-scale systems. In: *USENIX Annual Technical Conference*, vol. 8 (2010)
8. Kubiawicz, J., et al.: Oceanstore: an architecture for global-scale persistent storage. *ACM SIGOPS Oper. Syst. Rev.* **34**(5), 190–201 (2000)
9. Oh, K., Qin, N., Chandra, A., Weissman, J.: Wiera: policy-driven multi-tiered geo-distributed cloud storage system. *IEEE Trans. Parallel Distrib. Syst.* **31**(2), 294–305 (2019)
10. Qi, L., Chen, Y., Yuan, Y., Fu, S., Zhang, X., Xu, X.: A QoS-aware virtual machine scheduling method for energy conservation in cloud-based cyber-physical systems. *World Wide Web* **23**(2), 1275–1297 (2020)

11. Ren, K., Li, D., Abadi, D.J.: Slog: serializable, low-latency, geo-replicated transactions. *Proc. VLDB Endowment* **12**(11), 1747–1761 (2019)
12. Roselli, D.S., Lorch, J.R., Anderson, T.E., et al.: A comparison of file system workloads. In: *USENIX Annual Technical Conference, General Track*, pp. 41–54 (2000)
13. Ross, R.B., Thakur, R., et al.: PVFS: a parallel file system for linux clusters. In: *Proceedings of the 4th Annual Linux Showcase and Conference*, pp. 391–430 (2000)
14. Shvachko, K., Kuang, H., Radia, S., Chansler, R.: The hadoop distributed file system. In: *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, pp. 1–10. IEEE (2010)
15. Singh, H.J., Bawa, S.: Scalable metadata management techniques for ultra-large distributed storage systems-a systematic review. *ACM Comput. Surv. (CSUR)* **51**(4), 1–37 (2018)
16. Uluyol, M., Huang, A., Goel, A., Chowdhury, M., Madhyastha, H.V.: Near-optimal latency versus cost tradeoffs in geo-distributed storage. In: *17th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 20)*, pp. 157–180 (2020)
17. Weil, S.A., Brandt, S.A., Miller, E.L., Long, D.D., Maltzahn, C.: Ceph: a scalable, high-performance distributed file system. In: *Proceedings of the 7th Symposium on Operating Systems Design and Implementation*, pp. 307–320 (2006)
18. Weil, S.A., Pollack, K.T., Brandt, S.A., Miller, E.L.: Dynamic metadata management for petabyte-scale file systems. In: *SC'04: Proceedings of the 2004 ACM/IEEE Conference on Supercomputing*, p. 4. IEEE (2004)
19. Xu, X., et al.: Secure service offloading for internet of vehicles in SDN-enabled mobile edge computing. *IEEE Trans. Intell. Transp. Syst.* **22**(6), 3720–3729 (2020)
20. Xu, X., Zhang, X., Gao, H., Xue, Y., Qi, L., Dou, W.: Become: blockchain-enabled computation offloading for IoT in mobile edge computing. *IEEE Trans. Ind. Inf.* **16**(6), 4187–4195 (2019)
21. Yang, S., Wang, X., Wang, X., An, L., Zhang, G.: High-performance docker integration scheme based on openstack. *World Wide Web* **23**(4), 2593–2632 (2020)
22. Yu, H., Zhang, F., Wu, Y.: Granary: a sharing oriented distributed storage system. *Future Gen. Comput. Syst.* **38**, 47–60 (2014)
23. Yu, J., Wu, W., Yang, D., Huang, N., et al.: MRFS: a distributed files system with geo-replicated metadata. In: Sun, X. (ed.) *ICA3PP 2014, Part II. LNCS*, vol. 8631, pp. 273–285. Springer, Cham (2014). [https://doi.org/10.1007/978-3-319-11194-0\\_21](https://doi.org/10.1007/978-3-319-11194-0_21)
24. Zaker, F.: Online Shopping Store - Web Server Logs (2019). <https://doi.org/10.7910/DVN/3QBYB5>
25. Zhou, J., Chen, Y., Wang, W., He, S., Meng, D.: A highly reliable metadata service for large-scale distributed file systems. *IEEE Trans. Parallel Distrib. Syst.* **31**(2), 374–392 (2019)





# XTuning: Expert Database Tuning System Based on Reinforcement Learning

Yanfeng Chai<sup>1,2</sup>, Jiake Ge<sup>1</sup>, Yunpeng Chai<sup>1(✉)</sup>, Xin Wang<sup>3</sup>,  
and BoXuan Zhao<sup>1</sup>

- <sup>1</sup> Key Laboratory of Data Engineering and Knowledge Engineering, MOE, and School of Information, Renmin University of China, Beijing, China  
{yfchai, gejiake, ypchai, boxuan.zhao}@ruc.edu.cn
- <sup>2</sup> College of Computer Science and Technology, Taiyuan University of Science and Technology, Taiyuan, Shanxi, China
- <sup>3</sup> College of Intelligence and Computing, Tianjin University, Tianjin, China  
wangx@tju.edu.cn

**Abstract.** Database performance optimization has become a hot issue in recent years. Some works deeply reconstruct the database to achieve specified goals like throughput or latency. The others focus on the database's configuration knobs with reinforcement learning (RL) to improve the performance without any empirical knowledge. But the exhaustive offline training process costs plenty of time and resources due to the large inefficient configuration knobs combinations with trial-and-error methods. The most time-consuming part of the process is not the RL network training, but the database performance evaluation for acquiring the reward values of target performance like throughput or latency. So we propose an expert database tuning system (XTuning) which contains a correlation knowledge model to remove unnecessary training costs and a multi-instance mechanism (MIM) to support fine-grained tuning for diverse workloads. The models define the importance and correlations among these configuration knobs for the user's specified target. Then we implement the models as Progressive Expert Knowledge Tuning (PEKT) algorithm with an abstracted architectural optimization integrated into XTuning. Experiments show that XTuning can effectively reduce the training time and achieves extra performance promotion compared with the state-of-the-art tuning methods.

**Keywords:** Database optimization · Auto tuning · Expert knowledge rules · Reinforcement learning · Reduce training time

## 1 Introduction

Databases usually offer a larger number of configuration knobs for users. For example, MySQL and PostgreSQL have about 200+ knobs while the key-value

This work is supported by the National Key Research and Development Program of China (No. 2019YFE0198600), National Natural Science Foundation of China (No. 61972402, 61972275, and 61732014).

store RocksDB [7] still has more than 100+ knobs for performance tuning. Therefore, tuning the hundreds of knobs is an impossible mission even for the very experienced DBAs. In another word, finding the optimal knobs solution in the huge continuous space is an NP-hard problem [14]. Existing knobs-based auto-tuning methods usually have some weaknesses during the training process. First, there is no fine-grained tuning mechanism for a specified workload. Second, plenty of time (dozens of hours to days) [16] and resources are spent on the offline performance measurements, which include many invalid knobs combinations lacking in correlation and feasibility check. In fact, these costs could be avoided with the constraint rules based on existing experiences or knowledge. The contributions are summarized as follows:

- (1) We propose an expert database tuning system (XTuning) based on reinforcement learning, which contains the correlation rules module based on expert knowledge for different scenarios.
- (2) We extend XTuning with Progressive Expert Knowledge Tuning (PEKT) algorithm included an abstraction method of the architectural optimization and multi-instance mechanism (MIM) for further performance promotion.
- (3) Experiments show XTuning outperforms the SOTA auto-tuning method CDBTune both on training time reduction and on performance promotion.

## 2 Related Work

We summarize existing works related to the database auto-tuning as follows:

**Knobs-based Auto-tuning.** BestConfig [17] uses search-based methods to find the optimal knobs based on historical tuning data, which costs lots of time and needs to restart the process if a new request arrives. OtterTune [14] uses a learning-based method to recommend knobs based on the historical tuning experience. But this method requires higher quality samples with every necessary condition. CDBTune [16] adopts deep reinforcement learning (DRL) with a deep deterministic policy gradient method and a try-and-error strategy to find the optimal knobs through many performance tests. QTune [10] also utilizes a DRL model, focusing on the SQL’s pattern for fine-grained tuning at a different level.

*Knobs-based optimization will not be the final destination for auto-tuning.* With key-value stores emerging, databases like to use it as the storage engine like Snowflake [3] and MyRocks [11]. KVStores are widely used in distributed system, such as TiDB [8], CockroachDB [13], NebulaGraph [12], and HugeGraph [9].

**Architectural Optimization.** SILK [1] designs an I/O scheduler to deal with the latency spikes by dynamically allocating I/O resources according to the operations’ priorities. ALDC [2] focuses on LSM-tree’s compaction mechanism with controllable granularity methods to acquire specified goals. Monkey [5] focuses on the bloom filter for performance promotion. Dostoevsky [6] further proposed a hybrid merge policy to remove superfluous merging adaptively. Bourbon [4] uses machine learning to build a learned index to promote the lookup performance.

### 3 Expert Knowledge-Based Tuning Architecture

Figure 1 presents the overall architecture of XTuning, and the gray arrow represents the training process. First, the external expert rules module classifies workloads from the system module. Then it (MIM) passes the classified workload to the corresponding core tuning algorithm. It could support fine-grained workloads classification methods under different scenarios while CDBTune only supports coarse-grained way. Second, the auto-tuning module receives the classified workload and other parameters, then trains itself efficiently with internal expert rules. They could significantly reduce the RL network training times and enhance the practicability while others like CDBTune cost too long. Finally, the auto-tuning module recommends knobs to the system. The above steps are repeated until the RL algorithm converges. RL utilizes a try-and-error strategy to explore optimal knobs' combinations, normally ignoring some configuration knobs that naturally have some kind of positive or negative correlations.

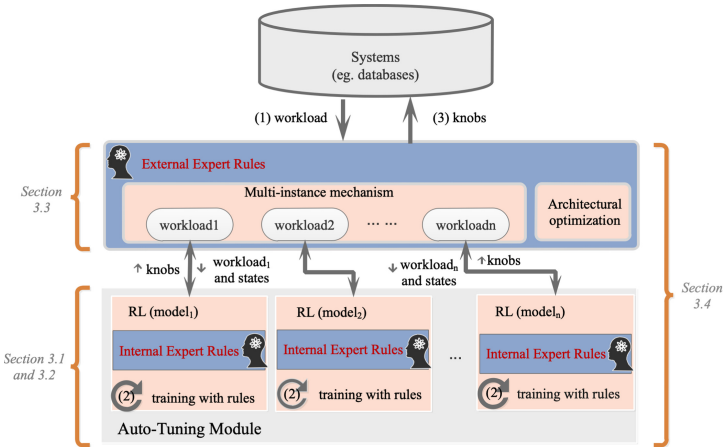


Fig. 1. System architecture of XTuning.

#### 3.1 Correlation Rules Table of Knobs

As Table 1 shown, the architecture of correlation rules in XTuning is something like a two-dimensional table. XTuning utilizes it to offer a fine-grained tuning optimization under diverse workloads.  $r_{i \rightarrow j(\cdot)}$  means the correlation between knob<sub>i</sub> and knob<sub>j</sub> with positive (+), negative (-), or uncorrelated ( $\phi$ ).

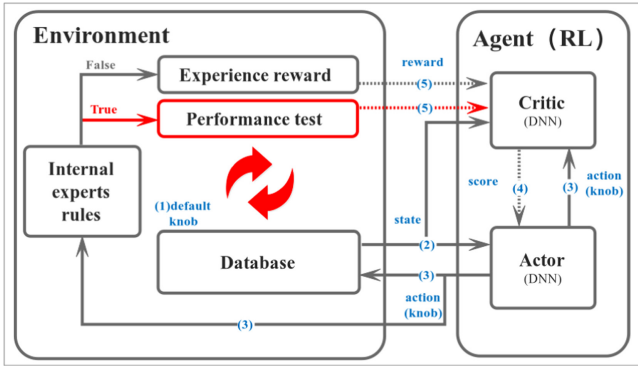
For example, LevelDB provides two knobs for the in-memory write buffer size and max file size. The buffer will be serialized into the disk as one or more files. Now we assume that we already know this expert knowledge, which means the  $knob_{write\_buffer\_size}$  and the  $knob_{max\_file\_size}$  should conform to positive correlation and  $knob_{write\_buffer\_size} \geq 2.0 \cdot knob_{max\_file\_size}$ . The correlation can be presented as (+)2.0, which leads to the least space-amplification for file systems. But if the two knobs have the wrong correlation, this meaningless performance testing will be skipped to reduce training time.

**Table 1.** Correlation rules table of knobs.

Workload <sub>i</sub>	knob <sub>1</sub>	knob <sub>2</sub>	...	knob <sub>n-1</sub>	knob <sub>n</sub>
knob <sub>1</sub>	-	-		-	-
knob <sub>2</sub>	$r_{2 \rightarrow 1(+)}$	-		-	-
⋮	⋮	⋮	⋮	⋮	⋮
knob <sub>n-1</sub>	$r_{n-1 \rightarrow 1(-)}$	$r_{n-1 \rightarrow 2(-)}$		-	-
knob <sub>n</sub>	$r_{n \rightarrow 1(+)}$	$r_{n \rightarrow 2(-)}$		$r_{n \rightarrow n-1(\phi)}$	-

### 3.2 Knobs Correlation with Internal Expert Knowledge

In Fig. 2, the red part represents the performance testing process, which provides the reward values to the RL network. To solve this problem, we embed an internal expert rules mechanism. Figure 2 shows that (1) If the RL outputs a knob that conforms to the internal expert rules, it seems to be a high-performance knob from an experiential perspective. The reward value is determined by the performance test module (red lines). (2) If the RL outputs a knob that does not conform to the internal expert rules, it seems to be a low-performance knob from an experiential perspective. So the real test is not needed, and it will be replaced by the experience reward (gray lines) for the system’s availability.



**Fig. 2.** Workflows of the auto-tuning module. (Color figure online)

### 3.3 Workloads Correlation with External Expert Knowledge

The workloads in real scenarios change dynamically. Existing auto-tuning methods like CDBTune cannot deal with these situations. In this part, we will describe how the external expert rules work in XTuning.

**Multi-instance Mechanism (MIM).** In Table 2, the external expert rules implement the classification based on the read/write ratio. XTuning trains each network individually based on these workloads. For example, MIM is monitoring

the real-time status of the workload. Once it reaches the threshold, MIM will re-select the auto-tuning modules to serve according to the next workload’s pattern, and recommend high-performance knobs again. Note that (1) the internal structure of the input and output and the auto-tuning module are fixed. Therefore, XTuning only needs to establish a general neural network framework to load the internal parameters of the network corresponding to different instance models rather than to reestablish the neural network. (2) When XTuning training multiple models, the workload proportion generated by the process can meet the random value for the corresponding model’s range. (3) MIM can generate the specified number of models according to the user’s demand. XTuning could dynamically recommend the optimal knobs which fit the current workload’s status. Therefore, comparing to CDBTune’s coarse-grained tuning, MIM could classify workloads with a fine-grained method for better tuning.

**Table 2.** Multi-instance mechanism for the fine-grained tuning.

Write (%)	$(0, A_1]$	$(A_1, A_2]$	$(A_2, A_3]$		$(A_{i-1}, A_i]$
Instance	$model_1$	$model_2$	$model_3$		$model_i$
Read : Scan	$a_1 : b_1$	$a_2 : b_2$	$a_3 : b_3$	.....	$a_i : b_i$
Threshold	$Th_1$	$Th_2$	$Th_3$		$Th_i$

**Abstract Architectural Optimization as Extra Knobs.** As we mentioned in Sect. 1, We import a Fine-grained Controllable Compaction (FCC) mechanism in LevelDB, which further improves the performance and system fluctuation by controlling the write amplification (WA) for different read/write ratio workloads. As shown in Eq. 1,  $R_m$  means the compaction ratio in Fine-grained Controllable Compaction (FCC) mechanism.  $\sum_{i=1}^n S_i$  is the current total size accumulated and  $F_{max}$  is the max file size. If the number exceeds  $N_{max}$  or  $R_m \geq R_{th}$ , then the compaction will be triggered immediately. So the ratio threshold  $R_{th}$  can control the compaction granularity to acquire specified performance. Therefore, we abstract the ratio as an expert knob in PEKT to achieve further performance promotion.

$$R_m = \begin{cases} (\sum_{i=1}^n S_i)/F_{max}, & i < N_{max}, \\ R_{th}, & otherwise. \end{cases} \quad (1)$$

### 3.4 Progressive Expert Knowledge Tuning Algorithm

Next, we integrate the FCC mechanism, the internal and external expert rules into XTuning as a progressive expert knowledge tuning algorithm (PEKT).

**Algorithm 1.** Progressive Expert Knowledge Tuning: Training Phase

---

**Input:**workloads sequence  $W_n\{W_1, W_2, \dots, W_\tau\}$

```

1:  $model_i \leftarrow external\ expert\ rules(W_n)$       8:  $reward_{RL_i} \leftarrow Eval(knob)$ 
2: In each episode:                                9: else
3:  $critical\ params \leftarrow Database(W_n)$       10:  $reward_{RL_i} \leftarrow Exp.\ reward$ 
4:  $RL_i \leftarrow model_i$                         11: end if
5: for each time step  $\beta$  do                       12:  $train\ RL_i.\ critic(reward\_RL)$ 
6:    $knob \leftarrow RL_i.\ actor(params)$           13:  $train\ RL_i.\ actor(score)$ 
7:   if  $internal\ expert\ rules(knob) ==$           14: end for
    $True\ or\ random() < \epsilon$  then

```

---

**External Expert Rules.** (1) In Algorithm 1, the external expert rules invoke a multi-instance mechanism to identify and classify the current workload. At the same time, the multi-instance mechanism forces the auto-tuning module to establish the corresponding RL network instance (Line 4). (2) In Algorithm 2, when XTuning receives the tuning signal and current operations' type, the external expert rules invoke the multi-instance mechanism to identify the current workload. Then the external expert rules inform the auto-tuning module to load the corresponding RL network instance (Line 3) for auto-tuning (Line 4).

**Algorithm 2.** Progressive Expert Knowledge Tuning: Running Phase

---

**Input:**workloads sequence  $W_n\{W_1, W_2, \dots, W_\tau\}$

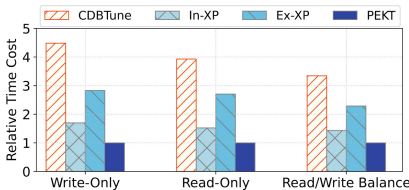
```

1:  $model_i \leftarrow external\ expert\ rules(W_n)$ 
2:  $important\ params \leftarrow Database(W_n)$ 
3:  $RL_i \leftarrow model_i$ 
4:  $knob \leftarrow RL_i.\ actor(params)$ 

```

---

**Internal Expert Rules.** (1) During the training phase in Algorithm 1, internal expert rules directly participate in and simplify the RL training process. If the knob does not conform to the internal expert rules, the performance testing phase will be skipped (Line 8). But there is still a tiny probability  $P(r < \epsilon)$  to have a random exploration for rules' self-improvement. Rewards are set based on the experience to reduce the training time (Line 10); (2) Internal expert rules only serve the training period of XTuning, rather than the actual tuning period.



	WO	RO	RWB
<b>CDBTune</b>	103h	106h	117h
<b>In-XP</b>	39h	41h	50h
<b>Ex-XP</b>	65h	73h	80h
<b>PEKT</b>	23h	27h	35h

**Fig. 3.** Comparison of training time cost with CDBTune, In-XP, Ex-XP and PEKT.

## 4 Experiment Study

We implement XTuning based on the CDBTune [16]. Our evaluation is based on YCSB [15] and LevelDB for architectural optimization. We generate 5 GB of data and 50 M operations with 5 threads for each testing round. Each key-value pair is set to have a 16-B key and a 1-KB value.

### 4.1 Training Time Reduction with Expert Rules

We evaluate the different modules in XTuning with the Internal Expert Rules (In-XP), the External Expert Rules (Ex-XP), and Progressive Expert Knowledge Tuning (PEKT) with full features. Then we make a comparison of offline training time reduction with the above groups under three workloads (*Write-Only*, *Read-Only*, and *Read/Write-Balance*). First, all three can effectively reduce the offline training time in Fig. 3. Moreover, PEKT can reduce the offline training time by 77.67%, 74.53%, and 70.09% under *WO*, *RO*, and *RWB* workloads. Second, internal expert rules could reduce the performance testing cost with the correlation rules to accelerate RL network training. That means In-XP still achieves time reductions by 62.14%, 61.32%, and 57.26% under the above three workloads. Third, due to the external expert rules (Sect. 3.3), Ex-XP still outperforms CDBTune with time reductions by 36.89%, 31.13%, and 31.62%.

### 4.2 Throughput Improvement

In Fig. 4, PEKT can achieve the best performance promotion under different read/write ratios workloads. Compared with the default configuration setting, PEKT can promote the throughput by 3.8x ~ 10.02x. Meanwhile, PEKT also can achieve about 4.58% ~ 64.26% higher throughput improvement than CDBTune.

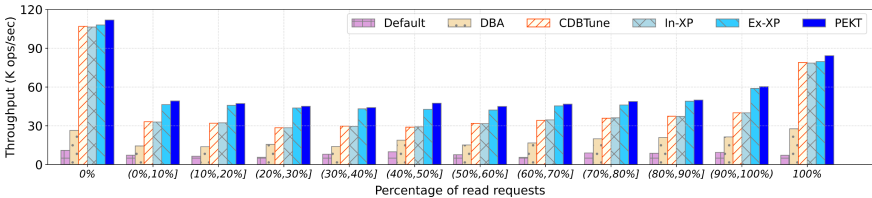


Fig. 4. Throughput comparison under the different read/write ratios workloads.

However, PEKT merely gets a tiny superiority under *RO* and *WO* workloads compared with the CDBTune. Because CDBTune focuses only on the two workloads, but PEKT utilizes the MIM for fine-grained read/write ratios workloads. Moreover, architectural optimization in XTuning further improves the throughput due to the FCC mechanism for controllable write amplification.

### 4.3 Latency Reduction

In this part, we evaluate the latency reduction under different read/write ratios workloads. We know the fluctuation usually ruins users' experience due to the terrible tail latency. In Sect. 3.4, we abstracted the FCC as an extra knob to import architectural optimization into XTuning. So PEKT could effectively reduce the tail latency by 53.88% ~ 94.39% and 23.47% ~ 63.45% compared with the Default and the CDBTune. Due to the FCC, PEKT can restrict latency into a more reasonable range to offer a smooth user experience (Fig. 5).

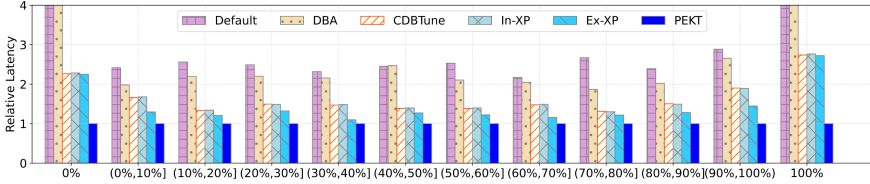


Fig. 5. Latency comparison under the different read/write ratios workloads.

### 4.4 Architectural Optimization Performance in XTuning

First, in Fig. 6(a), PEKT reduces internal I/O size by 52.9% and 19.02% compared with the Default and the CDBTune under *RO* workload. Second, PEKT reduces I/O size by 75.04% and 24.74% under *WR* workload in Fig. 6(d). Third, FCC effectively reduces the compaction I/O size by 80.21% and 54.01% compared with the Default and the CDBTune under *WO* workload in Fig. 6(g).

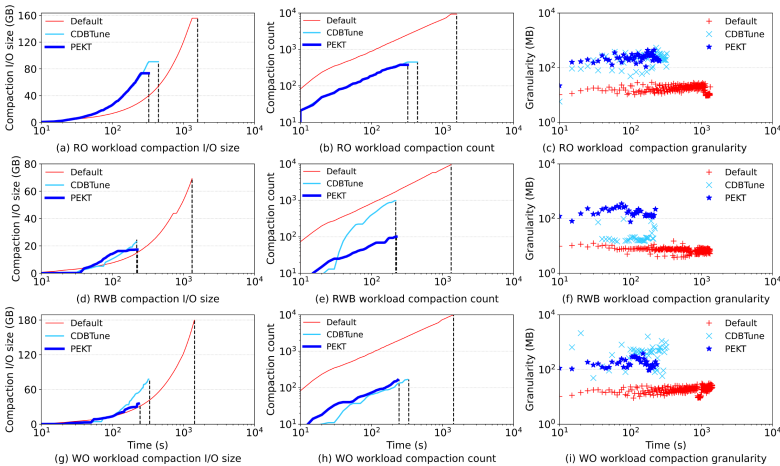


Fig. 6. Internal compaction I/O size, count and granularity under different workloads.



Figure 6(c), (f), and (i) describe the internal I/O granularity under *RO*, *RWB*, and *WO* workloads. The Default has the smallest compaction granularity because of its configuration space containing no optimizations for the workloads. Though the CDBTune can achieve close throughput performance with PEKT under *RO* and *WO* workloads, it still cannot control the compaction granularity, which may lead to terrible system fluctuations.

## 5 Conclusion

Existing auto-tuning methods usually ignore the correlations between the knobs and the workloads. Therefore, we propose XTuning with the internal and external expert knowledge modules to skip the unnecessary training rounds for reducing the training time with a fine-grained tuning for the complex workloads. Moreover, we integrate the architectural optimization into the XTuning, which leads to further performance promotion with the specified demands.

## References

1. Balmau, O., Dinu, F., Zwaenepoel, W., Gupta, K., Chandhiramoorthi, R., Didona, D.: SILK: preventing latency spikes in log-structured merge key-value stores. In: 2019 USENIX Annual Technical Conference, pp. 753–766, July 2019
2. Chai, Y., Chai, Y., Wang, X., Wei, H., Wang, Y.: Adaptive lower-level driven compaction to optimize LSM-tree key-value stores. *IEEE Trans. Knowl. Data Eng.* (2020, early access). <https://doi.org/10.1109/TKDE.2020.3019264>
3. Dageville, B., et al.: The snowflake elastic data warehouse. In: Proceedings of the 2016 International Conference on Management of Data, pp. 215–226 (2016)
4. Dai, Y., et al.: From wisckey to bourbon: a learned index for log-structured merge trees. In: 14th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 20), pp. 155–171 (2020)
5. Dayan, N., Athanassoulis, M., Idreos, S.: Monkey: optimal navigable key-value store. In: SIGMOD, pp. 79–94. ACM (2017)
6. Dayan, N., Idreos, S.: Dostoevsky: better space-time trade-offs for LSM-tree based key-value stores via adaptive removal of superfluous merging. In: Proceedings of the 2018 International Conference on Management of Data, pp. 505–520 (2018)
7. Dong, S., Callaghan, M., Galanis, L., Borthakur, D., Savor, T., Strum, M.: Optimizing space amplification in RocksDB. In: CIDR (2017)
8. Huang, D., et al.: TiDB: a raft-based HTAP database. *Proc. VLDB Endowment* **13**(12), 3072–3084 (2020)
9. Hugegraph (2021). <https://github.com/hugegraph/hugegraph>
10. Li, G., Zhou, X., Li, S., Gao, B.: Qtune: a query-aware database tuning system with deep reinforcement learning. *Proc. VLDB Endowment* **12**(12), 2118–2130 (2019)
11. Matsunobu, Y., Dong, S., Lee, H.: Myrocks: LSM-tree database storage engine serving facebook’s social graph. *Proc. VLDB Endowment* **13**(12), 3217–3230 (2020)
12. Nebula graph (2021). <https://github.com/vesoft-inc/nebula-graph>
13. Taft, R., et al.: Cockroachdb: the resilient geo-distributed sql database. In: Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, pp. 1493–1509 (2020)

14. Van Aken, D., Pavlo, A., Gordon, G.J., Zhang, B.: Automatic database management system tuning through large-scale machine learning. In: Proceedings of the 2017 ACM International Conference on Management of Data, pp. 1009–1024 (2017)
15. YCSB-C (2018). <https://github.com/basicthinker/YCSB-C>
16. Zhang, J., et al.: An end-to-end automatic cloud database tuning system using deep reinforcement learning. In: Proceedings of the 2019 International Conference on Management of Data, pp. 415–432 (2019)
17. Zhu, Y., et al.: Bestconfig: tapping the performance potential of systems via automatic configuration tuning. In: Proceedings of the 2017 Symposium on Cloud Computing, pp. 338–350 (2017)



# CELA: An Accurate Learned Cardinality Estimator with Strong Generalization Ability and Dimensional Adaptability

Weiqing Zhou, Siyu Zhan<sup>(✉)</sup>, Lei Guo, and Bo Dai

University of Electronic Science and Technology of China, No. 2006, Xiyuan Ave,  
West Hi-Tech Zone, Chengdu 611731, Sichuan, China  
wqzhou@std.uestc.edu.cn, {zhansy,leiguo,daibo}@uestc.edu.cn

**Abstract.** Accurate cardinality estimation contributes significantly to query optimization whereas traditional approaches such as histogram-based or sketching-based approaches relying on assumption of uniform distribution of data and appropriate pre-set parameters, often leading to dilemma in practical applications. In this paper, an accurate lightweight, dimensionally adaptive, strongly generalizable learned cardinality estimator for multi-dimensional range queries, CELA is proposed reflecting on the characteristics of desirable cardinality estimators. For the purpose of capturing relationship between dimensions, CELA raises a query-oriented approach of constructing constraint matrices to apply convolution. Experiments illustrates that CELA performs superbly on each defined indicators far superior to PostgreSQL. Furthermore, the strong generalization ability of CELA is demonstrated by the excellent performance trained with continuously scaled-down training set.

**Keywords:** Cardinality estimation · Dynamic architecture · Deep learning · Generalization ability · Dimensional adaptability

## 1 Introduction

There are many methods for database query optimization [3, 9, 10], and cost estimation is an inextricable part of them all [7], in which cardinality estimation is an important component. Cardinality estimation specifically refers to predicting the number of records in the query results. The cardinality estimation is so critical that it has been studied by researchers in the field of databases for decades, but there is still much room for improvement. There are already many widely used traditional methods. The performance of histogram-based methods [13] depends on whether the data conform to the assumption of a uniform distribution. What's more, advance settings for width or depth are also required. Sketching-based methods are often applied to estimate the number of different data, but requires extra space to store the bitmaps and carefully designed hash

---

Supported by the National Key R&D Program of China (grant No. 2019YFB1705601) and the Natural Science Foundation of China (grant No. 62072075).

functions for different workloads. Machine learning and deep learning are rapidly evolving in research and industry [1, 5, 6, 14]. With the explosion of this technology, some learned cardinality estimators emerge which have shown promising results.

An exquisite, lightweight, and easy-to-train cardinality estimation model with high generalization ability and dimensional adaptability, CELA is proposed in this paper. Specifically, the approach for dynamic vectorization and dynamic model architecture is proposed for adaption to queries with different dimensions. In addition, an efficient algorithm for data generation is explored. What’s more, three indicators for measuring the model are presented, and the performance of CELA is tested and compared with other estimators.

## 2 Related Work

Cardinality estimation is a crucial part of query optimization. However, the traditional methods are helpless to cope with multi-dimensional constraints. With the excellence of machine learning and deep learning, learned cardinality estimators emerge continuously. Andreas Kipf [4] extracts features of tables, joins, and predicates separately with fully connected networks and learn the cardinality with MLPs after aggregation. Furthermore Lucas Woltmann [12] transforms the query into a one-dimensional vector as input to the neural networks. Hasan [2] seeks to estimate the joint probability distribution from samples. Nevertheless, the researches have not yet explored in-depth specifically on the generalization ability of models and adaptability to queries with various dimensions.

## 3 Vectorization

For instance, a query  $q$ , “select  $a_1$  from  $A$  where  $a_1 > b_1$  and  $a_2 = b_2$  and ... and  $a_k < b_k$ ”, “ $a_i > b_i$ ” is defined as the constraint on the data in dimension  $w_i$  of the query, in which the predicate  $p_i$  and the qualifier  $b_i$  need to be parsed. For a query with  $k$  dimensions,  $k$  constraint triplets like  $\langle a_i, p_i, b_i \rangle$  are obtained.

Each constraint triplet is refined from the query  $q$ . Firstly,  $a_i$  is processed and a sequence is maintained to specify their order. In the constraint vector,  $p_i$  is encoded with 3 bits using one-hot and normalized  $b_i$  occupies 1 bit. The 4-bit vector is then passed through a fully connected layer containing  $n$  neurons and a ReLU layer resulting in a vectorized representation of  $n$  bits on  $w_i$ . The constraint matrix of size  $[k, n]$  is obtained by splicing constraint vectors in order and adjusting the size of the matrix. Such a multi-dimensional representation, instead of a  $[1, k * n]$  single-dimensional vector, allows better utilization of convolution to capture the correlation between different attributes.

## 4 Model

### 4.1 Considerations

Cardinality estimation is a part of physical execution plan generation. The time taken for a query from reaching the storage engine to returning the final result,

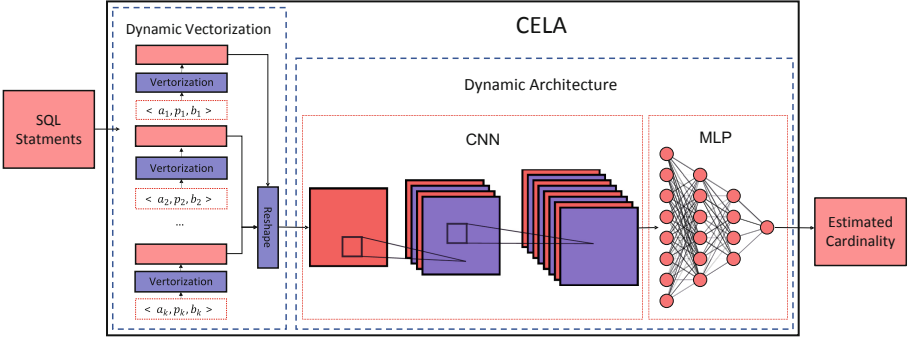


Fig. 1. The architecture of CELA.

includes the time for generating the physical plan and the time for executing the plan in the storage engine. Therefore, the key for learned estimators is to produce the accurate estimated results in a short time. From the above issues, a dynamic model architecture is designed, which is lightweight and can capture inter-dimensional relationship and have strong generalization ability.

## 4.2 Dynamic Vectorization

The number of bits encoded by one-hot for each constraint triplet in the query  $q$  is denoted as  $EB$ . The number of neurons in the single-layer MLP,  $n$  is related to the dimension  $k$  and the number of encoding bits  $EB$ . In terms of popularity, the more dimensions, the more neurons. When the encoding is determined,  $n$  is expected to be monotone polynomial with only one variable  $k$ . In this work,  $n$  is specifically equal to  $EB * K$  and  $EB$  is 4. For high computing performance, the size of the constraint matrix is adjusted to match data with different dimensions. The number of elements in the constraint matrix can be calculated as  $EB * K^2$ . Provided that  $EB$  is a perfect square number, constraint matrix can be reshaped to  $[k\sqrt{EB}, k\sqrt{EB}]$  as further input.

## 4.3 Dynamic Architecture

For the dynamic input scale brought by dynamic vectorization, a dynamic network architecture is designed to accommodate varying dimensions, as well as to avoid excessive overhead. The architecture is shown in Fig. 1, which contains two parts, CNNs with two convolutional layers and MLPs with four layers. The output channels of the two convolutional layers,  $Channel_{conv1}$  and  $Channel_{conv2}$  are set adaptively and separately shown as Formula 1 and Formula 2.

$$Channel_{conv1} = EB * k \quad (1)$$

$$Channel_{conv2} = EB^{3/2} * k \quad (2)$$

The output of CNNs is reshaped to a one-dimensional vector of size  $[1, EB^{5/2} * k^3]$  for further computation. The MLPs are also with adaptive layers according to  $k$ . The number of neurons in each layer of the network is expressed as Formula 3–5, and the output layer contains only one neuron.

$$Numel_{mlp1} = EB * FN(EB, k) = EB^2 * k^2 \quad (3)$$

$$Numel_{mlp2} = EB^{1/2} * FN(EB, k) = EB^{1/2} * k^2 \quad (4)$$

$$Numel_{mlp3} = k \quad (5)$$

## 5 Experiment

### 5.1 Generating Data

The following way of generating queries is often considered. For the constraints in the queries on each dimension, values are taken evenly within the range of the attributes in the database. This is followed by a random selection of predicates for that dimension, but the values obtained may not exist in the database and skewed distribution may bring troubles. Therefore, the proposed algorithm of generating data is illustrated as Table 1. The experiments are conducted on the 4GB TPC-H workload [11] in PostgreSQL [8] and data is generated separately according to Table 1. 211,721 samples including 102,789 training samples, 215,636 samples including 101,032 training samples and 200,936 samples including 100,869 training samples are separately generated with 2, 3 and 4 dimensions. It is of interest that in this work, both the training set and the test set is close to 50%, revealing the strong generalization ability.

**Table 1.** Generating data.

The algorithm of generating data	
Step 1	Sample randomly in the database to obtain the real records in the table;
Step 2	Extract the constraint values $b_1, b_2, b_3, \dots, b_k$ for each of the attributes
Step 3	Select randomly one of the three predicates “>”, “<” and “=”, as $p_i$ , which forms a constraint triplet $\langle a_i, p_i, b_i \rangle$ on dimension $w_i$ for a query;
Step 4	Repeat Step 2, 3 until the constraint triplet on all dimensions is formed.;
Step 5	Repeat Steps 1–4 until a sufficient amount of data is generated

### 5.2 Model Training

For all three dimensions of data L1Loss is set as the loss function and the loss in one training episode is shown in Fig. 2(a). It illustrates that the model converges fast without seeing much data no matter which dimension it is trained on. Figure 2(b) illustrates the average loss for 200 continuous episodes. Figure 2(b) demonstrates that the loss is reduced to a low level after just one training episode, although there is a steep drop during approximately 25 more training episodes.

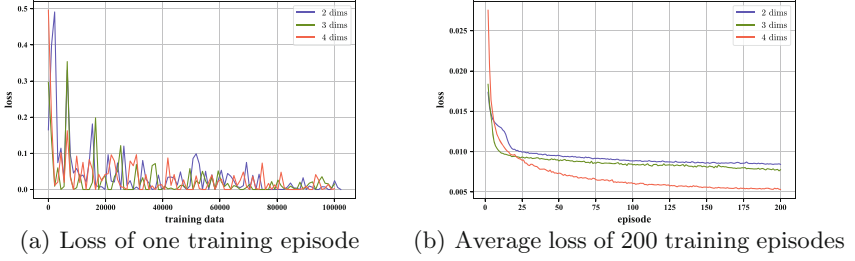


Fig. 2. Training loss.

### 5.3 Evaluation

In this work, three quantitative indicators are proposed as follow to measure the performance of the model. First, The total number of queries for which the estimated cardinality in the test set is exactly equal to the true cardinality expressed as Formula 6. Second, the average error cardinality for all queries in the test set expressed as Formula 7. Third, the average error rate of all queries in the test set expressed as Formula 8.

$$EQ = \sum_{i=1}^{QT} x_i, x_i = \begin{cases} 1, Car_{ie} = Car_{it} \\ 0, Car_{ie} \neq Car_{it} \end{cases} \quad (6)$$

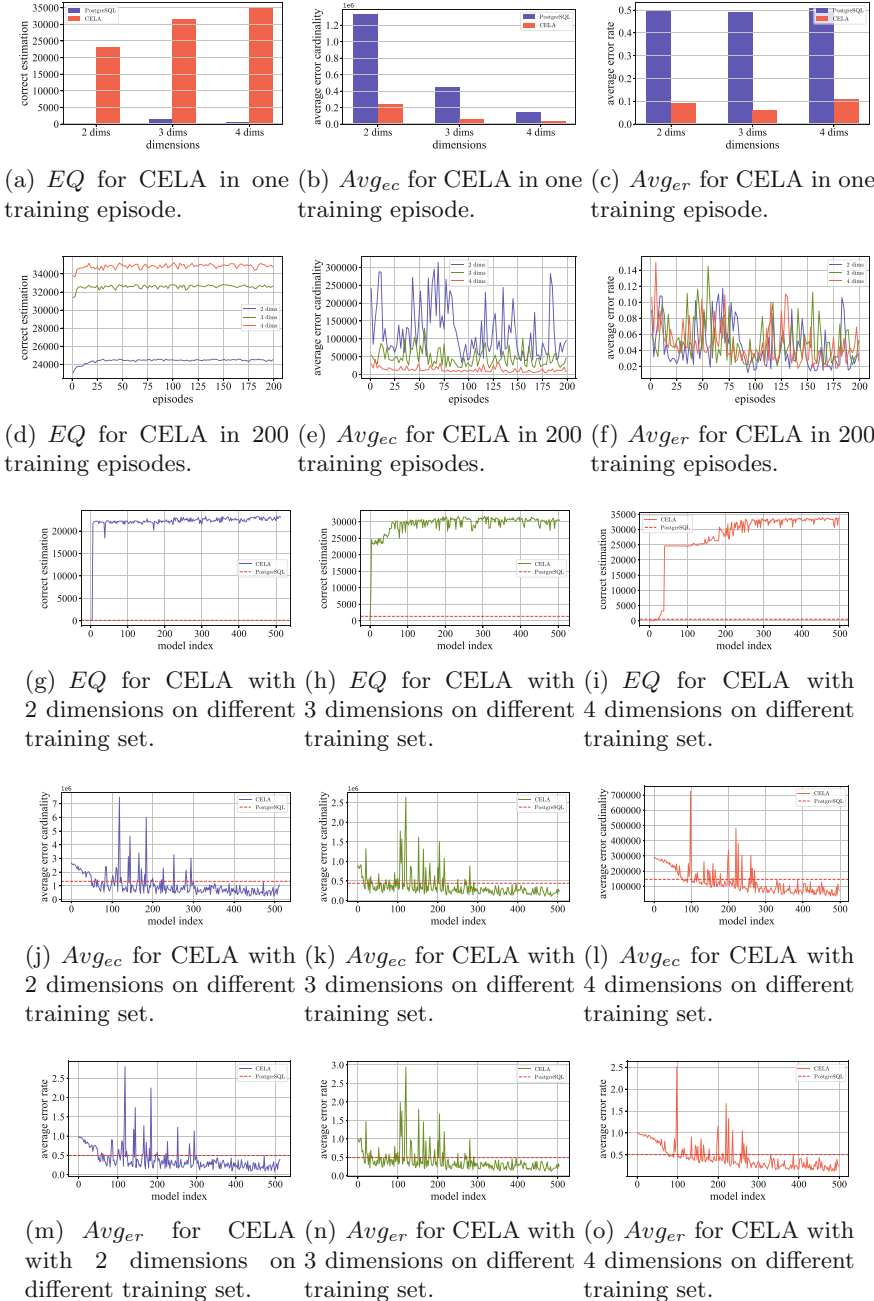
$$Avg_{ec} = \frac{\sum_{i=1}^{QT} |Car_{ie} - Car_{it}|}{QT} \quad (7)$$

$$Avg_{er} = \frac{\sum_{i=1}^{QT} |Car_{ie} - Car_{it}|}{\sum_{i=1}^{QT} Car_{it}} \quad (8)$$

From Fig. 3(a)–(c) it is seen that the proposed model outperforms the cardinality estimator in PostgreSQL for each indicators a lot. The Fig. 3(d)–(f) shows that there is only a small improvement in each dimensions after more episodes of training not matching the overhead of training.

To demonstrate the strong generalization ability of CELA, we keep the test set constant, continuously add 200 queries into training set, and then test the models. From Fig. 3(g)–(i), the performance of the first few dozen models on  $EQ$  can even approach the best performance. Shown in Fig. 3(j)–(o), about the 300th model outperforms PostgreSQL in all three dimensions, and the training set is about 600,000 queries, **37.5%** of the total data set. The fantastic results are full proof the strong generalization ability of CELA.

Overall, the performance of the models on varying dimensions is shown in Table 2 retaining three decimals. MSCN [4] only take predicate sets as input to adapt to our workload, so there is no need for concatenation after average pooling. Only the attributes need to be encoded in vectorization of LocalNN [12], so it can be applied directly. The two recent learned estimators are selected to



**Fig. 3.** Performance of CELA.



**Table 2.** Performance on different dimensions.

	Cardinality estimator	<i>EQ</i>	<i>Avg<sub>ec</sub></i>	<i>Avg<sub>er</sub></i>
Performance on 2 dims	PostgreSQL	91	1,326,542.005	0.497
	LocalNN	22,433	221,571.788	0.083
	MSCN	22,358	184,191.273	0.069
	<b>CELA</b>	<b>24,585</b>	<b>32,362.293</b>	<b>0.012</b>
Performance on 3 dims	PostgreSQL	1,341	441,115.440	0.491
	LocalNN	30,768	243,296.303	0.271
	MSCN	28,614	158,685.886	0.177
	<b>CELA</b>	<b>32,810</b>	<b>18,378.845</b>	<b>0.020</b>
Performance on 4 dims	PostgreSQL	519	145,902.568	0.506
	LocalNN	32,022	165,388.694	0.574
	MSCN	30,897	79,541.116	0.276
	<b>CELA</b>	<b>35,423</b>	<b>4,998.175</b>	<b>0.017</b>

compare with CELA. The performance of PostgreSQL on *EQ* is unstable, while CELA’s performance continuously improves with dimensions and exceeds other three estimators a lot. On *Avg<sub>ec</sub>* and *Avg<sub>er</sub>*, the performance of PostgreSQL is relatively stable but poor. As the dimensions increases, the performance of both MSCN and LocalNN decreases rapidly, and LocalNN even performs worse than PostgreSQL, while CELA is quite stable as the dimensions increase. In summary, there is tremendous progress brought by CELA with approximately **2.4%** – **–4%** error of PostgreSQL, **6.2%** – **–17.4%** error of MSCN, **3%** – **–14.5%** error of LocalNN.

## 6 Conclusions

In this work, an accurate lightweight learned model for multi-dimensional range queries, CELA is proposed with dynamic vectorization and model architecture adapted to the various dimensions allowing the complexity of CELA to match the input size. TPC-H workloads are implemented on PostgreSQL to generate abundant data based on real records. After only one training episode, CELA exceeds PostgreSQL, MSCN and LocalNN by far on all defined indicators and the performance is even improved after more training episodes. For further research, the strong generalization ability of the model is demonstrated by holding the test set constant and reducing the training set. In conclusion, our work is valuable and provides constructive ideas for other researches.

## References

1. Ding, J., et al.: Alex: an updatable adaptive learned index. In: Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, pp. 969–984 (2020)

2. Hasan, S., Thirumuruganathan, S., Augustine, J., Koudas, N., Das, G.: Deep learning models for selectivity estimation of multi-attribute queries. In: Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, pp. 1035–1050 (2020)
3. Kamatkar, S.J., Kamble, A., Vilorio, A., Hernández-Fernandez, L., Cali, E.G.: Database performance tuning and query optimization. In: Tan, Y., Shi, Y., Tang, Q. (eds.) DMBD 2018. LNCS, vol. 10943, pp. 3–11. Springer, Cham (2018). [https://doi.org/10.1007/978-3-319-93803-5\\_1](https://doi.org/10.1007/978-3-319-93803-5_1)
4. Kipf, A., Kipf, T., Radke, B., Leis, V., Boncz, P., Kemper, A.: Learned cardinalities: estimating correlated joins with deep learning. arXiv preprint [arXiv:1809.00677](https://arxiv.org/abs/1809.00677) (2018)
5. Marcus, R., et al.: Neo: a learned query optimizer. arXiv preprint [arXiv:1904.03711](https://arxiv.org/abs/1904.03711) (2019)
6. Marcus, R., Papaemmanouil, O.: Towards a hands-free query optimizer through deep learning. arXiv preprint [arXiv:1809.10212](https://arxiv.org/abs/1809.10212) (2018)
7. Markl, V., Raman, V., Simmen, D., Lohman, G., Pirahesh, H., Cilimdžić, M.: Robust query processing through progressive optimization. In: Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data, pp. 659–670 (2004)
8. Momjian, B.: PostgreSQL: Introduction and Concepts, vol. 192. Addison-Wesley, New York (2001)
9. Ortiz, J., Balazinska, M., Gehrke, J., Keerthi, S.S.: Learning state representations for query optimization with deep reinforcement learning. In: Proceedings of the Second Workshop on Data Management for End-To-End Machine Learning, pp. 1–4 (2018)
10. Panahi, V., Navimipour, N.J.: Join query optimization in the distributed database system using an artificial bee colony algorithm and genetic operators. *Concurr. Comput. Pract. Exp.* **31**(17), e5218 (2019)
11. Poess, M., Floyd, C.: New TPC benchmarks for decision support and web commerce. *ACM Sigmod Rec.* **29**(4), 64–71 (2000)
12. Woltmann, L., Hartmann, C., Thiele, M., Habich, D., Lehner, W.: Cardinality estimation with local deep learning models. In: Proceedings of the Second International Workshop on Exploiting Artificial Intelligence Techniques for Data Management, pp. 1–8 (2019)
13. Xu, J., Zhang, Z., Xiao, X., Yang, Y., Yu, G., Winslett, M.: Differentially private histogram publication. *VLDB J.* **22**(6), 797–822 (2013). <https://doi.org/10.1007/s00778-013-0309-y>
14. Yu, X., Li, G., Chai, C., Tang, N.: Reinforcement learning with tree-LSTM for join order selection. In: 2020 IEEE 36th International Conference on Data Engineering (ICDE), pp. 1297–1308. IEEE (2020)



# Cost-Based Lightweight Storage Automatic Decision for In-Database Machine Learning

Shuangshuang Cui<sup>1</sup>, Hongzhi Wang<sup>1,2(✉)</sup>, Haiyao Gu<sup>1</sup>, and Yuntian Xie<sup>1</sup>

<sup>1</sup> Harbin Institute of Technology, Harbin, China  
{20S103244,wangzh,1190201423,1181400603}@stu.hit.edu.cn

<sup>2</sup> Peng Cheng Laboratory, Shenzhen, China

**Abstract.** Storage structure decision for a database aims to automatically determine the effective storage structure according to the data distribution and workload. With the integration of machine learning and database becoming closer, complex machine learning tasks are directly executed in database, and need the support of efficient storage structure. The existing storage decision methods are mainly oriented to common workloads and rely on the decision of experienced DBAs, which has low efficiency and high risk of error. Thus, an automated storage structure decision method for in-database machine learning is urgently needed. We propose a cost-based lightweight row-column storage automatic decision system. To the best of our knowledge, this is the first storage structure selection for machine learning tasks. Extensive experiments show that the accuracy of the storage structure above 90%, shorten the task execution time by about 85%, and greatly reduce the risk of decision error.

**Keywords:** AI for DB · Row and column storage · Data partition

## 1 Introduction

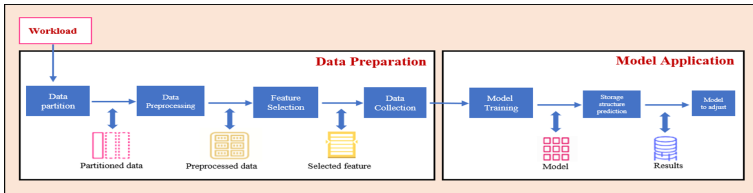
In-database machine learning can lead to orders-of-magnitude performance improvements over current state-of-the-art analytics systems [1]. Complex machine learning tasks require efficient storage structures support. However, The existing decision of storage structure mainly depends on experienced DBAs [2], they make no quantitative analysis of the cost of the storage structures. It is difficult for them to make sure that their experience is correct, and there is a great risk of wasting resources and failing in the execution of tasks [3]. There is an urgent need for a solution that can provide automatic storage structure decision for DBAs and even ordinary database users. There are three challenges: (1) How to partition workloads to make feature extraction most efficient? (2) How to establish a cost model for the storage structure? (3) How to select features and collect relevant feature data efficiency?

- We propose a cost-based intelligent decision system for row and column storage for machine learning in database. To the best of our knowledge, this is the first storage structure selection system for machine learning (Sect. 2).

- We propose a data partitioning algorithm for machine learning task load in database, which is beneficial to increase the data scale in machine learning and improve the accuracy of the model. (Sect. 3.1)
- We propose a feature selection method for machine learning load and a storage engine performance acquisition algorithm, which can help inexperienced users to efficiently decide the appropriate storage structure. (Sect. 3.2 to 3.3)

## 2 System Overview

The core problem of automating the decision on the least cost storage structure is when and how to preprocess and extract features, and how to build the cost model and apply it. We use the read and write execution time of the workload as the cost parameter (Fig. 1).



**Fig. 1.** Workflow of row-column storage intelligent decision method.

**Architecture.** The goal of the row and column storage decision system is to design the storage structure with the least cost for ML workload. The data preparation module mainly includes data partitioning and feature selection. The model application module is consists of module training and module adjusting.

**Workflow.** Too heavy machine learning workload can lead to efficiency problems. Thus, We first solve the data partitioning problem. As for the cost of the row and column storage structure, the key problem lies in the extraction of task features and generation of training data. We select five features that can most affect the execution efficiency and explain the reason. We also design a performance acquisition algorithm. Finally, the model is training.

## 3 Data Preparation

### 3.1 Data Partition

Since the excessive workload of machine learning in database, large-scale data will cause efficiency problems if preprocessed directly. The data partitioning algorithm can be specially oriented to non-uniformly distributed workload with addition of weighting factors for attributes, It can add weights to the workload according to the user’s specific attributes.

The main function of Algorithm 1 is deviding large-scale data. The key idea of the algorithm is described as follows: First, get a certain m-dimension data

---

**Algorithm 1.** Data partition

---

**input:**  $T$ : the data table;  $K$ : the number of partition;  $F_{i,j}$ : the  $j$ th element of the file of  $i$  in  $F$ ;  $a_n$ : the weighting factor;  $T_m$ :  $m$  dimension data selected from  $T$ ;  
**Output:**  $F$ : generated file;

```

1:  $n \leftarrow |T|$ 
2:  $num \leftarrow n/k$ 
3:  $T_m \leftarrow random(T, m)$ 
4: for  $i : 1 \rightarrow m$  do
5:   for  $j : 1 \rightarrow n$  do
6:      $T_m \leftarrow T_m * a_n$ 
7:   end for
8: end for
9: for  $i : 1 \rightarrow m$  do
10:  for  $j : 1 \rightarrow num$  do
11:     $F_{i,j} \leftarrow T_m[num * i + j]$ 
12:  end for
13: end for
14: return  $F$ ;

```

---

from the table to get the  $n * m$ -dimension matrix. Then, calculate the  $N$  data of each dimension of the  $m$ -dimension according to the weighting factor. According to the result of the weighted calculation, we can get the  $n * 1$ -dimension weighted result  $T_m$ . Finally, sort the result and divide it evenly into  $k$  areas.

**Algorithm Complexity.** The algorithm line 4–8 is executed  $m * n$  times, and the line 9–13 is executed  $m * num$  times. So the total time complexity is  $O(n)$ .

### 3.2 Feature Selection

To select the features that represent the cost of the storage engine’s selection, we analyzed and validated data patterns and workload-related characteristics. The influence of feature selection on performance prediction is shown in Table 1.

**Table 1.** Influence of feature selection on performance prediction.

feature influence	keyFieldSize	nonKeyFieldSize	numOffixedLengthField	numOfVarLengthField	numOfRows
The amount of data	√	√			
The amount of entries	√				
Database system processing			√	√	
Batch processing					√

- (1) Key field size and non-key field size: The storage operates in blocks, and the data size of single row affects the amount of data in a block.
- (2) the number of fixed-length fields and variable-length fields: the database system often has different processing methods for fixed-length fields and variable-length fields, which will affect the performance prediction.

- (3) The number of rows involved in a single operation: The single-row amortized performance of a batch operation in the storage engine is higher than that of single-row operation.

### 3.3 Data Collection

The features we choose are the ones that most affect performance prediction, so it is more difficult to capture. We develop the storage engine performance data collection algorithm. Line 2–37 of the algorithm are executed in the row storage database. In line 3<sup>1</sup>–22, data schema is generated randomly and the related performance data required are calculated. For each selected field, its type is assigned randomly as fixed-length or variable-length. Its length is randomly generated according to the following formula:  $x = x_0 + Exponential(\lambda)$ , where  $x$  is the length to be defined for the field in the new table, and  $x_0$  is the length of the field defined in the original table.

Line 11–15 and Line 20–24 calculate the key field size and non-key field size, the number of fixed length field, and the number of variable length field of the new data schema. Line 23 creates a new table based on the above information. Line 24–30 executes  $m$  insertions and records the average insertion time. Line 31–36 executes  $k$  random look-up query statements, recording the number of rows returned each time and query time.

**Complexity Analysis.** It is considered that the execution time of each query operation is  $O(1)$ , the total time complexity is  $O(n^2)$ .

## 4 Storage Decision Model

In the model application stage, there are two difficulties, i.e. how to establish a cost model for the collected performance data and select an appropriate model for training. We attempt to solve these problems in this section.

**Cost Model.** We use the performance data to train the cost model. We analyze the performance data collected above and design the cost model. For a given workload and data schema  $S$ , our model calculates the cost of row and column storage respectively as shown below:

$$Cost_{row}(S) = W_1 * V_{row-insert}(S) + W_2 * V_{row-select}(S)$$

$$Cost_{column}(S) = W_1 * V_{column-insert}(S) + W_2 * V_{column-select}(S)$$

Where  $W_1$  denotes the number of insert in the workload,  $W_2$  denotes the number of select in the workload, and  $V_x$  denotes the predicted value.

**Model Training.** The advantage of in-database machine learning is efficient. It requires the training process of cost model to be efficient while ensuring accuracy. We chose XGBoost learners [4] to train the regression model, which is a tradeoff between performance and prediction accuracy for lightweight.

---

<sup>1</sup> a - total key field size; b - total non-key field size; c - the number of fixed-length fields; d - the number of variable-length fields.

---

**Algorithm 2.** Storage engine performance data acquisition
 

---

**input:** *Table*: original data; *n*: the number of schema to be generated; *m*: the number of insert; *k*: the number of select.

**Output:**  $S_{row-insert}$ : insert execution time in row;  $S_{row-select}$ : select e time in row storage;  $S_{column-insert}$ : insert time in column;  $S_{column-select}$ : select time in column.

```

1:  $S_{row-insert}, S_{row-select}, S_{column-insert}$  and  $S_{column-select} \leftarrow \emptyset$ 
2: for  $i : 1 \rightarrow n$  do
3:    $a, b, c$  and  $d \leftarrow 0$ ;
4:    $D \leftarrow \emptyset$ ;
5:    $D \leftarrow D \cup Table.KeyFieldSet$ ;
6:    $p \leftarrow randomInt(0, |Table.NonKeyFieldSet|)$ ;
7:    $A \leftarrow random\ select\ p\ elements\ from\ Table.NonKeyFieldSet$ ;
8:    $D \leftarrow D \cup A$ ;
9:   for each column  $C_i \in D$  do
10:     $C_{i.type} \leftarrow random\ select\ from\ fixed - length, variable - length$ 
11:    if  $C_{i.type} == fixed - length$  then
12:       $c \leftarrow c + 1$ ;
13:    else
14:       $d \leftarrow d + 1$ ;
15:    end if
16:     $C_{i.length} \leftarrow random\ select\ from\ [Table.getFieldLength(C_i), +\infty)$ ;
17:    if  $C_i \in Table.KeyFieldSet$  then
18:       $a \leftarrow a + C_{i.length}$ ;
19:    else
20:       $b \leftarrow b + C_{i.length}$ ;
21:    end if
22:  end for
23:  create new table Table2 with D
24:   $timeInsert \leftarrow 0$ ;
25:  for  $j : 1 \rightarrow m$  do
26:    insert Table.row[j] into Table2;
27:     $timeInsert \leftarrow timeInsert + the\ execution\ time\ of\ line29$ ;
28:  end for
29:   $timeInsert \leftarrow timeInsert/m$ ;
30:   $S_{row-insert} \leftarrow S_{row-insert} \cup \{(a, b, c, d, 1, timeInsert)\}$ 
31:  for  $j : 1 \rightarrow k$  do
32:    random execute a select statement of SQL on Table2;
33:     $timeSelect \leftarrow the\ execution\ time\ of\ line35$ ;
34:     $count \leftarrow the\ number\ of\ rows\ returned\ in\ line35$ ;
35:     $S_{row-select} \leftarrow S_{row-select} \cup \{(a, b, c, d, count, timeSelect)\}$ 
36:  end for
37: end for
38: repeat line2 to line37 in column - database
39: return  $S_{row-insert}, S_{row-select}, S_{column-insert}, S_{column-select}$ ;

```

---

**Loss Function.** The loss function adopts the common loss function of XGBoost model, and its general form is as follows:

$$Obj^{(t)} = \sum_{i=1}^n l(y_i, y_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) + constant$$

The first term describes the training error, and the  $l$  function is used to measure the error between the predicted value and the true value.  $y_i$  is the true value,  $y_i^{(t-1)}$  is the predicted value of the model obtained from the previous  $t-1$  rounds of training, and  $f_t(x_i)$  is the function to be trained in the  $t$  round.

$$\Omega(f) = \gamma T + 1/2\lambda\|\omega\|^2$$

Where  $T$  represents the number of leaf nodes, and  $\omega$  represents the fraction of leaf nodes. The model training objective requires that the prediction error should be as small as possible.

**Lightweight.** To make the model more widely used, it is required that the model must be lightweight and migratable. We package the two proposed methods, and design the odbc interface, which can connect to database directly.

## 5 Experiments

### 5.1 Accuracy Evaluation

**Table 2.** Comparison of model accuracy.

	Our model		Reference model	
	Insert	Select	Insert	Select
Row-oriented	<b>95.04%</b>	<b>91.40%</b>	<b>97.25%</b>	<b>90.07%</b>
Column-oriented	<b>92.18%</b>	<b>96.77%</b>	<b>92.37%</b>	<b>93.65%</b>

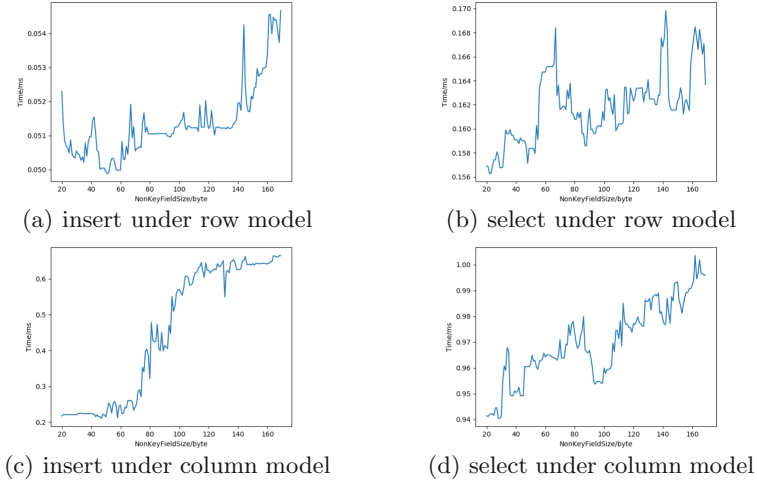
The experimental database is selected as OpenGauss1.1.0 [7]. The TPC-H public dataset was selected as benchmark.  $accuracy = 1 - (V_{predict} - V_{truth})/V_{truth}$  [8], where the  $V_{predict}$  is the time predicted. And the  $V_{truth}$  is the value of execution time of database feedback. The accuracy of our models are above 90% (Table 2). We compare the accuracy with the model proposed Wei et al. [5]. Under the row-oriented, the accuracy of our insert model is 2.21% lower than theirs. Because they use LSM storage engine, which has superior write performance.

### 5.2 Feature Section Effectiveness Evaluation

We test five selected features to verify the validity of selected features. Due to the length of the paper, the process of verifying the influence of “non-key field length” on SQL execution time is listed. Under the premise of controlling a single variable, we set the key field length = 16, the number of fixed-length fields and variable-length fields = 5. We record the predicted value in row/column storage with the change of non-key field length. The results are shown in Fig. 2.

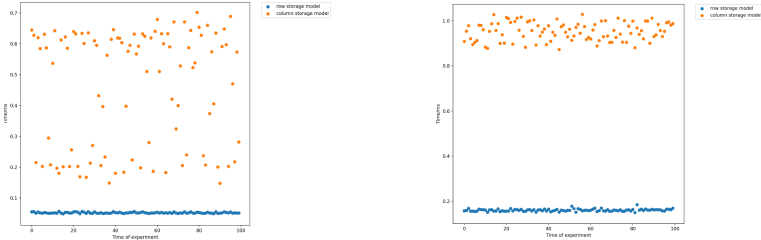
Although the predicted time of each models somewhat fluctuates, it generally increases with the augment of the length of non-key fields, because the storage engine operates on a block, and non-key fields affect the size of single row data. It affects the amount of data in a block which affects the execution time of the SQL in turn. It meets the expectation of feature selection, and the feature extraction time is the fastest while ensuring the model accuracy.





**Fig. 2.** The relationship between time predicted and non-key fields' length

### 5.3 Compare the Applicable Workloads of Row/Column Model



**Fig. 3.** Insert and select execution time predicted of row and column storage structure

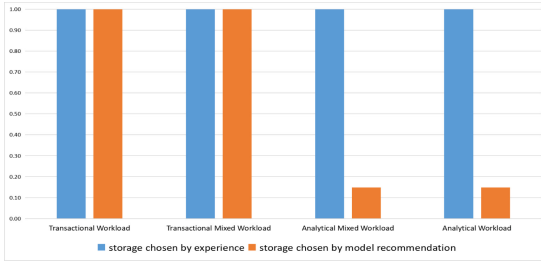
We design the experiment repeats 100 times. Each time randomly generating data schema features, and comparing the predicted time values given by the row and column storage model.

The results (Fig. 3) show row storage require less execution time and predicted results more consistently under the vast majority of insertion and selection loads. This fully demonstrates the importance and necessity of analysing the storage cost quantitatively.

### 5.4 Comparisons on Various Workloads

The experimental results were normalized, and the experimental results were shown in Fig. 4.

Under transactional and transactional mixed workloads, row storage is selected according to experience, and the model recommendation structure is consistent with experience. Under analytical and analytical mixed workloads, the application of the model recommendation structure will result in approximately 85% performance improvement.



**Fig. 4.** Performance comparison between the storage structure selected in experience and the storage structure recommended by the model under different workloads.

## 6 Conclusions

This paper proposes a cost-based intelligent decision for row-column storage, so that the database can efficiently choose a storage structure suitable for the data even when the performance of the database is unknown. From experimental results, using the method proposed in this paper to determine the storage structure can shorten the task execution time by about 85%, greatly reduce the risk of decision errors, and greatly improve the efficiency of task execution. In the future, we plan to verify our proposed algorithm on more row and column storage databases to further enhance the generalization ability of the model.

**Acknowledgements.** This paper was supported by NSFC grant (U1866602, 71773025). The National Key Research and Development Program of China (2020YFB1006104).

## References

1. Olteanu, D.: The relational data borg is learning. *PVLDB* **13**(12), 3502–3515 (2020)
2. De Marchi, F., Lopes, S., Petit, J.-M., Toumani, F.: Analysis of existing databases at the logical level: the DBA companion project. *ACM SIGMOD Rec.* **32**(1), 47–52 (2003)
3. Park, Y., Zhong, S., Mozafari, B.: Quicksel: quick selectivity learning with mixture models. In: *Proceedings of the 2020 SIGMOD* (2020)
4. Chen, T., Guestrin, C.: XGBoost: a scalable tree boosting system. In: *Proceedings of the 22nd ACM SIGKDD* (2016)
5. Wang, H., Wei, Y., Yan, H.: Automatic storage structure selection for hybrid workload (2020)

# **Data Mining and Applications**



# NP-PROV: Neural Processes with Position-Relevant-Only Variances

Xuesong Wang<sup>1</sup>, Lina Yao<sup>1</sup>(✉), Xianzhi Wang<sup>2</sup>, Feiping Nie<sup>3</sup>,  
and Boualem Benatallah<sup>1</sup>

<sup>1</sup> University of New South Wales, Sydney 2052, Australia  
{xuesong.wang1, lina.yao}@unsw.edu.au

<sup>2</sup> University of Technology Sydney, Sydney 2007, Australia

<sup>3</sup> Northwestern Polytechnical University, Xi'An 710060, China

**Abstract.** Neural Processes (NPs) families encode distributions over functions to a latent representation given a set of context data, and decode posterior mean and variance at unknown locations. Since mean and variance are derived from the same latent space, they may fail on out-of-domain tasks where fluctuations in function values amplify the model uncertainty. We present a new member named Neural Processes with Position-Relevant-Only Variances (NP-PROV). NP-PROV hypothesizes that a target point close to a context point has small uncertainty, regardless of the function value at that position. The resulting approach derives mean and variance from a function-value-related space and a position-related-only latent space separately. Our evaluation on synthetic and real-world datasets reveals that NP-PROV can achieve state-of-the-art likelihood while retaining a bounded variance when drifts exist in the function value.

**Keywords:** Uncertainty evaluation · Neural processes ·  
Position-relevant-only variance

## 1 Introduction

Neural networks (NNs) are proven effective in various machine learning tasks for making deterministic predictions. They have the flexibility of describing and fitting any type of data distributions. Nevertheless, most neural networks are incapable of evaluating model stochasticity. They cannot handle tasks where prediction uncertainty is equally crucial, e.g., autonomous driving [18], disease diagnosis [13], and stock market forecasting [3]. In contrast, Gaussian processes (GPs) that model data with an infinite sequence of correlated normal distributions are intrinsically suitable for such problems. Conditioned on some prior knowledge, e.g., context points with positions  $\mathbf{X}$  and function values  $\mathbf{Y}$ , they are able to infer the likelihood of target values  $\mathbf{Y}_*$  at unknown locations  $\mathbf{X}_*$ . Despite the advantages, GPs require designing data-specific kernel functions. Also, the cubic computational cost of matrix inversion impedes GPs from handling large-scale data.

The recent progress in models based on Neural Process (NP) [7] has advanced GPs with the fast forward propagation and powerful feature representations of NNs. Basic NPs represent a stochastic process with an encoder-decoder network structure. They encode context data  $(\mathbf{X}, \mathbf{Y})$  to a latent representation  $\mathbf{Z} \sim \mathbf{P}(\mathbf{Z}|\mathbf{X}, \mathbf{Y})$  and decode it to a posterior probability of the target data based on their relationships with the context points  $\mathbf{Y} \sim \mathbf{P}(\mathbf{Y}_*|\mathbf{Z}; \mathbf{X}_*)$ . Recent years have witnessed the success of a family of NP-variants, such as Attentive NP [10], Convolutional Conditional NP [9], and Sequential NP [16]. They improve NP via aggregating context knowledge non-linearly and induce spatial or temporal biases among target points. Nonetheless, they still decode mean and variance from the same latent variable—meaning that their variances are correlated to the context values  $\mathbf{Y}$ . When there are shifts in the future testing sets (e.g., stock market price with incremental trends), fluctuations in  $\mathbf{Y}$  can severely amplify model uncertainty. Therefore, we hypothesize that the precondition of a stabilized model for out-of-domain tasks is that the variance inference being reliant only on locations  $\mathbf{X}$  meanwhile unrelated to the function values  $\mathbf{Y}$ .

In this paper, we introduce a new member named Neural Processes with Position-Relevant-Only Variances (NP-PROV), which derives mean and variance functions from two coupled latent spaces. Mean values are related to context values  $\mathbf{Y}$ , self-correlation within context locations  $\mathbf{X}$ , and cross-correlations between context positions  $\mathbf{X}$  and target positions  $\mathbf{X}_*$ . Variance values exclude function values  $\mathbf{Y}$  yet are associated with the self-correlations within the target positions. Our main contributions are as follows:

- We designed an auto-encoder module associated with self-correlations between data points. It reconstructs context data through the module and reduces model uncertainty in high self-correlation scenarios.
- The approach derives mean and variance values from two coupled latent spaces. The position-relevant-only variances prevent the model from estimating oscillating uncertainty when function values are out of the training range.
- The proposed model achieves state-of-the-art performance over on-the-grid and off-the-grid datasets. Specifically, three GP-based kernels and a real-world time series are adopted for off-the-grid tasks. Four image inpainting tasks are evaluated: MNIST, SVHN, celebA and miniImageNet.

## 2 Background

*Gaussian Processes.* Let  $\mathbf{Y} = [y_1, \dots, y_N]^\top \in \mathbb{R}^N$  be a set of  $N$  observations (as what we call context set here) at input locations  $\mathbf{X} = [x_1, \dots, x_N]^\top \in \mathbb{R}^{N \times D}$ , a function  $f : \mathbb{R}^D \mapsto \mathbb{R}$ , and a likelihood  $p(\mathbf{Y}|\mathbf{F}; \mathbf{X})$ , where  $\mathbf{F} = f(\mathbf{X})$  denotes the function values at the input locations. A Gaussian process (GP) prior is placed on the function  $f$ ; it models all function values as a jointly Gaussian distribution to infer  $f$ . The prior has a mean function  $m(\cdot) : \mathbb{R}^D \mapsto \mathbb{R}$  and a covariance function  $\mathcal{K}(\cdot, \cdot) : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$ . The generative model of the corresponding GP is as follows:

$$p(\mathbf{Y}|\mathbf{F}; \mathbf{X}) = \mathcal{N}(m(\mathbf{X}), \mathcal{K}(\mathbf{X}, \mathbf{X}^\top)) \quad (1)$$

GP hypothesizes similar function values of two inputs that are highly correlated. Given a context set  $(\mathbf{X}, \mathbf{Y})$ , we can perform probabilistic inference and assign posterior distributions over unknown locations  $\mathbf{X}_* = [x_{*1}, \dots, x_{*M}]^\top$  from the same function. The posterior distribution also follows a joint Gaussian distribution  $p(\mathbf{Y}_*|\mathbf{F}; \mathbf{X}_*, \mathbf{X}, \mathbf{Y}) = \mathcal{N}(\mu_*, \Sigma_*)$ :

$$\begin{aligned} \mu_* &= m(\mathbf{X}_*) + \mathbf{K}_*^\top \mathbf{K}^{-1}(\mathbf{Y} - m(\mathbf{X})) \\ \Sigma_* &= \mathbf{K}_{**} - \mathbf{K}_*^\top \mathbf{K}^{-1} \mathbf{K}_* \end{aligned} \quad (2)$$

where  $\mathbf{K} = \mathcal{K}(\mathbf{X}, \mathbf{X}^\top)$ ,  $\mathbf{K}_* = \mathcal{K}(\mathbf{X}, \mathbf{X}_*^\top)$ , and  $\mathbf{K}_{**} = \mathcal{K}(\mathbf{X}_*, \mathbf{X}_*^\top)$ . Intuitively, when there is no trend in observations (i.e.,  $m(\mathbf{X}_*) = m(\mathbf{X}) = 0$ ), the mean of  $\mathbf{Y}_*$  is a linear combination of elements in  $\mathbf{Y}$ . The weights of those elements are associated with the self-correlation of  $\mathbf{X}$  (the  $\mathbf{K}^{-1}$  part) and cross-correlation between  $\mathbf{X}$  and  $\mathbf{X}_*$  (the  $\mathbf{K}_*^\top$  part). The variance equals the total uncertainty of  $\mathbf{X}_*$  (the  $\mathbf{K}_{**}$  part) minus the certainty induced from  $\mathbf{X}$  (the  $\mathbf{K}_*^\top \mathbf{K}^{-1} \mathbf{K}_*$  part) and is irrelevant to function values  $\mathbf{Y}$ .

*Convolutional Conditional Neural Processes.* As a type of latent model, NPs inherit distribution consistency from GPs, where the context set  $(\mathbf{X}, \mathbf{Y})$  and the target data set  $(\mathbf{X}_*, \mathbf{Y}_*)$  are sampled from the same function and share a latent variable.

Convolutional Conditional Neural Processes (ConvCNP) differ from other NP families in handling testing data outside the range of training data. They introduce a discretization of a continuous range of  $\mathbf{X} \cup \mathbf{X}_*$  named  $\mathbf{X}_t$ . Convolution operations enable NPs to focus on local receptive fields on the grid and to generalize out-of-range tasks.

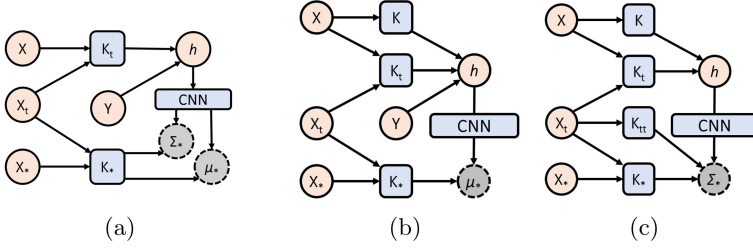
The posterior distribution of  $p(\mathbf{Y}_*|\mathbf{F}; \mathbf{X}_*, \mathbf{X}_t, \mathbf{X}, \mathbf{Y}) = \mathcal{N}(\mu_*, \Sigma_*)$  is also a Gaussian distribution:

$$\begin{aligned} \mu_* &= \psi_{*\mu} \{ \mathbf{K}_*^\top \text{CNN}[\psi_t(\mathbf{K}_t^\top \mathbf{Y})] \} \\ \Sigma_* &= \psi_{*\Sigma} \{ \mathbf{K}_*^\top \text{CNN}[\psi_t(\mathbf{K}_t^\top \mathbf{Y})] \} \end{aligned} \quad (3)$$

where  $\mathbf{K}_* = \mathcal{K}_*(\mathbf{X}_*, \mathbf{X}_*^\top)$ ,  $\mathbf{K}_t = \mathcal{K}_t(\mathbf{X}_t, \mathbf{X}_t^\top)$  are covariance functions with learnable length scales;  $\psi_t, \psi_{*\mu}$  and  $\psi_{*\Sigma}$  are multi-layer perceptrons (MLPs); the graphical model of ConvCNP is described in Fig. 1 (a);  $\psi_t(\mathbf{K}_t^\top \mathbf{Y})$  encodes prior knowledge about the function distribution  $h$  on the entire grid  $\mathbf{X}_t$ ;  $\text{CNN}(\cdot)$  enables model to exhibit translation invariance;  $\psi_{*\mu}(\mathbf{K}_*^\top \cdot)$  and  $\psi_{*\Sigma}(\mathbf{K}_*^\top \cdot)$  translate the knowledge to predict target mean and standard deviation for  $p(\mathbf{Y}_*)$ .

### 3 Neural Processes with Position-Relevant-Only Variances

$\mathbf{K}_t$  evaluates correlation between  $\mathbf{X}_t$  and  $\mathbf{X}$ , and  $\mathbf{K}_*$  evaluates cross correlation between  $\mathbf{X}_*$  and  $\mathbf{X}_t$ . Therefore,  $\psi_t(\mathbf{K}_t \mathbf{Y})$  and  $\psi_*(\mathbf{K}_*^\top \cdot)$  in ConvCNP act as two



**Fig. 1.** Graphical Models for (a) ConvCNP (b) NP-PROV mean and (c) NP-PROV variance

cascaded cross-correlation modules in GPs (the  $\mathbf{K}_*^\top$  part). We propose a new member Neural Processes with Position-Relevant-Only Variances (NP-PROV) to improve ConvCNP in two ways:

1. Considering self-correlations within context data. ConvCNP usually considers cross-correlations between target data and context data only. It is possible to further reduce the model uncertainty near the region where context data have high self-correlations.
2. Relying the variance only on the positions  $\mathbf{X}$ . By removing the original reliance on function values  $\mathbf{Y}$  in ConvCNP, we can effectively prevent the model from amplifying uncertainty caused by value fluctuations if it is made to be only associated with relative distances among locations (as in GP).

In the following, an off-the-grid scenario with continuous inputs and an on-the-grid scenario with intrinsically discretized inputs will be detailed.

### 3.1 Off-the-Grid Scenario

*Mean Function.* We follow convCNP to construct the model with two cascaded GP-alike layers. The first layer maps  $\mathbf{X}$  to a uniformly discretized grid space  $\mathbf{X}_t = [x_1, \dots, x_t]^\top$  built on the lower and upper range of  $\mathbf{X} \cup \mathbf{X}_*$ . The second layer maps the space back to  $\mathbf{X}_*$ . As illustrated in Fig. 1(b) first line of the workflow, an auto-encoder structure is computed to match  $\mathbf{K}^{-1}\mathbf{Y}$  in GPs first. Suppose  $\mathbf{K} = \mathcal{K}(\mathbf{X}, \mathbf{X}^\top) \in \mathbb{R}^{N \times N}$  is the self-covariance function;  $\psi_E$  and  $\psi_D$  are MLPs. Since in GPs,  $\mathbf{K}\mathbf{K}^{-1}\mathbf{Y} = \mathbf{Y}$ , we hence use an auto-encoder structure to map  $y_n$  to a representation  $\tilde{h}_n$  and then project it back to  $y_n$ . By minimizing the reconstruction loss  $\mathcal{L}_2$ , we are able to mimic matrix inversion through  $\tilde{h}_n$ :

$$\begin{aligned} \tilde{h}_n &= \psi_E(\mathbf{K}y_n) \\ \tilde{y}_n &= \psi_D(\mathbf{K}\tilde{h}_n), \quad \mathcal{L}_2 = \frac{1}{N} \sum_{n=1}^N (y_n - \tilde{y}_n)^2 \end{aligned} \quad (4)$$

Then, we can compute cross-correlation weights associated with  $\mathbf{K}_t = \mathcal{K}(\mathbf{X}_t, \mathbf{X}^\top) \in \mathbb{R}^{T \times N}$  and concatenated with the self-correlation part:

$$\begin{aligned} h'_t &= \mathbf{K}_t[y_n \quad \mathcal{I}] \\ h_t &= \psi_t\left(\left[\frac{1}{N} \sum_{n=1}^N \tilde{h}_n \quad h'_t\right]\right) \end{aligned} \quad (5)$$

where  $h'_t$  maps the  $n$ -th context point to the  $t$ -th data in  $\mathbf{X}_t$ .  $\mathcal{I}$  is an additional identity density channel appended to  $\mathbf{Y}$  to normalize the scale factor in functional values. A new linear transformation  $\psi_t$  fuses averaged self-correlation and cross-correlation information to a latent variable  $h_t$  on the  $t$ -th data of  $\mathbf{X}_t$  as shown the second line of workflow in Fig. 1 (b).

Then a CNN takes the obtained condition  $h_t$  to suffice translation invariance. It uses a UNet structure to capture global and local patterns. The CNN enables NP-PROV to predict out-of-domain tasks by handling locations outside the training range through filter sliding. When projecting the output back from  $\mathbf{X}_t$  to

$\mathbf{X}_*$ , cross-correlation between  $\mathbf{X}_*$  and  $\mathbf{X}_t$  is represented as  $\mathbf{K}_* = \mathcal{K}(\mathbf{X}_*, \mathbf{X}_t^\top) \in \mathbb{R}^{M \times T}$ . The overall mean function of the  $m$ -th target point in  $\mathbf{X}_*$  is as follows:

$$\mu_m^* = \psi_\mu^*(\mathbf{K}_* \text{CNN}(h_t)) \quad (6)$$

where  $\psi_*$  is the mean MLP transformation, as shown in the third line of the workflow in Fig. 1 (b).

*Covariance Function.* Based on (2), GP replaces the original values  $\mathbf{Y}$  in  $\mathbf{K}_*^\top \mathbf{K}^{-1} \mathbf{Y}$  with the covariance  $\mathbf{K}_*$  in  $\mathbf{K}_*^\top \mathbf{K}^{-1} \mathbf{K}_*$ , and add a module associated with self-correlations within the target data  $\mathbf{K}_{**}$ . Similarly, as shown in Fig. 1 (c), we define  $\mathbf{K}_{t'} = \mathcal{K}(\mathbf{X}, \mathbf{X}_t^\top) \in \mathbb{R}^{T \times N}$ ;  $\mathbf{K}_{tt} = \mathcal{K}(\mathbf{X}_t, \mathbf{X}_t^\top) \in \mathbb{R}^{T \times T}$  and two new MLP mappings  $\psi'_{t'}$  and  $\psi_{tt}$ :

$$y'_n = \psi'_{t'}(\mathcal{I} \mathbf{K}_{t'}), \quad h_{tt} = \psi_{tt}(\mathcal{I} \mathbf{K}_{tt}) \quad (7)$$

where  $\mathcal{I}$  is an identity matrix that helps calculating the sum of the rows of the followed matrix. In this way,  $y'_n$  maps  $\mathbf{X}_t$  back to  $\mathbf{X}$  and replaces values  $\mathbf{Y}$  in the mean function. And more importantly, it does not explicitly depend on the function values  $y$ . This value-irrelevant input  $y'_n$  will generate a new  $\tilde{h}_t$  using (5). After the convolution on the new  $\tilde{h}_t$ , the result is concatenated with  $h_{tt}$ . The overall covariance function is as follows:

$$\Sigma_m^* = \psi_\Sigma^*(\mathbf{K}_* [\text{CNN}(\frac{1}{T} \sum_{t=1}^T h_{tt}), \tilde{h}_t]) \quad (8)$$

Again, as in Fig. 1(c), the added elements in NP-PROV can retain reasonable variance regardless of drifts in function values when compared with ConvCNP.



*Inference and Learning.* With  $\mu_\star$  and  $\Sigma_\star$ , the posterior distribution of  $\mathbf{Y}_\star$  follows a multivariate normal distribution:  $\mathcal{N}(\mu_\star, \Sigma_\star)$ . The training objective is to maximize the log-likelihood of target outputs meanwhile minimizing the reconstruction loss on the context set, given all parameters defined as  $\Theta$ :

$$\theta^\star = \arg \max_{\theta \in \Theta} \sum_{m=1}^M \log p(y_m | \mathcal{N}(\mu_\star, \Sigma_\star)) - \frac{1}{N} \sum_{i=1}^N (y_i - \tilde{y}_i)^2 \quad (9)$$

### 3.2 On-the-Grid Scenario

CNN can be applied to On-the-grid data (e.g., images), which are discretized and represent better spatial correlations between context and target data with little efforts. We describe the context inputs  $\mathbf{X}$  using a context mask  $\mathbf{M}$ , where an element value is 1 if a pixel is revealed and 0 otherwise. The target inputs  $\mathbf{X}_\star$  formulate the whole image.

*Mean Function.* Similar to off-the grid data, self-correlations and cross-correlations weights are computed for the latent variable extraction. Instead of adopting kernel functions, convolutions are implemented on the context masks  $\mathbf{M}$  and the context values  $\mathbf{M} \odot \mathbf{Y}$ :

$$\begin{aligned} \tilde{h}_n &= \psi_E(\mathbf{M}), \quad \tilde{M} = \psi_D(\tilde{h}_n), \quad \mathcal{L}_2 = \|\mathbf{M} - \tilde{M}\|_2 \\ h_t &= \psi_t([\tilde{h}_n, \mathbf{M} \odot \mathbf{Y}]) \end{aligned} \quad (10)$$

where  $\mathbf{Y}$  represents a full image,  $\psi_E, \psi_D, \psi_t$  are convolutions.  $\psi_E$  aims to extract self-correlations within unmasked points.  $\psi_t$  shows cross correlations within unmasked pixel values. Then, a UNet CNN structure maps the concatenated latent variable to a translation invariant representation. Finally, a new convolution  $\psi_{\star\mu}$  maps the representation to the posterior mean:

$$\mu_m^\star = \psi_\mu^\star(\text{CNN}(h_t)) \quad (11)$$

*Covariance Function.* We directly use  $\mathbf{M}$  to generate the output-irrelevant latent variable  $h$  for variance functions. Also, we add a self-correlation layer on the target masks, i.e., an identity matrix  $\mathcal{I}$ :

$$\tilde{h}_t = \psi'_t(\mathbf{M}), \quad h_{tt} = \psi_{tt}(\mathcal{I}) \quad (12)$$

The overall function learns a new transformation  $\psi_\Sigma^\star$  that maps  $h$  and  $h_{\star\star}$  to the posterior covariance. The objective is to maximize a log-likelihood to recover the whole image and to minimize the reconstruction error of the mask simultaneously.

$$\Sigma_m^\star = \psi_\Sigma^\star(\text{CNN}[h_{tt}, \tilde{h}_t]) \quad (13)$$

## 4 Experiments

We conduct few-shot regression tasks on off-the-grid 1d datasets and on-the-grid images to evaluate the effectiveness of NP-PROV.

### 4.1 Off-the-Grid Datasets

We aim to maximize the likelihood of the outputs  $\mathbf{Y}_*$  at unknown locations  $\mathbf{X}_*$ , given observations  $\mathbf{Y}$  at input locations  $\mathbf{X}$ . We are interested in the following questions: (a) Are the prediction and uncertainty estimation reasonable? (b) How will the model perform when the testing range of  $X$  and  $Y$  goes beyond the training data, i.e., out-of-domain tasks? (c) How will the self-correlation affect model prediction? We compare the results of four state-of-the-art neural process members: Neural Process (NP) [7], Conditional Neural Process (CNP) [6], Attentive Neural Process (ANP) [10], and Convolutional Conditional Neural Process (ConvCNP) [9]. We use three challenging kernels to generate synthesised Gaussian processes functions, according to [9]:

- EQ:  $\mathcal{K}(x, x') = e^{-\frac{1}{2}(\frac{x-x'}{0.25})^2}$
- Matern  $-\frac{5}{2}$ :  $\mathcal{K}(x, x') = (1 + 4\sqrt{5}d + \frac{5}{3}d^2)e^{-\sqrt{5}d}$  with  $d = 4|x - x'|$
- Weakly periodic:  $\mathcal{K}(x, x') = e^{-\frac{1}{2}(f_1(x) - f_1(x'))^2 - \frac{1}{2}(f_2(x) - f_2(x'))^2} \cdot e^{-\frac{1}{8}(x - x')^2}$ , with  $f_1(x) = \cos(8\pi x)$  and  $f_2(x) = \sin(8\pi x)$

The training data range within  $x \in [-2, 2]$  for GP-sampled datasets. Also, a real-world time series dataset Smart Meter [1] is added. It contains energy consumption readings from a sample of 5,567 London Households between November 2011 and February 2014. We select timestamp in days as the input  $\mathbf{X}$  and consumption in kWh/ half-hour as the output  $\mathbf{Y}$ . The  $x$  range is set to  $[0, 2]$ , representing a relative 2-day window from a random clip on the time axis. The number of context points and target points in each task are uniformly distributed in  $\mathcal{U}(3, 50)$ . A batch of 16 tasks is generated and the total number of tasks in one epoch is 256. We train each model for 200 epochs and then test them using 2,048 new generated tasks for 6 times.

Table 1 shows the log-likelihood (on the probability density function) of five methods. Model performance is presented in Fig. 2, which supplement mean and variance results from the original GP as a reference for synthesized datasets. Table 1 shows both ConvCNP and NP-PROV outperform others significantly. The result of NP is quite unstable, due to stochastic encoding of the latent variable. The first column of Fig. 2 shows the results with the testing range  $[-5, 5]$  for the GP-sampled datasets. NP, CNP, and ANP predict oscillated mean values; therefore, we omit their variance values for display clarity. Both NP-PROV and ConvCNP match well with the original GP. This advantage sources from convolution, where filters can slide along the axis. NP-PROV usually predicts a narrower variance than ConvCNP when the target points are adjacent to context points. This might be attributed to more uncertainty reduction from context self-correlation. When Smart Meter is extended to the interval  $[-1, 5]$ ,

the predicted variance becomes much higher than NP-PROV (as it only adopts cross-correlation), and the target points become far from the context points. NP-PROV mitigates this issue by leveraging the self-correlation on target data.

**Table 1.** Log-likelihood of off-the-grid datasets (mean  $\pm$  standard deviation)

Model	EQ	Matern	Weakly periodic	Smart meter
NP	$-1.20 \pm 0.43$	$-0.82 \pm 1.95$	$-1.75 \pm 1.36$	$0.93 \pm 0.23$
CNP	$-1.10 \pm 0.02$	$-1.36 \pm 0.03$	$-2.04 \pm 0.02$	$1.81 \pm 0.06$
ANP	$-0.35 \pm 0.01$	$-0.75 \pm 0.02$	$-2.09 \pm 0.03$	$1.66 \pm 0.05$
ConvCNP	$2.15 \pm 0.05$	$0.83 \pm 0.06$	$-1.15 \pm 0.01$	<b><math>2.65 \pm 0.06</math></b>
NP-PROV	<b><math>2.20 \pm 0.02</math></b>	<b><math>0.90 \pm 0.03</math></b>	<b><math>-1.00 \pm 0.02</math></b>	$2.32 \pm 0.05$
ConvCNP (self)	<b><math>77.60 \pm 2.33</math></b>	$40.26 \pm 0.81$	$-47.63 \pm 0.75$	$0.95 \pm 0.00$
NP-PROV (self)	$77.55 \pm 2.61$	<b><math>44.14 \pm 1.03</math></b>	<b><math>-40.96 \pm 0.89</math></b>	<b><math>0.99 \pm 0.01</math></b>

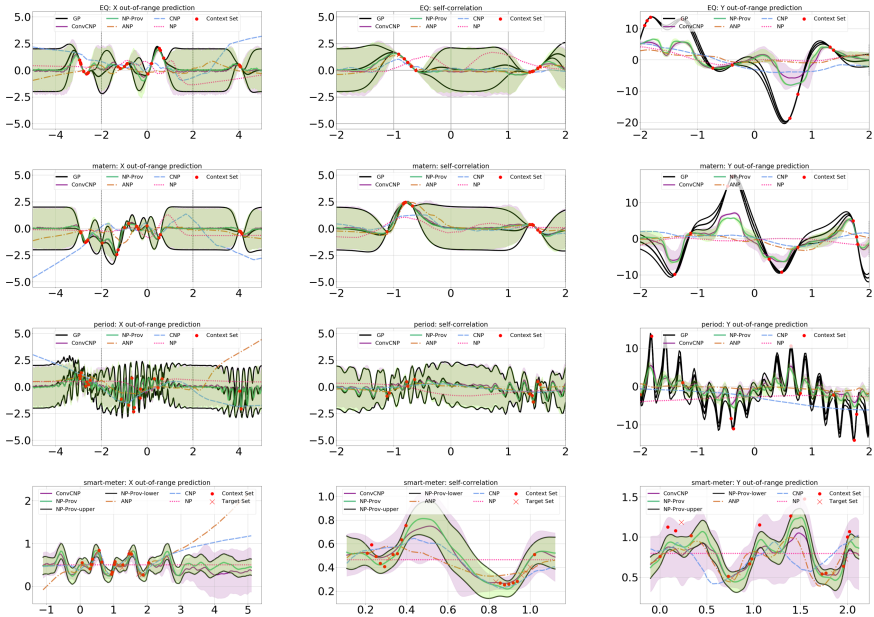
We analyze the impact of self-correlation on the model. In the second column of Fig. 2, we give context data in smaller intervals of  $[-1, -0.5] \cup [1.2, 1.7]$  for GP-sampled datasets and  $[0.2, 0.4] \cup [0.8, 1.1]$  for Smart Meter to get higher self-correlations between context points. Also, in Table 1, we present the log-likelihood of ConvCNP (self) and NP-PROV (self) where target points are adjacent to these compacted regions of context data. NP-PROV predicted lower variances on target points near compacted regions, indicating the ability to capture the self-correlation between context points.

Finally, we modify the testing GP generator to  $y = 10 \times \mathcal{GP}(x)$  and 1.5 times of the original Smart Meter to evaluate model performance when testing  $\mathbf{Y}$  goes beyond the training range. The third column of Fig. 2 reveals that all methods are too conservative in adaptation due to the normalization effect of activation layers. However, the variance of NP-PROV on the context data is close to zero, and it is not affected by the explosion of target values, which suffice the hypothesis of a stochastic process. Such an advantage is crucial when there are shifts in time series data.

## 4.2 On-the-Grid Datasets

Given different proportions of pixel values, the objective is to complete the whole image and estimate uncertainty at unknown pixels. Four image datasets are introduced: MNIST, SVHN, celebA  $32 \times 32$  and miniImageNet [5]. Except MNIST, all the other datasets are RGB images. MiniImageNet is used to verify the model capability of recovering images of unseen classes. The context points are sampled from  $\mathcal{U}(\frac{n_{total}}{100}, \frac{n_{total}}{2})$  and the target points are  $n_{total}$ , i.e., the whole image.

Table 2 demonstrates the log-likelihood of the comparing methods on testing sets. The results of miniImageNet are only displayed for ConvCNP and NP-PROV since other baselines are overwhelmed by the image size  $84 \times 84$ . Figure 3

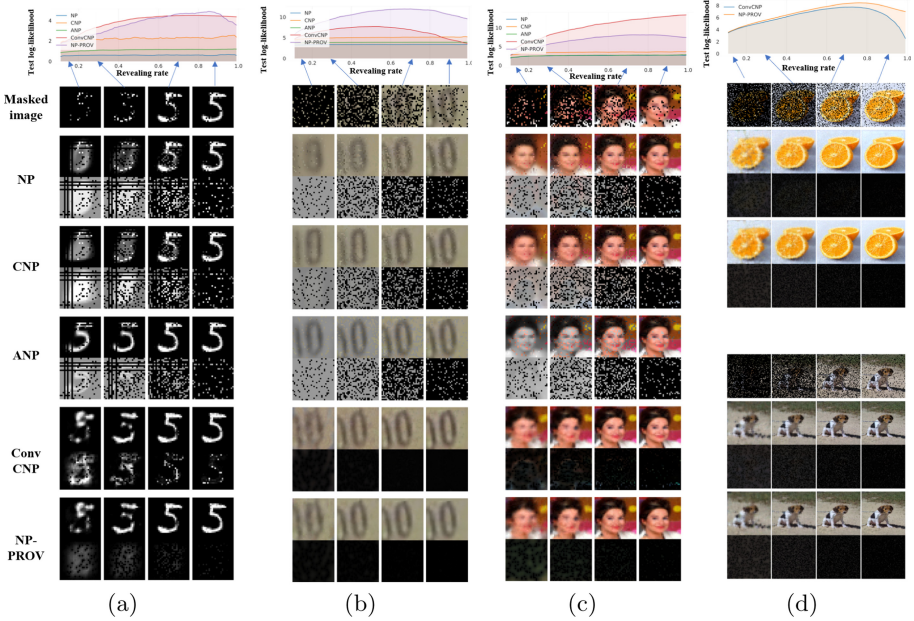


**Fig. 2.** Model predictions on 4 datasets: EQ (the first row), Matern (the second row), Weakly period (the third row) and Smart meter (the fourth row). The first column is out-of-x-range prediction, the second column is self-correlation prediction, the third column is out-of-y-range prediction.

shows the prediction results on the datasets. NP-PROV and miniImageNet massively outperform other baselines. NP-PROV achieves better results on SVHN and miniImageNet. From the variance results in Fig. 3, most of the methods don’t exhibit explainable variances in these two methods, because they are similar to “unsupervised” scenarios where one or a few images represent a new class, hence the methods are unable to gain enough variance information from the pixel values. Whereas there are only limited patterns for a digit or an outline of a face in MNIST and celebA, other baselines adopting pixel values can learn a stabilized mean and display variance on the edges.

**Table 2.** Log-likelihood of on-the-grid datasets (mean  $\pm$  standard deviation)

Model	MNIST	SVHN	celebA 32 $\times$ 32	miniImageNet
NP	$0.65 \pm 4e-4$	$3.21 \pm 6e-4$	$2.78 \pm 1e-3$	N/A
CNP	$1.94 \pm 4e-2$	$4.48 \pm 5e-3$	$3.09 \pm 4e-2$	N/A
ANP	$0.95 \pm 3e-3$	$3.50 \pm 5e-3$	$2.32 \pm 4e-2$	N/A
ConvCNP	<b><math>2.98 \pm 4e-2</math></b>	$6.03 \pm 2e-1$	<b><math>6.35 \pm 2e-1</math></b>	$3.65 \pm 4e-2$
NP-PROV	$2.66 \pm 3e-2$	<b><math>8.24 \pm 5e-2</math></b>	$5.11 \pm 1e-2$	<b><math>4.39 \pm 2e-1</math></b>



**Fig. 3.** Test log-likelihood w.r.t revealing rates (the upper row). The lower row presents some samples from (a) MNIST, (b) SVHN, and (c) celebA  $\times 32$  and miniImageNet (d) with different revealing rates: 10%, 30%, 70%, 90%. The first row in every method predicts the mean and the second row predicts variance.

## 5 Related Work

Since Gaussian processes suffer high computational cost on matrix inverse, recent work focuses on adopting fast forward-feeding neural networks to substitute original GP. A group of Neural Processes-based models is proposed based on variational inference and ELBO (Evidence Lower Bound) [15]. Neural process families abstract a latent variable or a function from context data and decode the function to the target data based on the relationships between target data and context data. The variations of NPs mostly depend on how the relationships are presented. The original NP [7] and conditional NP [6] adopt mean function to extract stochastic and deterministic latent variables that suffice the exchangeability of a stochastic process. Attentive NP [10] considers the self-attention between context points and cross-attention between context points and target points so that attentive aggregation towards latent variables can be used. Regarding nonstructural relationships between data points, Sequential NP [16] and recurrent NP [19] address the temporal correlations in time series. Convolutional conditional NP [9] utilizes the translation equivariance of convolution to predict on out-of-training range target locations. Functional NP [14] and Graph NP [2] exploit topological relationships between context and target nodes. Similar to Meta-Agnostic-Meta-Learning (MAML), residual NP [11] assumes that

few shot tasks share a unified latent variable with minor variations on each task. Thus, the latent variable of the context data is adapted based on the prediction residues. Besides, numerous work focuses on enhancing Gaussian Processes with non-linear feature extraction using neural networks. NNGP [12] and ConvNet [8] explore the relationships between Gaussian processes and one layer neural networks from the perspective of Bayesian inference and approximation. Deep Gaussian processes [4] and deep kernel learning [20] substitute manual designed kernels with neural networks to extract higher-dimensional features.

## 6 Conclusions and Discussions

We introduced NP-PROV, a new member of Neural Processes that derive variances from a position-relevant-only latent space. We verify that NP-PROV can estimate bounded uncertainty when context data have high self-correlations or function values are out-of-the-training range. We mitigate the fluctuations in prediction variances when there exists a shift in function values. We believe that NP-PROV opens a door of rethinking the relationships between the mean and variance in Neural Processes. This work leaves multiple avenues for future improvements. It would be interesting to see the mean derivation be more adaptive to out-of-the training range. Also, unifying on-and-off-the-grid version of NP-PROV to fit in higher-dimensional space.

### A Computational Complexity

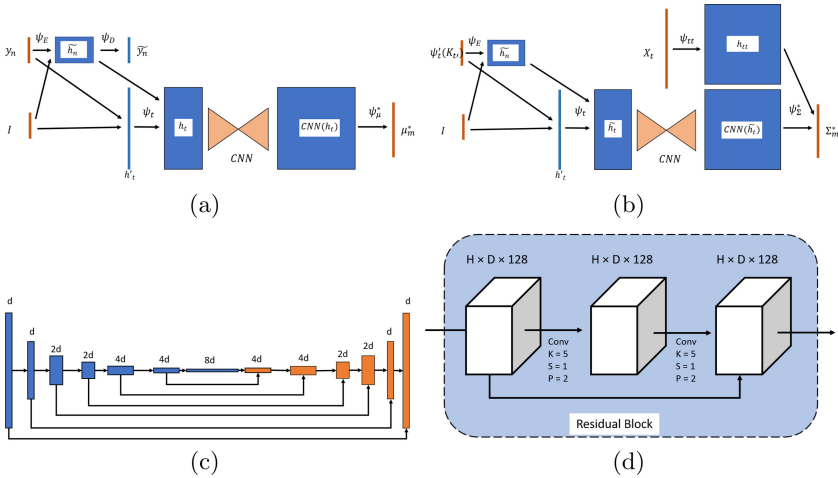
An attention layer takes  $\mathcal{O}(nmd)$ , where  $n$  and  $m$  are the key and query lengths, and  $d$  implies the weight dimension [17]. A convolutional layer costs  $\mathcal{O}(nkdf)$ , where  $k$  and  $f$  refer to the kernel size and channel depth. Therefore, two MLPs in encoder and decoder cost  $\mathcal{O}(msd)$  and  $\mathcal{O}(nsd)$ , where  $m$  and  $n$  refer to the context and target set sizes, and  $s$  refers to the grid length of the discretization ( $s \gg m, n$ ). The computationally expensive part CNN costs  $\mathcal{O}(skdf)$  for convolutions on the  $\mathcal{S}$  space. Compared with ConvCNP, the latent path in NP-PROV brought in multiple MLPs that costs  $\mathcal{O}(s)$ . By tuning the output dimensions of MLPs we can preserve the total number of parameters in convolution and are thus affordable.

### B Off-the-Grid Datasets Experiments

*Datasets Details.* All models were trained for 200 epochs for GP-based synthetic datasets. Each epoch contains 256 batches with the batch size being 16. After each epoch, a new batch of 60 tasks were validated and the number of testing tasks was 2048. The Smart Meter dataset is split into 1,710 training batches, 484 validating batches, and 239 testing batches (7 : 2 : 1) of batch size 16, with a sequence of 100 datapoints per batch. As the data is collected half-hourly, one

batch include approximately four days’ data. The numbers of context data and of target data are sampled from  $\mathcal{U}(3, 50)$  for all the datasets.

*Model Structures and Training Details.* We present model structures in Fig. 4, where Fig. 4(a–b) show how mean and variance are derived from related variables and Table 3 explains every linear transformation in details. The architecture of CNN is detailed in Fig. 4(c). The stride for each convolution is 2. The blue blocks represent convolution layers and the orange ones denote transposed convolution layers. Skipped concatenations are adopted on layers of the same channel depth. During the training process, we use a learning rate of  $3e-4$  for all the GP-based datasets and  $10^{-3}$  for the Smart Meter. Weight decay of  $10^{-5}$  is applied to all model parameters.



**Fig. 4.** Model structures on (a) mean derivations, (b) variance derivations (c) UNet (d) Residual structure in CNN

## C On-the-Grid Datasets Experiments

*Dataset Details.* We use MNIST, SVHN, celebA, and miniImageNet datasets in our experiments. The image sizes and the numbers of training and testing samples are presented in Table 4. We split the original training datasets into 7:3 training sets and validating sets for model selection. For RGB images, we scale the pixel values to  $[0, 1]$  by dividing them by 255. For miniImageNet dataset, we use 4 samples per batch with 5-way-5-shot images and train the model on 200 batches, resulting in overall  $4 \times 5 \times 5 \times 200 = 20,000$  images. The testing set was set to 1-way-1-shot of 63 samples. The batch size for all the other datasets is 16.

**Table 3.** Notations for off-the-grid datasets

Module	Off-the-grid	On-the-grid	Notation
$\psi_E$	Linear (2, 8)	Conv2d (c, 128, 9, 1)	Encoder for mimicking matrix inversion
$\psi_D$	Linear (8, 1)	Conv2d (128, c, 9, 1)	Decoder for mimicking matrix inversion
$\psi_t$	Linear (10, 8)	Conv2d (c, 127, 9, 1)	Context to grid mapping
$\psi_\mu^*$	Linear (16, 1)	Conv2d (c, 128, 9, 1)	Grid to target mean mapping
$\psi_{t'}$	Linear (2, 1)	Conv2d (128, c, 1, 1)	Grid to context mapping
$\psi_{tt}$	Linear (2, 16)	Conv2d (c, 128, 9, 1)	Grid self-correlation
$\psi_\Sigma^*$	Linear (32, 1)	Conv2d (256, c, 1, 1)	Grid to target variance mapping

**Table 4.** Data descriptions for on-the-grid datasets

Dataset	Image size	Training size	Validating size	Testing size
MNIST	(28, 28, 1)	42,000	18,000	10,000
SVHN	(28, 28, 3)	51,279	21,978	26,032
celebA	(32, 32, 3)	141,819	60780	2,592
miniImageNet	(84, 84, 3)	20,000	N/A	63

*Model Structures and Training Details.* The linear transformations (used for on-the-grid datasets) were replaced by convolution operations. The weights and filter sizes are given in Table 3. The padding value is 2 for all the filters, which keeps the output size equalling the input size. The inner CNN structure cascaded four blocks of residual networks detailed in Fig. 4 (d). The learning rate was set to be 5e-4 for all the datasets. The training epoch is 100 with early stopping set to 15 epochs.

## References







- 3springs: Neural processes for sequential data (2019). <https://github.com/3springs/attentive-neural-processes>
- Carr, A.N., Wingate, D.: Graph neural processes: towards Bayesian graph neural networks. arXiv preprint [arXiv:1902.10042](https://arxiv.org/abs/1902.10042) (2019)
- Christou, C., Cunado, J., Gupta, R., Hassapis, C.: Economic policy uncertainty and stock market returns in Pacificrim countries: evidence based on a Bayesian panel VAR model. *J. Multinat. Financial Manage.* **40**, 92–102 (2017)
- Damianou, A., Lawrence, N.: Deep Gaussian processes. In: *Artificial Intelligence and Statistics*, pp. 207–215 (2013)
- Deleu, T., Würfl, T., Samiei, M., Cohen, J.P., Bengio, Y.: Torchmeta: A Meta-Learning library for PyTorch (2019). <https://arxiv.org/abs/1909.06576>. <https://github.com/tristandeleu/pytorch-meta>
- Garnelo, M., et al.: Conditional neural processes. arXiv preprint [arXiv:1807.01613](https://arxiv.org/abs/1807.01613) (2018)
- Garnelo, M., et al.: Neural processes. arXiv preprint [arXiv:1807.01622](https://arxiv.org/abs/1807.01622) (2018)



8. Garriga-Alonso, A., Rasmussen, C.E., Aitchison, L.: Deep convolutional networks as shallow Gaussian processes. arXiv preprint [arXiv:1808.05587](https://arxiv.org/abs/1808.05587) (2018)
9. Gordon, J., Bruinsma, W.P., Foong, A.Y., Requeima, J., Dubois, Y., Turner, R.E.: Convolutional conditional neural processes. arXiv preprint [arXiv:1910.13556](https://arxiv.org/abs/1910.13556) (2019)
10. Kim, H., et al.: Attentive neural processes. arXiv preprint [arXiv:1901.05761](https://arxiv.org/abs/1901.05761) (2019)
11. Lee, B.J., Hong, S., Kim, K.: Residual neural processes. In: AAAI 2020 (2020)
12. Lee, J., Bahri, Y., Novak, R., Schoenholz, S.S., Pennington, J., Sohl-Dickstein, J.: Deep neural networks as gaussian processes. arXiv preprint [arXiv:1711.00165](https://arxiv.org/abs/1711.00165) (2017)
13. Lorenzi, M., Filippone, M., Frisoni, G.B., Alexander, D.C., Ourselin, S., Alzheimer’s Disease Neuroimaging Initiative, et al.: Probabilistic disease progression modeling to characterize diagnostic uncertainty: application to staging and prediction in alzheimer’s disease. *NeuroImage* **190**, 56–68 (2019)
14. Louizos, C., Shi, X., Schutte, K., Welling, M.: The functional neural process. In: *Advances in Neural Information Processing Systems*, pp. 8743–8754 (2019)
15. Rudner, T.G., Fortuin, V., Teh, Y.W., Gal, Y.: On the connection between neural processes and Gaussian processes with deep kernels. In: *Workshop on Bayesian Deep Learning, NeurIPS* (2018)
16. Singh, G., Yoon, J., Son, Y., Ahn, S.: Sequential neural processes. In: *Advances in Neural Information Processing Systems*, pp. 10254–10264 (2019)
17. Vaswani, A., et al.: Attention is all you need. In: *NIPS* (2017)
18. Wei, J., Dolan, J.M., Snider, J.M., Litkouhi, B.: A point-based MDP for robust single-lane autonomous driving behavior under uncertainties. In: *2011 IEEE International Conference on Robotics and Automation*, pp. 2586–2592 (2011)
19. Willi, T., Masci, J., Schmidhuber, J., Osendorfer, C.: Recurrent neural processes. arXiv preprint [arXiv:1906.05915](https://arxiv.org/abs/1906.05915) (2019)
20. Wilson, A.G., Hu, Z., Salakhutdinov, R., Xing, E.P.: Deep kernel learning. In: *Artificial Intelligence and Statistics*, pp. 370–378 (2016)



# A Minority Class Boosted Framework for Adaptive Access Control Decision-Making

Mingshan You<sup>1</sup> , Jiao Yin<sup>2</sup> , Hua Wang<sup>1</sup>  , Jinli Cao<sup>2</sup> ,  
and Yuan Miao<sup>1</sup> 

<sup>1</sup> Institute for Sustainable Industries and Liveable Cities,  
Victoria University, Melbourne, Australia  
Hua.Wang@vu.edu.au

<sup>2</sup> Department of Computer Science and Information Technology,  
La Trobe University, Melbourne, Australia

**Abstract.** Access control is an effective way to prevent data exfiltration from insiders. Recently, machine learning algorithms have been widely used in access control decision-making. However, these algorithms usually fail to consider the dynamic class imbalance in access control problems and thus achieve poor performance on minority classes. In addition, concept drift problems caused by evolving user and resource attributes, user behaviours and access environments are also challenges to tackle. This paper proposes a minority class boosted framework for adaptive access control methods. Specifically, this framework uses a continuous incremental batch learning strategy instead of a batch learning approach to handle the concept drift problem adaptively. Furthermore, a boosting window (BW) algorithm within the framework is specially designed to boost the performance of the minority class, thus, to decrease false positive decisions. The proposed framework is evaluated on a well-known Amazon employee access dataset and results demonstrate the effectiveness and flexibility of the proposed framework and BW algorithm.

**Keywords:** Access control · Online learning · Data imbalance · Boosting window

## 1 Introduction

Data exfiltration has in recent years been becoming a top concern across most security-conscious communities, such as governments, armies and other organizations with high-value data [17]. It is also known as data breach or data theft, and many other names. But in essence, it refers to carrying out an unauthorized data transfer from an information system by malware or a malicious actor. According to a report [12] from McAfee in 2019, they found that most IT professionals have experienced at least one data breach during their career after surveying 700 IT security professionals from different industry and country. On average, they have dealt with six breaches over the course of their professional lives. Data

breaches come in different forms, but many of them are related to insider threats. According to the 2020 Verizon Data Breach Investigations Report [21], 30% of data breaches involve internal actors. So, how to prevent data exfiltration from insiders has now become a research hotspot.

Access control is a technology that is typically employed as the first line of defence for the protection of data [15, 17, 22, 24], guaranteeing that only authorized users can gain access to sensitive resources. The latest access control strategy, attribute-based access control (ABAC), grants or denies an operation request based on assigned attributes of the subject and object, environmental conditions and a set of policies specified in terms of those attributes and conditions [17, 20, 23]. As ABAC can create different policies in accordance with administrative requirements, it is more flexible than traditional role-based access control systems [19, 28]. However, the size of the policy increases dramatically as the increasing diversity of roles, data access environments (e.g., homes and cars) and access devices (e.g., smartphones and tablets). The huge policy scale not only reduces the efficiency of the system, but also leads to frequent misconfiguration of policies [8, 27]. Therefore, more and more researchers are beginning to employ machine learning (ML) methods to develop access control strategies.

From the perspective of machine learning, access control decision-making is a binary classification problem. For an access control system, log files record a large number of historical data of resource requests and system responses, which is a perfect labelled data resource for machine learning algorithms to learn potential data distribution and decision patterns [25, 32]. However, due to the unique features of access control, simple algorithm transplants without proper tailoring or improvement may lead to severe performance degradation.

Specifically, there are two main challenges when building a high-performance machine learning based access control decision-making model. The first challenge is the concept drift problem [11]. Access control patterns are changing over time due to the evolving user and resource attributes, user behaviours and environments. Thus, traditional batch learning strategies are not suitable for access control problems. One possible solution is to adjust the learning strategy online to capture new decision-making patterns, and modify the parameters of the model based on the latest label data over time. Data imbalance is another problem which may cause severe performance decrease besides concept drift for ML algorithms. For access control tasks, there are two classes, access approved (class 1) and access denied (class 0). In most cases, resource access applications are routine working requests and should be approved. Only a very few applications are due to misoperation or illegal access initiated by malicious users. Therefore, the datasets collected from log files are usually imbalanced and lack negative samples. To make matters worse, in terms of security protection systems, negative samples are often more important than the positive sample. It needs more effort to boost the ML models' performance on negative samples (minority class).

To tackle the above-mentioned problems, this paper proposes a minority-based adaptive access control decision-making framework. In brief, our main contributions are as follows:

- (1) We propose an online machine learning framework for access control problems, which adopts consecutive incremental batch learning to adjust the parameters of the ML classifier. The framework is capable of capturing and adapting to possible concept drift and real-time changes in system pattern.
- (2) We design a boosting window (BW) algorithm within each consecutive batch to tackle the severe data imbalance problem. BW algorithm sets a fixed-size window to hold the samples used to update the classifier parameters. The boosting window selects only the misclassified samples and controls the class ratio within the window with a preset sample rate of 0 to 1. The designed BW algorithm can effectively boost the performance of the minority class, the more important class in access control problems.
- (3) We evaluate the proposed framework and BW algorithm on a real-world Amazon employee access dataset and the results demonstrate their effectiveness and flexibility. We also discuss the influence of the hyper-parameters of BW algorithm on the performance of the minority class.

The remainder of this article is divided into the following sections. Section 2 introduces related works. Section 3 describes the implementation details of the proposed framework and the working principle of the boosting window algorithm. Section 4 presents and discusses the experiment results. Finally, Sect. 5 outlines conclusions and future works.

## 2 Related Work

This section reviews the related works from the perspective of access control and machine learning. Past and recent research progress and gaps between existing research and real-world applications will be presented.

Role-based access control (RBAC) and attribute based access control (ABAC) are two widely used access control methods. The former assigns permissions of systems, resources and networks based on a user's role within an organization [16]. Due to its simple implementation, RBAC has long been one of the main access control methods. E. Bertino et al. proposed a temporary RBAC model, employing a role triggers strategy to deal with periodic role activations or other temporary roles [1]. To better suit the complicated system environment and provide more fine-grained access control policies, M. J. Moyer designed a generalized ABAC paradigm to create and maintain rich access control policies, which incorporates traditional user roles with subject roles, object roles and environment roles into access control decisions [14]. However, with the development of information system, the permissions of roles are gradually refined and RBAC tends to give users unnecessarily enlarged permissions. Besides, RBAC allows multiple users to share the same permissions, with no restrictions other than roles, which may lead to malicious use of this vulnerability for unauthorized access [26, 33].

In recent years, attribute-based access control has gradually become a mainstream access control strategy because it takes into account additional attributes from both users and requested resources [6]. ABAC is a policy-based method which generate fine-grained context-aware access control policies rather than roled-based static permissions [17]. A series of general or domain-specific policy-based ABAC models have been successfully used in cloud computing, real-time systems, collaborative environments, mobile environments, grid computing, web services etc. [2, 4, 34, 35].

However, the policy scale has become larger and more complex due to the increase in attributes and the pursuit of system flexibility and versatility. Policy-based ABAC models also caused a number of issues, such as policy misconfiguration and slow response [10, 17]. Therefore, some researchers have started to explore hybrid ABAC models based on policy generating rules and ML algorithms. For example, S. Dutta et al. proposed a privacy via anomaly-detection system (PALS) to leverage machine learning algorithms to capture physical context collected from attributes and generate context-driven policies [3].

Besides, pure ML approaches have attracted more and more attention in various applications and domains [5, 7, 9, 18]. The main problems for pure ML access control decision models are concept drift and data imbalance. Concept drift is a phenomenon that the statistical distributions of what an ML algorithm tries to describe and predict change over time in an arbitrary way, which often exists in data streams or sequences of data organized in chronological order [11]. Data imbalance refers to that the number of samples belonging to each class in a labelled dataset is not at an equivalent level [30]. This paper is a practice of trying to deal with both concept drift and data imbalance problems in pure ABAC ML models.

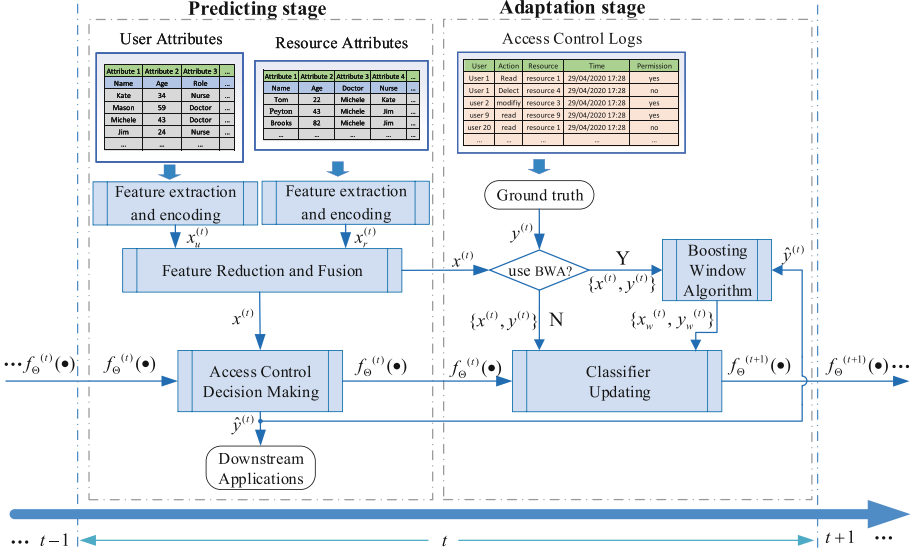
### 3 Methodology

This section first presents the workflow of the proposed consecutive incremental batch learning framework, aiming to tackle the possible concept drift for access control decision-making. Then, we illustrate the principle and implementation details of the boosting window algorithm, which is designed to improve the performance of the minority class, thereby reducing false positives.

#### 3.1 Workflow of the Proposed Framework

The proposed framework is essentially a classifier-agnostic consecutive incremental batch learning process for access control applications. Data is divided into a sequence of consecutive batches and a ML classifier working as an ABAC model in this framework would be pretrained and updated according to these data batches. Figure 1 shows the main processes in time step  $t$ , which includes two stages, the predicting stage and the adaptation stage.

At the predicting stage, when a new access request from a user to a resource happens, the trained model can give a predictive result for downstream applications. Take time step  $t$  ( $t \geq 0$ ) as an example. Firstly, features are extracted



**Fig. 1.** Workflow of the proposed consecutive incremental batch learning framework

and encoded from user and resource attributes separately, denoted as  $x_u^{(t)}$  and  $x_r^{(t)}$ . The available user attributes may include, for example, user name, age, department, group, position, service years, etc. Resource attributes may include, for example, resource id, name, owner, date created and modified, text description, etc. For non-numeric attributes, appropriate feature extraction or encoding methods are applied to extract features. Then, feature reduction and fusion algorithms are applied to generate the final feature set  $x^{(t)}$  from  $x_u^{(t)}$  and  $x_r^{(t)}$ . Let  $f_{\theta}(\cdot)$  is a randomly initialized binary ML classifier applied in this framework,  $\theta$  is the parameter set of  $f(\cdot)$ ,  $f_{\theta}^{(t)}(\cdot)$  ( $t \geq 1$ ) is the status at time step  $t$ , which is actually the classifier updated from  $f_{\theta}^{(t-1)}$  at time step  $t-1$ . The access control decision for the data batch at time step  $t$  would be made according the result of Eq. (1), which will be used to guide downstream applications.

$$\hat{y}^{(t)} = f_{\theta}^{(t)}(x^{(t)}), (t \geq 1). \quad (1)$$

At the adaptation stage, the ground truth  $y^{(t)}$  corresponding to  $x^{(t)}$  would be extracted from the verified access control logs. If using BW algorithm to deal with the data imbalance problem, the labelled dataset  $D^{(t)} = \{x^{(t)}, y^{(t)}\}$  as well as the predicted result  $\hat{y}^{(t)}$  calculated by Eq. (1) are fed as the inputs of BW algorithm. The detailed implementation of BW algorithm will be presented in the following subsection. The output of BW algorithm is a deliberately tailored dataset  $D_w^{(t)} = \{x_w^{(t)}, y_w^{(t)}\}$ . Finally, the parameters of classifier  $f_{\theta}^{(t)}(x)$  would be fine-tuned based on the tailored dataset  $D_w^{(t)}$  and turns into a new status,  $f_{\theta}^{(t+1)}(x)$ , which would be used by the predicting stage of time step  $t+1$ . If the

BW algorithm is not used,  $f_{\Theta}^{(t)}(x')$  would be fine-tuned on the original labelled dataset  $D^{(t)} = \{x^{(t)}, y^{(t)}\}$  directly. No matter if BW algorithm is applied,  $f_{\Theta}^{(t)}(x')$  can adapt to any possible concept drifts existing in  $D^{(t)}$ , because it can be fine-tuned by the latest data batch  $D^{(t)}$ .

### 3.2 Boosting Window Algorithm

BW algorithm is the key component of the proposed framework, aiming to boost the performance of the minority class, especially to decrease the false positive rate. The main idea of BW algorithm is to deliberately select a fixed size of samples as the boosting window samples to update the classifier at each time step. Algorithm 1 demonstrates how boosting window is implemented at time step  $t$  ( $t \geq 1$ ).

---

#### Algorithm 1. Boosting Window Algorithm

---

**Input:**  $D^{(t)} = \{x^{(t)}, y^{(t)}\}$ ,  $\hat{y}^{(t)}$ ,  $N_w$ ,  $r$ ,  $F_r$ .

**Output:**  $D_w^{(t)} = \{x_w^{(t)}, y_w^{(t)}\}$ .

```

1: if  $t=1$  then
2:    $x_{old} = \emptyset$ ;  $y_{old} = \emptyset$ 
3: end if
4: find out the indexes of all samples  $idx$  which satisfy that  $y^{(t)} == \hat{y}^{(t)}$ 
5:  $x^{(t)} = x^{(t)}[idx]$ ,  $y^{(t)} = y^{(t)}[idx]$ 
6:  $x_w^{(t)} = \text{concatenate}(x_{old}, x^{(t)})$ 
7:  $y_w^{(t)} = \text{concatenate}(y_{old}, y^{(t)})$ 
8:  $D_w^{(t)} = \{x_w^{(t)}, y_w^{(t)}\}$ 
9: if  $F_r == \text{True}$  then
10:  find out the indexes of all negative samples  $idx0$  in  $D_w^{(t)}$ 
11:  if  $\text{len}(idx0) > r * N_w$  then
12:     $idx0 = idx0[-r * N_w : -1]$ 
13:  end if
14:  find out the indexes of all positive samples  $idx1$  in  $D_w^{(t)}$ 
15:  if  $\text{len}(idx1) > (1-r) * N_w$  then
16:     $idx1 = idx1[-(1-r) * N_w : -1]$ 
17:  end if
18:   $idx = idx0 \cup idx1$ 
19:   $x_w^{(t)} = x_w^{(t)}[idx, :]$ ;  $y_w^{(t)} = y_w^{(t)}[idx]$ 
20: else
21:  if  $\text{len}(x_w^{(t)}) > N_w$  then
22:     $x_w^{(t)} = x_w^{(t)}[-N_w :, :]$ ;  $y_w^{(t)} = y_w^{(t)}[-N_w :]$ 
23:  end if
24: end if
25:  $x_{old} = x_w^{(t)}$ ;  $y_{old} = y_w^{(t)}$ 
26: return  $D_w^{(t)} = \{x_w^{(t)}, y_w^{(t)}\}$ 

```

---

The number of samples within the boosting window is called boosting window size, denoted as  $N_w$ , where  $N_w$  should be bigger than the step batch size  $N_s$ . The negative sample rate within the boosting window is denoted as  $r$  ( $0 < r < 1$ ), which can be used to adjust the boosting strength of the negative samples. The higher  $r$  is, the stronger boosting strength would be. The total number of negative samples within the boosting window is  $r * N_w$  and the rest are positive samples.

The inputs of BW algorithm include the labelled dataset  $D^{(t)} = \{x^{(t)}, y^{(t)}\}$  at time step  $t$ , the corresponding predicted output  $\hat{y}^{(t)}$  calculated by  $f_{\Theta}^{(t)}(\cdot)$ , the boosting window size  $N_w$  and the negative sample rate  $r$ . The output is the selected dataset  $D_w^{(t)} = \{x_w^{(t)}, y_w^{(t)}\}$  to boost the negative (minority) samples.  $F_r$  is a Boolean variable to indicate if applying the negative sample rate strategy in BW algorithm.

Steps 1–3 in Algorithm 1 show how to initialize the boosting window, where  $x_{old}$  and  $y_{old}$  denotes the old samples in the boosting window before the time step  $t$ .

Steps 4–8 show that the algorithm only picks the misclassified samples in time step  $t$  and put them into the boosting window.

If  $F_r$  is set to True, BW algorithm will apply a negative sample rate strategy as shown in steps 10–19. Specifically, steps 10–13 show how to crop the negative sample indexes so that the total number of negative samples in the boosting window is less than or equal to  $r * N_w$ . Similarly, steps 14–17 are to tailor the indexes of positive samples to make the number of positive samples less than or equal to  $(1 - r) * N_w$ . Steps 18–19 set the final samples in the boosting window,  $\{x_w^{(t)}, y_w^{(t)}\}$ .

Otherwise, if  $F_r$  is set to False, the BW algorithm will only keep the total number of samples in the boosting window equals  $N_w$  regardless of the ratio of negative samples, as shown in steps 21–23.

Step 25 is to prepare data for the following time step  $t + 1$ . The current samples in the boosting windows will be the old samples for the next time step.

To conclude, the boosting window algorithm boosts the minority class via (1) focusing on and boosting the misclassified samples as shown in steps 4–8, (2) adjusting the negative sample rate in the boosting window as shown in steps 10–19.

## 4 Experiment Results

This section reports the experimental methodology and the evaluation results in detail.

### 4.1 Dataset

This research uses a real-world Amazon employee access dataset<sup>1</sup> to conduct experiments and evaluate the performance of the proposed framework and BW

<sup>1</sup> <https://www.kaggle.com/c/amazon-employee-access-challenge>.



algorithm. It contains 32,769 extremely imbalanced labelled samples, including 30,872 positive samples (access approval) and 1,897 negative samples (access rejection). Each sample contains eight user attributes and one resource attribute. Figure 2 shows the data imbalance status over time, using a sliding window imbalance factor (SWIF) [30] as the indicator, where the sliding window size equals to 100.

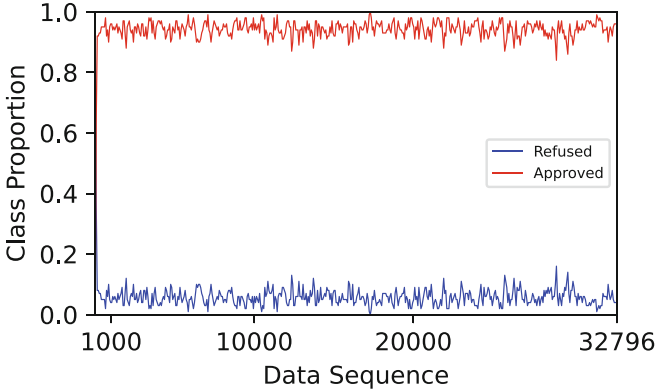


Fig. 2. Data imbalance status over time

## 4.2 Evaluation Metrics

Accuracy, Precision, Recall and F1 Score are in general four basic metrics widely employed in classification model evaluation. Among them, Accuracy, the percentage of correct predictions, is usually the most important metric to evaluate a model’s performance. However, it becomes less instructive when data is extremely imbalanced. Taking the dataset used in this paper as an example, the positive samples are almost count for 95% of the total dataset. Even if the model just directly make all predictions equal 1, its accuracy can reach 95%. Therefore, Precision and Recall, which separately indicate a model’s exactness and completeness in each class, are used as a supplement. F1 Score is the harmonic mean of Precision and Recall, thus it is usually considered as the decisive measure to decide which classifier is better.

For some special applications, such as access control decision-making and biomedical event extraction [13], Recall is a more significant metric to evaluate the performance of a model. For example, for the access control problem, a false negative result (access request should be approved but refused) only leads to a re-application or manual review. However, a false positive result (access request should be refused but approved) may cause severe consequences, such as data breaches, privacy theft or cyber attacks.

Considering the specificity of the access control problem, we calculate Precision, Recall and F1 Score on class 0 instead of class 1 to show the classifier’s

performance on negative samples. Besides, we define a relative model cost  $C$  as Eq. (2) to represent the total misclassification cost of an access control model.

$$C = N_{FN} * 1 + N_{FP} * p, \quad (2)$$

where  $N_{FN}$  and  $N_{FP}$  represent the total number of false negative samples and false positive samples accordingly. Penalty factor  $p$  is the ratio of cost caused by a false positive sample and cost of a false negative sample. For access control problems penalty factor  $p \geq 1$ .

To facilitate comparing the model cost of a series of access control models under different penalty factor settings, we define a normalized relative model cost  $\widehat{C}_i$  for the  $i$ -th model as Eq. (3).

$$\widehat{C}_i = \frac{C_i}{\max(C_1, C_2, \dots, C_n)}, i = \{1, 2, \dots, n\}, \quad (3)$$

where  $n$  is the total number of compared access control models.

### 4.3 Experimental Setting

The step batch size is a hyperparameter of the proposed framework. A larger step batch size means a lower model update frequency. Therefore, the access control decision model will have a slower response to existing concept drift. On the other hand, a smaller step batch size means more frequent model updates, which means a higher computational cost. Information systems can set different step batch sizes according to their own preference. In this article, for definiteness and without loss of generality, we set the step batch size to 100.

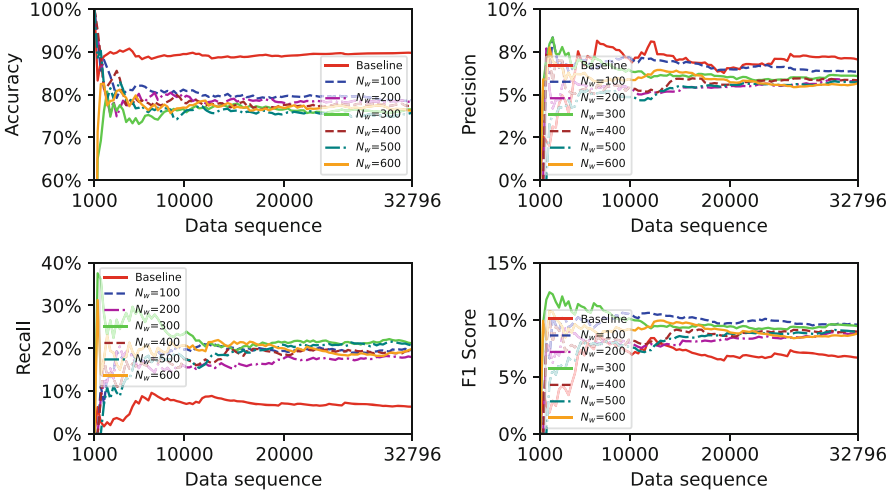
As the proposed framework is classifier-agnostic, the classifier  $f_{\Theta}(\cdot)$  could be any binary ML classifiers, such as Neural Networks [29, 31], Logistic Regression, Support Vector Machine and Random Forest. Considering the verified universal approximation property, we adopt a full-connected three-layer neural network with ten hidden nodes as the classifier used in this framework.

### 4.4 Performance of Boosting Misclassified Samples

To demonstrate the performance of boosting misclassified sample strategy alone, we set  $F_r$  to False, as shown in Algorithm 1.

Figure 3 shows the real-time performance on the minority class (class 0) when BW algorithm adopts boosting misclassified sample strategy alone. Baseline is the proposed framework shown in Fig. 1 without applying the BW algorithm. Data sequence starts from 1000 because the first 1000 samples are used to pre-train the classifier at time step  $t = 1$  and are not used for evaluation.

Unsurprisingly, Baseline achieves the best Accuracy, but gets the worst Precision, Recall and F1 Score. Because there are much more positive samples in the data sequence, if no action is taken to boost the minority class, the classifier will tend to overfit on class 1 and underfit on class 0.



**Fig. 3.** Real-time performance comparison on different boosting window size (class 0)

When applying BW algorithm and boosting the misclassified samples, we can see a significant increase in Recall and F1 Score, which are much more important metrics for access control problems with all boosting window size settings. Therefore, the strategy of boosting misclassified samples alone is effective to boost the performance of the minority class and decrease the false positive rate. Among all boosting window size settings, when  $N_w = 300$ , the access control decision-making model records the best Recall and when  $N_w = 100$ , it achieves the best F1 Score.

Accordingly, Table 1 summarises the overall performance among the whole dataset on the minority class when BW algorithm adopts boosting misclassified samples strategy alone.

**Table 1.** Overall performance comparison on different boosting window size (class 0)

Metrics	Accuracy	Precision	Recall	F1 Score	$\widehat{C}$ (when $p =$ )			
					1	10	100	1000
<b>Baseline</b>	<b>0.8964</b>	0.0691	0.0638	0.0664	<b>0.4256</b>	<b>0.8999</b>	1.0000	1.0000
$N_w = 100$	0.7872	0.0633	0.1947	0.0955	0.8738	0.9627	0.8829	0.8625
$N_w = 200$	0.7832	0.0576	0.1794	0.0872	0.8904	0.9809	0.8996	0.8789
$N_w = 300$	0.7636	0.0605	<b>0.2132</b>	<b>0.0943</b>	0.9709	0.9840	<b>0.8678</b>	<b>0.8432</b>
$N_w = 400$	0.7720	0.0580	0.1936	0.0893	0.9364	0.9867	0.8868	0.8640
$N_w = 500$	0.7565	0.0569	0.2067	0.0892	1.0000	1.0000	0.8759	0.8503
$N_w = 600$	0.7615	0.0561	0.1979	0.0874	0.9797	0.9994	0.8842	0.8595

As shown in Table 1, Baseline achieves the best overall Accuracy at 0.8964, which has an obvious advantage compared with others. However, models with all settings have very poor Precision performance, ranging from 0.0561 to 0.0691, which means that there are only about 5–7 truly negative samples within every 100 predicted negative samples. The good thing is that with BW algorithm, the Recall on the minority class significantly increases from 0.0638 to 0.2131 when  $N_w = 300$  and the F1 Score has also increased from 0.0664 to 0.0943.

Apart from the above-mentioned four metrics, Table 1 also analyses the normalized relative model cost  $\hat{C}$  with different penalty factors.  $\hat{C}$  is a much more straightforward metric for decision-makers to choose the best access control decision-making model. When the penalty factor  $p = 1$  or  $p = 10$ , the baseline achieves the best normalized relative model cost. In this case, the cost of false negative samples equals to or slightly less than the cost of false positive samples. Therefore, the most important thing for a classifier is to increase the overall accuracy instead of improving the performance of the minority class. As  $p$  increases to 100 or even 1000, the false positive samples cost hundreds of or even thousands of times more than the false negative samples. Accordingly, the models achieving the best Recall can get the lowest  $\hat{C}$ , when  $N_w = 300$ . The BW algorithm provides the flexibility for decision-makers to choose the best parameter based on their actual penalty factors.

Although both Fig. 3 and Table 1 have shown the effectiveness of boosting misclassified samples in boosting the minority class, the performance of Recall is far from enough to meet the actual requirements of the access control problem. Therefore, BW algorithm further designed a negative sample rate to adjust the sample ratio in the boosting window. The results with different negative sample rates are discussed in the following section.

#### 4.5 Performance Comparison with Different Negative Sample Rates

When setting  $F_r = \text{True}$ , as shown in Algorithm 1, BW algorithm applies a negative sample rate to adjust the sample ratio in the boosting window. We set  $N_w = 300$ , negative sample rate  $r = \{0.2, 0.4, 0.6, 0.8, 0.9, 0.95\}$ . Baseline is still the consecutive incremental batch learning framework shown in Fig. 1 without applying the BW algorithm.

Figure 4 shows the real-time performance comparison on different negative sample rates on the minority class. Obviously, with the increase of negative sample rate  $r$ , the Accuracy of the access control decision-making models decrease monotonically, while their recall increases monotonically. All of the Precisions are at an equivalent low level. Baseline gets the worst F1 Score, followed by the model when  $r = 0.2$ . Others' F1 Scores are at a similar level.

Figure 4 demonstrates the capability of BW algorithm to boost the Recall of the minority class to a very high level. Table 2 gives more details and quantity analyses on the overall performance comparison on different negative sample rates.

As shown in Table 2, as the negative sample rates used to update the classifier at each time step increases from around 0.05 (Baseline) to 0.95, the Accuracy

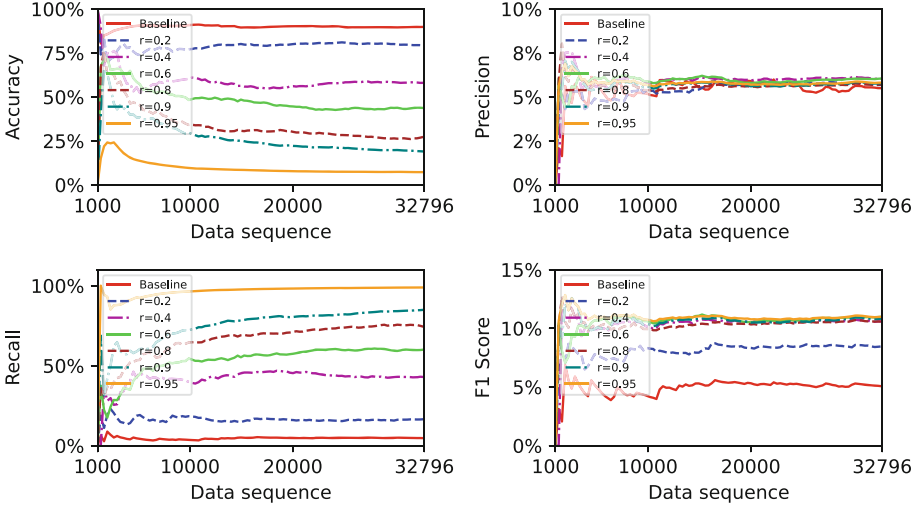


Fig. 4. Real-time performance comparison on different negative sample rates (class 0)

Table 2. Sample rate comparison on the minority class

Metrics	Accuracy	Precision	Recall	F1 Score	$\widehat{C}$ (when $p =$ )			
					1	10	100	1000
<b>Baseline</b>	<b>0.8964</b>	0.0691	0.0638	0.0664	<b>0.1095</b>	<b>0.6395</b>	1.0000	1.0000
$r = 0.2$	0.7940	0.0568	0.1647	0.0845	0.2221	0.6858	0.8975	0.8786
$r = 0.4$	0.5798	0.0601	0.4286	0.1054	0.4531	0.7685	0.6643	0.6061
$r = 0.6$	0.4375	0.0603	0.5998	0.1096	0.6065	0.8257	0.5136	0.4293
$r = 0.8$	0.2742	0.0569	0.7432	0.1057	0.7825	0.9208	0.3953	0.2821
$r = 0.9$	0.1902	0.0577	0.8490	0.1080	0.8731	0.9519	0.3015	0.1729
$r = 0.95$	0.0726	0.0580	<b>0.9891</b>	0.1096	1.0000	1.0000	<b>0.1784</b>	<b>0.0283</b>

decreases from 0.8999 to 0.0726. Because with  $r$  increases, fewer positive samples are selected to update the classifier and the model will perform worse on the majority class, which would lead to a decrease in Accuracy. On the other hand, as the negative sample rate  $r$  increases, the model can identify more negative samples. Thus, the Recall on the negative class dramatically increases from 0.0638 to 0.9891. This demonstrates the capability of BW algorithm to control the Recall of the minority class. The F1 Score also increases from 0.0664 to 0.1096.

As for normalized relative model cost  $\widehat{C}$ , when penalty factor  $p = 1$  or 10, Baseline with a high positive sample rate achieves the best while the model with  $r = 0.95$  gets the worst performance. By contrast, when  $p \geq 100$ , models with  $r = 0.95$  achieves the best  $\widehat{C}$  while Baseline gets the worst.

For a real-world application, it is hard to say which setting is the best. A recommended practice is to calculate the  $\widehat{C}$  of different models according to Eq. (2) and (3) based on actual penalty factors.

## 4.6 Discussion

As shown in Table 1 and Table 2, the performance on class 1 and class 0 are conflicting. In other words, the performance improvement in class 0 will hurt the performance of class 1. In real applications, trade-offs must be made to choose the most appropriate model for a particular application. In this case, specially designed domain-dependent metrics, such as normalized relative model cost  $\widehat{C}$ , could act as a better decisive metric for model selection.

## 5 Conclusion

In conclusion, there is an urgent need for technology to deal with various forms of internal data breaches due to concerns about data security and confidentiality. An accurate access control model based on machine learning can effectively prevent data leakage in companies or organizations around the world. To lead an intelligent ML-based access control decision-making model, this paper proposes a consecutive incremental batch learning framework to tackle the possible concept drift in real-world applications. Within the framework, a BW algorithm is specifically designed to deal with the severe data imbalance problem in the access control problem. As the minority class is much more important for the systems data security and privacy protection, BW algorithm focuses on the misclassified sample and designs a boosting window to boost the performance of the minority class. Experimental results on a real-world dataset demonstrate the effectiveness and flexibility of the proposed framework and BW algorithm.

## References

1. Bertino, E., Bonatti, P.A., Ferrari, E.: TRBAC: a temporal role-based access control model. In: Proceedings of the Fifth ACM Workshop on Role-Based Access Control, pp. 21–30 (2000)
2. Ding, S., Cao, J., Li, C., Fan, K., Li, H.: A novel attribute-based access control scheme using blockchain for IoT. *IEEE Access* **7**, 38431–38441 (2019)
3. Dutta, S., Chukkapalli, S.S.L., Sulgekar, M., Krithivasan, S., Das, P.K., Joshi, A.: Context sensitive access control in smart home environments. In: 2020 IEEE 6th International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing, (HPSC) and IEEE International Conference on Intelligent Data and Security (IDS), pp. 35–41. IEEE (2020)
4. Gupta, M., Awaysheh, F.M., Benson, J., Al Azab, M., Patwa, F., Sandhu, R.: An attribute-based access control for cloud-enabled industrial smart vehicles. *IEEE Trans. Ind. Inform.* (2020)
5. He, J., Rong, J., Sun, L., Wang, H., Zhang, Y., Ma, J.: A framework for cardiac arrhythmia detection from IoT-based ECGs. *World Wide Web* **23** (2020). <https://doi.org/10.1007/s11280-019-00776-9>
6. Hu, V.C., Kuhn, D.R., Ferraiolo, D.F., Voas, J.: Attribute-based access control. *Computer* **48**(2), 85–88 (2015)

7. Jiang, H., Zhou, R., Zhang, L., Wang, H., Zhang, Y.: Sentence level topic models for associated topics extraction. *World Wide Web* **22** (2019). <https://doi.org/10.1007/s11280-018-0639-1>
8. Kabir, E., Mahmood, A., Wang, H., Mustafa, A.: Microaggregation sorting framework for k-anonymity statistical disclosure control in cloud computing. *IEEE Trans. Cloud Comput.* **PP**, 1 (2015). <https://doi.org/10.1109/TCC.2015.2469649>
9. Li, H., Wang, Y., Wang, H., Zhou, B.: Multi-window based ensemble learning for classification of imbalanced streaming data. *World Wide Web* **20**, 1–19 (2017). <https://doi.org/10.1007/s11280-017-0449-x>
10. Li, J., Zhang, B.: An ontology-based approach to improve access policy administration of attribute-based access control. *Int. J. Inf. Comput. Secur.* **11**(4–5), 391–412 (2019)
11. Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., Zhang, G.: Learning under concept drift: a review. *IEEE Trans. Knowl. Data Eng.* **31**(12), 2346–2363 (2018)
12. McAfee: Grand theft data II: the drivers and shifting state of data breaches. Technical report, McAfee (2019). <https://www.mcafee.com/enterprise/en-us/assets/reports/restricted/rp-data-exfiltration-2.pdf>
13. Miwa, M., Ananiadou, S.: Adaptable, high recall, event extraction system with minimal configuration. *BMC Bioinform.* **16**(10), 1–11 (2015)
14. Moyer, M.J., Abamad, M.: Generalized role-based access control. In: *Proceedings 21st International Conference on Distributed Computing Systems*, pp. 391–398. IEEE (2001)
15. Paci, F., Squicciarini, A., Zannone, N.: Survey on access control for community-centered collaborative systems. *ACM Comput. Surv. (CSUR)* **51**(1), 1–38 (2018)
16. Sandhu, R.S.: Role-based access control. In: *Advances in Computers*, vol. 46, pp. 237–286. Elsevier (1998)
17. Servos, D., Osborn, S.L.: Current research and open problems in attribute-based access control. *ACM Comput. Surv. (CSUR)* **49**(4), 1–45 (2017)
18. Srivastava, K., Shekokar, N.: Machine learning based risk-adaptive access control system to identify genuineness of the requester. In: Gunjan, V.K., Zurada, J.M., Raman, B., Gangadharan, G.R. (eds.) *Modern Approaches in Machine Learning and Cognitive Science: A Walkthrough*. SCI, vol. 885, pp. 129–143. Springer, Cham (2020). [https://doi.org/10.1007/978-3-030-38445-6\\_10](https://doi.org/10.1007/978-3-030-38445-6_10)
19. Sun, X., Wang, H., Li, J., Pei, J.: Publishing anonymous survey rating data. *Data Min. Knowl. Discov.* **23**, 379–406 (2011). <https://doi.org/10.1007/s10618-010-0208-4>
20. Sun, X., Wang, H., Plank, A.: An efficient hash-based algorithm for minimal k-anonymity. In: *Proceedings of the Thirty-first Australasian Conference on Computer Science*, vol. 74, pp. 101–107 (2008). <https://doi.org/10.1145/1378279.1378297>
21. Verizon: Data breach investigations report. Technical report, Verizon (2020). <https://enterprise.verizon.com/resources/reports/2020-data-breach-investigations-report.pdf>
22. Vimalachandran, P., Liu, H., Lin, Y., Ji, K., Wang, H., Zhang, Y.: Improving accessibility of the Australian my health records while preserving privacy and security of the system. *Health Inf. Sci. Syst.* **8** (2020). <https://doi.org/10.1007/s13755-020-00126-4>
23. Wang, H., Cao, J., Zhang, Y.: Ticket-based service access scheme for mobile users. *Australian Comput. Sci. Commun.*, 285–292 (2002). <https://doi.org/10.1145/563857.563834>

24. Wang, H., Cao, J., Zhang, Y.: A flexible payment scheme and its role-based access control. *IEEE Trans. Knowl. Data Eng.* **17**, 425–436 (2005). <https://doi.org/10.1109/TKDE.2005.35>
25. Wang, H., Sun, L.: Trust-involved access control in collaborative open social networks. In: 2010 Fourth International Conference on Network and System Security, pp. 239–246. IEEE, September 2010. <https://doi.org/10.1109/NSS.2010.13>
26. Wang, H., Sun, L., Bertino, E.: Building access control policy model for privacy preserving and testing policy conflicting problems. *J. Comput. Syst. Sci.* **80** (2014). <https://doi.org/10.1016/j.jcss.2014.04.017>
27. Wang, H., Wang, Y., Taleb, T., Jiang, X.: Editorial: Special issue on security and privacy in network computing. *World Wide Web* **23** (2019). <https://doi.org/10.1007/s11280-019-00704-x>
28. Wang, H., Zhang, Y., Cao, J.: Effective collaboration with information sharing in virtual universities. *IEEE Trans. Knowl. Data Eng.* **21**, 840–853 (2009). <https://doi.org/10.1109/TKDE.2008.132>
29. Yin, J., Tang, M., Cao, J., Wang, H.: Apply transfer learning to cybersecurity: predicting exploitability of vulnerabilities by description. *Knowl. Based Syst.* **210**, 106529 (2020)
30. Yin, J., Tang, M.J., Cao, J., Wang, H., You, M., Lin, Y.: Adaptive online learning for vulnerability exploitation time prediction. In: Huang, Z., Beek, W., Wang, H., Zhou, R., Zhang, Y. (eds.) *WISE 2020*. LNCS, vol. 12343, pp. 252–266. Springer, Cham (2020). [https://doi.org/10.1007/978-3-030-62008-0\\_18](https://doi.org/10.1007/978-3-030-62008-0_18)
31. Yin, J., Tang, M., Cao, J., Wang, H., You, M., Lin, Y.: Vulnerability exploitation time prediction: an integrated framework for dynamic imbalanced learning. *World Wide Web* **1**(1), 1–23 (2021). <https://doi.org/10.1007/s11280-021-00909-z>
32. Zhang, F., Wang, Y., Liu, S., Wang, H.: Decision-based evasion attacks on tree ensemble classifiers. *World Wide Web* **23** (2020). <https://doi.org/10.1007/s11280-020-00813-y>
33. Zhang, J., Li, H., Liu, X., Luo, Y., Chen, F., Wang, H.: On efficient and robust anonymization for privacy protection on massive streaming categorical information. *IEEE Trans. Dependable Secure Comput.* **PP**, 1 (2015). <https://doi.org/10.1109/TDSC.2015.2483503>
34. Zhang, Y., Zheng, D., Deng, R.H.: Security and privacy in smart health: efficient policy-hiding attribute-based access control. *IEEE Internet Things J.* **5**(3), 2130–2145 (2018)
35. Zhong, H., Zhou, Y., Zhang, Q., Xu, Y., Cui, J.: An efficient and outsourcing-supported attribute-based access control scheme for edge-enabled smart healthcare. *Future Gener. Comput. Syst.* **115**, 486–496 (2021)





# Recognizing Hand Gesture in Still Infrared Images by CapsNet

Hongwang Xiao, Yun Yang, Ke Yu, Jiao Tian, Xinyi Cai, Ying Zhao, Kai Zhang, Na Guo, and Jinjun Chen<sup>(✉)</sup>

Swinburne University of Technology, Hawthorn, VIC 3122, Australia  
{hxiao, yyang, keyu, jiaotian, xcai, yingzhao, kaizhang, nguo, jchen}@swin.edu.au

**Abstract.** In recent years, gesture recognition has been becoming a hot topic that attracts much attention from the computer vision community because of its great potential in many real-world applications. There is a need to design a robust hand-based gesture recognition algorithm to cope with hand gesture recognition tasks. Infrared image recognition has the characteristic of not being disturbed by illumination variation. As a promising alternative to Convolutional Neural Networks (CNN), Capsule Networks (CapsNet) can represent the orientations of features and capture the spatial relationships between features of an entity, which makes CapsNet possess higher generalization ability. In this paper, we propose IRHGR-CapsNet to investigate hand gesture recognition in still infrared images. To evaluate the testing accuracy, the convergence ability and the generalization ability of IRHGR-CapsNet comprehensively, we split the original dataset into three subsets according to different split proportions and get four different dataset split modes for experiments. We can achieve almost 100.00% testing accuracy on all the four dataset split modes. Meanwhile, we can also demonstrate that our proposed IRHGR-CapsNet has a strong convergence ability and generalization ability.

**Keywords:** Hand gesture recognition · Infrared image recognition · Capsule Networks

## 1 Introduction

Nowadays, gesture recognition has been drawing much research attention from the computer vision community because it has a great potential in many real-world applications, like human computer interface, service robots, smart home and smart factories, etc. For instance, with the development of automobile industry, modern intelligent vehicles are providing ample functionalities inside a car for drivers. As a result, the drivers need human-computer interactive commands to experience these inside-car functions. In general, it is an easy and acceptable way to generate these commands by hand gestures which are basically based on human hands and fingers motions. Therefore, there is a need to design robust hand gesture recognition approaches that can work efficiently

even in complex situations or can avoid other disturbance. “Ideally, gesture recognition should be based on a photo of a still hand showing only a single gesture against a clear background in well-lit conditions. But real-life conditions are hardly ever like that. We don’t always get the comfort to use solid, clear backgrounds when presenting gestures.” said Marusarz [1]. To sum up, challenges that are related to gesture recognition include complex background, movement, diversity of gestures, etc.

Infrared image recognition has the characteristics of not being disturbed by low light, sun glare, and can penetrate smoke and haze, etc. Therefore, infrared image recognition offers a potentially better method for gesture recognition in a scenario with complex background or hard illumination condition.

Since 2012 when the work of [2] proposed a large and deep Convolutional Neural Networks (CNN) to deal with ImageNet [3] classification, CNN has been successfully applied in many computer vision tasks [2, 4–6]. While admitting the great success of CNN, one of its inherent drawbacks should not be ignored, i.e., CNN’s intermediate scalar neurons cannot represent the orientations of features and capture the spatial relationships between features of an entity, which makes CNN to be fooled easily [7]. To address this shortcoming of CNN, CapsNet was proposed and was taken as a promising alternative to conventional CNN. Actually, the concept of “capsule” was first proposed by Hinton et al. [8] in 2011, but there was difficulty in implementing the algorithm then. Then later in 2017, Sabour et al. [9] proposed a dynamic routing algorithm which enables capsules to communication with each other. By this dynamic routing mechanism, the work of [9] designed a novel networks architecture, named CapsNet.

Contribution of this paper includes:

- This work is the first one to apply CapsNet to hand gesture recognition in still infrared images. The proposed approach has potential to be utilized in gesture recognition in night or dark condition, not only in daylight condition.
- We present IRHGR-CapsNet architecture which can achieve almost 100.00% recognition accuracy on a benchmark dataset. Compared with CNN, we can use less training samples to get a more accurate recognition model by IRHGR-CapsNet, which is very suitable for situations where there is not enough labeled training data.
- Although we demonstrated the generalization ability of CapsNet in sign language symbols recognition task in our previous work [10], in this paper we further demonstrate this characteristic of CapsNet by conducting comprehensive experiments.
- Through extensive experiments we demonstrate that our proposed IRHGR-CapsNet is able to converge stably and fast as well as maintaining a high classification accuracy.

The rest of this paper goes as follows: Sect. 2 introduces the related work of CapsNet and hand gesture recognition. The proposed approach is described in Sect. 3 in details. In Sect. 4, we introduce the experimental settings, followed by comprehensive experimental results and analysis in Sect. 5. Then Sect. 6 presents an ablation study and discussion. Lastly, Sect. 7 summarizes this paper and presents future work.

## 2 Related Work

### 2.1 CapsNet

The most basic component of CapsNet is capsule which is a group of neurons and whose output is an activity vector instead of a scalar [9]. The length of the output vector stands for the existence probability of an entity in an image, and its orientation denotes instantiation parameters (e.g., direction, hue, texture, location) of this entity. Active capsules in lower levels interact with those in higher levels via transformation matrices. A higher-level capsule becomes active when it agrees with the output from lower-level capsules.

The main difference between CapsNet and CNN includes two aspects: 1) the inter-layers of CNN output scalar neurons, while in CapsNet it outputs vector neurons (i.e., capsules); 2) CapsNet adopts dynamic routing between capsule layers instead of convolution and pooling mechanism between convolutional and pooling layers in CNN. Since the advent of CapsNet in 2017, it has been applied successfully in various computer vision tasks, such as image classification [11], object segmentation [12], object detection [13], etc.

### 2.2 Hand Gesture Recognition

There is not much work directly related to hand gesture recognition by CapsNet. Our previous work of [10] is one of them that addresses sign language digits and alphabets recognition by CapsNet and achieves state-of-the-art results, which is a similar work that uses CapsNet to address gesture recognition task. Some other relevant works are for action detection or recognition [14–18]. The work of [14] presented a novel approach by CapsNet to address multi-view and multi-label facial action unit detection. The method was verified by FERA 2017 facial expression dataset and achieved satisfactory results in its corresponding field. The work of [15] proposed 3D CapsNet (i.e., taking a sequence of videos as input) which is a unified framework that can perform action segmentation and action classification simultaneously. The work of [16] proposed an multimodal approach by CapsNet for pixel-wise localization of actors and actions in a video based on natural language queries where an actor and an action are specified. The work of [17] proposed a method by CapsNet for skeleton-based action recognition which takes skeleton heatmap and skeleton coordinates as input. This architecture consists of two types of capsules obtained by spatial encapsulation and temporal encapsulation respectively. The work of [18] proposed a CapsNet model for human action recognition in individual video frames. The method achieved 73.52% and 90.47% on KTH<sup>1</sup> and UCF datasets<sup>2</sup> respectively, compared with 61.41% and 68.18% by baseline CNN.

## 3 Proposed Approach

To solve the problem of hand gesture recognition in infrared images, we design a modified CapsNet architecture, named IRHGR-CapsNet which contains a capsule module and a reconstruction module shown in Figs. 1 and 2 respectively.

<sup>1</sup> <https://www.csc.kth.se/cvap/actions/>.

<sup>2</sup> <https://www.crcv.ucf.edu/research/data-sets/>.

### 3.1 Capsule Module

As shown in Fig. 1, the first two layers of the capsule module are both ReLU convolutional layers that have 256 kernels with a stride of 3, but with kernel size of  $9 \times 9$  and  $7 \times 7$  respectively. The third layer (i.e., primary capsule layer) is a convolutional capsule layer which has 32 channels, each of which has 8 convolutional filters of size  $3 \times 3$  and stride 1 to the whole input, i.e., each channel is composed of 8D capsule in a  $6 \times 6$  grid. The last layer, HandGesture Capsule layer, is a fully connected layer which has ten capsules each of which is in the length of 32D, i.e., each gesture class has a 32D capsule. We should note that the parameters here we adopt are dedicated to the hand gesture recognition task in this paper, but the capsule architecture is a genetic one to deal with classification tasks.

### 3.2 Reconstruction Module

Besides the capsule module, the reconstruction module is another important component of a CapsNet. The reconstruction module shown in Fig. 2 is used to reconstruct a hand gesture image from the HandGesture Capsule layer representation. It contains 1 fully connected layer, 5 upsampling layers with bilinear interpolation and 6 convolutional layers with  $3 \times 3$  kernels. Among them each upsampling layer is connected by a  $3 \times 3$  convolutional layer. The last (i.e., the 6th one)  $3 \times 3$  convolutional layer is fed into the Sigmoid function, after which it outputs a reconstructed  $120 \times 120 \times 3$ -dimension image.

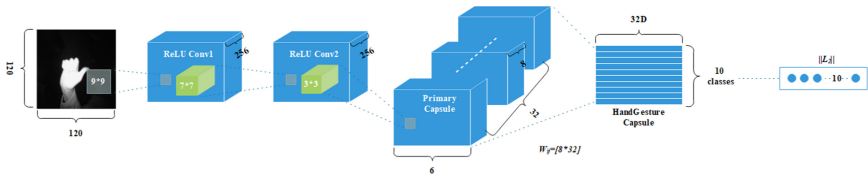


Fig. 1. Capsule module of IRHGR-CapsNet for infrared hand gesture recognition

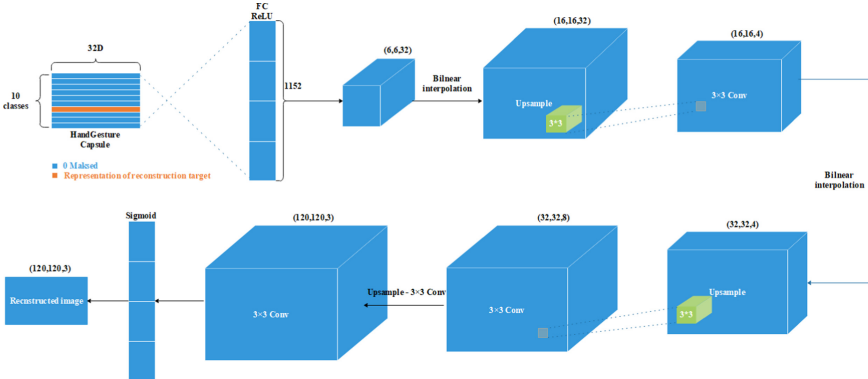
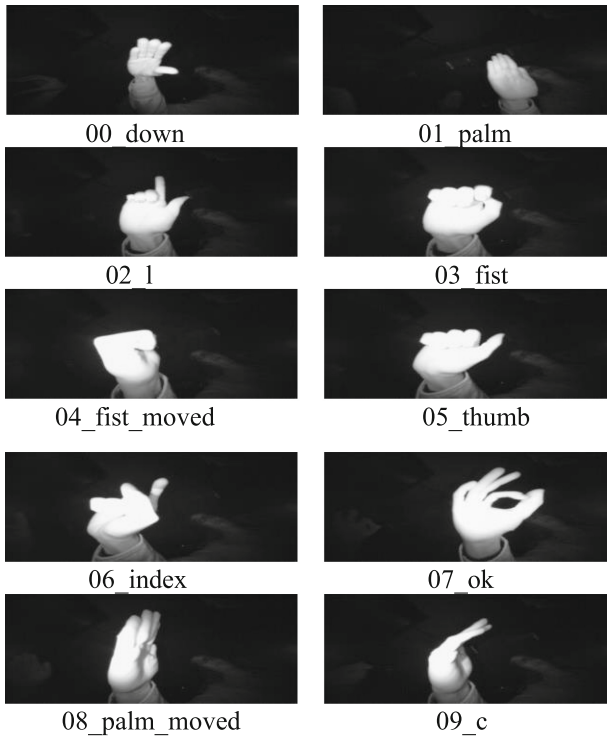


Fig. 2. Reconstruction module of IRHGR-CapsNet for infrared hand gesture recognition

## 4 Experimental Settings

### 4.1 Dataset

The dataset we use in the paper is Hand Gesture Recognition Database (HGRD)<sup>3</sup>, which was first presented in [19]. The images in HGRD are near infrared images with size of  $240 \times 640$  acquired by a Leap Motion sensor. HGRD consists of 10 classes of hand gestures that were performed 200 times by each of 10 different persons (5 men and 5 women), thus containing 20,000 samples in total. Figure 3 shows examples of this dataset. Thank the authors of [19] for presenting this dataset.



**Fig. 3.** Examples of HGRD (hand gesture recognition database)

### 4.2 Dataset Split Modes

In our experiments, the original dataset is divided into three subsets, training set, validation set and testing set, by different proportions. When the original dataset is split by the proportion of 80%, 10% and 10% respectively, we name it 8:1:1 split in the paper. We design four different dataset split modes, which are shown in Table 1.

<sup>3</sup> <https://www.kaggle.com/gti-upm/leapgestrecog>.

**Table 1.** Dataset split modes

Split mode name	Split proportion	Number of training samples	Number of validation samples	Number of testing samples
8:1:1 split	80%, 10%, 10%	16,000	2,000	2,000
5:2:3 split	50%, 20%, 30%	10,000	4,000	6,000
3:2:5 split	30%, 20%, 50%	6,000	4,000	10,000
1:1:8 split	10%, 10%, 80%	2,000	2,000	16,000

### 4.3 Training Parameters of the Proposed Approach

We trained our networks using TensorFlow [20] and Tesla P100. In our experiments we adopt the proposed IRHGR-CapsNet and set up training step to 20,000. For the training procedure, we adopt Adam optimizer [21] and learning rate of 0.0001. Also, we use dropout with a rate of 0.5 as regularization on the ReLU Conv2 layer.

## 5 Experimental Results and Analysis

### 5.1 Experimental Results of Our Approach

In our experiments, we resize the original  $240 \times 640$  infrared images to  $120 \times 120$  as input to speed up the training. In experiments on the aforementioned four different dataset split modes, we can achieve all the final testing accuracy of almost 100.00% with a low loss on testing set. More details can be found in Table 2. The testing CPU time includes data I/O time and model inference time. The processing time per sample means the sum of data I/O time and model inference time for recognizing a gesture from a single image. Our trained model achieves millisecond-level inference performance.

**Table 2.** Experimental results on four different dataset split modes

Split mode	Training CPU time	Testing accuracy	Testing loss	Testing samples	Testing CPU time	Processing time/sample
8:1:1 split	4174.53 s	100.00%	0.00019	2,000	12.23 s	6.12 ms
5:2:3 split	4379.26 s	99.98%	0.00028	6,000	58.59 s	9.77 ms
3:2:5 split	4388.20 s	99.96%	0.00045	10,000	155.13 s	15.51 ms
1:1:8 split	4317.16 s	99.96%	0.00055	16,000	378.27 s	23.64 ms

\* Input  $120 \times 120$  infrared image with data augmentation, training steps = 20k, batch size = 50, routing iteration = 1.

### 5.2 Comparison of Confusion Matrix of Different Dataset Split Modes

In the experiments, we get a normalized confusion matrix from which we can see our approach achieves a very high recall score. Figures 4 and 5 are the confusion matrixes of gesture recognition for the 8:1:1 split, 5:2:3 split and 3:2:5 split and 1:1:8 split respectively. In the confusion matrix, each column represents the predicted class of 10 kinds of gestures (i.e., it corresponds to “00\_down, 01\_palm, 02\_l, 03\_fist, 04\_fist\_moved, 05\_thumb, 06\_index, 07\_ok, 08\_palm\_moved, 09\_c” respectively from the 1<sup>st</sup> column to the 10<sup>th</sup> column), and the sum of the numbers in each column represents the number of samples predicted to be in this class. Each row represents the true class to which the sample belongs (i.e., it corresponds to “00\_down, 01\_palm, 02\_l, 03\_fist, 04\_fist\_moved, 05\_thumb, 06\_index, 07\_ok, 08\_palm\_moved, 09\_c” respectively from the 1<sup>st</sup> row to the 10<sup>th</sup> row), and the sum of the numbers in each row represents the number of samples in the corresponding class. For example, in the right part of Fig. 4, the 6<sup>th</sup> row means there are 602 samples belonging to the class “05\_thumb”, but the “1” (noted by a red circle) means one sample that belongs to the class “05\_thumb” is finally wrongly predicted to be in the class “06\_index”.

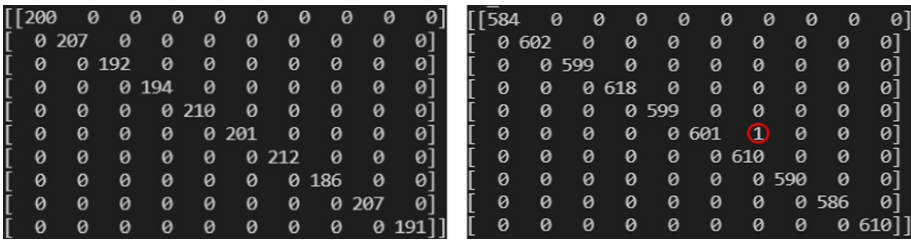


Fig. 4. Confusion matrix for 8:1:1 split (left) and 5:2:3 split (right) (Color figure online)

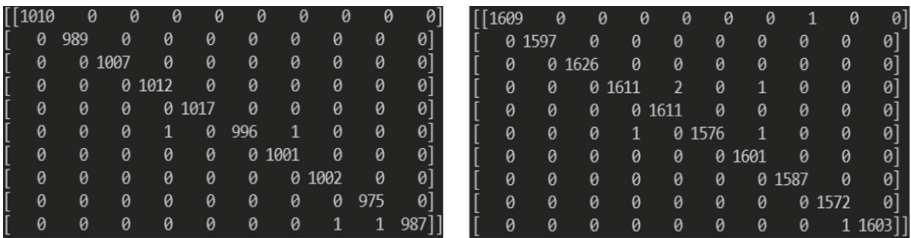


Fig. 5. Confusion matrix for 3:2:5 split (left) and 1:1:8 split (right)

### 5.3 Comparison with Existing Approaches

Also, we compare the result of our approach with those in [19] and [22] in Table 3. from which we can see that our approach achieves 100.00% recognition accuracy on all the ten classes of gestures, outperforming both methods in [19] and [22]. Although the testing

accuracy of the approach in [19] is very close to 100.00%, it is not a deep learning based method, but a feature-based method using Depth Spatiograms of Quantized Patterns feature descriptor [23]. Our approach is a deep learning based method with end-to-end neural networks.

**Table 3.** Comparison of our approach with existing approaches

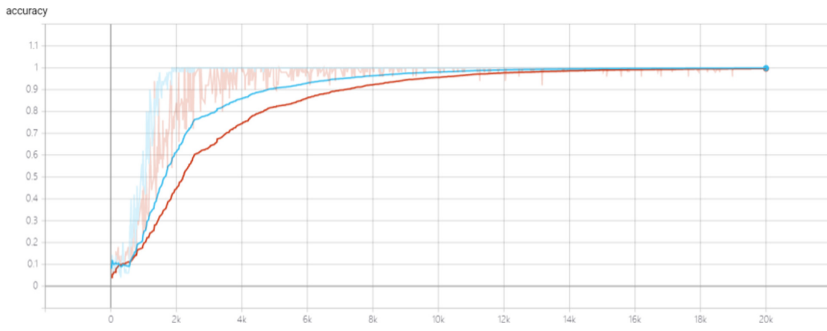
Gesture class	Approaches		
	Ours	Approach in [19]	Approach in [22]
00_down	1.00	1.00	0.87
01_palm	1.00	1.00	0.95
02_l	1.00	0.99	1.00
03_fist	1.00	0.99	0.87
04_fist_moved	1.00	1.00	1.00
05_thumb	1.00	0.99	0.95
06_index	1.00	1.00	1.00
07_ok	1.00	0.99	0.87
08_palm_moved	1.00	1.00	0.95
09_c	1.00	1.00	0.87

\* This result is from the experiment on 8:1:1 split mode.

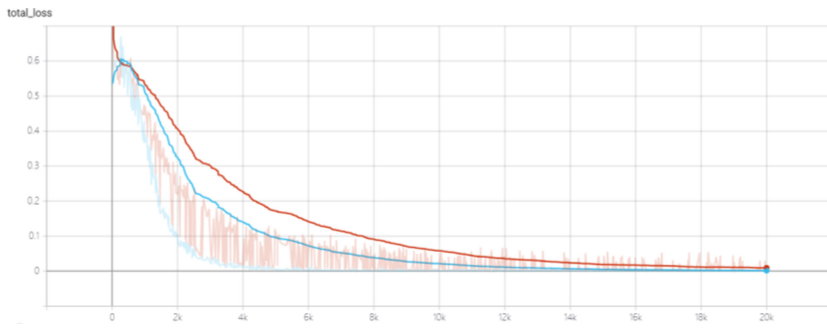
#### 5.4 Comparison of Training Curve and Validation Curve for Different Dataset Split Modes

Moreover, we plot training curve and validation curve to analyze the convergence ability of our approach. Figures 6 and 7 illustrate the accuracy and total loss for training procedure (dark orange curve) and validation procedure (sky blue curve) respectively for the experiment on 8:1:1 split. Figures 8 and 9 (dark blue curve for training, red curve for validation), Figs. 10 and 11 (gray curve for training, orange curve for validation), and Figs. 12 and 13 (light blue curve for training, purple curve for validation) show the corresponding accuracy and total loss for the experiments on 5:2:3 split, 3:2:5 split, and 1:1:8 split, respectively. All the figures show that our proposed IRHGR-CapsNet is able to converge stably and fast as well as maintaining a high classification accuracy.

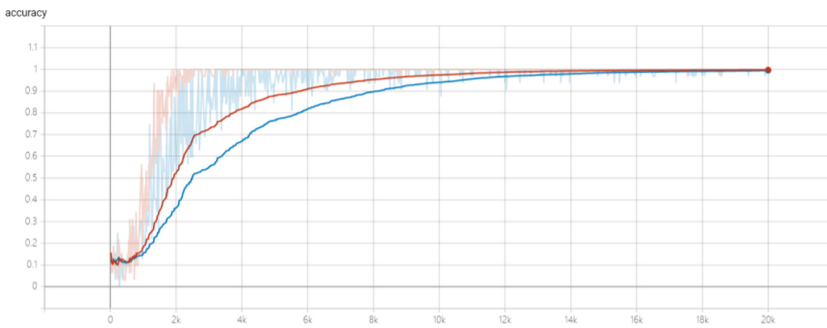




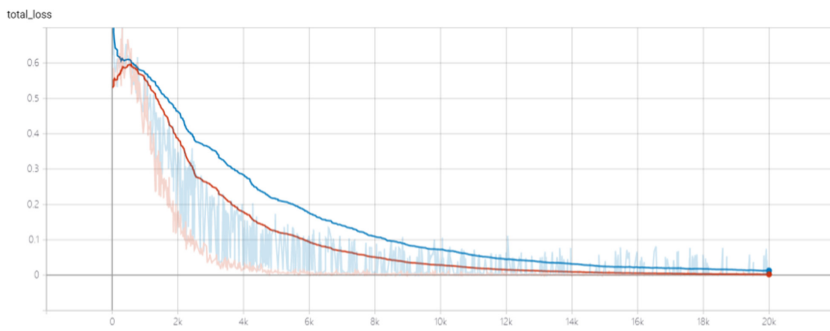
**Fig. 6.** Experimental results of HGRD recognition for 8:1:1 split: accuracy (Color figure online)



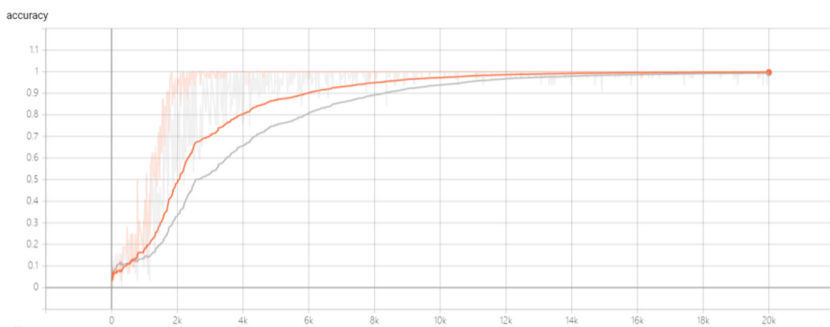
**Fig. 7.** Experimental results of HGRD recognition for 8:1:1 split: total loss (Color figure online)



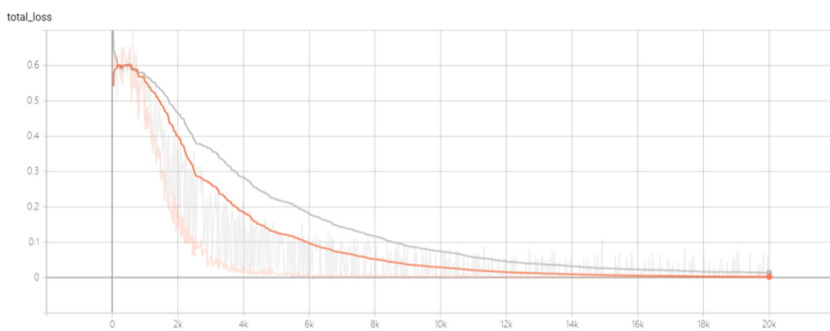
**Fig. 8.** Experimental results of HGRD recognition for 5:2:3 split: accuracy (Color figure online)



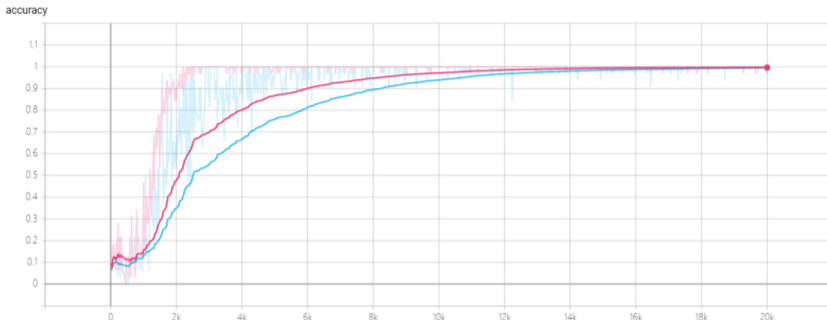
**Fig. 9.** Experimental results of HGRD recognition for 5:2:3 split: total loss (Color figure online)



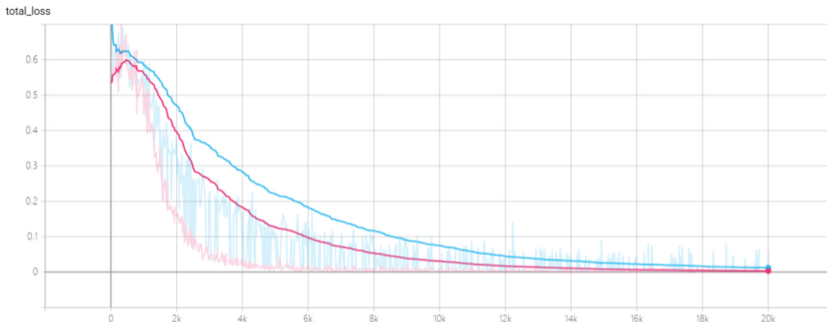
**Fig. 10.** Experimental results of HGRD recognition for 3:2:5 split: accuracy (Color figure online)



**Fig. 11.** Experimental results of HGRD recognition for 3:2:5 split: total loss (Color figure online)



**Fig. 12.** Experimental results of HGRD recognition for 1:1:8 split: accuracy (Color figure online)



**Fig. 13.** Experimental results of HGRD recognition for 1:1:8 split: total loss (Color figure online)

## 6 Ablation Study

We carry out ablation study of the experiment on HGRD to evaluate the generalization ability, the convergence ability and the testing accuracy of our approach. To be specific, we compare IRHGR-CapsNet with CNN, and analyze the effect routing iteration on IRHGR-CapsNet. Generalization ability means a trained model can be successfully applied to unseen data for inference.

### 6.1 Evaluation on Different Networks

We compare our IRHGR-CapsNet and CNN with the same experimental settings to evaluate the generalization ability of our approach on HGRD recognition. The comparison results are shown in Table 4. For 8:1:1 split, we both achieve 100.00% testing accuracy by CapsNet and CNN. However, for the experiments on 5:2:3 split, 3:2:5 split and 1:1:8 split, the generalization ability of IRHGR-CapsNet outperforms CNN. Especially on 1:1:8 split, IRHGR-CapsNet achieves testing accuracy of 99.96% and testing loss of 0.00055, far better than CNN's accuracy 99.27% and testing loss 0.03563. By that it means our IRHGR-CapsNet model has a better generalization ability on unseen data than CNN.

**Table 4.** Experimental results on different networks for generalization ability

Split mode	Networks	Testing accuracy	Testing loss
8:1:1 split	IRHGR-CapsNet	100.00%	0.00019
	CNN	100.00%	0.00004
5:2:3 split	IRHGR-CapsNet	99.98%	0.00028
	CNN	99.88%	0.00596
3:2:5 split	IRHGR-CapsNet	99.96%	0.00045
	CNN	99.90%	0.00670
1:1:8 split	IRHGR-CapsNet	99.96%	0.00055
	CNN	99.27%	0.03563

\* Input 120 \* 120 infrared image with data augmentation, training steps = 20k, batch size = 50, routing iteration = 1.

## 6.2 Evaluation on Different Routing Iterations

We test various routing iterations to evaluate how it will affect the convergence and testing accuracy of our approach on HGRD recognition. From the aforementioned analysis we know that IRHGR-CapsNet can converge fast and be stable with only 1 dynamic routing iteration. To further evaluate the effect of the routing iteration, we conduct four experiments with routing iterations of 1, 10, 20, and 30 respectively. From the comparison results shown in Table 5, we conclude that only 1 routing iteration is enough for IRHGR-CapsNet to converge without overfitting as well as achieve high testing accuracy.

**Table 5.** Comparison of different routing iterations

Routing iteration	Testing accuracy	Testing loss	Training CPU time	Testing CPU time	Overfitting
1	99.96%	0.00055	4317.16 s	378.27 s	No
10	99.97%	0.00055	4198.47 s	373.81 s	No
20	99.97%	0.00055	4302.11 s	364.12 s	No
30	99.98%	0.00063	4214.08 s	369.30 s	No

\* Input 120 \* 120 infrared image with data augmentation, training steps = 20k, batch size = 50, and dataset split mode = 1:1:8.

## 6.3 Discussion

HGRD is a high-quality dataset, so we can see that both IRHGR-CapsNet and CNN can achieve very high testing accuracy on this dataset. However, if we look into the different

experimental results between IRHGR-CapsNet and CNN, we can see that IRHGR-CapsNet shows a better generalization ability. In other words, compared with CNN, we can use less training samples to train and get a more accurate recognition model. Our approach has a big potential to be applied in cloud environment [24–26] to provide deep learning based API by cloud service vendors [27]. Moreover, our approach is suitable for scenarios where there is not large amount of labelled training data i.e., few-shot learning. Therefore, it is expected to be applied in cloud-based systems which has high demand of efficiency on computation and storage resource [28–30].

## 7 Conclusion and Future Work

In this paper, for the first time, we apply CapsNet to hand gesture recognition in still infrared images. We propose a modified CapsNet architecture (i.e., IRHGR-CapsNet) containing a capsule module and a reconstruction module. Extensive experiments have demonstrated that the proposed IRHGR-CapsNet has a strong generalization ability, a stable and fast convergence ability and achieve almost 100.00% testing accuracy than conventional CNN architectures. By that it means CapsNet is suitable for the situations where there is not enough labelled training data. Therefore, this work can be taken as an exploration on few-shot learning which is expected to provide a future research direction of few-shot learning by CapsNet.

## References

1. Marusarz, W.: The challenges and opportunities of gesture recognition. <https://nexocode.com/blog/posts/gestures-recognition-challenges-and-opportunities/>. Accessed 19 Apr 2021
2. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: NIPS, pp. 1–9 (2012). <https://doi.org/10.1145/3065386>
3. Deng, J., Dong, W., Socher, R., Li, L.-J., Kai, L., Li, F.-F.: ImageNet: a large-scale hierarchical image database. In: CVPR, pp. 248–255 (2009). <https://doi.org/10.1109/cvpr.2009.5206848>
4. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR, pp. 3431–3440 (2015). <https://doi.org/10.1109/CVPR.2015.7298965>
5. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: ICCV, pp. 2980–2988 (2017). <https://doi.org/10.1109/ICCV.2017.322>
6. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: OverFeat: integrated recognition, localization and detection using convolutional networks. In: ICLR (2014). <https://doi.org/10.1016/j.visres.2006.11.009>
7. Yosinski, J., Clune, J., Nguyen, A., Yosinski, J., Clune, J.: Deep neural networks are easily fooled: high confidence predictions for unrecognizable images. In: CVPR, pp. 427–436 (2015). <https://doi.org/10.1109/CVPR.2015.7298640>
8. Hinton, G.E., Krizhevsky, A., Wang, S.D.: Transforming auto-encoders. In: Honkela, T., Duch, W., Girolami, M., Kaski, S. (eds.) ICANN 2011. LNCS, vol. 6791, pp. 44–51. Springer, Heidelberg (2011). [https://doi.org/10.1007/978-3-642-21735-7\\_6](https://doi.org/10.1007/978-3-642-21735-7_6)
9. Sabour, S., Frosst, N., Hinton, G.E.: Dynamic routing between capsules. In: NIPS, pp. 1–11 (2017). <https://doi.org/10.1177/1535676017742133>
10. Xiao, H., et al.: Sign language digits and alphabets recognition by capsule networks. *J. Ambient Intell. Humaniz. Comput.* 1–11 (2021). <https://doi.org/10.1007/s12652-021-02974-8>

11. Hoogi, A., Wilcox, B., Gupta, Y., Rubin, D.L.: Self-attention capsule networks for image classification. arXiv Prepr. [arXiv:1904.12483](https://arxiv.org/abs/1904.12483) (2019)
12. LaLonde, R., Bagci, U.: Capsules for object segmentation. In: 1st Conference on Medical Imaging with Deep Learning (MIDL), pp. 1–9 (2018)
13. Neelavathy Pari, S., Mohana, T., Akshaya, V.: Real-time traffic sign detection using capsule network. In: Proceedings of the 11th International Conference on Advanced Computing (ADCOM), pp. 193–196 (2019). <https://doi.org/10.1109/ICoAC48765.2019.247140>
14. Ertugrul, I.O., Jeni, L.A., Cohn, J.F.: FACSCaps: pose-independent facial action coding with capsules. In: CVPR Workshops, pp. 2211–2220 (2018). <https://doi.org/10.1109/CVPRW.2018.00287>
15. Duarte, K., Rawat, Y.S., Shah, M.: VideocapsuleNet: a simplified network for action detection. In: NeurIPS, pp. 7610–7619 (2018)
16. McIntosh, B., Duarte, K., Rawat, Y.S., Shah, M.: Multi-modal capsule routing for actor and action video segmentation conditioned on natural language queries. arXiv Prepr. [arXiv:1812.00303](https://arxiv.org/abs/1812.00303) (2018)
17. Yu, Y., Tian, N., Chen, X., Li, Y.: Skeleton capsule net: an efficient network for action recognition. In: Proceedings of 8th International Conference on Virtual Reality and Visualization (ICVRV), pp. 74–77. IEEE (2018). <https://doi.org/10.1109/ICVRV.2018.00022>
18. Algamdi, A.M., Sanchez, V., Li, C.-T.: Learning temporal information from spatial information using CapsNets for human action recognition. In: International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3867–3871 (2019). <https://doi.org/10.1109/icaasp.2019.8683720>
19. Mantecón, T., del-Blanco, C.R., Jaureguizar, F., García, N.: Hand gesture recognition using infrared imagery provided by leap motion controller. In: Blanc-Talon, J., Distant, C., Philips, W., Popescu, D., Scheunders, P. (eds.) ACIVS 2016. LNCS, vol. 10016, pp. 47–57. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-48680-2\\_5](https://doi.org/10.1007/978-3-319-48680-2_5)
20. Abadi, M., et al.: TensorFlow: large-scale machine learning on heterogeneous distributed systems. arXiv Prepr. [arXiv:1603.04467](https://arxiv.org/abs/1603.04467) (2016)
21. Kingma, D.P., Ba, J.L.: Adam: a method for stochastic optimization. In: ICLR, pp. 1–15 (2015)
22. Huang, D.-Y., Hu, W.-C., Chang, S.-H.: Gabor filter-based hand-pose angle estimation for hand gesture recognition under varying illumination. *Expert Syst. Appl.* **38**, 6031–6042 (2011). <https://doi.org/10.1016/j.eswa.2010.11.016>
23. Mantecón, T., Mantecón, A., Del-Blanco, C.R., Jaureguizar, F., García, N.: Enhanced gesture-based human-computer interaction through a compressive sensing reduction scheme of very large and efficient depth feature descriptors. In: 2th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS) (2015). <https://doi.org/10.1109/AVSS.2015.7301804>
24. Qi, L., Dou, W., Zhang, X., Chen, J.: A Qos-aware composition method supporting cross-platform service invocation in cloud environment. *J. Comput. Syst. Sci.* **78**(5), 1316–1329 (2012). <https://doi.org/10.1016/j.jcss.2011.12.016>
25. Wang, L., Jie, W., Chen, J.: *Grid Computing: Infrastructure, Service, and Applications*, 528 p. CRC Press, Boca Raton (2009). ISBN 13:978-1420067668. <https://doi.org/10.1201/9781315218854>
26. Qi, L., Dou, W., Chen, J.: Weighted principal component analysis-based service selection method for multimedia services in cloud. *Computing* **98**(1–2), 195–214 (2014). <https://doi.org/10.1007/s00607-014-0413-x>
27. Liu, X., Yuan, D., Zhang, G., Chen, J., Yang, Y.: Swindow-C: a peer-to-peer based cloud workflow system. In: Borko, F., Armando, E. (eds.) *Handbook of Cloud Computing*, pp. 309–332. Springer, Boston (2010). ISBN 978-1-4419-6523-3. [https://doi.org/10.1007/978-1-4419-6524-0\\_13](https://doi.org/10.1007/978-1-4419-6524-0_13)

28. Song, X., Dou, W., Chen, J.: A workflow framework for intelligent service composition. *Futur. Gener. Comput. Syst.* **27**(5), 627–636 (2011). <https://doi.org/10.1016/j.future.2010.06.008>
29. Chen, J., Yang, Y.: Temporal dependency based checkpoint selection for dynamic verification of fixed-time constraints in grid workflow systems. In: *Proceedings of ACM/IEEE 30th International Conference on Software Engineering (ICSE)*, pp. 141–150 (2008). <https://doi.org/10.1145/1368088.1368108>
30. Puthal, D., Nepal, S., Ranjan, R., Chen, J.: DLSeF: a dynamic key-length-based efficient real-time security verification model for big data stream. *ACM Trans. Embed. Comput. Syst (TECS)* **16**(2), Article 51 (2017). <https://doi.org/10.1145/2937755>



# Vertical Federated Principal Component Analysis on Feature-Wise Distributed Data

Yiu-ming Cheung<sup>1</sup>(✉) , Jian Lou<sup>2</sup>, and Feng Yu<sup>1</sup>

<sup>1</sup> Department of Computer Science, Hong Kong Baptist University, Kowloon Tong, Hong Kong SAR, China  
ymc@comp.hkbu.edu.hk

<sup>2</sup> Guangzhou Institute of Technology, Xidian University, Guangzhou, China  
jlou@xidian.edu.cn

**Abstract.** Despite the wide attention to federated learning (FL) in the literature, the existing studies mostly focus on supervised federated learning under the horizontally partitioned local dataset setting. This paper will study the unsupervised FL under the vertically partitioned dataset setting. Accordingly, we propose the vertically dataset partitioned federated principal component analysis (VFedPCA) method, which reduces the dimensionality across the joint datasets over all the clients and extracts the principal component feature information for downstream data analysis. VFedPCA features efficient local computation, communication efficiency, and privacy-preserving. Further, we study two communication topologies. The first is a server-client topology where a semi-trusted server coordinates the federated training, while the second is the fully-decentralized topology which eliminates the requirement of the server by allowing clients to communicate with their neighbors. Extensive experiments conducted on real-world datasets justify the efficacy of VFedPCA under vertical partitioned FL setting.

**Keywords:** Principal component analysis · Federated learning · Vertical distributed data

## 1 Introduction

Federated learning (FL) [15] has been receiving increasing attention in the literature, which enables collaborative machine learning in a communication-efficient and privacy-preserving way. FL provides a general-purpose collaborative learning framework, which consists of a coordinating central server and multiple participating clients with their local datasets, e.g., organizations (cross-silo setting) or devices (cross-device setting). More recent FL methods also propose fully-decentralized learning, where clients directly communicate with their neighbors. It eliminates the need of the coordinating server which can sometimes expose security and privacy risks to the collaborative training [26]. During the FL training, the raw local datasets of all clients are kept locally and restricted to be



exchanged or transferred over network for privacy-preserving purpose. Instead, it suffices to communicate only intermediate training variables (e.g., local gradients or local model parameters). Due to its generality and superiority in the privacy-preserving aspect, FL paves its way to a growing number of application areas. Nevertheless, most existing FL methods focus almost exclusively on the supervised learning problems under the horizontally distributed local dataset setting. As far as we know, unsupervised FL learning under the settings of vertically distributed datasets has yet to be well explored.

In fact, the vertically partitioned local dataset setting is common in many practical applications [16]. Under this setting, different clients hold a local partition of features, instead of samples as in the horizontally partitioned setting. It naturally arises when the features/attributes describing the same group of entities are collected and stored among multiple clients. For example, the financial features of a person can split among multiple financial companies s/he has dealt with, e.g., banks, credit card companies, stock market. Similar to the horizontal FL setting, it is important to collaboratively train the model based on all clients' data partition to maximize the global model performance. For example, when a bank refers to a machine learning (ML) model for deciding whether to grant a loan application of a customer, it is ideal for the model training to take account into all the financial records of the customer, not only the transactions history held by this bank. It is apparent from the example that raw local datasets should not be exchanged because the vertically partitioned datasets can contain sensitive information, i.e., the financial status of the customer. Furthermore, unsupervised learning is practically appealing because it need not the labels for model training. The labels can be expensive and time-consuming to mark, and even require domain expertise [13, 19]. The label can also contain sensitive information, which will incur privacy leakage if not handled properly, especially under the vertically partitioned setting where the labels need to be shared among all clients [5, 8].

Under the circumstances, this paper will concentrate on principal component analysis (PCA), which is one of the most fundamental and useful unsupervised learning task with multiple clients holding the vertically partitioned datasets under the FL framework. PCA plays a pivotal role in high-dimensional data analysis by extracting principal components, which are uncorrelated and ordered, with the leading ones retaining most of the data variations [18]. Very recently, federated PCA is studied on the sample-wise distributed (also known as horizontally distributed) data [9, 10]. However, to the best of our knowledge, there is no federated PCA tailored to the feature-wise distributed (also known as vertically distributed) data. The two data distribution settings, though seemed subtle, has fundamental difference as summarized in the review [23]. To this end, we propose a vertically partitioned federated PCA method, abbreviated VFedPCA, which features computational efficiency, communication efficiency and privacy-preserving. In VFedPCA, clients keep their datasets local and only requires the communication of model parameters. Such model exchange suffices

to occur periodically to reduce communication overhead. Within each communication round, clients run the computationally efficient local power iteration [1, 11, 17], and the warm-start power iteration method can further improve performance. Furthermore, we consider the relationship between each independent subset and the full set based on the weight ratio of eigenvalue. Subsequently, we supplement the weight scaling method to further improve the accuracy and performance of the algorithm. In addition, we consider both of two settings: (1) a centralized server coordinates the learning, and (2) the fully decentralized setting which eliminates the need of the server. In summary, the main contributions of this paper are below:

1. We first attempt to study PCA under the vertically-partitioned FL setting.
2. We propose VFedPCA, featuring computational efficiency, communication efficiency, and privacy-preserving. VFedPCA comes with two variants: one requires a central server for coordination and the other performs fully decentralized learning where a client communicates directly with its neighbors.
3. Extensive experiments on real datasets justify the favorable features of VFedPCA under the vertically-partitioned FL setting, while maintaining accuracy similar to unseparated counterpart dataset.

## 2 Notations and Background

### 2.1 Notation

We use boldfaced lower-case letters, e.g.,  $\mathbf{x}$ , to denote vectors and boldfaced upper-case letters, e.g.,  $\mathbf{X}$ , to denote matrix. The superscript is associated with the number of iterations, e.g.,  $\mathbf{X}^t$  denotes the decision variable at iteration  $t$ , while the subscript is associated with indices of different parties, e.g., the data matrix  $\mathbf{X}_i \in \mathfrak{R}^{n \times p_i}$  denotes the  $i$ -th party that has  $p_i$  variables (i.e., features) and  $n$  samples (i.e., users). We use  $\mathbf{X}^*$  to denote a dual space and  $\|\mathbf{x}\|$  denotes the standard Euclidean norm for the vector  $\mathbf{x}$ . For image dataset, the data matrix  $\mathbf{X}_i \in \mathfrak{R}^{n \times m \times m}$  is the  $i$ -th party's input dimension, where  $m$  denotes the pixel size. The summary of the used symbols in this paper is shown in Table 1.

**Table 1.** Summary of frequently used notation

Notation	Description	Notation	Description
$n$	Number of samples	$m$	Number of features
$p$	Partitions	$l$	Local iterations
$t$	Federated communications	$f_i$	The features of the $i$ -th party
$s_i$	The samples of the $i$ -th party	$\alpha_i$	The eigenvalue of the $i$ -th party
$\mathbf{a}_i$	The eigenvector of the $i$ -th party	$\omega_i$	The weight of the $i$ -th party
$\mathbf{u}$	The federated eigenvector	$\mathbf{u}_G$	The global eigenvector

## 2.2 PCA and Power Iteration

In this subsection, we describe the basic PCA setup and present the centralized power iteration. Let  $\mathbf{X} \in \mathfrak{R}^{m \times n}$  be the data matrix. The goal of PCA is to find the top eigenvectors of the symmetric positive semidefinite (PSD) matrix  $\mathbf{A} = \frac{1}{n} \mathbf{X} \mathbf{X}^\top \in \mathfrak{R}^{m \times m}$ , which is the covariance matrix. We assume that the target matrix  $\mathbf{A}$  has eigenvalues  $1 \geq \alpha_1 \geq \alpha_2 \cdots \geq \alpha_m \geq 0$  with corresponding normalized eigenvectors  $\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_m$ , then  $\mathbf{A} \mathbf{a} = \alpha \mathbf{a}$ . In general,  $\mathbf{A}^k \mathbf{a} = \alpha^k \mathbf{a}$  for all  $k = 1, \dots, m$ , which is the basis of the power iteration method. Let  $\alpha^{(0)}$  be an approximation to an eigenvector of  $\mathbf{A}$ , and  $\mathbf{a}^{(0)}$  as an initial vector. The power method [21] estimates the top eigenvector by repeatedly applying the update step below:

$$\boldsymbol{\omega} = \mathbf{A} \mathbf{a}^{(k-1)}, \mathbf{a}^{(k)} = \frac{\boldsymbol{\omega}}{\|\boldsymbol{\omega}\|}, \quad (1)$$

where at each iteration,  $\mathbf{a}^{(k)}$  will get closer to the top principal eigenvector  $\mathbf{a}_1$ . When  $k \rightarrow \infty$ ,  $\mathbf{a}^{(k)}$  will converge to  $\mathbf{a}_1$ .

Recently, [17] has proposed a distributed SVD algorithm based on the local power iteration, which incorporates the periodic communication for communication-efficiency. The LocalPower method can save communication by  $l$  times without much impact on the accuracy. To guarantee the convergence, they partition the data matrix  $\mathbf{X}$  randomly. They assume the local correlation matrix  $\mathbf{A}_i = \frac{1}{s_i} \mathbf{X}_i \mathbf{X}_i^\top$  is a good approximation to the global correlation matrix  $\mathbf{A} = \frac{1}{n} \mathbf{X} \mathbf{X}^\top$ , which is equivalent to  $\|\mathbf{A} - \mathbf{A}_i\| \leq \rho \|\mathbf{A}\|$ , where  $\mathbf{A}_i \in \mathfrak{R}^{s_i \times m}$  is the  $i$ -th partition,  $\rho$  bounds the difference between  $\mathbf{A}_i$  and  $\mathbf{A}$ , which is assumed as small as  $\epsilon$ . However, our setting allows all parties to independently calculate the principal component vector based on the basic PCA method, and derive the federated principal component vector in the global center according to the weights of the parties, which can reduce the complexity of communication calculations and ensure the privacy of participants.

## 2.3 Federated Learning

FL is data-private collaborative learning, where multiple clients jointly train a global ML model with the help of a central coordination server. The goal of FL is achieved through the aggregation of intermediate learning variables, while the raw data of each customer is stored locally without being exchanged or transferred. According to different data partition structures, the distribution can be generally categorized into two regimes, namely horizontally and vertically partitioned data [7]. Currently, there is relatively less attention paid to the vertically partitioned data. Most of the existing cross-silo FL work is based on the supervised datasets, including trees [4], linear and logistic regression [20, 25], and neural networks [15]. These supervised FL works rely on the labels, which are expensive to acquire and require domain expertise. For example, diabetes patients may wish to contribute to the FL with their everyday health monitor data like glucose level and blood press. Since patients lack advanced medical

expertise, their local data are often unlabeled without a medical expert’s evaluation. As far as we know, the current work on the unlabelled data and vertical learning has yet to be explored. The only existing related papers focus on the semi-supervised FL [12] and weakly-supervised FL [14], respectively.

### 3 Federated-PCA on Privacy-Preserving Vertical-Partitioned Data

In this section, we first formulate the problem of federated PCA on the vertically partitioned dataset. Then, we propose the VFedPCA algorithm to collaboratively extract principal components with the participation of all clients, while ensuring privacy by avoiding sharing of raw local datasets. Our method devises the local power iteration for efficient local computation, along with periodic communication for better communication efficiency.

In particular, we consider two types of communication topologies. The first is the server-clients topology, which follows the most existing FL methods by introducing a semi-trusted server to coordinate the training. The second is the fully-decentralized topology, where the clients communicate in a peer-to-peer manner with their neighbors. It eliminates the need of the server, which itself can be malicious and sometimes considered unpractical provided that such a semi-trusted server exists.

#### 3.1 Problem Formulation

Let  $\mathbf{X} \in \mathfrak{R}^{m \times n}$  be the data matrix, which have  $m$  features and  $n$  samples, we partition  $\mathbf{X}$  into  $p$  clients as  $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_p]^\top$ , where  $\mathbf{X}_i \in \mathfrak{R}^{f_i \times n}$  contains  $f_i$  features of  $\mathbf{X}$  and  $\sum_{i=1}^p f_i = m$ . Let  $\mathbf{S} = \frac{1}{m} \mathbf{X}^\top \mathbf{X} \in \mathfrak{R}^{n \times n}$ . Let  $\mathbf{Z} = \mathbf{X} \mathbf{U}^\top \in \mathfrak{R}^m$ , which can be considered as the coordinate of the projection along the direction given by  $\mathbf{U}$ . Note that  $\mathbf{Var}(\mathbf{Z}) = \mathbf{U}^\top \mathbf{S} \mathbf{U}$ , where  $\mathbf{S}$  is the covariance matrix. Our purpose is to find the leading  $k$ -dimensional subspace such that the projection of the data onto the subspace has the largest variance. The problem could be formulated as follows:

$$\max_{\mathbf{U} \in \mathbb{R}^{m \times k}, \mathbf{U}^\top \mathbf{U} = \mathbf{I}} \|\mathbf{U}^\top \mathbf{S} \mathbf{U}\| = \alpha_G \mathbf{U}_G^\top \mathbf{U}_G = \alpha_G, \quad (2)$$

where  $\alpha_G$  are the leading eigenvalues of  $\mathbf{S}$  and  $\mathbf{U}_G$  are the eigenvectors corresponding to  $\alpha_G$ . Our aim is to minimize the distance between the global eigenvector  $\mathbf{U}_G$  that is computed as if all features were centralized together, and each client’s  $\mathbf{U}_i$ , as follows

$$\min_{\mathbf{U}_G, \mathbf{U}_i \in \mathfrak{R}^{m \times k}} \text{dist}(\mathbf{U}_G, \mathbf{U}_i), \text{ for all } i = 1, \dots, p. \quad (3)$$

We define the squared  $\chi$ -distance [3] as the distance between  $\mathbf{U}_G$  and  $\mathbf{U}_i$  by:

$$\text{dist}(\mathbf{U}_G, \mathbf{U}_i) := \sum_{j=1}^m \sum_{h=1}^k \frac{((\mathbf{U}_G)_{j,h} - (\mathbf{U}_i)_{j,h})^2}{(\mathbf{U}_G)_{j,h} + (\mathbf{U}_i)_{j,h}}, \quad (4)$$

where  $\mathbf{U}_G$  is the leading eigenvectors of the global covariance matrix  $\mathbf{S}$ , and  $\mathbf{U}_i$  is the leading eigenvectors of each client's (say  $i$ -th) local covariance matrix.

### 3.2 Local Power Method

Let us consider the data partition among  $p$  clients. In each client, we use power iteration algorithm (also known as the power method) to produce the greatest (in absolute value) eigenvalue of  $\mathbf{A}_i = \frac{1}{f_i} \mathbf{x}_i^\top \mathbf{x}_i$ , and a nonzero vector  $\mathbf{a}_i$ , which is a corresponding eigenvector of  $\alpha_i$ , i.e.  $\mathbf{A}_i \mathbf{a}_i = \alpha_i \mathbf{a}_i$ . For  $l = 1, 2 \dots L$ , each client will compute locally until convergence by

$$\mathbf{a}_i^{l+1} = \frac{\mathbf{A}_i \mathbf{a}_i^l}{\|\mathbf{A}_i \mathbf{a}_i^l\|}, \quad (5)$$

where the vector  $\mathbf{a}_i^l$  is multiplied by the matrix  $\mathbf{A}_i$  and normalized at every iteration. Throughout the paper, we use  $l \in [L]$  to denote local iterations and reserve  $t \in [T]$  to denote global rounds.

If  $\mathbf{a}_i^l$  is an eigenvector of  $\mathbf{A}_i$ , its corresponding eigenvalue is given by

$$\alpha_i^l = \frac{\mathbf{A}_i (\mathbf{a}_i^l)^\top \mathbf{a}_i^l}{(\mathbf{a}_i^l)^\top \mathbf{a}_i^l}, \quad (6)$$

where  $\mathbf{a}_i^l$  and  $\alpha_i^l$  represent the largest eigenvector and eigenvalue of the  $i$ -th client, respectively.

### 3.3 Federated Communication

Each client uploads the eigenvector  $\mathbf{a}_i^t$  and the eigenvalue  $\alpha_i^t$  to the central sever. First, the server computes the weight  $\omega_i$  of each client. Then, the server merges all clients' results and computes the federated eigenvector  $\mathbf{u}^t$

$$\omega_i^t = \frac{\alpha_i^t}{\sum_{i=1}^p \alpha_i^t}, \mathbf{u}^t = \omega_1^t \mathbf{a}_1^t + \omega_2^t \mathbf{a}_2^t + \dots \omega_p^t \mathbf{a}_p^t, \quad (7)$$

where  $\omega_i^t$  is the weight of the  $i$ -th client and  $\mathbf{u}^t$  is the shared projection feature vector by merging all clients'  $\mathbf{a}_i^t$ 's.

### 3.4 Sever-Clients Architecture

All clients share parameters with the help of the central coordination server. In our setting, this third-client coordinator can be trusted, which means that it will honestly conduct the designated functionality and will not attempt to breach the raw data of any clients. After adjusting the new parameters, this federated-parameter is returned to all clients. Then, each client calculates locally based on this new federated-parameter. Moreover, this "communication-and-local-computation" cycle is iterative. The framework of this model is shown in

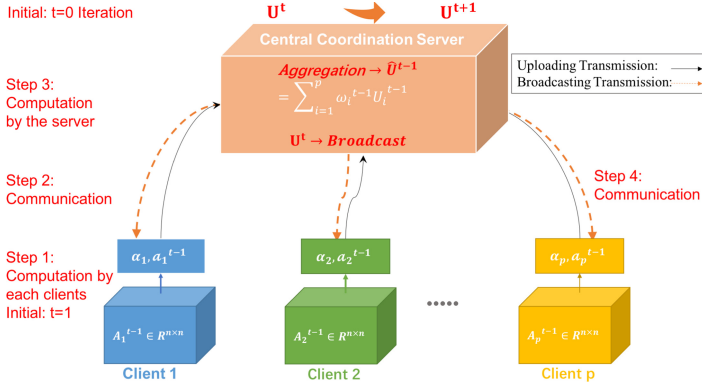


Fig. 1. The framework of VFedPCA with central server.

Fig. 1. The algorithm is summarized in Algorithm 1. The key steps of this method are as follows:

**Local Training:** All clients perform local power method independently. Each client first calculates the covariance matrix  $\mathbf{A}$ . Subsequently, the largest eigenvector  $\mathbf{a}$  and eigenvalue  $\alpha$  are calculated (see steps 4–10 of Algorithm 1);

**Model Integration:** The central server calculates the weight  $\omega_i$  occupied by each client in the federated model by receiving the eigenvalue  $\alpha_i$  of each client, and then combines the received eigenvector  $\mathbf{a}_i, i \in \{1, \dots, p\}$ , to derive a federated feature vector  $\mathbf{u}^t$ , where the communication is cyclical (see steps 12–16 of Algorithm 1);

**Parameters Broadcasting and Updating:** The central coordination server broadcasts the aggregated parameters  $\mathbf{u}^t$  to the  $p$  clients. Each client updates local eigenvector with the new returned federated feature vector  $\mathbf{u}^t$  (see step 17 of Algorithm 1). The advantage of this model is relatively more efficient, although it relies on the help of the server which can be potentially malicious.

### 3.5 Local Power Iteration with Warm Start

The general power iteration algorithm starts with an initialization vector  $\mathbf{a}^{(0)}$ , which may be an approximation to the dominant eigenvector or a random unit vector. Here, we initialize the algorithm from a “warm-start” vector  $\mathbf{a}^{(0)}$ , which is based on the federated vector from the previous communication round. It is natural to reach convergence faster after consecutive iterations, which will reduce the actual running time and improve the performance of the local power iteration algorithm in Sect. 3.2. The algorithm is summarized in Algorithm 2. The main step is as follows:

**Local Training:** Each node uses the federated feature vector  $\mathbf{u}$  as the initialize value to perform local power method, then calculate the  $k+1$  to  $2k$  eigenvector  $\mathbf{a}$  and eigenvalue  $\alpha$  and send to the server (see steps 2–8 of Algorithm 2).

---

**Algorithm 1: VFedPCA Learning with Central Server**


---

```

1 ⇒ Run on the  $i$ -th node
2 Input: Data  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{f_i}\} \in \mathfrak{R}^{n \times f_i}$  belongs to client  $i$ 
3 Output: Federated principal eigenvector  $\mathbf{u}$ 
4 Initial: Let  $\bar{\mathbf{x}} = 0$ ,  $\mathbf{a}_i^{(0)} \in \mathfrak{R}^n$  randomly
5 for  $l=1$  to  $L$  do
6   | while each client  $i, i = \{1, 2 \dots p\}$  do
7     |    $\mathbf{a}_i^{l+1} = \frac{\mathbf{A}_i \mathbf{a}_i^l}{\|\mathbf{A}_i \mathbf{a}_i^l\|}$ ,  $\alpha_i^l = \frac{\mathbf{A}_i (\mathbf{a}_i^l)^\top \mathbf{a}_i^l}{(\mathbf{a}_i^l)^\top \mathbf{a}_i^l}$ 
8     | end
9     | Send  $(\alpha_i^l, \mathbf{a}_i^l)$  to the central server.
10 end
11 ⇒ Run on central coordination server
12 for  $t=1$  to  $T$  do
13   | Receive  $(\alpha_i^t, \mathbf{a}_i^t) = (\alpha_i^l, \mathbf{a}_i^l)$  from  $n$  clients
14   | for  $i = 1$  to  $p$  do
15     |    $\omega_i^t \leftarrow \alpha_i^t$  and  $\mathbf{u}^t \leftarrow Merge(\omega_i^t, \mathbf{a}_i^t)$  by eq.(7)
16     | end
17   | Broadcast and update eigenvector  $\mathbf{u}^t \leftarrow \mathbf{u}^{t+1}$ 
18 end

```

---

This method considers the use of an adaptive loop idea which allows communication between clients to reduce the error to a certain extent, especially for the case where the error between the general federated result  $\mathbf{u}$  and the global result  $\mathbf{u}_G$  is large.

### 3.6 Weight Scaling Method

The federated result may be sometimes not beneficial to all clients. In general, the client with the larger eigenvalue has a greater influence on the federated result, which hints a natural intuition that following the direction of the clients with larger eigenvalues tends to reach the global consensus faster and costs less communication rounds. Inspired by this intuition, we introduce a weight scaling factor to further improve the federation communication. The improved weight scaled federated average is formulated by

$$\begin{aligned} \mathbf{u}^t = & (1 + \eta_1^t) \omega_1^t \mathbf{a}_1^t + \dots + (1 + \eta_{\lceil p/2 \rceil}^t) \omega_1^t \mathbf{a}_{\lceil p/2 \rceil}^t \\ & + (1 - \eta_{\lceil p/2 \rceil + 1}^t) \omega_1^t \mathbf{a}_{\lceil p/2 \rceil + 1}^t \dots (1 - \eta_p^t) \omega_p^t \mathbf{a}_p^t, \end{aligned} \quad (8)$$

where  $\sum_{i=1}^{\lceil p/2 \rceil} \eta_i^t = \sum_{i=\lceil p/2 \rceil + 1}^p \eta_i^t$ . In brief, we gradually increase the impact of the first half of the clients with larger eigenvalues, while further decrease the impact of the clients of the second half with smaller eigenvalues. The algorithm is summarized in Algorithm 3. The main step is as follows:

**Model Integration:** The central server calculates the weight  $\omega_i$  occupied by each client in the federated model based on the received eigenvalue  $\alpha_i$  of each

---

**Algorithm 2: Local Power Iteration with Warm Start**

---

```

1 ⇒ Run on the  $i$ -th node
2 Initial: Let  $\mathbf{a}_i^{(0)} = \mathbf{u}$  -warm-start
3 for  $l=1$  to  $L$  do
4   while each client  $i, i = \{1, 2 \dots p\}$  do
5      $\mathbf{a}_i^{l+1} = \frac{\mathbf{A}_i \mathbf{a}_i^l}{\|\mathbf{A}_i \mathbf{a}_i^l\|}$   $\alpha_i^l = \frac{\mathbf{A}_i (\mathbf{a}_i^l)^\top \mathbf{a}_i^l}{(\mathbf{a}_i^l)^\top \mathbf{a}_i^l}$ 
6   end
7   Send  $(\alpha_i^l, \mathbf{a}_i^l)$  to the central server.
8 end

```

---



---

**Algorithm 3: Weight Scaling Method**

---

```

1 ⇒ Run on central collaborative server
2 for  $t=1$  to  $T$  do
3   Receive  $\alpha_i^t, \mathbf{a}_i^t$  from  $n$  clients
4   for  $i = 1$  to  $p$  do
5      $\omega_i^t \leftarrow \alpha_i^t$  by eq.(7),  $\eta_i^t = \eta$ 
6      $\mathbf{u}^t \leftarrow \text{Merge}(\eta_i^t, \omega_i^t, \mathbf{a}_i^t)$  by eq. (8)
7   end
8 end

```

---

client, and then adds  $\eta$  parameter to further adjust the weight scale of each client. Then, it combines the received eigenvector  $\mathbf{a}_i, i \in \{1, \dots, p\}$ , to derive a federated feature vector  $\mathbf{u}^t$  (i.e., steps 2–8 of Algorithm 3). This method is especially suitable for situations where some clients have a larger weight in the federated process.

### 3.7 Fully Decentralized Architecture

In this variant, we eliminate the need of the central server, where all clients only share principal component parameters with each other, and compute the results locally. The framework of this model is shown in Fig. 2. The algorithm is summarized in Algorithm 4. The main step of this method is as follows:

**Parameters Communication and Updating:** Communication among clients is connected and share intermediate results of the parameters  $\alpha_i$  and  $\mathbf{a}_i$ . Each client obtains the extracted final data based on calculating the weights  $\omega_i$ 's occupied by all clients and updates locally (see steps 3–5 of Algorithm 4).

This model circumvents the need for third-client agencies to help participants further reduce some external risks, especially for two-client scenarios. However, the principal components of all clients need to be calculated independently, and the computing efficiency will be greatly affected.



### 3.8 Complexity Analysis

In each client, let  $L$  be the total number of local iterations, which only depends on the eigen-gap  $\Delta$  after full passes over the data between the top two eigenvalues of  $\mathbf{A}_i$ . In Algorithm 1, let  $T$  be the total number of federated aggregations executed by the server.

**Theorem 1 (Local Iteration Complexity).** *After  $O(\frac{1}{\Delta} \log \frac{n}{\epsilon})$  steps, the local power method can achieve  $\epsilon$ -accuracy.*

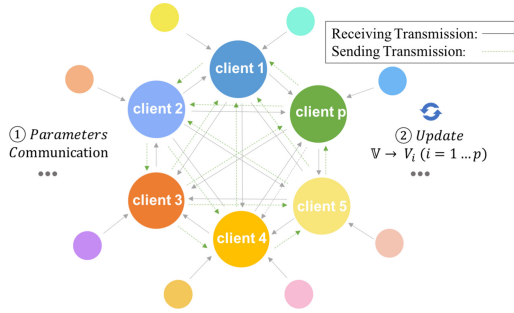


Fig. 2. The framework of fully decentralized (peer-to-peer) VFPCA learning

---

#### Algorithm 4: Fully decentralized VFedPCA learning

---

```

1 ⇒ Run on the  $i$ -th node
2 Parameters communication between connected clients
3 for  $i = 1$  to  $p$  do
4   |  $\omega_i^l \leftarrow \alpha_i^l, \mathbf{a}^l \leftarrow Merge(\omega_i^l, \mathbf{a}_i^l)$ 
5   | Update eigenvector  $\mathbf{a}_i^l \leftarrow \mathbf{a}^l$ 
6 end
    
```

---

*Remark 1.* We show the proof in the appendix<sup>1</sup>. According to Theorem 1, the normalized iterate  $\frac{\mathbf{A}_i \mathbf{a}_i^l}{\|\mathbf{A}_i \mathbf{a}_i^l\|}$  is an  $\epsilon$ -accurate estimate of the top principal component. Hence, the total local power iterations complexity is  $O(\frac{1}{\Delta} \log \frac{n}{\epsilon})$ .

## 4 Experimental Results

This section will empirically evaluate the proposed method. In the experiments, we utilized four groups of real-world datasets: 1) structured datasets from different domains [6]; 2) medical image dataset [24]; 3) Face image dataset [2]; 4) Gait

<sup>1</sup> Link to Appendix: <https://www.comp.hkbu.edu.hk/~ymc/papers/conference/wise21/appendix.pdf>.

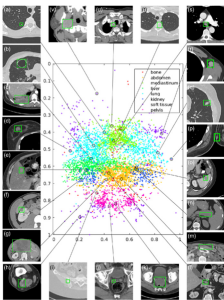
image dataset [22]. The feature and sample size information of all datasets are summarized in Table 2. In addition, illustrative images of the image dataset are shown in Figs. 3, 4 and 5, respectively. In all experiments, we measure the communication cost by calculating the bits of all the variables transmitted throughout the algorithm execution. As a preliminary study, we set  $T = 1$ , i.e., one-shot communication between the clients and the server, in our experiments, unless otherwise stated.

#### 4.1 Experiment on Structured Dataset

**Semi-synthetic Datasets.** We use 8 structured datasets from different domains, which are the real data that are publicly available at the UCI Machine Learning Repository [6]. We compare the communication cost and estimation error by the different settings of  $p$  and  $l$  based on the feature-wise setting. We change the number of partitions  $p = 3, 5, 10$ ,  $p = 10, 50, 100$ , and the local iterations  $l = 5, 10, 20$ ,  $l = 50, 100, 200$ , and we vary the number of communication period to

**Table 2.** Summary of datasets

Dataset	# of features	# of samples
College	9	990
GlaucomaM	54	196
PimaIndiansDiabetes	33	351
Musk	166	476
Vehicle	18	846
Sonar	60	208
Swarm	2400	2000
TCGA	20500	800
DeepLesion	262144	100
Face	10000	225
CASIA	65536	240



**Fig. 3.** The DeepLesion dataset.



**Fig. 4.** The YaleFace dataset



Fig. 5. The CASIAGait database.

$t = 5, t = 10$ . The number of each client’s features is split and configured according to the different settings of  $p$ .

**Communication Cost and Estimation Error** In this part, we simulate the communication between the clients and the server on a single machine, where the CPU time are wall-clock time.

- **The effect of local feature size.** In Fig. 6, we fix  $l = 10$  and  $l = 100$  but change the number of involved clients  $p = 3, 5, 10$  and  $p = 10, 50, 100$ . Then, we plot the distance error and the running time cost between the global eigenvector and the federated eigenvector with the different partitions  $p$ . In all experiments, compared with the un-communicated situation, the distance error after the communication has been significantly reduced and convergence has been achieved. It can be observed that the smaller  $p$  will lead to less communication cost.

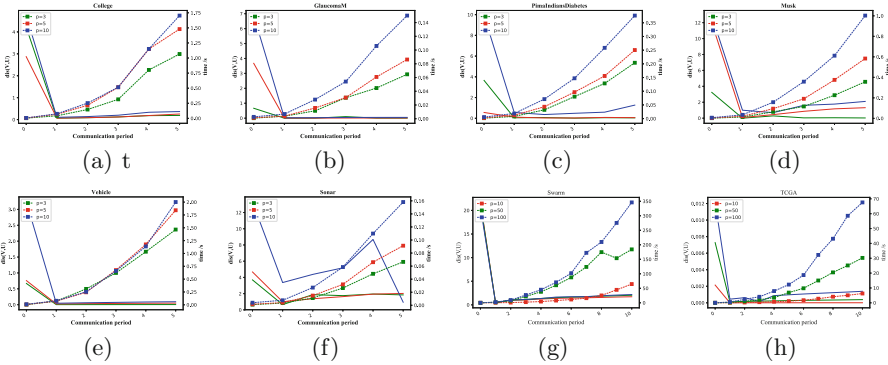
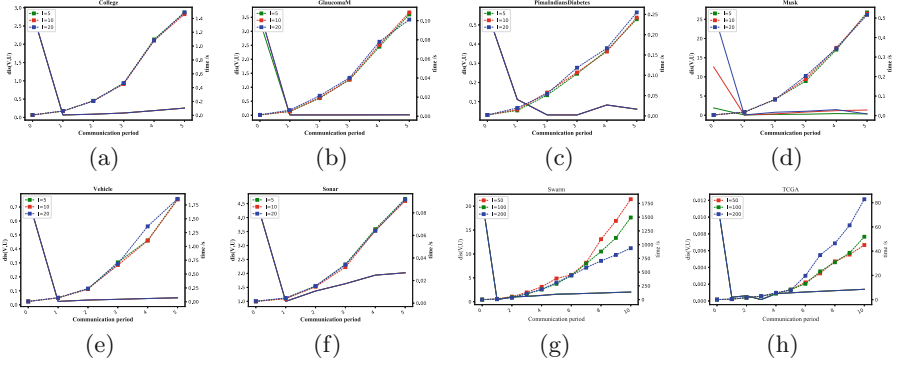
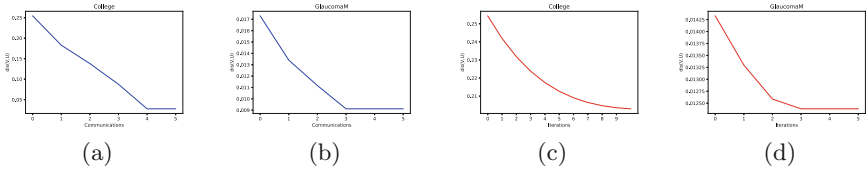


Fig. 6. The distance error (left vertical axis) and the running time (right vertical axis) of Algorithm 1 with respect to the number of parties on structured datasets: (a) College, (b) GlaucomaM, (c) PimaIndiansDiabetes, (d) Musk, (e) Vehicle, (f) Sonar, (g) Swarm, and (h) TCGA, where  $l = 10, p = 3, 5, 10$  for (a)–(f), and  $l = 100, p = 10, 50, 100$  for (g)–(h).

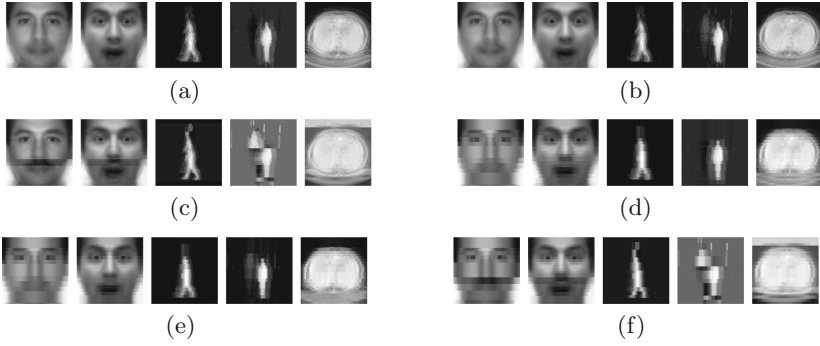


**Fig. 7.** The distance error (left vertical axis) and the running time (right vertical axis) of Algorithm 1 with respect to the number of local iterations on structured datasets: (a) College, (b) GlaucomaM, (c) PimaIndiansDiabetes, (d) Musk, (e) Vehicle, (f) Sonar, and (g) Swarm (h) TCGA; For (a)–(f),  $p=5$ ,  $l=5, 10, 20$ ; for (g)–(h),  $p=50, 100$ ,  $l=50, 100, 200$ .

- **The effect of local iterations.** In Fig. 7, we fix  $p=5$  and  $p=100$ , but change the number of local power iterations to  $l=5, 10, 20$  and  $l=50, 100, 200$ . Then, we plot the distance error and the running time cost between the global eigenvector and the federated eigenvector with the different local iterations  $l$ . All experimental results show that, after the communication, the error eventually decreases and the algorithm converges. It can also be seen that the result will remain stable when  $l$  is up to 20.
- **The effect of warm-start power iterations.** In Fig. 8, we fix  $p=5$  and local power iterations  $l=100$ . Then, we plot the distance error between the global eigenvector and the federated eigenvector after communications  $t=5$ . Experimental results show the trend that the error further decreases and converges.
- **The effect of  $\eta$ .** In Fig. 8, we fix  $p=5$  and local power iterations  $l=100$ . Then, we plot the distance error between the global eigenvector and the federated eigenvector after iterations  $t=5, 10$ . Experimental results show the trend that the error further decreases and converges compared with the one without using the adjustment factor  $\eta$ .



**Fig. 8.** The distance error of Algorithms 2 and 3 with respect to the number of local iterations on structured datasets: (a) College, (b) GlaucomaM, where  $p=5$ ,  $l=100$ .



**Fig. 9.** The final results of comparative experiment on image datasets: YaleFace (center-light and surprised), CasiaGait (sequence 1 and 10) and DeepLesion, where (a) PCA on the un-split data, (b) VFedPCA on the split data, (c) PCA on the isolated data. Image segmentation ( $k = 10$ ) results: (d) PCA on the un-split data, (e) VFedPCA on the split data, (f) PCA on the isolated data.

## 4.2 Case Studies

**Medical Image Dataset.** The DeepLesion dataset [24] is a CT slices collection from 4427 unique patients, which contains a variety of lesions (e.g., lung nodules, liver lesions). We selected 100 samples from the dataset and each image is normalized to an  $512 \times 512$  gray image. We set  $p = 8$  clients, re-split the features, and assign them equally to each client. We used the commonly clustering method: k-means, with the number  $k = 20$  of clusters.

**Yale Face Dataset.** We use the Yale Face Dataset [2], which contains 165 grayscale images of 15 subjects. Each subject configures 11 different facial expressions and  $n = 15$  samples for each facial expression, where each face image is normalized to a  $100 \times 100$  gray image and we set  $p = 10$  clients. We use the k-means with  $k = 10$ .

**Gait Estimation.** The CASIA is a gait database [22] for gait recognition, including 20 persons. We have image sequences, 4 sequences for each of the three directions and  $n = 20$  samples for each direction, where each image is resized to the  $256 \times 256$  scale. We set  $p = 16$  clients. We used the k-means with  $k = 10$ .

**Comparative Results.** We first perform PCA on image datasets. Figure 9 (a)(b)(c) show that the final images after federating is almost the same as the final images after un-split image data. Then, to further verify that the final image results extract useful features, we also perform the clustering experiment on the image dataset. Figure 9 (d)(e)(f) show the clustering result (also known as image segmentation) after using the federated PCA method, which

also shows improvement over the isolated PCA (i.e., independently run PCA on splitted local datasets). It will help each client to further perform local training tasks.

## 5 Concluding Remarks

In this paper, we have proposed VFedPCA algorithm, which can obtain a collaborative model that improves over the local models learned separately by each client. Through extensive comparative studies on various tasks, we have verified that the collaborative model achieves comparative accuracy with the centralized model as if the dataset were un-splitted. In addition, we propose two strategies, i.e., the local power iteration warm start method and the weight scaling method, to further improve the performance and accuracy of the model. In terms of the federated communication, the full decentralized model has lower communication cost and more flexible application value.

From the practical point of view, it is not uncommon that there exists a non-linear relationship between various data, especially for the image data, most of which possess a non-linear relationship. Under the circumstances, the PCA method which is essentially linear is inapplicable to find an appropriate representative direction. In the future, we will therefore study the vertical federated kernel PCA learning on datasets with the different types of non-linearity.

**Acknowledgment.** This work was supported by HKBU AI-Info Communication Study (AIS) Scheme with Project No. AIS 21-22/03.

## References

1. Balcan, M.-F., Du, S.S., Wang, Y., Yu, A.W.: An improved gap-dependency analysis of the noisy power method. In: Conference on Learning Theory, pp. 284–309. PMLR (2016)
2. Belhumeur, P.N., Hespanha, J.P., Kriegman, D.J.: Eigenfaces vs. fisherfaces: recognition using class specific linear projection. *IEEE Trans. Pattern Anal. Mach. Intell.* **19**(7), 711–720 (1997)
3. Cha, S.: Comprehensive survey on distance/similarity measures between probability density functions (2007)
4. Cheng, K., Fan, T., Jin, Y., Liu, Y., Chen, T., Yang, Q.: SecureBoost: a lossless federated learning framework. [arXiv:1901.08755](https://arxiv.org/abs/1901.08755) (2019)
5. Du, W., Han, Y.S., Chen, S.: Privacy-preserving multivariate statistical analysis: linear regression and classification. In: SIAM International Conference on Data Mining. SIAM (2004)
6. Dua, D., Graff, C.: UCI machine learning repository (2017)
7. Fan, J., Wang, D., Wang, K., Zhu, Z.: Distributed estimation of principal eigenspaces. *Ann. Stat.* **47**(6), 3009 (2019)
8. Gascón, A., et al.: Secure linear regression on vertically partitioned datasets. *IACR Cryptol. ePrint Arch.* 2016:892 (2016)
9. Grammenos, A., Smith, R.M., Crowcroft, J., Mascolo, C.: Federated principal component analysis. In: NeurIPS (2020)

10. Guo, X., Li, X., Chang, X., Wang, S., Zhang, Z.: Privacy-preserving distributed SVD via federated power. [arXiv:2103.00704](#) (2021)
11. Hardt, M., Price, E.: The noisy power method: a meta algorithm with applications. In: *NeurIPS* (2014)
12. Jeong, W., Yoon, J., Yang, E., Hwang, S.J.: Federated semi-supervised learning with inter-client consistency. [arXiv:2006.12097](#) (2020)
13. Ji, Z., Zou, X., Huang, T., Wu, S.: Unsupervised few-shot learning via self-supervised training. [arXiv:1912.12178](#) (2019)
14. Jin, Y., Wei, X., Liu, Y., Yang, Q.: Towards utilizing unlabeled data in federated learning: a survey and prospective. [arXiv:Learning](#) (2020)
15. Kairouz, P., McMahan, H.B., Avent, B., et al.: Advances and open problems in federated learning. [arXiv:1912.04977](#) (2019)
16. Kargupta, H., Huang, W., Sivakumar, K., Johnson, E.: Distributed clustering using collective principal component analysis. *Knowl. Inf. Syst.* **3**(4), 422–448 (2001)
17. Li, X., Wang, S., Chen, K., Zhang, Z.: Communication-efficient distributed SVD via local power iterations. [arXiv:2002.08014](#) (2020)
18. Mishra, S.P., et al.: Multivariate statistical data analysis-principal component analysis (PCA). *Int. J. Liv. Res.* **7**(5), 60–78 (2017)
19. Nian, K., Zhang, H., Tayal, A., Coleman, T., Li, Y.: Auto insurance fraud detection using unsupervised spectral ranking for anomaly. *J. Finance Data Sci.* **2**(1), 58–75 (2016)
20. Nock, R., et al.: Entity resolution and federated learning get a federated resolution. [arXiv:1803.04035](#) (2018)
21. Saad, Y.: *Numerical methods for large eigenvalue problems: revised edition*. SIAM (2011)
22. Wang, L., Tan, T., Ning, H., Weiming, H.: Silhouette analysis-based gait recognition for human identification. *IEEE Trans. Pattern Anal. Mach. Intell.* **25**(12), 1505–1518 (2003)
23. Wu, S.X., Wai, H.-T., Li, L., Scaglione, A.: A review of distributed algorithms for principal component analysis. *Proc. IEEE* **106**(8), 1321–1340 (2018)
24. Yan, K., Wang, X., Lu, L., Summers, R.M.: Deeplesion: automated mining of large-scale lesion annotations and universal lesion detection with deep learning. *J. Med. Imaging* **5**(3), 036501 (2018)
25. Yang, Q., Liu, Y., Chen, T., Tong, Y.: Silhouette analysis-based gait recognition for human identification. *ACM Trans. Intell. Syst. Technol. (TIST)* **10**(2), 1–19 (2019)
26. Zantedeschi, V., Bellet, A., Tommasi, M.: Fully decentralized joint learning of personalized models and collaboration graphs. In: *International Conference on Artificial Intelligence and Statistics*, pp. 864–874 (2020)



# Anchoring-and-Adjustment to Improve the Quality of Significant Features

Eunkyung Park<sup>1</sup>(✉), Raymond K. Wong<sup>1</sup>, Junbum Kwon<sup>2</sup>,  
and Victor W. Chu<sup>3</sup>

- <sup>1</sup> Computer Science and Engineering, University of New South Wales,  
Sydney, Australia  
{eunkyung.park, ray.wong}@unsw.edu.au
- <sup>2</sup> School of Marketing, University of New South Wales, Sydney, Australia  
junbum.kwon@unsw.edu.au
- <sup>3</sup> Computer Science and Engineering, Nanyang Technological University,  
Singapore, Singapore  
wchu@ntu.edu.sg

**Abstract.** There is an enormous demand for Explainable Artificial Intelligence to obtain human-understandable models. For example, advertisers are keen to understand what makes video ads successful. In our investigation, we have analysed heterogeneous visual, auditory, and textual content features from YouTube video ads. This paper proposes a two-stage anchoring-and-adjustment approach. In the first stage, we search for the optimum penalized value in the regularization path of Lasso that maximizes the number of Significant Features (SFs). After that, we improve the quality of SFs by dropping features with high Variance-Inflation-Factor (VIF) because high VIF often makes a spurious set of SFs. Experiments show that, compared to the one-stage approach without the adjustment stage, our proposed two-stage approach results in a smaller number of SFs but a higher ability to identify true features that appeal to ad viewers from human evaluation. Furthermore, our approach can identify much more SFs while maintaining similar prediction accuracy as what Lasso and Elastic-net can obtain.

**Keywords:** Explainable models · Content features · Anchoring-and-adjustment · Variance-inflation-factor · Significance test

## 1 Introduction

Recently, there is an enormous demand for Explainable Artificial Intelligence (XAI) methods [4], as decision-makers often need to fully understand what factors drive the outcomes in many practical applications.

This paper proposes a two-stage anchoring-and-adjustment approach to improve the quality of Significant Features (SFs). In the first stage, we maximize the number of the candidate of SFs by adopting SFLasso [10] and SFLasso-SI (Selective Inference) [9] that search for the optimum penalized value  $\lambda$  in the



regularization path of Lasso [14] that maximizes the number of SFs. Considering two opposing factors: model size (i.e., the number of active variables) and the correlations among the active variables, SFLasso-SI chooses  $\lambda$ , generating the biggest number of SFs via statistical significance test using SI [7, 13] in training data. After that, we further improve the quality of SFs by dropping features with high Variance-Inflation-Factor (VIF), which measures the amount of correlation with other features, because high VIF can inflate either the magnitude of coefficients or its variance, and thus often makes a spurious set of SFs.

Post-hoc explainability approach typically runs complex DNN first for high prediction accuracy and then runs the post model to explain the first model’s prediction by selecting input features (DeepLIFT [11], L2X [2], and ACD [12]). Given that our data have small observations, which would not be enough to train DNN, we extend Lasso, an intrinsic explainable model. This allows us to do a statistical test for selecting features using recently developed selective inference.

Most Lasso variants focus on improving prediction accuracy, such as Elastic-net[16], and Enumerate Lasso [3] or finding more or better active variables, but not finding more SFs. Recently, SFLasso [10], and SFLasso-SI [9] was developed to find the maximum number of SFs.

OLS post-Lasso [1] proposed to rerun OLS with the active variables resulted from Lasso. However, this naive approach results in many false SFs [5, 7, 13]. Covariance Test [8] pioneered for significance test after variable selection when signal variables are not too correlated with noise variables [5]. For more general explanatory variables, Lee et al. [7] derived closed-form p-values for selected active variables after fitting Lasso with a fixed value of hyperparameter  $\lambda$ . While this selective inference can exclude some false SFs, we screen such false SFs using VIF iteration to drop highly correlated features with other explanatory variables.

Experiments using YouTube video ads show that, compared to the one-stage approach without adjustment stage (SFLasso-SI), our proposed two-stage approach (SFLasso-SI+VIF) results in a smaller number of SFs but much higher accuracy in identifying true features that appeal to ad viewers from human evaluation. Furthermore, our approach can identify more SFs while maintaining similar prediction accuracy as Lasso and Elastic-net can obtain.

## 2 Proposed Two-Stage Anchoring-and-Adjustment Approach

Anchoring-and-Adjustment heuristic is one of the strategies to estimate unknown quantities starting with information one does know and then adjust until an acceptable value is reached [15]. To find many true features, we propose a two-stage anchoring-and-adjustment method in the framework of the Explainability maximized method.

### 2.1 Anchoring Stage Using Explainability Maximized Method

Lasso penalizes non-zero coefficients by adding a regularization term  $\lambda$  in the objective function of OLS as follows:  $\min_{\beta} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1$ , where

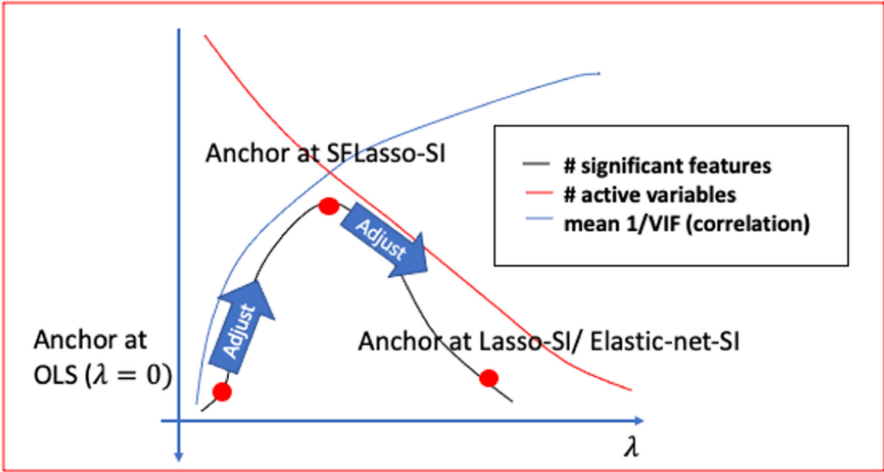


Fig. 1. Visualization of anchoring and adjustment

$X \in \mathbb{R}^{n \times p}$  (e.g., visual objects, text),  $y \in \mathbb{R}^n$  (e.g., the number of likes of YouTube video ads), and  $\beta \in \mathbb{R}^p$  is a vector of regression coefficient. Figure 1 shows that as  $\lambda$  decreases, the number of active variables increases, but the correlation among the active variables also increases (i.e., the mean of the inverse of VIF decreases). This relationship suggests that too big or too small values of  $\lambda$  are not good at identifying SFs. When  $\lambda$  is very big (e.g., Lasso, Elastic-net), only a small number of active variables is generated, so multicollinearity is not likely to happen. However, only a few active variables result in even fewer SFs. On the other hand, when  $\lambda$  is very small (e.g.,  $\lambda = 0$  for OLS), there are many active variables, so the correlation among active variables is high. This multicollinearity reduces the number of SFs.

Considering those two competing forces, one can expect that the maximum number of SFs is likely to occur at some point of  $\lambda$  where there is a good enough number of active variables while the correlation among  $x$  variables is not too high. Based on the above behaviors, SFLasso and SFLasso-SI search for the optimum penalized value in the regularization path of Lasso that maximizes the number of SFs as follows.  $f(\lambda) = \max \sum_{i=1}^p I(p\text{-value}_{\beta_i} \leq 0.05)$ , where  $I = 1$  if  $p\text{-value}_{\beta_i} \leq 0.05$  or  $I = 0$  otherwise. For a given  $\lambda$ , the SFLasso selects active variables  $A$  with non-zero coefficients that minimize the objective function. For the testing of each selected variable's significance, while SFLasso runs OLS with the selected active variables and uses  $p$ -value, SFLasso-SI applies the selective inference [7, 13]. The selective test does more conservative test by making false SFs insignificant [5, 7, 13]. Then, the number of SFs is counted. This process is repeated within a range of  $\lambda$  values. Finally, the  $\lambda$  that generates the maximum number of SFs is chosen.

## 2.2 Adjustment Stage

While SFLasso-SI finds many SFs, SFLasso-SI might include false SFs because it tends to choose features with an inflated magnitude of the coefficient to pass the significance test. James et al. [6] suggested that VIF that exceeds 10 indicates a problematic amount of multicollinearity. To adjust the solution of SFLasso-SI, we drop the feature with the highest VIF in the active variable set if the VIF is higher than 10. Note that we do not drop all the features with VIF greater than 10 at one time because the VIF of a focal variable is affected by the other variables. Usually, once a variable with very high VIF is dropped, VIFs for the remaining variables become much smaller. Then, we drop another feature with the highest VIF among the remaining  $x$  variables. We repeat this VIF iteration until the highest VIF becomes less than 10.

Figure 1 illustrates the adjustments from the three anchoring points from OLS, SFLasso-SI, and Lasso-SI. As OLS has many highly correlated variables, dropping the most correlated variables by VIF iteration reduces the variance of the coefficient, resulting in increases in the number of SFs. However, for SFLasso-SI, which is likely to generate false SFs with inflated coefficients, VIF iteration is likely to reduce the number of SFs. This reduction is the evidence to show that the correlation among active  $x$  variables of SFLasso-SI contributes to the inflation of magnitude of the coefficient in the numerator more than that of its variance in the denominator in the t-statistics, leading to some false SFs being included in the set of SFs. Lastly, Lasso-SI often has a few SFs due to the small number of active variables, making a low correlation among active  $x$  variables. Therefore, the highest VIF is often smaller than the threshold (10), resulting in no adjustment.

## 3 Empirical Evaluation

We conduct experiments using YouTube video ads. We divide our datasets into the training (70%), validation (15%), and test (15%) sets (Table 1). We extract visual objects, speech tones, and spoken words. We use a bag-of-words model to count word frequency. We include features with occurrences of more than 5 across video ads. The first block in Table 2 shows the result of World Vision US YouTube videos. The first 4 models run only the first anchoring stage without VIF iteration. Among these 4 models, SFLasso-SI identifies the biggest number

**Table 1.** Size of training, variables, and unmatched ratio

	# train ( $n$ )	# vars ( $p$ )	Unmatched ratio (%)
World Vision US	370	1,629	21.42
World Vision CA	318	1,374	25.98

(68) of SFs. The next 4 models do the second adjustment stage by doing the VIF iterations. As expected, our proposed SF<sub>Lasso-SI+VIF</sub> generates a smaller number of SFs than SF<sub>Lasso-SI</sub> but the biggest number of SFs among all other competing models that go through the VIF iteration. Furthermore, SF<sub>Lasso-SI+VIF</sub> shows a similar prediction accuracy (RMSE) level with other predictability maximized methods such as Lasso+VIF. Table 3 shows that our SF<sub>Lasso-SI+VIF</sub> outperforms all the other models in the F1 score. In particular, compared to SF<sub>Lasso-SI</sub>, adding VIF iteration improve precision substantially from 26 to 69 for the US and from 28 to 63 in Canada, while recall increases only slightly in Canada. The recall is low across all models, which means that identifying truly appealing features among video content features is a challenging problem. Nevertheless, the VIF iteration helps improve the quality of identified SFs (i.e., higher precision).

**Table 2.** Results from models - World Vision US and Canada

		$\lambda$	$\alpha$	RMSE (train)	RMSE (test)	VIF Mean	VIF Max	#act	#sig
US	OLS	0.0		0.57	2840.62	1374.35	106505.60	368	0
	Lasso-SI	37.9		44.40	202.95	1.27	1.66	4	2
	Elastic-SI	285.8	0.1	44.04	204.13	1.69	4.00	6	1
	<b>SFLasso-SI</b>	0.6		20.77	209.29	3.78	30.15	151	68
	OLS+VIF			23.65	219.56	5.50	9.79	237	16
	Lasso-SI+VIF			44.40	202.95	1.27	1.66	4	2
	Ela-SI+VIF			44.04	204.13	1.69	4.00	6	1
	<b>SF-SI+VIF</b>			20.99	207.47	3.35	9.57	144	26
CA	OLS	0.0		2.20	3622.45	664.79	139282.80	316	0
	Lasso-SI	140.7		148.82	277.41	1.13	1.17	3	1
	Elastic-SI	1406.6	0.1	148.82	277.41	1.13	1.17	3	0
	<b>SFLasso-SI</b>	2.7		87.85	302.01	2.96	18.28	107	58
	OLS+VIF			58.29	468.40	5.66	9.58	218	15
	Lasso-SI+VIF			148.82	277.41	1.13	1.17	3	1
	Ela-SI+VIF			148.82	277.41	1.13	1.17	3	0
	<b>SF-SI+VIF</b>			65.85	295.85	2.70	9.64	102	27

OLS identifies zero SF. Through the VIF iteration (threshold 10), many highly correlated variables are dropped from 368 to 237. As a result, 16 SFs are identified. This VIF process confirms that multicollinearity hides SFs. On the contrary, Lasso-SI and Elastic-SI drop many variables due to the mismatch between train and validation data. As a result, only 4 (Lasso-SI) and 6 (Elastic-SI) features are active, and only 2 (Lasso-SI) and 1 (Elastic-SI) features have significance. Since VIF is already low enough (i.e., smaller than threshold 10), there are no additional variables to be dropped with VIF criteria. Therefore, Lasso-SI+VIF and Elastic-SI+VIF do not gain additional SFs. As discussed earlier, SF<sub>Lasso-SI</sub> finds the biggest number (68) of SFs by trading off the reduction in the amount of correlation against the size of active features. Specifically, SF<sub>Lasso-SI</sub> still keeps 151 active variables after dropping many correlated vari-

**Table 3.** Results from human evaluation

	US			Canada		
	Precision	Recall	F1	Precision	Recall	F1
OLS	0	0	0	0	0	0
Lasso-SI	50.00	0.19	0.37	100.00	0.20	0.39
Elastic-SI	0.00	0.00	0.00	0.00	0.00	0.00
<b>SFLasso-SI</b>	26.47	3.36	5.96	27.59	3.16	5.66
OLS+VIF	43.75	1.31	2.54	53.33	1.58	3.07
Lasso-SI+VIF	50.00	0.19	0.37	100.00	0.20	0.39
Ela-SI+VIF	0.00	0.00	0.00	0.00	0.00	0.00
<b>SFLasso-SI+VIF</b>	69.23	3.36	6.41	62.96	3.35	6.37

ables. As a result, its max VIF is about 30, which substantially reduces OLS (106,505), while it keeps much more active variables than Lasso-SI (151 vs. 4).

**Table 4.** Significant features - World Vision US

OLS (0/0)	
Lasso-SI (1/2)	<visual objects>mammal <spoken words> <b>water</b>
Elastic-SI (0/1)	<visual objects>mammal
<b>SFLasso-SI (18/68)</b>	<visual objects>indoor man sky <b>tree grass boy</b> woman girl people <b>smiling</b> field road mountain beach plant athleticGame bed house orange <b>eating</b> red room snow suit militaryUniform old box chicken tattoo <speech tones>joy <spoken words> <b>action</b> address <b>area believe</b> bone call <b>continue</b> doctor Donna famine <b>gift happen</b> heart information involve <b>kid leader</b> leave letter life local <b>make</b> people poverty problem program reach really <b>right</b> sense sponsor <b>stand stop</b> teacher <b>vital</b> water wonderful worker
OLS+VIF (7/16)	<visual objects> <b>tree animal little</b> <spoken words> <b>hope leave end</b> challenge man teacher sit cool plan follow ready malaria <b>development</b>
Lasso-SI+VIF (1/2)	<visual objects > mammal <spoken words> <b>water</b>
Elastic-SI+VIF (0/1)	<visual objects>mammal
<b>SFLasso-SI+VIF (18/26)</b>	<visual objects> <b>ground tree rollingcredits boy little</b> window <b>smiling</b> playing <b>eating</b> <spoken words> <b>action day god</b> <b>happen</b> information <b>kid leader</b> main <b>make</b> plan problem <b>right</b> Sabina <b>stand stop vital</b> wonderful

More importantly, the max VIF of SFLasso-SI is still relatively high compared to the suggested threshold (30 > 10). After several VIF iterations until its threshold, our proposed SFLasso-SI+VIF obtains only 26 SFs among 144 active variables. Surprisingly, more than half of the initial SFs are gone. This is because SFLasso-SI favours the inflated magnitudes of the coefficient to maximize the

number of SFs. This VIF iteration substantially improves the identified SFs' quality by cutting off the false SFs with such inflated coefficients.

Our SFLasso-SI+VIF identifies much more SFs than OLS+VIF (26 vs. 16), although SFLasso-SI+VIF uses a smaller number of active variables than OLS+VIF (144 vs. 237). Note that the max VIFs in both are lower than 10. This surprising result can be explained by the difference in the initial anchoring set of active variables. While OLS+VIF starts with all the 368 variables, SFLasso-SI+VIF does with 144 active variables from SFLasso-SI. The correlation structure among variables is complex. VIF measures the total redundancy of a focal x variable with the rest of the active variables. Depending on the set of active variables, VIF iteration can journey very different paths and result in very different final sets of active variables and SFs. This result emphasizes (1) the important role of the first anchoring stage to identify many candidates of SFs, and (2) the second adjustment stage to drop false SFs via VIF iteration. Table 4 shows the SFs from each model. Although SFLasso-SI+VIF identifies smaller SFs than SFLasso-SI (26 vs. 68), both models include the same number (18) of true features. In other words, SFLasso-SI+VIF has a higher quality (i.e., precision) of SFs (69 vs. 26).

**Table 5.** Deleted false SFs and added true SFs by VIF iteration - World Vision US

False significant features (48)	True significant features (5)
⟨visual objects⟩indoor man sky woman girl people field road mountain beach plant athleticGame bed house orange red room snow suit militaryUniform old box chicken tattoo ⟨speech tones⟩joy ⟨spoken words⟩address bone call doctor Donna famine heart information involve leave letter life local people poverty program reach really sense sponsor teacher water worker	⟨visual objects⟩ground rollingcredits little ⟨spoken words⟩day god

The next question is how SFLasso-SI+VIF can achieve higher accuracy in hitting truly appealing features to ad viewers than SFLasso-SI, although SFLasso-SI+VIF identifies a smaller number of SFs than SFLasso-SI. As discussed above, SFLasso-SI tends to have false SFs with inflated coefficients. Through the VIF iteration, 48 false SFs are excluded, as seen in Table 5. The visual objects ‘indoor’, ‘man’, and ‘sky’ are examples. As a result, precision increases. Furthermore, recall that high VIF can increase the coefficient variance, leading to the insignificance of a focal feature. In other words, high VIF can hide true features. Hence, the VIF process help reveal true features. The visual objects ‘ground’, ‘little’, and ‘rolling credits’ and the spoken words ‘day’ and ‘god’ are those features. These additions improve recall as well as precision. In summary, the second adjustment stage via the VIF iterations increases identified SFs by excluding false SFs and adding new true features. The results on World Vision Canada are similar to those on World Vision US.

## 4 Conclusion

In this paper, we propose a two-stage anchoring-and-adjustment approach. In the first stage, we adopt the recently developed SFLasso-SI to find many candidates of SFs. After then, through the VIF iterations, we adjust SFs by dropping false SFs and adding true SFs. Human evaluations show that our proposed two-stage approach achieves higher accuracy in identifying true features than SFLasso-SI without the VIF iterations. Furthermore, our approach maintains similar prediction accuracy as what Lasso and Elastic-net can obtain.

**Acknowledgements.** This research is supported by the Australian Government Research Training Program Scholarship.

## References

1. Belloni, A., Chernozhukov, V.: Least squares after model selection in high-dimensional sparse models. In: Bernoulli, vol. 19, pp. 521–547 (2013)
2. Chen, J., Song, L., Wainwright, M.J., Jordan, M.I.: Learning to explain: an information-theoretic perspective on model interpretation. In: ICML, vol. 80, pp. 882–891 (2018)
3. Hara, S., Maehara, T.: Enumerate lasso solutions for feature selection. In: AAAI, pp. 1985–1991 (2017)
4. Harder, F., Bauer, M., Park, M.: Interpretable and differentially private predictions. In: AAAI, pp. 4083–4090 (2020)
5. Hastie, T., Tibshirani, R., Wainwright, M.: Statistical Learning with Sparsity: The Lasso and Generalizations (2015)
6. James, G., Witten, D., Hastie, T., Tibshirani, R.: An Introduction to Statistical Learning: With Applications in R. Springer, New York (2014). <https://doi.org/10.1007/978-1-4614-7138-7>
7. Lee, J.D., Sun, D.L., Sun, Y., Taylor, J.E.: Exact post-selection inference, with application to the lasso. *Ann. Stat.* **44**(3), 907–927 (2016)
8. Lockhart, R., Taylor, J., Tibshirani, R.J., Tibshirani, R.: A significance test for the lasso. *Ann. Stat.* **42**(2), 413–468 (2014)
9. Park, E., Wong, R.K., Kwon, J., Chu, V.W.: Maximizing explainability with sf-lasso and selective inference for video and picture ads. In: Advances in Knowledge Discovery and Data Mining, pp. 566–577 (2021)
10. Park, E., Wong, R.K., Kwon, J., Chu, V.W., Rutz, O.J.: Video ads content analysis using significant features lasso. In: The 43rd ISMS Marketing Science Conference (2021)
11. Shrikumar, A., Greenside, P., Kundaje, A.: Learning important features through propagating activation differences. In: ICML, vol. 70, pp. 3145–3153 (2017)
12. Singh, C., Murdoch, W.J., Yu, B.: Hierarchical interpretations for neural network predictions. In: ICLR (2019)
13. Taylor, J., Tibshirani, R.: Post-selection inference for  $l_1$ -penalized likelihood models. *Can. J. Stat.* **46**(1), 41–61 (2018)
14. Tibshirani, R.: Regression shrinkage and selection via the lasso. *J. Roy. Stat. Soc. Ser. B (Methodol.)* **58**(1), 267–288 (1996)

15. Tversky, A., Kahneman, D.: Judgment under uncertainty: heuristics and biases. *Science* **185**(4157), 1124–1131 (1974)
16. Zou, H., Hastie, T.: Regularization and variable selection via the elastic net. *J. Roy. Stat. Soc. B* **67**, 301–320 (2005)





# Data Mining Based Artificial Intelligent Technique for Identifying Abnormalities from Brain Signal Data

Md. Nurul Ahad Tawhid<sup>1</sup>(✉), Siuly Siuly<sup>1</sup>, Kate Wang<sup>2</sup>, and Hua Wang<sup>1</sup>

<sup>1</sup> Institute for Sustainable Industries and Liveable Cities, Victoria University, Melbourne, Australia

md.tawhid1@live.vu.edu.au, {siuly.siuly,hua.wang}@vu.edu.au

<sup>2</sup> School of Health and Biomedical Sciences, RMIT University, Melbourne, Australia  
kate.wang@rmit.edu.au

**Abstract.** Analysis of brain signal data like Electroencephalography (EEG) plays an important role in efficient diagnosis of neurological disorders and treatment. EEG records electrical activity of the brain and contains huge volume of multi-channel time-series data that are visually analyzed by neurologists to identify abnormalities within the brain, which is time-consuming, error-prone, and subject to fatigue. Therefore, an automatic data mining system is always in need to detect abnormality from those large volume of data. To meet the requirements, in this study, a time-frequency spectrogram image-based classification framework is developed using texture feature extractor and machine learning (ML) based classifiers. At first, signals are filtered to remove noises and artifacts and normalized. Then signals are segmented into small chunks and spectrogram images are generated from those segments using short-time Fourier transform. After that, histogram based textural features are extracted and significant features are selected using principal component analysis. Finally, those features are fed into three ML based classifiers for categorizing the signals into different groups. The proposed system is tested on EEG brain signal data and have obtained promising results in identifying different abnormality groups, which indicates that the proposed system can be used for mining large volume of brain signal data.

**Keywords:** Brain signal data · EEG · Time-Frequency spectrogram image · Machine learning · Data mining

## 1 Introduction

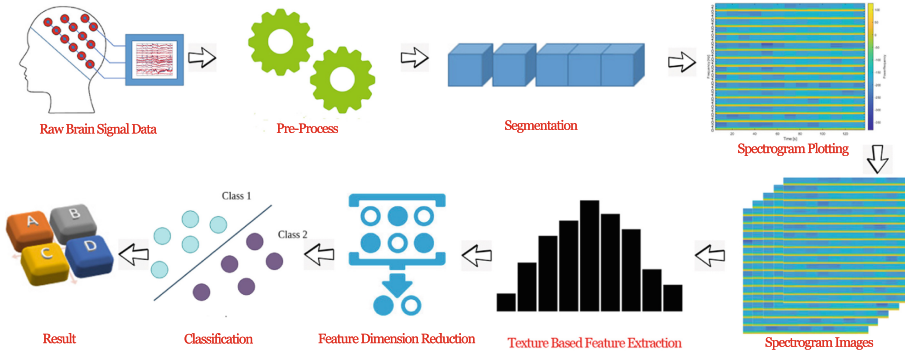
In recent years, comprehensive studies have made on brain signal data, particularly using electroencephalogram (EEG) data due to its significant importance in health and medicine related applications [3]. Efficient and effective analysis of EEG signal is useful for various purposes like neurological diseases diagnosis and treatment [4, 18, 24, 27], brain computer interface [20], sleep stage detection [21], emotion/fatigue detection [6, 16] etc. EEG records the spontaneous electrical activity of the brain, which is large volume of time-series data. These data are

aperiodic, non-stationary and dynamic in nature and contains patterns related to the subject's mental health state [17]. Analysis of those large-scale aperiodic and non-stationary EEG signals is currently a challenging task. Data mining system can extract biomarkers from brain signal data and use those biomarkers to create computer aided diagnostic (CAD) systems that can classify brain states based on abnormalities.

Typically, EEG signal mining process can be divided into two steps: feature extraction and classification of the extracted features using different classifiers. Several techniques for analyzing and classifying large EEG signal data have been developed in recent years [11, 17, 19, 22, 25]. Most of these studies have used different statistical information as features for the signal classification with different classifiers. These traditional methods are often not feasible in extracting significant and discriminative features from large EEG data. Moreover, statistical analysis of larger recording data may overlook short-term changes in signal characteristics, which are important for abnormality detection. Using visual representation of short-term signal segments can solve this issue as it generates visual representation of raw data and works on small segments. Furthermore, most of the studies in this field have only tested their approaches on a single dataset, therefore their application to other datasets is debatable. Nonetheless, most investigations have focused on identifying one neurological disorder from EEG data (2-class problem). Few research have looked at identifying two neurological disorders from healthy control (HC) subjects (3-class) in the same system as, authors in [10, 13] worked in detection of mild cognitive impairment and Alzheimer's disease patients from HC subjects. Authors of [2, 9] have worked in classifying autism spectrum disorder (ASD) and epilepsy (EP) from HCs. But to the best of our knowledge, no research has conducted to detect more than two disorders in a single framework from HC subjects. This is because EEG data volume are huge in nature and the data has overlapping biomarkers for various diseases.

Therefore, to perform classification on these kind of overlapping feature-based data into multi-class requires special data mining techniques. Furthermore, a generic mining framework to conduct classification tasks and identify different types of abnormalities from EEG signals is required. This study aims to fill this gap by developing a data mining framework using short-term visual representation of the brain signal data to classify into multiple abnormality classes based on the biomarker presented in the visual representation of the signal data.

To achieve the aforementioned goal, we have developed a data mining framework for brain signal data, specifically for EEG, to identify four neurological abnormalities namely, ASD, EP, Parkinson's disease (PD), and Schizophrenia (SZ) from HC subjects (5 class) using time-frequency (T-F) spectrogram image and ML based classifiers. The brain signal data is initially filtered to remove noise and artifacts. The signals are then divided into small time frame windows, and spectrogram plotting images are created using a short-time fourier transform (STFT). Completed CENTRIST (cCENTRIST), a histogram-based feature extraction technique is used to extract textural information from those



**Fig. 1.** Overview of the proposed framework.

images. The dimension of the retrieved features is then reduced using principal component analysis (PCA). Finally, three ML based classifiers are used to categorize the reduced extracted features: support vector machine (SVM),  $k$ -nearest neighbor ( $k$ NN) and random forest (RF).

This paper is organized as follows: Sect. 2 presents details workflow of the system. Section 3 states about used datasets and evaluation parameters. The experimental results are given in Sect. 4. Finally, Sect. 5 presents conclusion.

## 2 Workflow

In this study, we have used T-F based spectrogram image to classify brain signal data using cCENTRIST based feature extraction technique with three different ML based classification approaches:  $k$ NN, SVM and RF. The process consists of several steps: pre-processing, segmentation and spectrogram image generation, feature extraction and dimension reduction, and classification. Figure 1 depicts an overview of the proposed method. Details of those steps are given below.

### 2.1 Pre-processing the Brain Signal Data

In this step, we have pre-processed the brain signal data for removing the noise and artifacts introduced by the recording environment and the muscle movement of the subject during the recording time. These filtering processes are done due to some noise and artifacts are very much similar to some disease related signal patterns and may mislead the diagnosis process [25]. To perform the filtering, at first, we used the common average referencing (CAR) technique to remove the common noise and signals from all channels by removing the average signal from all electrodes. After that, artifacts introduced by muscle activity, eye movement and external noise are removed by passing the signal into a low pass infinite impulse response (IIR) filter with a cutoff frequency 40 Hz. Finally, the signals are normalized to a distribution of zero mean and a variance of one to reduce the individual signal differences and to reduce the computational complexity.

## 2.2 Spectrogram Image Generation

Here, the pre-processed signal data are converted into spectrogram images in two steps: at first the brain signal data are segmented into small chunks of three seconds (3 s) to increase the dataset size and as well as extract maximum number of features from the small signal segments. In this segmentation process original signals are segmented into small data chunks and given the level of original data, which makes an increase in the sample size. Then, spectrogram images are generated from those small chunks using STFT based plotting technique. These images provide a visual depiction of the signal data where different color represents the power and amplitude variation of the signal.

## 2.3 Feature Extraction and Dimension Reduction

Features from spectrogram images are extracted using cCENTRIST, an image feature extraction technique developed by Dey *et al.* [7] that combines Completed Local Binary Pattern (CLBP) and CENSus TRanform hISTogram (CENTRIST). It was developed by replacing Linear Binary Pattern (LBP) of CENTRIST [26] with CLBP and performed well on garments texture classification [7] and gender categorization from facial image [23]. cCENTRIST breaks the images into pyramid structure blocks and CLBP-based 3D histograms of those blocks are created and concatenated to produce a special histogram as an image feature. The retrieved feature's dimensions are reduced using PCA, and finally, the reduced feature set is utilized as input to various ML based classifiers.

## 2.4 Classification of the Extracted Features

To classify the cCENTRIST based histogram data of the spectrogram images, we have used three different ML based classifiers: RF,  $k$ NN and SVM. In  $k$ NN based classification, we have tested for 10 different  $k$  values (1 to 10) and for SVM, we used the same LibSVM [5] as the authors of cCENTRIST [7] used. Finally, these classifiers perform a multi-class classification for different neurological disorders and their performance are evaluated using different evaluation techniques.

# 3 Performance Evaluation Materials and Parameters

To validate the proposed model, we have used EEG brain signal data from four different neurological disorders: ASD, EP, PD and SZ. We have used these four datasets to perform a five-class classification using the proposed method (ASD vs EP vs PD vs SZ vs HC). Performance of the proposed method is evaluated using different evaluation matrices that are popular in this field of study.

## 3.1 Datasets

We have used four publicly available datasets of four different neurological abnormalities (ASD, EP, PD, SZ) to validate the proposed brain signal mining system. A brief information of those datasets are given in Table 1. Detail description of those datasets can be found in [1, 4, 12, 14].

**Table 1.** Brief information of the used datasets.

Datasets	No of patients	No of HCs	Recording frequency	No of channels
ASD [1]	12	4	256	16
EP [14]	7	7	256	20
PD [4]	14	14	500	64
SZ [12]	14	14	250	19

### 3.2 Classification Performance Measure

This model is validated using a 5-fold cross-validation technique in order to reduce its classification bias and predict its overall accuracy across the entire dataset. This process involves dividing the dataset into five parts; four of which are used to train the classifier and the remainder one to test the learned system. This step is done five times so that each image in the dataset belongs to the test set exactly once. Finally, the 5-fold results are used to evaluate the system's performance using five parameters: sensitivity (Sen), specificity (Spec), precision (Prec), F1 score (F1), and accuracy (Acc). These criteria allow to predict the behavior of the classifiers on the test data [8, 15, 28].

## 4 Experimental Results

In this study, we have developed a brain signal data mining framework using spectrogram images of the signal data and ML based approaches. The proposed framework has tested on four neurological disease related EEG datasets and performed a five-class classification task. This section describes and visualizes the obtained results in detail with experimental setups.

### 4.1 Experimental Setup

Since EEG recordings contain varying sample rates and channels, the datasets must be standardized to make the data comparable. For this, we kept the ASD dataset (has 16 channel) as base and converted PD, EP, SZ datasets into that format by keeping data from standard 16 channels (Fp1, Fp2, F3, F4, F7, F8, C3, C4, T3, T4, P3, P4, T5, T6, O1, O2) and discarding other channel data and finally, resampled those 256 Hz. After that, EEG signals are pre-processed, segmented and spectrogram images are generated using STFT. This resulted in a total of 19417 images from four datasets, with ASD, EP, PD and SZ contributing 5437 (3825 ASD, 1612 HC), 2483 (1248 EP, 1235 HC), 1745 (864 PD, 881 HC) and 9752 (5312 SZ, 4440 HC) images, respectively. We combined all HC images to create a class of 8168 HC subjects, resulting in a 5-class classification problem.

### 4.2 Results

In this brain signal mining framework, we have used histogram-based cCENTRIST method to extract textural features from spectrogram images. The

**Table 2.** Five round average Sen, Spec, Prec, F1 and Acc for SVM,  $k$ NN and RF.

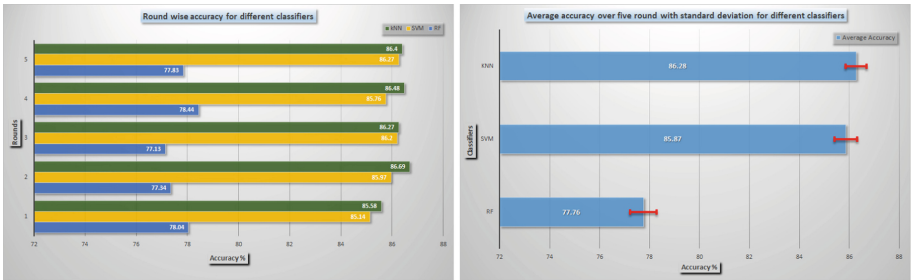
Disease	SVM				$k$ NN				RF			
	Sen	Spec	Prec	F1	Sen	Spec	Prec	F1	Sen	Spec	Prec	F1
ASD	90.72	96.80	87.45	0.89	87.59	97.59	90.03	0.89	75.77	97.40	87.69	0.81
EP	83.59	98.51	79.46	0.81	77.14	98.06	73.29	0.75	31.87	99.97	98.56	0.48
Normal	84.77	87.83	83.49	0.84	88.89	86.56	82.77	0.86	92.29	70.17	69.20	0.79
PD	83.99	99.61	91.01	0.87	69.11	99.95	98.37	0.81	24.26	100.00	100.00	0.39
SZ	84.91	96.22	89.43	0.87	86.27	97.06	91.72	0.89	76.35	96.09	88.02	0.82
<b>Avg</b>	<b>85.59</b>	<b>95.79</b>	<b>86.17</b>	<b>0.86</b>	<b>81.80</b>	<b>95.85</b>	<b>87.23</b>	<b>0.84</b>	<b>60.11</b>	<b>92.72</b>	<b>88.69</b>	<b>0.66</b>
<b>Acc</b>	<b>85.87 ± 0.45</b>				<b>86.28 ± 0.42</b>				<b>77.76 ± 0.53</b>			

extracted features are then reduced in dimension using PCA, and classified using three ML-based classifiers: SVM, RF, and  $k$ NN ( $k = 1$  to  $10$ ). Table 2 shows the 5-round average results of three classifiers for 5-fold cross validation technique. For  $k$ NN, we have just given the results of  $k=9$  as it was the best of the ten different  $k$  values we tested.

Table 2 shows that  $k$ NN has the highest overall accuracy of 86.28%, whereas RF has the lowest overall accuracy of 77.76%. The accuracy of the SVM classifier is similar to that of the  $k$ NN. We have compared the accuracy of the three classifiers in Fig. 2, where Fig. 2a illustrates the round-wise accuracy and Fig. 2b depicts the average accuracy with standard deviation (SD).

For further evaluation of the proposed model, we have calculated and plotted the sensitivity, specificity, precision and F1 score for the classifiers, as shown in Fig. 3a, 3b, 3c and 3d.

Figure 3a shows that, although  $k$ NN has the best classification performance, SVM clearly outperforms all other classifiers in terms of sensitivity, with a much higher round wise sensitivity value. SVM has the highest single round sensitivity value of 86.17% and an overall 5-fold average value of 85.59%(±0.55). RF



(a) Round wise accuracy (b) Average accuracy with SD

**Fig. 2.** Accuracy comparison for different classifiers.

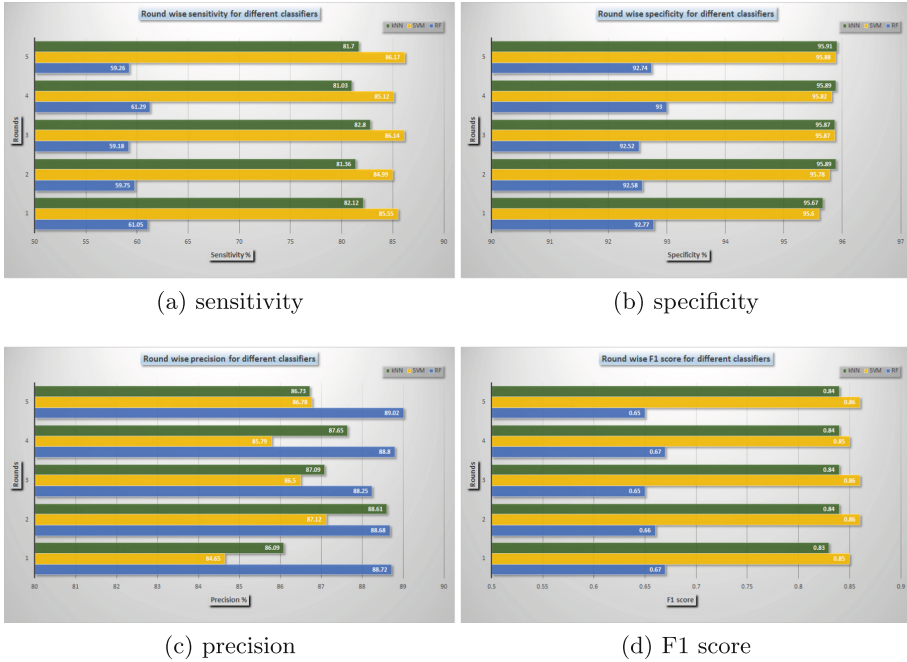


Fig. 3. Round wise Sen, Spec, Prec and F1 score comparison for different classifiers

has the lowest single round sensitivity of 59.18% with 5-fold average value of 60.11% ( $\pm 1.00$ ). For  $k$ NN, those values are 82.80% and 81.80% ( $\pm 0.69$ ), respectively. This result indicates that the SVM classifier is highly sensitive in detecting diseases than other classifiers.

Figure 3b shows the round-wise specificity of the used ML-based classifiers, showing that SVM and  $k$ NN have similar specificity values across rounds. The highest specificity values produced by  $k$ NN are 95.91% for a single round and 95.85% ( $\pm 0.01$ ) for a 5-round average. For SVM and RF, those values are 95.88%, 95.79 ( $\pm 0.11$ ), and 93.00%, 92.72% ( $\pm 0.19$ ), respectively. Higher specificity value indicates the model’s ability to differentiate the healthy subjects from patients.

Round wise precision values of the three classifiers are plotted in Fig. 3c. Despite its poor overall performance, the RF classifier has a high precision for all rounds compared to other classifiers. This is because, despite its low sensitivity, the photos it identifies as the patient’s image are true in general compared to other classifiers. Overall five round average precision for RF,  $k$ NN, SVM are 88.69% ( $\pm 0.28$ ), 87.23% ( $\pm 0.96$ ) and 86.17% ( $\pm 0.98$ ), respectively.

F1 score is the harmonic mean of precision and recall and an useful metric for evaluating classifier performance. Figure 3d depicts the round wise F1 score for the used classifiers, where SVM classifier outperforms other classifiers in all round values. Overall SVM has an average F1 score of 0.86 ( $\pm 0.005$ ) while for  $k$ NN, it is 0.84 ( $\pm 0.005$ ), and RF has the lowest average of 0.66 ( $\pm 0.01$ ).

## 5 Conclusion

In this study, a T-F spectrogram image with ML based data mining technique for brain signal data is proposed. To evaluate the proposed method, we have used EEG brain signal data for multi-category neurological diseases classification. The signal data is initially filtered to remove noise and artifacts, and segmented into small chunks. Then T-F based spectrogram images are generated from those segments using STFT. Textural features are extracted using cCENTRIST and PCA is used to reduce the extracted features dimension. Finally,  $k$ NN, SVM and RF classifiers are used for classifying those features into five classes: ASD, EP, PD, SZ, HC. Among the tested classifiers,  $k$ NN achieved the highest accuracy of 86.28%. Deep learning-based models, like convolutional neural networks, can be used in the future to classify the generated T-F based spectrogram images for mining brain signal data and improve classification performance.

## References

1. Alhaddad, M.J., et al.: Diagnosis autism by fisher linear discriminant analysis FLDA via EEG. *Int. J. Bio-Sci. Bio-Technol.* **4**(2), 45–54 (2012)
2. Alturki, F.A., AlSharabi, K., Abdurraqeab, A.M., Aljalal, M.: EEG signal analysis for diagnosing neurological disorders using discrete wavelet transform and intelligent techniques. *Sensors* **20**(9), 2505 (2020)
3. Alvi, A.M., Siuly, S., Wang, H.: Neurological abnormality detection from electroencephalography data: a review. *Artif. Intell. Rev.*, 1–38 (2021). <https://doi.org/10.1007/s10462-021-10062-8>
4. Anjum, M.F., Dasgupta, S., Mudumbai, R., Singh, A., Cavanagh, J.F., Narayanan, N.S.: Linear predictive coding distinguishes spectral EEG features of Parkinson's disease. *Parkinsonism Relat. Disord.* **79**, 79–85 (2020)
5. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. *ACM Trans. Intell. Syst. Technol. (TIST)* **2**(3), 1–27 (2011)
6. Demir, F., Sobahi, N., Siuly, S., Sengur, A.: Exploring deep learning features for automatic classification of human emotion using EEG rhythms. *IEEE Sens. J.* **21**(13), 14923–14930 (2021)
7. Dey, E.K., Tawhid, M., Ahad, N., Shoyaib, M.: An automated system for garment texture design class identification. *Computers* **4**(3), 265–282 (2015)
8. He, J., Rong, J., Sun, L., Wang, H., Zhang, Y., Ma, J.: A framework for cardiac arrhythmia detection from IoT-based ECGs. *World Wide Web* **23**(5), 2835–2850 (2020)
9. Ibrahim, S., Djemal, R., Alsuwailam, A.: Electroencephalography (EEG) signal processing for epilepsy and autism spectrum disorder diagnosis. *Biocybernetics Biomed. Eng.* **38**(1), 16–26 (2018)
10. Ieracitano, C., Mammone, N., Hussain, A., Morabito, F.C.: A novel multi-modal machine learning based approach for automatic classification of EEG recordings in dementia. *Neural Netw.* **123**, 176–190 (2020)
11. Oh, S.L., et al.: A deep learning approach for Parkinson's disease diagnosis from EEG signals. *Neural Comput. Appl.* **32**(15), 10927–10933 (2020)
12. Olejarczyk, E., Jernajczyk, W.: Graph-based analysis of brain connectivity in schizophrenia. *PLoS One* **12**(11), e0188629 (2017)



13. Oltu, B., Akşahin, M.F., Kibaroglu, S.: A novel electroencephalography based approach for Alzheimer's disease and mild cognitive impairment detection. *Biomed. Sig. Process. Control* **63**, 102223 (2021)
14. Pereira, A., Fiel, J.: Resting-state interictal EEG recordings of refractory epilepsy patients (2019). <https://doi.org/10.17632/6HX2SMC7NW.1>
15. Sarki, R., Ahmed, K., Wang, H., Zhang, Y.: Automated detection of mild and multi-class diabetic eye diseases using deep learning. *Health Inf. Sci. Syst.* **8**(1), 1–9 (2020). <https://doi.org/10.1007/s13755-020-00125-5>
16. Şengür, D., Siuly, S.: Efficient approach for EEG-based emotion recognition. *Electron. Lett.* **56**(25), 1361–1364 (2020)
17. Siuly, S., Alcin, O.F., Bajaj, V., Sengur, A., Zhang, Y.: Exploring Hermite transformation in brain signal analysis for the detection of epileptic seizure. *IET Sci. Meas. Technol.* **13**(1), 35–41 (2018)
18. Siuly, S., et al.: A new framework for automatic detection of patients with mild cognitive impairment using resting-state EEG signals. *IEEE Trans. Neural Syst. Rehabil. Eng.* **28**(9), 1966–1976 (2020)
19. Siuly, S., Khare, S.K., Bajaj, V., Wang, H., Zhang, Y.: A computerized method for automatic detection of schizophrenia using EEG signals. *IEEE Trans. Neural Syst. Rehabil. Eng.* **28**(11), 2390–2400 (2020)
20. Siuly, S., Li, Y.: Discriminating the brain activities for brain-computer interface applications through the optimal allocation-based approach. *Neural Comput. Appl.* **26**(4), 799–811 (2015)
21. Supriya, S., Siuly, S., Wang, H., Zhang, Y.: EEG sleep stages analysis and classification based on weighed complex network features. *IEEE Trans. Emerg. Top. Comput. Intell.* **5**(2), 236–246 (2018)
22. Supriya, S., Siuly, S., Wang, H., Zhang, Y.: Automated epilepsy detection techniques from electroencephalogram signals: a review study. *Health Inf. Sci. Syst.* **8**(1), 1–15 (2020). <https://doi.org/10.1007/s13755-020-00129-1>
23. Tawhid, M.N.A., Dey, E.K.: A gender recognition system from facial image. *Int. J. Comput. Appl.* **180**(23), 5–14 (2018)
24. Tawhid, M.N.A., Siuly, S., Wang, H.: Diagnosis of autism spectrum disorder from EEG using a time-frequency spectrogram image-based approach. *Electron. Lett.* **56**(25), 1372–1375 (2020)
25. Tawhid, M.N.A., Siuly, S., Wang, H., Whittaker, F., Wang, K., Zhang, Y.: A spectrogram image based intelligent technique for automatic detection of autism spectrum disorder from EEG. *Plos One* **16**(6), e0253094 (2021)
26. Wu, J., Rehg, J.M.: Centrist: a visual descriptor for scene categorization. *IEEE Trans. Pattern Anal. Mach. Intell.* **33**(8), 1489–1501 (2010)
27. Yin, J., Cao, J., Siuly, S., Wang, H.: An integrated mci detection framework based on spectral-temporal analysis. *Int. J. Autom. Comput.* **16**(6), 786–799 (2019)
28. Zhang, F., Wang, Y., Liu, S., Wang, H.: Decision-based evasion attacks on tree ensemble classifiers. *World Wide Web* **23**(5), 2957–2977 (2020). <https://doi.org/10.1007/s11280-020-00813-y>



# Where Should I Go? A Deep Learning Approach to Personalize Type-Based Facet Ranking for POI Suggestion

Esraa Ali<sup>1</sup>, Annalina Caputo<sup>2</sup>, Séamus Lawless<sup>1</sup>,  
and Owen Conlan<sup>1</sup>

<sup>1</sup> ADAPT Centre, School of Computer Science and Statistics, Trinity College Dublin,  
Dublin, Ireland

{esraa.ali,seamus.lawless,owen.conlan}@adaptcentre.ie

<sup>2</sup> ADAPT Centre, School of Computing, Dublin City University, Dublin, Ireland  
annalina.caputo@adaptcentre.ie

**Abstract.** In a faceted search system, type-based facets (t-facets) represent the categories of the resources being searched. Ranking algorithms are needed to select and promote the most relevant t-facets. However, as these are extracted from large multi-level taxonomies, they are impossible to show entirely to the user. Facet ranking is usually employed to filter out irrelevant facets for the users. Existing facet ranking methods neglect both the hierarchical structure of t-facets and the user historical preferences. This research introduces a personalized t-facet ranking that addresses both issues. During a first step, a Deep Neural Network (DNN) model is trained to assign a relevance score to each t-facet based on three groups of relevance features. The score reflects the t-facet relevance to the user, the input query, and its general importance in the dataset. Subsequently, these scores are aggregated and the t-facets are re-organised into a smaller sub-tree to be presented to the user. Our approach aims at minimizing the effort required by the user to reach their intended search target. This is measured in terms of number of clicks the user has to perform on the t-facet tree to reach a relevant resource. The approach is applied to a Point-Of-Interest suggestion task. We solve the problem by ranking the categories of the venues as t-facets. The evaluation compares our DNN-based approach with other existing baselines and investigates the individual contribution of each group of features. Our experiment has demonstrated that the proposed personalized deep learning model leads to better t-facet rankings and minimized user effort.

**Keywords:** Facet ranking · Deep Neural Networks · Personalization

## 1 Introduction

The problem of Point-of-Interest (POI) suggestion has gained lot of research attention recently due to the spread of Location Based Social Networks (LBSN). The problem is concerned with recommending interesting places for the users to

visit. POI suggestion algorithms personalize the recommendations according to the current user’s context as well as the recorded user’s history.

In this research, we solve the POI suggestion problem in the context of Faceted Search Systems (FSS), where the categories of the POIs are used as type-based facets (t-facets) that help the user to navigate the information space. Literature showed that the categories of POIs play a key role in solving this problem [2,3]. FSS provide users with a set of t-facets to help them in filtering and narrowing down search results and locating the intended POI quickly. When POIs’ categories are derived from large, multi-level taxonomies, the FSS need to implement methods to identify and prioritise the most relevant t-facet to show to the users to avoid overwhelming them.

In this work, we focus on analysing the role of personalization in t-facet ranking in isolation from other FSS aspects. We aim at answering the following research question: To what extent a Deep Neural Network (DNN) model can learn to rank t-facets in order to minimize user effort to reach the search target?

This study contributes to the research in this area by introducing a novel ranking algorithm for type-based facets relying on a deep neural network. It uses a combination of collaborative filtering (CF), query relevance, and personalization features to rank the facets. The CF features reflect the general importance of the t-facet among users. The query relevance derives the relevance of t-facets to the query from the relevance of the resources to which they belong. Finally, the personalization features exploit the user’s past preferences to build a vector which represents the user interests. The extracted features are fed into a DNN model. This is trained to combine these features into a t-facet ranking score. In a following step, the approach provides a t-facet construction strategy to decide the final tree to be portrayed to the searcher.

## 2 Facet Ranking Related Research

Several approaches have been proposed in literature to solve the problem of personalized facet ranking, which make use of individual user models, collaborative filtering, or a mixture between the two. Factic [11] is a FSS that personalizes by building user models from semantic usage logs. Several layers of user adaptation are implemented and integrated with different weights to enhance the facet relevance model. Koren et al. [9] suggested a CF approach by leveraging explicit user feedback about the facets, which is used to build a facet relevance model for individuals. They also use the aggregated ratings to build a collaborative model for the new users in order to provide initial good facets in absence of a user profile. A personalized ranking based on CF methods was suggested by Chantamunee et al. [4,5]. They used user ratings and Matrix factorization with SVM and Autoencoders to predict facet ranks. The Adaptive Twitter search system generates user models from Twitter to personalize facet-values ordering [1]. The user model contains entities extracted from the user’s tweets. The facet-values are weighted higher if they exist in the user profile. Le et al. [10] also collect user profiles from social networks. The profile is learned from user activities and preferences using a TF-IDF feature vector model. Important facets are highlighted

through matching with the vector model. All the approaches discussed so far use the same strategy to rank all types of facets. We believe it is important to distinguish between the types of facets during the ranking process as they support the user in different ways in finding their intended target. Ali et al. [8] proposed a probabilistic model to personalize t-facet ranking. Topic-based user profiles are collected from users' historical interactions with the system. A recent approach utilized Rocchio formula to build a vector representing the user interests for t-facet ranking [7]. In this model, the user's profile is expressed in a category space through vectors that capture the users' past preferences. The BERT embeddings are used as t-facet representation in vector space. The t-facet score is the cosine similarity between its BERT vector and the user profile vector.

### 3 Proposed Approach

Our approach to t-facet ranking is based on the intuition there are multiple 'signals' surrounding the user's interaction with the system that capture the relevance of a given t-facet. We define then the problem in terms of learning to rank t-facets based on a number of features, each capturing a different relevance aspect. A Deep Neural Network model is trained over these features, to capture the intricacy of the relationship between these features and user's profile.

When the user inputs a search query, the underlying search engine retrieves a set of relevant places. We assume that the set of retrieved results are relevant, and accordingly also the set of t-facets linked to them. This set of t-facets is the input to our ranking approach. Considering the t-facet hierarchy, the proposed approach starts by generating features for each leaf t-facet node. Then, a trained DNN model predicts a score which reflects the relevance of t-facets from user, query and collaborative perspective. Using the generated t-facet score, the second and final step constructs the final t-facet tree to be displayed to the user.

In the first step, this method collects a set of features for each user and query aiming to capture the deemed relevance of a given t-facet. Three groups of features are computed for each t-facet, query and user tuple. **Group-CF** contains collaborative filtering features, **Group-P** contains personalization features and **Group-Q** includes features reflecting the query relevance. These three groups are then fed into a DNN trained to predict a t-facet ranking score. The higher the score, the higher the relevance of the t-facet to the user and the query.

*Personalization Features (Group-P)* This group of features incorporates individual user preferences into the ranking process. Previous user ratings are used to build a preference profile for each user. When the user rates a visited POI positively, it is assumed that the user also likes its categories. The same also applies for POIs rated as neutral or negative. Based on this assumption, we compute the following features:

- ***uf\_positive\_prob***: The probability that user rates the t-facet as positive in general. It is the number of t-facets positively rated by the user, divided by the number of all t-facets rated by the user.

- ***uf\_positivity\_rate***: The probability that the user rates this t-facet as positive when she/he sees it. It is different from the previous feature as it considers only the ratings given by the user to this specific t-facet. To obtain it, the number of times the user rated this t-facet as positive is divided by the total number of times the user rated this specific t-facet. This features along with the previous one reflect whether the user has a strong attitude towards the t-facet on its own, or whether it depends on the individual document content.
- Similarly, features for the neutral t-facets (***uf\_neutral\_prob*** and ***uf\_neutrality\_rate***), and the negative t-facets (***uf\_negative\_prob*** and ***uf\_negativity\_rate***) are obtained.

In addition to these, other features characterizing the user are also added:

- ***user\_gender***, ***user\_age\_group***. User age is mapped into fixed intervals (for example 10 to 15 years). Both gender and age reflect the user demographic interest in a specific t-facet.
- Additional features like ***user\_avg\_ratings***, ***user\_reviews\_count***, ***user\_likes\_count*** can also be added if they are available in the dataset.

The features are calculated for each individual user and t-facet pair. They can be pre-calculated offline for all the t-facet taxonomy and stored in the user profile in advance. This profile will be updated as user rates new POIs.

*Collaborative Filtering Features (Group-CF)*. This group of features reflect the general searchers attitude towards the t-facets. Similarly to the user-based features, we compute the following list of collaborative-based features:

- ***cf\_positive\_prob***: The probability that users rate the t-facet as positive in general. It is the number of t-facets rated positively by all the users, divided by the number of all t-facets in the system.
- ***cf\_positivity\_rate***: The probability that the users rate this t-facet as positive when they see it. It is calculated as the number of times users rated this t-facet as positive divided by how many times the users rate this specific t-facet.
- In the same way, four features for neutral t-facets (***cf\_neutral\_prob*** and ***cf\_neutrality\_rate***), and negative t-facets (***cf\_negative\_prob*** and ***cf\_negativity\_rate***) are also obtained.

The features are calculated for each t-facet in the system, and can be pre-computed offline and stored in a global profile and updated periodically. Additional collaborative filtering features are extracted for each relevant POI and aggregated by averaging their values on the t-facet level, as listed below:

- ***avg\_rating***: Average ratings for POIs associated to this t-facet.
- ***avg\_rating\_count***: Average number of ratings the POI received.
- ***avg\_reviews\_count***: Average number of reviews the POI received.
- ***avg\_reviews\_sent***: Average reviews polarity. In order to calculate this feature, sentiment analysis is performed for the most recent reviews of the POI. Then, the sentiments are averaged into one overall polarity score.

- **price\_group**: This feature rates the price group of this POI belong. Some users prefer cheaper places or more expensive ones which should be considered during the ranking process.
- **avg\_cat\_count**: The category count (i.e. the different number of category types associated with this POI).
- **avg\_cat\_depth**: It reflects the depth of the category in the hierarchy type tree. It considers the count of all categories regardless their level. This includes the parent categories as well. If the document belongs to more than one category, the sum of the depth of all categories is taken. Common ancestors are added only once.

*Query Relevance Features (Group-Q)*. This group of features reflect the relevance of the t-facet to the input query and to the set of relevant results returned by the search engine in response to it. The features are:

- **max\_sim( $f_i, q$ )**: Maximum cosine similarity between t-facet name and each keyword in the query. This feature seeks to capture the direct mention of the category in the query.
- **avg\_sim( $f_i, q$ )**: Average semantic similarity between the t-facet and each keyword in the query. If multiple keywords are similar to the t-facet this will results in higher average and ultimately higher t-facet importance. The semantic similarities are computed using the cosine between the BERT vectors representing the two input texts.

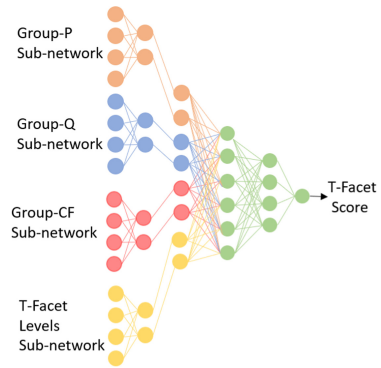
Another set of feature are related to the t-facet information gain. It measures how much information is gained when the user selects this t-facet. This feature set employs the POI-level search engine score in several ways. It assumes that the relevance travels from the POIs to their t-facets:

- **info\_gain**: the average score of the POIs associated with this t-facet.
- **mutual\_info\_gain**: similar to the previous one, but calculated only on POIs which are not covered by any earlier t-facets. A greedy approach is followed to calculate this feature. This feature highlights the importance of POIs which belong only to this specific t-facet.
- **info\_gain@1**: It is the score of the top ranked POI seen by the user at first position, if they filter the results using this t-facet. This feature highlights t-facets associated with a POI with a high relevance score.
- **info\_gain@k**: Information gain at  $k$  is the average for the  $k$  top POIs seen by the user at the top of the result page, after they filter the results using this t-facet.  $k$  is the size of the search result page. The feature favors t-facets with many highly ranked POIs in the first page.
- **popularity( $f_i, F_q$ )** is the t-facet popularity among the t-facets associated with the query results, computed as the number of relevant POIs that belong to this facet type divided by the total number of POIs in the result set.
- **mutual\_popularity( $f_i, F_q$ )** is the t-facet mutual popularity among the t-facets associated with query results, computed as the number of unseen relevant POIs belonging to this t-facet divided by the total number of POIs in the result set.

Finally, a vector is computed with features from all the three groups. It includes also the t-facet and its ancestors names added as categorical features. Features are extracted only for t-facets associated with relevant POIs returned by the search engine after the user submits the query.

*DNN Model Architecture.* In order to build the Deep Neural Network, each group of features is fed into a separate sub-network first, whose role is to reduce the features to a fixed number of nodes  $n$  per network. Regardless the different number of features in each group, the group’s sub-network will encode it in exactly  $n = 4$  nodes. The sub-networks are disconnected from each other, while their internal layers are fully connected. The sub-network outputs are then concatenated in a single layer used as input to the final prediction network. During training, the final network will utilize each sub-network output to predict the final t-facet relevance score.

Figure 1 demonstrates the suggested architecture. The groups P, Q and CF features are first passed to the sub-networks. In addition the T-Facet and its ancestors names are encoded using one-hot encoding used as inputs to the yellow sub-network used to generate a dense vector, which acts as t-facet identifier during the training process. This architecture ensures an equal contribution from each feature group to the final scoring network, represented in green. It also avoids groups containing large number of features to dominate the training process. On the other end, the final network (green) determines the appropriate way to combine the group features into a single overall score. After generating the prediction score, a final step is needed to build the t-facets tree to be provided to the user. To build a final t-facet tree with  $v$  levels, we adopted a *Fixed Level (Max)* strategy [8]. The strategy uses a predefined fixed page size for each t-facet level. It starts by grouping t-facets at level- $v$  by their parent. Then, it sorts the parent nodes at level- $(v - 1)$  by the maximum score of their children, generated in step one (DNN Model), and so on up to level-1.



**Fig. 1.** The DNN architecture. (Color figure online)

## 4 Evaluation

*Datasets.* Our approach is evaluated on a customized versions of TREC-CS 2016 dataset and Yelp Open Data dataset, following the personalised t-facet ranking methodology described in [6]. In TREC-CS, the t-facet taxonomy is derived from the Foursquare category hierarchy. The dataset has 27 users, 61 requests and an average of 208 t-facets per request to be ranked. The second dataset is customized from Yelp Open Dataset. It contains 1,456 users (requests) and average of 168 t-facets per request to be ranked. The taxonomy is derived from Yelp category

taxonomy. Only the first two levels of both taxonomies are included. Query is formulated using tags collected from user previous ratings.

*Evaluation Metrics.* We follow the strategy used in Faceted Search task of INEX 2011 Data-Centric Track [12]. We report two metrics suggested by the organizers. The number of actions (#Actions) counts how many clicks the user has to perform on the ranked facets in order to reach the first relevant POIs in the first page of results. This metric is a proxy for the user’s effort, which will help in answering our research question. We also include F-NDCG@9 metric. It reflects how many unique relevant POIs are covered by the first populated t-facet tree. The t-facet page size for both dataset was set to nine.

*DNN Training Setup.* The DNN model is implemented using Keras TensorFlow 2.4 Functional API<sup>1</sup>. The network is trained to learn the target t-facet rank computed as the number of relevant POIs belonging to this t-facet, collected from the POI level relevance judgment. The model uses Mean Square Error (MSE) as a loss function. All hidden layers in the model utilize the rectified linear activation function (ReLU). We used a pre-trained BERT model to compute the semantic similarity between the query and the t-facets. The ranking approach was evaluated using cross-validation on the request level. Learning rate is set to 0.0001, and the number of epochs is set to 200 for TREC-CS and 100 for Yelp.

*Results and Discussion.* Table 1 summarizes the results for the two datasets included in our experiments. At the top, we report the results of our DNN approach using all features (*Group-All*), each group individually (*Group-P*, *Group-CF*, and *Group-Q*), and their combinations (*Group-CF+Q*, *Group-CF+P*, and *Group-Q+P*). The second part of the table reports the results of personalized t-facet ranking baselines.

**Table 1.** Evaluation results using Fixed Levels (Max) tree building strategy.

Scoring method	TREC-CS		Yelp	
	#Actions	F-NDCG@9	#Actions	F-NDCG@9
Group-All	1.368	<b>0.223</b>	<b>2.480</b>	0.058
Group-P	1.333	0.222	2.500	<b>0.059</b>
Group-CF	1.316	0.210	2.530	0.054
Group-Q	1.298	0.219	2.505	0.056
Group-P+CF	1.316	0.221	2.503	<b>0.059</b>
Group-Q+CF	1.333	0.214	2.536	0.054
Group-P+Q	1.579	0.179	2.512	<b>0.059</b>
Prob. Scoring [8]	1.333	0.141	2.982	0.034
VSM Scoring [7]	<b>1.281</b>	0.099	2.668	0.038
MF-SVM [4]	1.474	0.101	4.729	0.007
Most Prob. (Person) [9]	1.684	0.207	2.557	<b>0.059</b>
Most Prob. (Collab) [9]	1.386	0.204	2.506	0.053
TF.IDF [10]	1.316	0.157	3.223	0.035

<sup>1</sup> Python implementation code for the DNN available at <https://bit.ly/3AkCTGF>.



From results, we can see that DNN-based models could effectively minimize the user effort on both datasets. Considering the number of actions metric, Group-All is the most performing system on the Yelp dataset, while Group-Q is the best system for the TREC-CS on which, however, Group-all has the extra benefit of better F-NDCG values. DNN-based models maximizes F-NDCG values across both datasets. They outperform all the baselines in TREC-CS dataset, and produce comparable values with the best two baselines in Yelp datasets. This reflects how the DNN-based models have the ability to generate trees with a collection of relevant t-facets, rather than producing a tree with a single relevant t-facet at its top, as in the case of VSM Scoring method. Indeed, VSM scoring minimized the # of Actions but performs poorly in terms of F-NDCG, which means that the tree provided in the first page contains more irrelevant facets. Considering the individual groups of features, the performance of the groups seems comparable, it is fair to assume that the groups contribute equally to the results of Group-All model. Considering Group-P, in TREC-CS dataset, each user profile has either 30 or 60 ratings. Such a small number of ratings impacts negatively on the performance of this group of features. Conversely, Yelp dataset contains more user preference samples and interaction data. This helps in building richer user profiles to aid the personalization process. This is evident also across all the baselines that make use of user profiles, i.e. Prob. Scoring, VSM Scoring and Most Prob. (Person).

## 5 Conclusions

In this work, we introduced a DNN-based approach for learning to rank personalized type-based facets. The approach extracts features that reflect the t-facet relevance from multiple perspectives. Personalization was achieved in two ways. At feature level, by computing a set of user-based features. At the dataset level, by training the DNN model on target t-facet ranks, which reflect the user interests. The results have shown that our method improves the ranking process, especially with rich users' profiles. In the future, we intend to develop property-based facet ranking methods to generate an integrated facet ranking framework.

**Acknowledgements.** This work was supported by the ADAPT Centre, funded by Science Foundation Ireland Research Centres Programme (Grant 13/RC/2106; 13/RC/2106.P2) and co-funded by the European Regional Development Fund.

## References

1. Abel, F., Celik, I., Houben, G.J., Siehndel, P.: Leveraging the semantics of tweets for adaptive faceted search on twitter. *Seman. Web* (2011)
2. Aliannejadi, M., Mele, I., Crestani, F.: A cross-platform collection for contextual suggestion. In: *SIGIR*. ACM (2017)
3. Bayomi, M., Lawless, S.: *Adapt.tcd*: an ontology-based context aware approach for contextual suggestion. In: *TREC* (2016)

4. Chantamunee, S., Wong, K.W., Fung, C.C.: Collaborative filtering for personalised facet selection. In: IAIT (2018)
5. Chantamunee, S., Wong, K.W., Fung, C.C.: Deep autoencoder on personalized facet selection. In: Neural Information Processing (2019)
6. Ali, E., Caputo, A., Lawless, S., Conlan, O., et al.: Dataset creation framework for personalized type-based facet ranking tasks evaluation. In: Candan, K.S. (ed.) CLEF 2021. LNCS, vol. 12880, pp. 27–39. Springer, Cham (2021). [https://doi.org/10.1007/978-3-030-85251-1\\_3](https://doi.org/10.1007/978-3-030-85251-1_3)
7. Ali, E., Annalina Caputo, S.L., Conlan, O.: Personalizing type-based facet ranking using BERT embeddings. In: SEMANTiCS 2021
8. Ali, E., Annalina Caputo, S.L., Conlan, O.: A probabilistic approach to personalize type-based facet ranking for poi suggestion. In: ICWE 2021
9. Koren, J., Zhang, Y., Liu, X.: Personalized interactive faceted search. In: WWW. ACM (2008)
10. Le, T., Vo, B., Duong, T.H.: Personalized facets for semantic search using linked open data with social networks. In: IBICA (2012)
11. Tvarožek, M., Bieliková, M.: Factic: personalized exploratory search in the semantic web. In: Benatallah, B., Casati, F., Kappel, G., Rossi, G. (eds.) ICWE 2010. LNCS, vol. 6189, pp. 527–530. Springer, Heidelberg (2010). [https://doi.org/10.1007/978-3-642-13911-6\\_44](https://doi.org/10.1007/978-3-642-13911-6_44)
12. Wang, Q., Ramírez, G., Marx, M., Theobald, M., Kamps, J.: Overview of the INEX 2011 data-centric track. In: Geva, S., Kamps, J., Schenkel, R. (eds.) INEX 2011. LNCS, vol. 7424, pp. 118–137. Springer, Heidelberg (2012). [https://doi.org/10.1007/978-3-642-35734-3\\_10](https://doi.org/10.1007/978-3-642-35734-3_10)



# Modeling Without Sharing Privacy: Federated Neural Machine Translation

Jianzong Wang, Zhangcheng Huang, Lingwei Kong<sup>(✉)</sup>, Denghao Li,  
and Jing Xiao

Ping An Technology (Shenzhen) Co., Ltd., Shenzhen, China  
{wangjianzong347,huangzhangcheng624,konglingwei630,lidenghao842,  
xiaojing661}@pingan.com.cn

**Abstract.** Training neural machine translation models requires large amount of diverse training corpora. It poses a challenge for collecting sufficient data. In addition, labeling monolingual corpus demands professional knowledge in certain domain. Building collaboration between different institutes produces other problems such as legality of data exchange and commercial data leakage.

In this paper, we proposed a federated neural machine translation model *FedNMT* to train a robust machine translation system without sharing raw data from participants. By applying *FedNMT*, neural machine translation (NMT) systems can be ameliorated from the corpus held by different contributors without directly exposing them to one another. This approach preserves the user privacy by utilizing the federated learning framework, encryption techniques. In the federated learning paradigms, a global model is distributed to user clients, and a central server is built to aggregate the learning parameters and update the gradients. Experimental results show the effectiveness of our model in comparison with the data-centralized model.

**Keywords:** Machine translation · Federated learning · Privacy preserving

## 1 Introduction

The achievements of Neural Machine Translation [2, 7, 16] have drawn the attention of the professionals ever since its appearance. The training resource - a parallel corpus is considered as a key component for modeling faithfulness and fluency. However, researchers are rarely aware of parallel corpus privacy. In certain fields like health care, finance, and engineering, there are a few available data resources to be exploited. Exchanging training corpora obtained by individual institutions is prohibited due to data security policy and concerns.

There are some previous research focused on the privacy of NLP, such as federated learning in language modeling [5] and the privacy in NMT [8]. Unarguably, it is necessary to implement research on neural machine translation with respect to data privacy protection. To alleviate the problem of lack of data

source, previous research [9, 17], proposed to exploit the monolingual corpora to import the model faithfulness and fluency. However, they are based on custom training corpora or general-domain publicly available corpora. Except for a few areas, data storage mechanisms are loosely regulated. In particular, transferring private data between hosts elevates the possibility of data breaches or the communication process to be hacked. Moreover, it also leads to inefficiency. Federated learning framework [4] has been developed to transmit encrypted intermediate model parameters between parties without sharing local data.

To enhance the performance of individual agents and build shared vocabularies without sharing training data, this paper proposes a federated neural machine translation model with exploited and jointly trained corpus held by different institutes under a data privacy-preserving scheme. Since each participants receive an identical copy of a *FedNMT* model, the training on particular model can continue locally using a new set of training corpus. We implement a series of experiments to demonstrate the feasibility of the privacy-preserving scheme. This paper has the following contributions:

- Present the first privacy preserving model in machine translation with high quality NMT model without sharing private data to our knowledge.
- We propose a privacy-preserving vocabulary generation method.
- Demonstrate competitive performance with data-centralised non-encrypted NMT methods.

## 2 Proposed Method

### 2.1 Problem Definition

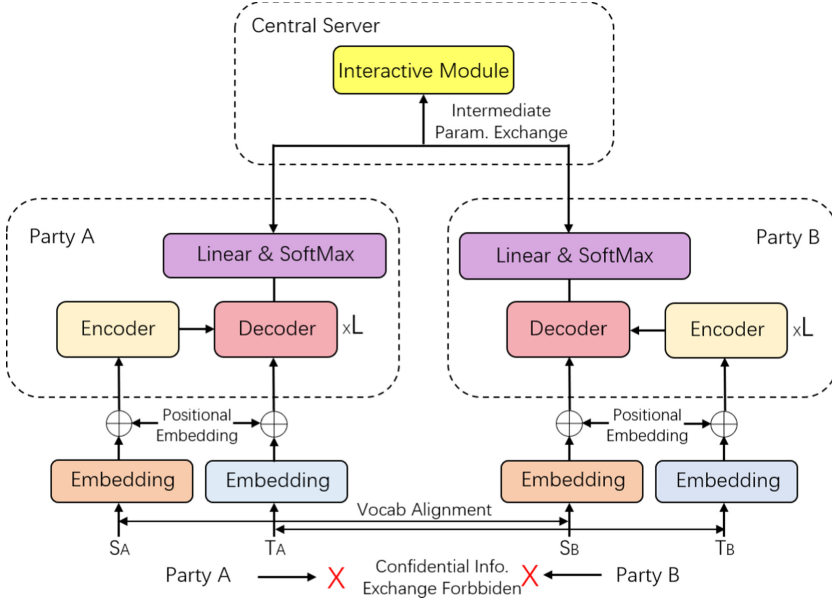
In the setting of federated neural machine translation, the architecture is constructed by a server and  $N$  parties. Each party holds a private corpus  $D_n = \{(x_i, y_i)\}_{i=1}^{K^{D_n}}$ . When collaboratively training a deep neural network (DNN) model, the objective function of global model is formulated as

$$\min_{\Theta_G} \mathcal{L}(\underbrace{\sum_{n=1}^N D_n}_{\text{locally fixed}}; \Theta_G) = \min_{\Theta_n} \sum_{n=1}^N \frac{1}{N} \mathcal{L}(\underbrace{D_n}_{\text{locally fixed}}; \Theta_n) \quad (1)$$

where  $\Theta_G$  is the weights of global model and  $\Theta_n$  is the weights of private model. Global model weights  $\Theta_G$  benefit from all the training corpus  $\sum_{n=1}^N D_n$ . The goal is to fit the model weights  $\Theta_G$  effectively and achieve better model performance without sharing the private data. In this paper, we design a framework to train the NMT model and verify the effectiveness of our method.

### 2.2 Architecture of FedNMT

The proposed *FedNMT* framework is shown in Fig. 1, where  $S_A, T_A, S_B$  and  $T_B$  are the source and target sentences of Party A and Party B respectively. Each of



**Fig. 1.** Illustration of proposed FedNMT framework.

the party has similar layers where two of the embedding layers is for converting original data into the vectors, followed by  $N$  layers of encoders and decoders. Thus, each party uses self-attention mechanism to obtain the high-dimensional vectors that represents the semantic relationship between the source language and target language. The process of word embedding is referenced by a federated vocabulary which will be described in Sect. 2.3. The local model conforms with a transformer architecture, where the depth of encoder-decoder is  $L$ . At each step, source and target languages are embedded by each party, the semantic relationship between words is computed by the self-attention module which integrates into the encoder-decoder module. The server contains an interactive module which requires difference of model weights  $\Delta\theta_n = \theta_n^{(N^{local},t)} - \theta_n^{(N^{local},t-1)}$  and momentum variable  $\beta$  as inputs. We adopt the momentum restart mechanism [10] at each aggregation step. The output of interactive module is defined as

$$Z(\theta_G^t, \beta^t) = Z(\theta_G^{t-1}, \beta^{t-1}) + \Delta Z^{t-1} \tag{2}$$

$$\Delta Z^{t-1} = \frac{1}{\sum N_n^{local}} \sum N_n^{local} \cdot (\Delta\theta_n, \Delta\theta_\beta) \tag{3}$$

where  $N_n^{local}$  is the local training steps of Party  $n$ . After the parties receive updated  $Z(\theta_G^t, \beta^t)$ , they re-initialize the momentum variables in local model and train with a new  $\theta_G^t$ .

### 2.3 Federated Vocabulary

The embedding process in Transformer is a combination of word embedding and positional embedding which requires a joint vocabulary to generate a vector representation of input language vocabulary pairs in the same space domain. This process refers to the vocabulary alignment process [5] in Fig. 1. In the step of building the federated vocabulary, we employed the method of Paillier homomorphic encryption [13] which allows secure computation over encrypted data. Given  $\{[[E_1]], \dots, [[E_i]]\}$  as the cipher-text and  $\{k_1, \dots, k_i\}$  as the scalar constants, homomorphic encryption supports the calculation of  $(k_1 \otimes [[E_1]]) \oplus \dots \oplus (k_i \otimes [[E_i]])$ .

We propose federated vocabulary alignment, which denotes parallel corpus owned by  $P_A$  and  $P_B$ , respectively. Firstly, party A and party B introduce local vocabularies  $V_A$  and  $V_B$  independently. Then, all local vocabularies are encrypted and denoted as  $[[V_A]]$  and  $[[V_B]]$ . The federated vocabulary only provides order of words based on the total frequency of clients, and statistical word frequency is not transmitted from the server.

### 2.4 Secure Model Training

The scheme of Federated NMT model training is illustrated in Algorithm 1. For each training iteration  $N^{local}$ , the parties send the updated model parameters  $\Delta\Theta^{local}$  to server which is responsible for aggregating the local parameters using *FedAVG* [12] and return back the updated model parameters.

The intermediate parameters exchange between central server and parties is confidential. At each round of federated aggregation, server only transmit the gains of parameters denoted as  $\Delta\Theta^{local}$  between every two federated aggregation steps. To protect the privacy of the parties, the process of parameter updating is carried out under differential privacy [1, 6], the transmission of model parameters at each federated step is accompanied with an additional noise  $Lap(\frac{\sigma}{\epsilon})$ , where  $\sigma$  is the sensitivity constant and  $\epsilon$  is the privacy threshold. Thus, server only receives an encrypted value  $\Delta\Theta^{local} + Lap(\frac{\sigma}{\epsilon})$  from the parties.

### 2.5 Domain Expert

In a real-world setting, the parties participating in a federated translation task may come from various domains. These conditions lead to a noise injection from the other parties. Moreover, the differential privacy updating policy may degrade the model performance. To alleviate this condition, we introduce the domain expert method inspired by [3, 15].

Instead of using the global model to translate an input sentence directly, a private model is trained synchronously. The difference between a data-centralised training and our method is that the vocabulary is replaced with the federated vocabulary built in the federated training process in the federated learning scheme. Hence, for each party, the final output is a combination of joint training model and a private model with domain adaption. Let  $M_G$  be the global model trained by the federated learning and  $M_P$  be the private

---

**Algorithm 1.** Method of privacy-preserving model training

---

**Require:** Parallel Corpus  $D_A=\{x_i,y_i\}^N$  and  $D_B=\{x_j,y_j\}^N$  held by separate parties, Learning rate  $\eta$ , Proportion of model parameters to share  $\varpi$ , Noisy threshold  $\epsilon$ .

```

1: procedure: Server initializes a global model  $\Theta^{global}$ 
2: Start the federated training with global step T
3: for  $t \leftarrow 1$  to T do
4:   Server distribute the updated parameters to parties
5:   for  $i \leftarrow 1$  to  $N^{local}$  do
6:     Training with a batch of corpus  $D_{local}$ 
7:     Update  $\Theta^{(i,t)}$ 
8:   end for
9:   Compute difference:  $\Delta\Theta^{(local,t)} = \Theta^{(N^{local},t)} - \Theta^{(N^{local},t-1)}$ 
10:  Add Laplacian Noise  $Lap(\frac{\sigma}{\epsilon})$ 
11:  Privacy preserving transmission:  $\Delta\Theta^{(local,t)} + Lap(\frac{\sigma}{\epsilon})$ 
12:  Privacy preserving federated aggregation
13: end for
14: end procedure

```

---

model optimized in a standard way for a specific domain. The final prediction  $\hat{y} = \lambda(x)M_G(\Theta_G) + (1 - \lambda(x))M_P(\Theta_P)$ , where  $\Theta_G$  and  $\Theta_P$  denote the model parameters of global and private model respectively.  $\lambda(x)$  is a gating function following the Mixture of Experts(MoE) architecture. We set this gating function as  $\lambda(x) = \tanh(\theta \cdot x)$ . The MoE architecture determines the influence ratio between the global model and private model in individual translation cases.

### 3 Experiment

#### 3.1 Experiment Setups

The experiments are implemented on English-Chinese and English-German translation. The training datasets for task En-De are Europarl V9<sup>1</sup> and News Commentary v14<sup>2</sup> and newstests 2015 as the test set. The training dataset for task En-Zh is from CWMT<sup>3</sup>, NEU2017 held by one of the participants and Casis2015 by the other wand newstests 2017 as test set. We randomly drew 2k samples from the training dataset as a validation set. Word segmentation is used for Chinese sentences with an open sourced tool called THULAC<sup>4</sup> [11]. After preprocessing, the resulting training corpus includes 2.07M and 3.5M language pairs for En-De and EN-Zh tasks respectively. In En-De task, Party A contains 1.75M sentence pairs and Party B contains 320K sentence pairs. In Zh-En task,

<sup>1</sup> <http://www.statmt.org/europarl/v9/training/europarl-v9.de-en.tsv.gz>.

<sup>2</sup> <http://data.statmt.org/news-commentary/v14/news-commentary-v14.tsv.gz>.

<sup>3</sup> <http://mteval.cipsc.org.cn:81/agreement/wmt>.

<sup>4</sup> <https://github.com/thunlp/THULAC-Python>.

Party A and Party B consists 1.8M and 1.7M sentence pairs. The language tool for evaluation is uncased 4-gram BLEU [14].

The experiment depth of encoder and decoder is 6. The amount of attention heads is 8 and the hidden embedding size is 512. The filter size for feed forward layer is 2048. A learning rate decay policy is also applied along with 4000 warm-up steps. In decoding, we set beam size to 6 and a length normalization weight of 1.5. The language tool for evaluation is uncased 4-gram BLEU [14]. *FedNMT* is the FL model without weighted average. *FedNMT+WA* denotes the model with weighted average at each federated aggregation step. We also conducted two experiments *FedNMT+DE* and *FedNMT+WA+DE* to augment the FL model with domain expert. In the privacy-preserving setup, we also train the *FedNMT+WA* model with low noise  $\epsilon = 1$  and high noise  $\epsilon = 2$ . The baseline systems are based on Transformer<sup>5</sup>. All models are trained more than 10 epochs to ensure convergence and trained on two NVIDIA Tesla V100 GPUs.

**Table 1.** BLEU scores on English-German and English-Chinese translation.

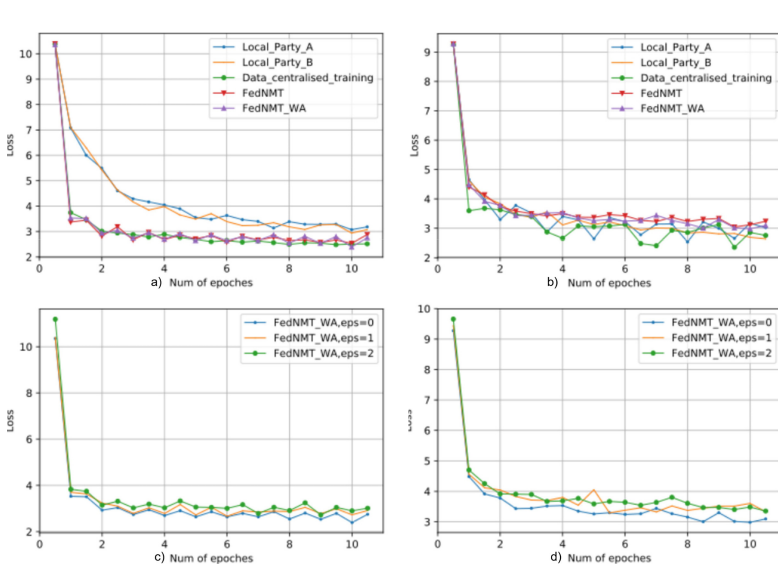
System		En-De		En-Zh	
		Dev	Test	Dev	Test
Baseline system	Transformer <sub>PartyA</sub>	19.44	19.81	14.28	14.13
	Transformer <sub>PartyB</sub>	19.20	18.23	11.57	10.91
	+Data-centralised training	21.07	21.84	16.05	<b>16.53</b>
FedNMT system	FedNMT	20.72	20.91	16.08	16.02
	FedNMT+WA	20.79	21.12	16.52	16.21
FedNMT with Domain Expert	FedNMT+DE <sub>PartyA</sub>	20.82	22.06	16.27	15.92
	FedNMT+DE <sub>PartyB</sub>	20.38	21.35	16.54	16.43
	FedNMT+WA+DE <sub>PartyA</sub>	<b>21.64</b>	<b>22.43</b>	<b>16.85</b>	16.39
	FedNMT+WA+DE <sub>PartyB</sub>	20.99	21.45	16.50	16.56

### 3.2 Performance

The training loss is illustrated in Fig. 2a and Fig. 2b, which plot training loss by number of epochs for different experiment setups. The evaluation results of the experiment are shown in Table 1, the precision loss of FedNMT is small compared with the local baseline model. The comparison of BLEU score with different noise level is demonstrated in Table 2. The BLEU score of *FedNMT+WA* is better than the simple *FedNMT* system, while the model parameters of *FedNMT+WA* is adopted with weighted average scheme. From the result of FL system adopted with domain expert, it shows that the final outputs is partly determined by the private model that the individual party trained. The training loss with different noise constant  $\epsilon$  is shown in Fig. 2c and Fig. 2d. As anticipated, there is a trade-off between model accuracy and privacy protection. Increment of  $\epsilon$  indicates a higher level of user privacy protection, where the value determines the amount of noise added to the transmission.

<sup>5</sup> <https://github.com/Kyubyong/transformer>.





**Fig. 2.** The training loss of a) En-De, b) En-Zh task on test set: FL vs. non-FL and comparison of c) En-De, d) En-Zh training loss on test set with different noise level.

**Table 2.** FedNMT+WA (F.WA) Model performance by varying the noise level.

System	En-De		En-Zh	
	Dev	Test	Dev	Test
F.WA, $\epsilon = 0$	20.79	21.12	16.52	16.21
F.WA, $\epsilon = 1$	20.51	20.93	16.47	16.05
F.WA, $\epsilon = 2$	20.34	20.48	16.55	15.79

For English-German system, the setup of *FedNMT* with weighted average and domain expert outperforms the baseline system, the BLEU score is higher than the data-centralised model by 0.59 BLEU points in En-De test set. For English-Chinese system, the transformer model with data-centralised training results in the best BLEU score. The BLEU of FL systems is slightly smaller than the local training model, and it still has an improvement versus the model with training corpus from Party A or Party B. Experiment results show that our model is competitive with data-centralised model. As expected, each party benefit from federated training. There is a noticeable increasing of BLEU score compared with the model with only the local training corpus. There is a trade-off between user privacy protection and translation system performance.

## 4 Conclusion

In this paper, we presented a neural machine translation model under the framework of federated learning, enumerate and examine the efficiency and accuracy

of this model. The experiment validates the benefit from of *FedNMT* without data leakage. Experiment results show that the precision loss of our model is relatively small compared to the local model which holds all the training corpus and the effectiveness of *FedNMT*. Despite the successes of applying federated learning into deep learning, we expect more future research to be conducted on natural language processing with encryption strategy to secure user privacy.

**Acknowledgement.** This paper is supported by National Key Research and Development Program of China under grant No. 2018YFB0204403.

## References

1. Abadi, M., Chu, A., Goodfellow, I.J., et al.: Deep learning with differential privacy, pp. 308–318. ACM (2016)
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate (2015)
3. Ben-David, S., Blitzer, J., Crammer, K., et al.: A theory of learning from different domains. *Mach. Learn.* **79**(1-2), 151–175 (2010)
4. Bonawitz, K., Eichner, H., Grieskamp, W., et al.: Towards federated learning at scale: system design. *CoRR abs/1902.01046* (2019)
5. Chen, M., Suresh, A.T., Mathews, R., et al.: Federated learning of N-gram language models, pp. 121–130 (2019)
6. Cheng, H.-P., et al.: Towards decentralized deep learning with differential privacy. In: Da Silva, D., Wang, Q., Zhang, L.-J. (eds.) *CLOUD 2019. LNCS*, vol. 11513, pp. 130–145. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-23502-4\\_10](https://doi.org/10.1007/978-3-030-23502-4_10)
7. Gehring, J., Auli, M., Grangier, D., et al.: Convolutional sequence to sequence learning. *Proc. Mach. Learn. Res.* **70**, 1243–1252 (2017)
8. Hisamoto, S., Post, M., Duh, K.: Membership inference attacks on sequence-to-sequence models: is my data in your machine translation system? *Trans. Assoc. Comput. Linguistics* **8**, 49–63 (2020)
9. Lample, G., Ott, M., Conneau, A., et al.: Phrase-based & neural unsupervised machine translation. In: *EMNLP* (2018)
10. Li, W., et al.: Privacy-preserving federated brain tumour segmentation. In: Suk, H.-I., Liu, M., Yan, P., Lian, C. (eds.) *MLMI 2019. LNCS*, vol. 11861, pp. 133–141. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-32692-0\\_16](https://doi.org/10.1007/978-3-030-32692-0_16)
11. Li, Z., Sun, M.: Punctuation as implicit annotations for Chinese word segmentation. *Comput. Linguistics* **35**(4), 505–512 (2009)
12. McMahan, B., Moore, E., Ramage, D., et al.: Communication-efficient learning of deep networks from decentralized data. *Proc. Mach. Learn. Res.* **54**, 1273–1282 (2017)
13. Moriai, S.: Privacy-preserving deep learning via additively homomorphic encryption. In: 26th IEEE Symposium on Computer Arithmetic, *ARITH 2019*, Kyoto, Japan, 10–12 June 2019, p. 198. IEEE (2019)
14. Papineni, K., Roukos, S., Ward, T., Zhu, W.: BLEU: a method for automatic evaluation of machine translation, pp. 311–318. *ACL* (2002)
15. Peterson, D., Kanani, P., Marathe, V.J.: Private federated learning with domain adaptation. *CoRR abs/1912.06733* (2019)
16. Vaswani, A., Shazeer, N., Parmar, N., et al.: Attention is all you need (2017)
17. Wang, S., Liu, Y., Wang, C., et al.: Improving back-translation with uncertainty-based confidence estimation, pp. 791–802 (2019)

# **Knowledge Graph and Entity Linking**



# Encoding the Meaning Triangle (Object, Entity, and Concept) as the Semantic Foundation for Entity Alignment

Kaisheng Zeng<sup>1</sup>(✉), Chengjiang Li<sup>2</sup>, Yan Qi<sup>3</sup>, Xin Lv<sup>1</sup>, Lei Hou<sup>1</sup>,  
Guozheng Peng<sup>4</sup>, Juanzi Li<sup>1</sup>, and Ling Feng<sup>1</sup>

<sup>1</sup> Department of Computer Science and Technology,  
Beijing National Research Center for Information Science and Technology,  
Tsinghua University, Beijing 100084, China  
{zks19,lv-x18}@mails.tsinghua.edu.cn,  
{houlei,lijuanzi,fengling}@tsinghua.edu.cn

<sup>2</sup> Meituan, Beijing 100102, China  
lichengjiang@meituan.com

<sup>3</sup> State Grid Tianjin Electric Power Company, Tianjin 300010, China

<sup>4</sup> China Electric Power Research Institute, Beijing 100192, China  
pengguozheng@epri.sgcc.com.cn

**Abstract.** Entity alignment intends to find equivalence relations between entities that exist in different knowledge graphs but semantically represent the same real-world object. Despite great progress has been made in entity alignment, **concept** - an indispensable piece in the meaning triangle (real-world object, symbolic entity, and concept) is inadequately addressed by the current approaches. In this study, we adopt the three-way distinction (so called meaning triangle [18]) in knowledge representation as the **semantic foundation** for entity alignment, and present a framework called C4EA (**C**oncept for **E**ntity **A**lignment) to integrate and unify concept and entity into entity alignment. C4EA uses a graph neural network and knowledge representation model to encode entity and entity relationship in a low-dimensional vector, and uses Box Embedding to represent concept as a hyper-rectangle in the vector space. We treat entity as a hyper-rectangle with a range of zero (a point in space), so that entities which are challenging to align through finite equivalent entity links can be drawn closer to each other in the representation space. Our experimental results on three public multilingual entity alignment datasets demonstrate the effectiveness of the proposed solution.

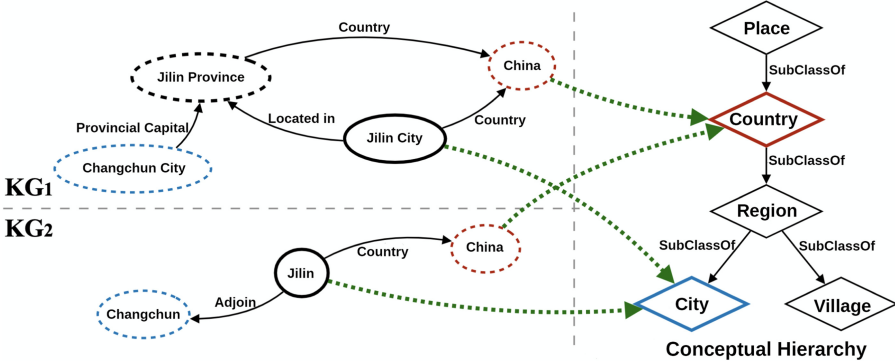
**Keywords:** Concept · Entity · Knowledge graph · Entity alignment

## 1 Introduction

Knowledge Graphs (KGs) have gained in importance in many Artificial Intelligence applications such as information extraction [11] and intelligent question

answering [7]. XLORE [12], DBpedia [14], YAGO [22] are a few widely used KGs in recent years. As KGs are usually extracted from multilingual data sources and built by people with different knowledge backgrounds, fusing different KGs that may coincide or complement each other via entity alignment (EA) has become an important issue, attracting the interests of many scholars [36].

The goal of EA is to find equivalence relations between entities that exist in different KGs, but semantically represent the same real-world object. Such entities form identical entity pairs.



**Fig. 1.** An example of the structure of entities Jilin and its neighbor entities in two heterogeneous KGs (dashed circles of the same color indicate pre-aligned entity pairs; green dotted arrows indicate entity-to-concept affiliation, namely ‘instanceOf’).

Traditional EA methods mostly rely on characteristics of entities, such as entity names, entity description summaries, etc. [5], and calculate their similarity based on these characteristics as the equivalence measurement [24]. As the scale and heterogeneity of KGs continue to increase, both the effectiveness and efficiency of these methods degrade greatly.

With the recent development of representation learning techniques, learning KG representations for EA has sparked vigorous discussions. The main idea based on representation learning is to encode various information of an entity in a low-dimensional vector representation, and then calculate the similarity based on two vector representations to obtain identical entity pairs. We categorize these methods into three groups according to the types of the information they use: (1) only use the structural information (i.e., entities and entity relations in KGs) to align entities; (2) fuse the attribute information into EA; and (3) merge the side information into the alignment process.

Despite great progress has been made in EA, **concept** - an indispensable piece in knowledge representation is inadequately addressed by the current approaches. In human cognition and logic inference, with the reference to a **physical object** through a **printable and computable symbol** (i.e., information objects to denote the physical object), mental processes meanwhile rely

on another iconic suggestion of the **unprintable mental concept** associated with the object [26].

These concepts define abstract ideas or general notions that occur in our mind and thoughts. They are understood to be the fundamental building blocks of thoughts and beliefs, and play a significant important role in all aspects of cognition. [18] popularized the term *meaning triangle* (comprised of *object*, *symbol*, and *concept*), but Aristotle was the first to make the distinction [26].

In this study, we adopt the three-way distinction (so called meaning triangle [18]) from the pioneers and use it as the **semantic foundation** for EA. In KGs, a concept is an abstract description of a class of objects with certain similar characteristics. The ‘*SubClassOf*’ relationship between concepts and concepts constitute a conceptual hierarchy. Entities in KGs symbolize physical, mental, fictional, or hypothetical objects in real-world, and link to concepts via the ‘*instanceOf*’ relationships.

There is no doubt that involving the lost *concept* piece in knowledge representation could lead us to a better understanding of symbolic entities in KGs. Concepts and conceptual hierarchies benefit EA from another perspective different from equivalent entity links. Figure 1 gives an example of using conceptual information to help align two entities. Suppose **Jilin City** in  $KG_1$  and **Jilin** in  $KG_2$  are a pair of equivalent entities that we expect to align. However, due to the heterogeneity of the knowledge graphs, there are some differences in their names and neighboring structures. Also, it is hard to transfer the existing equivalent pair information (**Changchun City**, **Changchun**) to (**Jilin City**, **Jilin**). Unlike entity **Changchun** which only refers to a city, entity **Jilin** can stand for either a city or a province, and a wrong alignment link (**Jilin**, **Jilin Province**) may also be generated. An effective way to resolve the problem is to introduce concept explicitly into EA. Based on the associated concept **city** of entity **Jilin**, we are able to narrow down the distance between **Jilin City** and **Jilin**, and distinguish **Jilin Province** from **Jilin** at the conceptual level.

Despite the advantage of meaning triangle (object, entity, concept) in knowledge representation, integration of conceptual information into EA is not a trivial task, and a number of issues need to be addressed.

1. How to represent entity and entity relationship, as well as concept and conceptual hierarchy for EA?
2. How to uniformly integrate both representations in an EA framework and make it work?
3. In the absence of concepts and conceptual hierarchies in KGs, how to conduct concept-aware EA? Some well-established KGs like DBpedia [14] and YAGO [22] encompass concepts and conceptual hierarchies, but not all KGs provide.

To address the above questions, we propose a framework called C4EA (Concept for Entity Alignment), which integrates and unifies concept and entity into EA. We use a graph neural network to encode the structural information of KGs, and represent entity in a low-dimensional vector in the same vector space. A knowledge representation model is exploited to learn the representation of entity

relationships and encode it into the corresponding entity representation. Meanwhile, we employ Box Embedding [23] to represent concept as a hyper-rectangle in the vector space, and treat entity as a hyper-rectangle with a range of zero (a point in space). Constraints induced by the conceptual hierarchy are finally enforced upon the representation vectors of entities, so that entities which are challenging to align through finite equivalent entity links can be drawn closer to each other in the representation embedding space. In this way, representations of concept and conceptual hierarchy can be naturally merged into the embeddings of entities and entity relationships. The later play an important role in EA.

For those KGs which lack concept and conceptual hierarchy, we use equivalent entity links to communicate the relationship between entities and concepts. C4EA implicitly learns the affiliation between entities and concepts (*instanceOf*), thereby helping entities without the *instanceOf* relationship to improve their alignment efficiency at the conceptual level.

The key contributions of the paper can be summarized as follows:

1. Taking meaning triangle (object, entity, concept) as the semantic foundation for EA, we propose a framework C4EA (Concept for Entity Alignment), which integrates and unifies concept and entity into EA. To our knowledge, this is the first neural EA model to use the information of concept and conceptual hierarchy.
2. We present neural methods to learn the representations of entity and entity relationship, as well as concept and conceptual hierarchy for EA. Our experimental results on three public multilingual EA datasets demonstrate the effectiveness of the proposed solution.

The remainder of the paper is organized as follows. We formulate the problem in Sect. 2, and detail our solution in Sect. 3. We report our performance study in Sect. 4. We review some closely related work in Sect. 5, and conclude the paper in Sect. 6.

## 2 Problem Formulation

We formalize two heterogeneous KGs that need to be associated via EA as  $G_1 = (E_1, R_1, T_1)$  and  $G_2 = (E_2, R_2, T_2)$ , where  $E_i, R_i, T_i$  respectively represent the entity set, relation set, and fact triple set of the knowledge graph  $G_i$  ( $i = 1, 2$ ). Let  $\mathcal{N}_e$  be the set of neighbor entities of entity  $e$ , namely, the set of entities that are directly connected to entity  $e$  in the same knowledge graph  $G \in \{G_1, G_2\}$  via fact triples, where  $\mathcal{N}_e = \{e' \mid (e, r, e') \in T \wedge (e, e' \in E)\} \cup \{e' \mid (e', r, e) \in T \wedge (e, e' \in E)\}$ .

We formally represent the set of concepts to which entities in  $E_1$  and  $E_2$  refer as  $\mathbb{C}$ , and represent the conceptual hierarchy as  $L = L_{instanceOf} \cup L_{subclassOf}$ , where

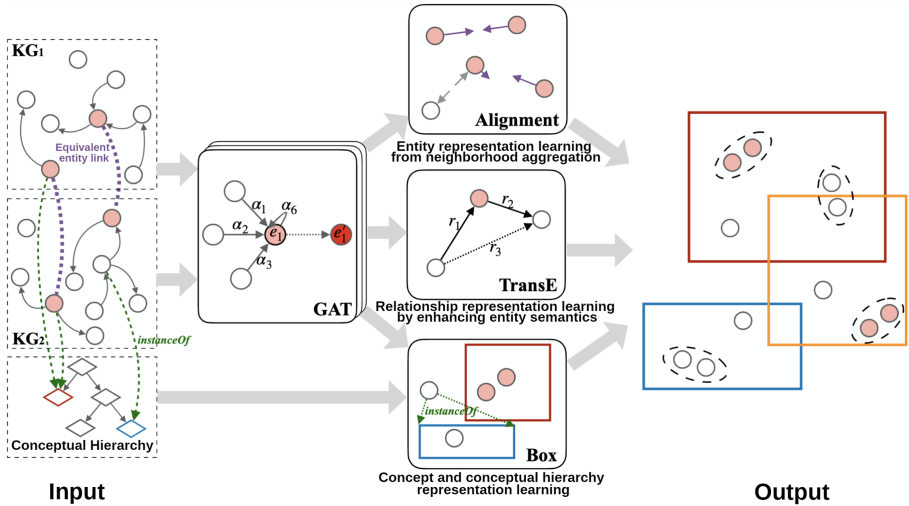
- $L_{instanceOf} = \{(e, c) \mid (e, instanceOf, c) \wedge (e \in E_1 \cup E_2) \wedge (c \in \mathbb{C})\}$  is the set of *instanceOf* relationships between entities and concepts.  $(e, instanceOf, c)$  indicates that entity  $e$  is associated with concept  $c$ .

- $L_{subclassOf} = \{(c_1, c_2) \mid (c_1, subclassOf, c_2) \wedge (c_1, c_2 \in \mathbb{C})\}$  represents the subordinate relationships between concepts,  $(c_1, subclassOf, c_2)$  indicates that concept  $c_1$  is a sub-concept of concept  $c_2$ .

Let  $S = \{(e_1, e_2) \in E_1 \times E_2 \mid e_1 \leftrightarrow e_2\}$  be a finite set of entity pairs with the equivalence relationship. Let  $S_{ea} \subset S$  be a set of pre-aligned seed entity pairs in  $S$ . The task of EA task is to identify the remaining equivalent entity pairs in the set  $(S - S_{ea})$ .

### 3 The C4EA Framework

To integrate and unify concept and entity into EA, we present a novel EA framework C4EA. Figure 2 shows its two major modules, responsible for entity representation learning and concept representation learning, respectively.



**Fig. 2.** The overall framework of C4EA. The purple dotted line in the input connects the pre-aligned equivalent entities in the two KGs, and the green dotted arrow indicates the relationship between the entity and the concept ('instanceOf'); the colored rectangle in the output represents different concepts, and the circled node is the potential equivalent entity pairs we want to discover. (Color figure online)

#### 3.1 Entity Representation Learning

**Learning from Neighborhood Aggregation.** The purpose of entity representation learning from neighborhood aggregation is to encode entities according to the structure of knowledge graph and obtain the entity representation embedding in the same vector space. To aggregate the characteristics of entities and



transfer equivalent information between entities in the knowledge graph structure better, we use a multi-layer Graph Attention Network (GAT) [32] to encode entities information. The vector representation  $\mathbf{h}_i^{(l+1)} \in \mathbb{R}^{n \times d}$  of entity  $e_i$  at the  $(l+1)^{th}$  GAT layer is calculated as below:

$$\mathbf{h}_i^{(l+1)} = \sigma \left( \sum_{e_j \in \mathcal{N}_{e_i} \cup \{e_i\}} \alpha_{ij}^{(l)} \mathbf{W}^{(l)} \mathbf{h}_j^{(l)} \right) \quad (1)$$

In Eq. 1,  $\mathbf{W}^{(l)} \in \mathbb{R}^{d \times d}$  is the weight matrix of the  $l^{th}$  layer in GAT. The initial entity vector (for example,  $\mathbf{h}_i^{(0)}$ ) comes from the entity representation matrix  $\mathbf{X} \in \mathbb{R}^{n \times d}$ ,  $d$  is the entity representation vector's dimension,  $\sigma(\cdot)$  is the nonlinear activation function. In our model, we use  $\text{ReLU}(\cdot) = \max(0, \cdot)$  as  $\sigma(\cdot)$ ,  $\mathcal{N}_{e_i}$  is the set of neighbour nodes for entity  $e_i$ ,  $\alpha_{ij}^{(l)}$  is the attention weight between entities  $e_i$  and  $e_j$  in the  $l^{th}$  neural network. Following the method of Sun et al. [30], we use two different matrices to perform linear transformations on the entities at both ends to calculate the attention coefficient  $\beta_{ij}^{(l)}$ :

$$\beta_{ij}^{(l)} = \text{LeakyReLU} \left[ (\mathbf{M}_1^{(l)} \mathbf{h}_i^{(l)})^\top (\mathbf{M}_2^{(l)} \mathbf{h}_j^{(l)}) \right] \quad (2)$$

In Eq. 2,  $\mathbf{M}_1^{(l)}, \mathbf{M}_2^{(l)} \in \mathbb{R}^{d \times d}$  are the parameter matrices of two different linear transformations,  $(\cdot)^\top$  is the transpose operation of the matrix,  $\text{LeakyReLU}(\cdot)$  is the nonlinear activation function [17]. By normalizing the attention coefficients of entity  $e_i$  and its different neighbor entities, the attention weight  $\alpha_{ij}^{(l)}$  between entities can be obtained as:

$$\alpha_{ij}^{(l)} = \text{softmax}(\beta_{ij}^{(l)}) = \frac{\exp(\beta_{ij}^{(l)})}{\sum_{e_k \in \mathcal{N}_{e_i} \cup \{e_i\}} \exp(\beta_{ik}^{(l)})} \quad (3)$$

After the process of  $L$  layers graph neural networks, we obtain the vector representation  $\mathbf{h}_i^L \in \mathbf{H}^L$  of the entity  $e_i$  fused with neighborhood information. We call  $\mathbf{H}^L$  as the entity representation from neighborhood aggregation; then we get a low-dimensional vector embedding of the entity.

Although the same graph neural networks are used to encode the entities, the two knowledge graphs' entities are even located in different vector spaces. In order to represent them in the same vector space, we use the pre-aligned set of equivalent entity pairs  $S_{ea}$  to reduce the distance between each pair of equivalent entity vector representations to align the entities of the two KGs. We define the distance evaluation function between two entities as  $\text{dist}(e_i, e_j) = \|\mathbf{h}_i - \mathbf{h}_j\|_{1/2}$ , where  $\|\cdot\|_{1/2}$  represents the  $L_1$  or  $L_2$  norm of the vector, we use Margin-based Ranking Loss as the optimization target for EA  $O_E$ :

$$O_E = \sum_{(e_i, e_j) \in S_{ea}} \sum_{(e'_i, e'_j) \in S'} \left[ \text{dist}(e_i, e_j) + \gamma_1 - \text{dist}(e'_i, e'_j) \right]_+ \quad (4)$$

where  $[\cdot]_+ = \max\{0, \cdot\}$  means to take the maximum value between the input vector and zero,  $\gamma_1 > 0$  is the margin hyperparameter, the representation embeddings of entity  $e_i$  and  $e_j$  come from the entity representation matrix  $\mathbf{H}^{(L)}$ , the negative sample set  $S'$  used for training is generated by sampling the nearest neighbors of the entity in set  $S_{ea}$  according to the pre-aligned equivalent entities [13].

### Enhancing Entity Semantics with Entity Relationship Representation.

The purpose of relationship representation learning is to model the relationships between entities, enrich the semantic information encoded by the entities, and improve the model’s discrimination. Following previous work [15], we choose the classic knowledge representation translation model TransE [1] to introduce the representation of relationships. TransE embeds entities and relationships into the same vector space. The relationship is regarded as a vector translation from the head entity to the tail entity. It will calculate a reasonableness score  $f(e_h, r, e_t) = \|\mathbf{h}_{e_h} + \mathbf{r} - \mathbf{h}_{e_t}\|_{L_1/L_2}$  for each relational triple  $(e_h, r, e_t) \in T$ . Similar to the entity representation learning from neighborhood aggregation; we also use Margin-based Ranking Loss as the optimization target for EA  $O_R$ :

$$O_R = \sum_{(e_h, r, e_t) \in T} \sum_{(e'_h, r', e'_t) \in T'} [f(e_h, r, e_t) + \gamma_2 - f(e'_h, r', e'_t)]_+ \quad (5)$$

The representation vectors of entity  $e_h$  and  $e_t$  come from the entity representation matrix  $\mathbf{H}^{(L)}$ , the vector representation of the relationship  $r$  is taken from the relation representation matrix  $\mathbf{R} \in \mathbb{R}^{|R| \times d}$  which needs to be learned,  $\gamma_2 > 0$  is the margin hyperparameter. The negative sample set  $T'$  used for training is generated by sampling the nearest neighbors of the relational triple set  $T$  [13].

## 3.2 Concept Representation Learning

The purpose of concept and conceptual hierarchy representation learning is to learn the representation of the concept so that it can naturally establish a connection between the entities, and help align the entity by constraining the entity representation. Since there is a hierarchical relationship between entities and concepts, and between different concepts (*instanceOf*, *subclassOf*), the selected concept representation model should be capable of hierarchical representation. Inspired by Ren et al. [23], we use the Box Embedding representation model to represent concepts. The Box Embedding representation model represents an object with a hyper-rectangle aligned with the axis in space. A hyper-rectangle has an internal space to express the concept’s property, and the entities belonging to the specific concept are represented as points in it, as shown in the output part of Fig. 2.

Formally, we define the concept  $c$  as the vector  $\mathbf{c} = [Cen(\mathbf{c}), Off(\mathbf{c})] \in \mathbb{R}^{2d}$  in the real Euclidean space. The area a concept contains could be described as  $\mathbf{Box}_{\mathbf{c}} \equiv \{\mathbf{v} \in \mathbb{R}^d | Cen(\mathbf{c}) - Off(\mathbf{c}) \leq \mathbf{v} \leq Cen(\mathbf{c}) + Off(\mathbf{c})\}$ , where  $\leq$  is the bit-wise partial order relationship,  $Cen(\mathbf{c}) \in \mathbb{R}^d$  is the center of the hyper-rectangle

$\mathbf{c}$ ,  $Off(\mathbf{c}) \in \mathbb{R}_{\geq 0}^d$  is the range offset of  $\mathbf{c}$ . For example, when  $d = 2$ ,  $Cen(\mathbf{c})$  is the center of a rectangle on the plane, and  $Off(\mathbf{c})$  is half of the rectangle’s length and width. When each element in  $Off(\mathbf{c})$  is zero,  $\mathbf{Box}_{\mathbf{c}}$  degenerates into a  $d$ -dimensional vector, which is a point in the  $d$ -dimensional space as same as the entity representation. Therefore, we use points and hyper-rectangles in the representation vector space to describe the relationship between entities and concepts respectively, and the entity vectors belonging to concept  $c$  can be expressed as  $\{\mathbf{e}_i \in \mathbf{Box}_{\mathbf{c}} | e_i \in E\}$ . To determine whether an entity belongs to the concept or not, it is measured by the distance between the entity representation point and the concept hyper-rectangle. Therefore, given an entity  $e$  and a concept  $c$ , the distance between them can be defined as:

$$\begin{aligned} \text{dist}_{box}(e, c) &= \text{dist}_{outside}(e, c) + \beta \cdot \text{dist}_{inside}(e, c) \\ &= \|\text{Max}(\mathbf{h} - \mathbf{c}_{max}, \mathbf{0}) + \text{Max}(\mathbf{c}_{min} - \mathbf{h}, \mathbf{0})\|_{L_1/L_2} \\ &\quad + \beta \|\text{Cen}(\mathbf{c}) - \text{Min}(\mathbf{c}_{max}, \text{Max}(\mathbf{c}_{min}, \mathbf{h}))\|_{L_1/L_2} \end{aligned} \quad (6)$$

In Eq. 6,  $\mathbf{c}_{max} = Cen(\mathbf{c}) + Off(\mathbf{c})$ ,  $\mathbf{c}_{min} = Cen(\mathbf{c}) - Off(\mathbf{c})$ . We measure the distance between the entity and the concept from the external distance  $\text{dist}_{outside}(\cdot, \cdot)$  and the internal distance  $\text{dist}_{inside}(\cdot, \cdot)$ . External distance represents the distance from the entity to the boundary of the hyper-rectangle where the concept is located, and the internal distance represents the distance between the entity and the center of the hyper-rectangle, and  $0 < \beta < 1$  is a hyperparameter that balances the distance proportions of those two parts. Here we set a minimum value of  $\beta$  so that the distance from the entity in the hyper-rectangle to the center of the hyper-rectangle can be reduced to  $\beta$  times to weaken the internal distance and emphasize the external distance. However, we still need to measure the inner distance ( $\beta \neq 0$ ) because we do not want the range of any hyper-rectangle to expand indefinitely. We expect that the entity representations belong to the same concept are as close as possible to each other, and the entity representations that do not belong to the concept are as far away as possible from each other so that we can define the optimization goal of the ‘*instanceOf*’ relationship between the entity and the concept  $O_I$ :

$$O_I = \sum_{(e,c) \in L} \sum_{(e',c') \in L'} [\text{dist}_{box}(e, c) + \gamma_3 - \text{dist}_{box}(e', c')]_+ \quad (7)$$

The vector representation of entity  $e$  comes from the entity representation matrix  $\mathbf{H}^{(L)}$  taken from the neighborhood aggregation. The vector representation of concept  $c$  is taken from the concept representation matrix  $\mathbf{C} \in \mathbb{R}^{|C| \times 2d}$  that needs to be learned,  $\gamma_3$  is a predefined hyperparameter. Following the work of Bordes et al. [1] and Lin et al. [16], the negative training sample set  $L'_{instanceOf}$  is obtained by random uniform sampling from  $L_{instanceOf}$ . From here, we can learn about the relationship between entities and concepts.

In order to place make a hierarchical structure into the learned concept representations, which is the natural property of the ‘*subclassOf*’ relationship between concepts, we constrain each group of concept representations with the subclass

relationship so that the hyper-rectangle of the hypernym can contain the hyper-rectangle of the hyponym. It naturally conforms to the feature that the higher-level concepts in the conceptual hierarchy have a more massive description range than the lower-level concepts. Therefore, we can define the distance function of the hypernym and hyponym  $\langle c_i, subclassOf, c_j \rangle$ :

$$f_{box}(c_i, c_j) = \|\text{Max}(\mathbf{c}_{j.min} - \mathbf{c}_{i.min}, \mathbf{0}) + \text{Max}(\mathbf{c}_{i.max} - \mathbf{c}_{j.max}, \mathbf{0})\|_{L_1/L_2} \quad (8)$$

Where  $\mathbf{c}_{y.max} = \text{Cen}(\mathbf{c}_y) + \text{Off}(\mathbf{c}_y)$  and  $\mathbf{c}_{y.min} = \text{Cen}(\mathbf{c}_y) - \text{Off}(\mathbf{c}_y)$ ,  $y \in \{i, j\}$ . If the hyper-rectangle of the concept  $c_j$  entirely contains concept  $c_i$ , then the concept distance between them is  $f_{box}(c_i, c_j) = 0$ . Similarly, we define the optimization goal of the ‘*subclassOf*’ relationship between concepts as  $O_S$ :

$$O_S = \sum_{(c_i, c_j) \in L_{subclassOf}} f_{box}(c_i, c_j) \quad (9)$$

### 3.3 Model Optimization

Based on the above discussion, we can get the overall optimization goal of C4EA  $O$ :

$$O = \alpha_1 O_E + \alpha_2 O_R + \alpha_3 O_I + \alpha_4 O_S \quad (10)$$

Among them,  $O_E$ ,  $O_R$ ,  $O_I$ , and  $O_S$  correspond to the optimization goals of EA, relationship representation, ‘*instanceOf*’ relationship, and ‘*subclassOf*’ relationship, respectively.  $\alpha_1, \alpha_2, \alpha_3, \alpha_4 > 0$  are the weight parameters for balancing each part of the target. We use Python and PyTorch framework to implement C4EA and set the number of layers of the graph neural network to three (including the input layer), that means  $L = 3$ . The entities’ initial vector representation matrix  $\mathbf{X}$ , the relation vector representation matrix  $\mathbf{R}$ , and the concept vector representation matrix  $\mathbf{C}$  are all initialized randomly using Xavier [9] uniform distribution. We use the AdaGrad [8] algorithm to optimize the model, which adjusts the learning rate in each dimension according to the size of the gradient value of the independent variable so as to avoid the problem of a uniform learning rate which is difficult to adapt to all dimensions.

## 4 Experiments

### 4.1 Datasets

To test the performance of C4EA on the EA task, we use the DBP15K [27] multilingual datasets supplemented with the concept and conceptual hierarchy and compare it with seven EA methods for comparative experiments, and analyze the results in details following the common practice of recent works [27, 28]. DBP15K contains the structural knowledge of three language pairs of encyclopedias in Chinese-English, Japanese-English, and French-English. Each language pair has 15,000 equivalent entity links. Based on DBP15K, we supplement the conceptual hierarchy taxonomy built by DBpedia, and match the ‘*instanceOf*’ relationship of the concept to which each entity belongs to in the datasets. Detailed statistical data are shown in Table 1.

Due to the lack of information about the category of entities in Chinese DBpedia, we based on the Japanese and French entities linked by Japanese-English and French-English two datasets with equivalent entities that have ‘*instanceOf*’ relationship in

**Table 1.** DBP15K statistical information (values in parentheses are obtained from another language through equivalent entity linking).

DataSets		Entities	Relations	Triples	Concepts	instanceOf	subclassOf
DBP15K <sub>ZH-EN</sub>	ZH	19,388	1,701	70,414	(149)	(10,450)	769
	EN	19,572	1,323	95,142	171	18,610	
DBP15K <sub>JA-EN</sub>	JA	19,814	1,299	77,214	59	11,872	
	EN	19,780	1,153	93,484	161	18,987	
DBP15K <sub>FR-EN</sub>	FR	19,661	903	105,998	111	16,432	
	EN	19,993	1,208	115,722	144	19,175	

61.65% and 83.88%; from Chinese-English datasets of equivalent entities linked, an ‘*instanceOf*’ relationship link with an average value of 72.77% was randomly selected from English entities and passed to the corresponding Chinese entities. In this way, the evaluation results in DBP15K<sub>ZH-EN</sub> can also verify the alignment enhancement effect of C4EA on a single knowledge graph lacking the concepts and conceptual hierarchy.

## 4.2 Experimental Setup

**Comparison Models.** We use some EA methods based on representation learning to compare with C4EA, including: MTransE [4], JAPE [27], SEA [21], AlignEA (the non-iterative extended training data version of BootEA [28]), GCN-Align [33], KECG [15], RSN4EA [10], MuGNN [2] and AliNet [30]. First, we aim to explore the concept and conceptual hierarchy information from the perspective of purely using structural information, and how much improvement this information can bring to EA, we do not include comparisons with models that are beyond the scope. Thus, RDGCN [34], BERT-INT [31] and others are not included in our experiment, such models that combine entity text information and powerful side information into the representation embedding. Intuitively, we classify concept and conceptual hierarchy information as pure structural information as well, and how to better integrate it with rich semantic side information such as name label string will be the focus of our future work. Second, TransEdge [29], NAEA [37] and other methods that use an iterative expansion of training data techniques also have not been included, because of the iterative expansion of training data is a general strategy that can be applied to any EA model. For a fair comparison, we only compare C4EA with the non-iterative alignment strategy models.

**Model Variants.** We separate an ablation model C4EA (w/o Box) from C4EA, which is the C4EA after removing the concept and conceptual hierarchy module to explore the impact of adding concept and conceptual hierarchy information on entity alignment.

**Metrics.** Following previous studies, we randomly select 30% of equivalent entity links and all conceptual information for training for all models, and the remaining 70% of identical entity links are used for testing. Following convention, we use **MRR** and **Hits@N** as our evaluation metrics. Among them, **MRR** is the Mean Reciprocal Rank of all correctly aligned entities, and **Hits@N** is the proportion of correctly aligned entities whose rank is not greater than **N** (usually **N** is 1, 5, 10). The higher the **Hits@N** and **MRR**, the better the performance. Each evaluation is repeated five times with different random seeds and averaged results are reported.

**Implementation Details.** In terms of model parameters, we select the learning rate  $\gamma_L$  of the AdaGrad algorithm from  $\{0.001, 0.005, 0.01, 0.05\}$  and the number of epochs is set to 2,000. The entity representation vector dimension  $d$  is selected from  $\{100, 150, 200\}$ . Margin hyperparameters  $\gamma_1, \gamma_2, \gamma_3$  are selected from  $\{1.0, 2.0, 3.0, 5.0\}$ ; the balance hyperparameters  $\alpha_1, \alpha_2, \alpha_3, \alpha_4$  are selected from  $\{0.2, 0.4, 0.6, 0.8, 1.0\}$ . In Eq. 6  $\beta = 0.02$  is selected from  $\{0.001, 0.01, 0.02, 0.05, 0.1, 0.2\}$ . For the entity representation learning from the neighborhood aggregation, the sampling number of negative examples corresponding to a positive example is 25. For the relationship representation learning of the enhanced entity semantics, the concept and conceptual hierarchy representation learning, the sampling number of negative examples corresponding to a positive example is 2. To accelerate model training, we update the negative samples every five epochs until the model converges or gets the max number of epochs. During the test, we use the CSLS strategy [6] and the bidirectional alignment strategy in Shi et al. [25] to find the nearest entity. Through our experiments, we find the optimal parameters combination as:  $\lambda = 0.005, d = 200, \gamma_1 = 3.0, \gamma_2 = 3.0, \gamma_3 = 1.0, \beta = 0.02, \alpha_1 = 1.0, \alpha_2 = 0.8, \alpha_3 = 0.6, \alpha_4 = 0.6$ , the distance metric in the method uses  $L_2$  norm. All experiments are conducted on a Linux server with GPU(GeForce RTX 2080 Ti) and CPU(Intel Xeon E5-2680 v4), 256GB memory.

### 4.3 Experimental Results

Table 2 lists the experimental results of C4EA and its comparison methods. It can be seen that C4EA is superior to the benchmark methods in all evaluation indicators of the three datasets.

**Table 2.** The experimental results of the EA of each model. “†” means we reproduced the results using their source code; “★” means the comparison results are taken from Sun et al. [30].

Models \ Datasets	DBP15K <sub>ZH-EN</sub>			DBP15K <sub>JA-EN</sub>			DBP15K <sub>FR-EN</sub>		
	Hit@1	Hit@10	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10	MRR
MTransE*	0.308	0.614	0.364	0.279	0.575	0.349	0.244	0.556	0.335
JAPE*	0.412	0.745	0.490	0.363	0.685	0.476	0.324	0.667	0.430
SEA*	0.424	0.796	0.548	0.385	0.783	0.518	0.400	0.797	0.533
AlignEA*	0.472	0.792	0.581	0.448	0.789	0.563	0.481	0.824	0.599
GCN-Align*	0.413	0.744	0.549	0.399	0.745	0.546	0.373	0.745	0.532
KECG†	0.478	0.835	0.598	0.490	0.844	0.610	0.486	0.851	0.610
RSN4EA†	0.508	0.745	0.591	0.507	0.737	0.590	0.516	0.768	0.605
MuGNN*	0.494	0.844	0.611	0.501	0.857	0.621	0.495	0.870	0.621
AliNet*	0.539	0.826	0.628	0.549	0.831	0.645	0.552	0.852	0.657
C4EA(w/o Box)	0.531	0.857	0.636	0.551	0.858	0.656	0.550	0.871	0.664
C4EA	<b>0.576</b>	<b>0.875</b>	<b>0.682</b>	<b>0.568</b>	<b>0.892</b>	<b>0.682</b>	<b>0.578</b>	<b>0.899</b>	<b>0.698</b>

C4EA, which integrates relational representation learning and conceptual hierarchy representation learning to enhance entity semantics, can better utilize the structural information in the knowledge graph. As a result, entities will have more semantic representation information and are more conducive to distinguishing confusing entities.

The results on  $\text{DBP15K}_{\text{FR-EN}}$  show that C4EA’s Hits@1 is 0.026 higher than AliNet (the increase rate is 4.82%), which is the state-of-the-art model using purely structural information, and 0.334 more heightened than the earliest EA method MTransE (the increase rate is 136.89%). C4EA utilizes GAT to learn the representation of entities from neighborhood aggregation, which weakens the impact of locally hard-to-align entities on the global alignment effect, thus mitigating the negative impact of knowledge graph heterogeneity on EA. On  $\text{DBP15K}_{\text{JA-EN}}$ , C4EA’s Hits@10 reaches 0.892, which is much higher than the 0.745 achieved by GCN-Align which also uses graph neural networks but fails to overcome the structural heterogeneity of KGs.

Among all benchmark methods, AliNet has the highest Hits@1 and MRR, which are 0.539 and 0.628 on  $\text{DBP15K}_{\text{JA-EN}}$ , significantly outperforming the second ranked method RSN4EA. It is a result of AliNet incorporating multi-layer neighborhood entities’ features and constraining equivalent entities that have the same hidden state in the neural network. MuGNN achieves the highest Hits@10, reaching 0.870 in  $\text{DBP15K}_{\text{FR-EN}}$ , due to the fact that the relationship mining algorithm used by MuGNN alleviates knowledge graph’s sparsity, and the entity incorporates more semantics; even if it is not accurately aligned, it is also in a higher alignment ranking position. KECG incorporates knowledge graph representation learning methods for EA, which also results in better Hits@10, but the percentage of accurate alignment is lower, Hits@1 is only 0.478 in  $\text{DBP15K}_{\text{ZH-EN}}$ . RSN4EA uses the strategy of Random Walk to capture global structural features from the perspective of the knowledge graph paths. Although it achieves an excellent Hits@1, which is 0.508 in  $\text{DBP15K}_{\text{ZH-EN}}$ , its Hits@10 is significantly lower than those graph neural network-based methods, which indicates that graph neural networks have a more vital ability to capture global structural information for EA.

#### 4.4 Analysis

**The Effectiveness of Concept and Conceptual Hierarchy.** To investigate the effect of adding the concept and conceptual hierarchy information on EA, we isolate an ablation model C4EA(w/o Box) from C4EA. The results listed in Table 2 reflect the positive effect of the concept and conceptual hierarchy representation learning on EA. In  $\text{DBP15K}_{\text{FR-EN}}$  C4EA’s Hits@1 increases from 0.550 to 0.578, and MRR improves from 0.664 to 0.698. The improvement of MRR well reflects the contribution brought by the joint concept and conceptual hierarchy information to EA as a whole.

Besides, we use the existing equivalent entity links to supplement the Chinese knowledge graph in  $\text{DBP15K}_{\text{ZH-EN}}$  that lacks information about the entity’s category. We verify the alignment effect of C4EA on a single knowledge graph lacking the concept and conceptual hierarchy. It can be found that the concept and conceptual hierarchy information exerts a higher impact on the Chinese-English dataset than the Japanese-English and French-English datasets, Hits@1 increased from 0.531 to 0.576, and MRR improved from 0.636 to 0.682. However,  $\text{DBP15K}_{\text{FR-EN}}$  contains more abundant fact triples than  $\text{DBP15K}_{\text{ZH-EN}}$ , and the English and French languages are more similar. Why is the effect of combining the concept and conceptual hierarchy information not as significant as in  $\text{DBP15K}_{\text{ZH-EN}}$ ? Through statistics on the concepts introduced in  $\text{DBP15K}_{\text{FR-EN}}$  and the ‘instanceOf’ relationship and equivalent entity links, we find that two entities with the same semantics may belong to different concepts in the conceptual hierarchy. For example, “Wang\_Lee-hom” in French DBpedia belongs to the concept “Singer”, while “Wang\_Leehom” in English DBpedia belongs to the concept “Artist”. Such inconsistencies are mostly caused by the classification of entities

according to different granularities, and only a few are caused by incorrect labeling. The ‘instanceOf’ relationship introduced by Chinese DBpedia through the equivalent entity linking is consistent with English DBpedia, so the joint concept and conceptual hierarchy information bring a more significant effect on DBP15K<sub>ZH-EN</sub>. This also inspires us to focus on the inconsistency of the concept classification of entities in future work.

**The Size of the Hyper-rectangle.** We calculate the  $L_2$  norm of each concept range offset in DBP15K<sub>JA-EN</sub> as the size of the concept, and calculate the number of entities contained in each concept (the number of entities whose external distance from the entity to the concept hyper-rectangle is zero) to explore what the model has learned the concept is characterized by the hyper-rectangle. We count the top six concepts with the largest size, and the results are shown in Table 3. Through data analysis, we find that the hyper-rectangle’s size has a strong correlation with the number of entities contained in the hyper-rectangle. Larger the hyper-rectangle often includes more entities, so the more abstract these concepts are. This is also in line with the characteristics of C4EA, the concept at the upper level of the conceptual hierarchy has a larger scope and includes more entities.

**Table 3.** The top six concepts with the largest size in DBP15K<sub>JA-EN</sub>.

The top six largest concepts	Size of the concept	Entities
<a href="http://www.w3.org/.../owl#Thing">www.w3.org/.../owl#Thing</a>	39.74	22,750
<a href="http://dbpedia.org/.../Agent">dbpedia.org/.../Agent</a>	35.79	15,548
<a href="http://dbpedia.org/.../Place">dbpedia.org/.../Place</a>	32.20	9,083
<a href="http://dbpedia.org/.../Person">dbpedia.org/.../Person</a>	30.96	8,462
<a href="http://schema.org/MusicGroup">schema.org/MusicGroup</a>	30.63	38
<a href="http://dbpedia.org/.../Work">dbpedia.org/.../Work</a>	29.57	1,681

## 5 Related Work

In recent years, embedding-based methods have emerged as viable means for EA. These models can be divided into three categories according to the type of knowledge they use: (1) purely use the structural information of the knowledge graph to align entities, (2) merge attributes and attribute value information for EA, and (3) combine side information for EA. The first type of methods only use the structural information of the knowledge graph. Among these methods, the knowledge representation model such as TransE [1] is extended to make it more targeted at EA scenarios [4, 10, 25], and the use of adversarial learning to reduce the representation for spatial differences [19, 20], some models use graph neural networks to aggregate neighbor entity information [2, 37], and some iteratively increase training data [28, 29]. The second type is to integrate attributes and attribute value information for EA, learn the vector representation of entity attributes and attribute values, and then combine entity structure information to align with multiple strategies [3, 27]. The third type of models integrate powerful side information for EA, which add text information such as entity names label and descriptive abstracts for alignment [31, 35]. Most current models ignore a piece of important structural information in the knowledge graph: concept and conceptual hierarchy. We find that using those information can also improve the effectiveness of the EA model.



## 6 Conclusion

In this work, we propose a novel framework C4EA, which uses the concept and conceptual hierarchy to help entities alignment through the representation of joint concepts. We use the Box Embedding representation model to represent the concept as a hyper-rectangle in the vector representation space. The entities belonging to the concept are represented as points in the hyper-rectangle, thereby establishing the relationship between the entity and the concept (*'instanceOf'*). Secondly, we use the containment relationship represented by the hyper-rectangle to model the hierarchical relationship between concepts (*'subclassOf'*). By introducing conceptual representations, more constraints are added to entity representations to increase the possibility of alignment from a conceptual level perspective. The experimental results on three public multilingual EA datasets demonstrate the effectiveness of the proposed solution.

**Acknowledgments.** This work is supported by the Key Technology Develop and Research Project (SGTJDK00DWJS1900242) in State Grid Corporation of China. We would like to express our deepest gratitude to Shunxin Lin for her suggestions about the paper writing.

## References

1. Bordes, A., Usunier, N., et al.: Translating embeddings for modeling multi-relational data. In: NIPS, pp. 2787–2795 (2013)
2. Cao, Y., Liu, Z., Li, C., Li, J., Chua, T.S.: Multi-channel graph neural network for entity alignment. arXiv (2019)
3. Chen, M., Tian, Y., Chang, K.W., et al.: Co-training embeddings of knowledge graphs and entity descriptions for cross-lingual entity alignment. arXiv (2018)
4. Chen, M., Tian, Y., Yang, M., Zaniolo, C.: Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. arXiv (2016)
5. Chen, X., Schallehn, E., Saake, G.: Cloud-scale entity resolution: current state and open challenges. Open J. Big Data (OJBD) **4**(1), 30–51 (2018)
6. Conneau, A., Lample, G., Ranzato, M., Denoyer, L., Jégou, H.: Word translation without parallel data. arXiv (2017)
7. Cui, W., Xiao, Y., Wang, H., Song, Y., Hwang, S.w., Wang, W.: KBQA: learning question answering over QA corpora and knowledge bases. arXiv (2019)
8. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. JMLR **12**(7) (2011)
9. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: JMLR (2010)
10. Guo, L., Sun, Z., Hu, W.: Learning to exploit long-term relational dependencies in knowledge graphs. arXiv (2019)
11. Han, X., Liu, Z., Sun, M.: Neural knowledge acquisition via mutual attention between knowledge graph and text. In: AAAI (2018)
12. Jin, H., Li, C., et al.: Xlore2: large-scale cross-lingual knowledge graph construction and application. Data Intell. **1**(1), 77–98 (2019)
13. Kotnis, B., Nastase, V.: Analysis of the impact of negative sampling on link prediction in knowledge graphs. arXiv (2017)
14. Lehmann, J., Isele, R., et al.: DBpedia-a large-scale, multilingual knowledge base extracted from Wikipedia. Semantic Web **6**(2), 167–195 (2015)

15. Li, C., Cao, Y., Hou, L., Shi, J., Li, J., Chua, T.S.: Semi-supervised entity alignment via joint knowledge embedding model and cross-graph model. In: EMNLP (2019)
16. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: AAAI (2015)
17. Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models. In: ICML (2013)
18. McElvenny, J.: Ogden and Richards' the meaning of meaning and early analytic philosophy. *Lang. Sci.* **41**, 212–221 (2014)
19. Pei, S., Yu, L., Hoehndorf, R., Zhang, X.: Semi-supervised entity alignment via knowledge graph embedding with awareness of degree difference. In: WWW (2019)
20. Pei, S., Yu, L., Zhang, X.: Improving cross-lingual entity alignment via optimal transport. *IJCAI* (2019)
21. Pei, S., Yu, L., et al.: Semi-supervised entity alignment via knowledge graph embedding with awareness of degree difference. In: WWW, pp. 3130–3136 (2019)
22. Pellissier Tanon, T., Weikum, G., Suchanek, F., et al.: YAGO 4: a reason-able knowledge base. In: Harth, A. (ed.) *ESWC 2020. LNCS*, vol. 12123, pp. 583–596. Springer, Cham (2020). [https://doi.org/10.1007/978-3-030-49461-2\\_34](https://doi.org/10.1007/978-3-030-49461-2_34)
23. Ren, H., Hu, W., Leskovec, J.: Query2box: Reasoning over knowledge graphs in vector space using box embeddings. *arXiv* (2020)
24. Shao, C., Hu, L.M., Li, J.Z., et al.: RiMOM-IM: a novel iterative framework for instance matching. *J. Comput. Sci. Technol.* **31**(1) (2016)
25. Shi, X., Xiao, Y.: Modeling multi-mapping relations for precise cross-lingual entity alignment. In: EMNLP (2019)
26. Sowa, J.: *Knowledge Representation - Logical, Philosophical, and Computational Foundations*. Brooks/Cole Thomson Learning (2000)
27. Sun, Z., Hu, W., Li, C.: Cross-lingual entity alignment via joint attribute-preserving embedding. In: ISWC (2017)
28. Sun, Z., Hu, W., Zhang, Q., Qu, Y.: Bootstrapping entity alignment with knowledge graph embedding. In: *IJCAI* (2018)
29. Sun, Z., Huang, J., Hu, W., Chen, M., Guo, L., Qu, Y.: TransEdge: translating relation-contextualized embeddings for knowledge graphs. In: ISWC (2019)
30. Sun, Z., Wang, C., Hu, W., Chen, M., Dai, J.: Knowledge graph alignment network with gated multi-hop neighborhood aggregation. In: AAAI (2020)
31. Tang, X., Zhang, J., Chen, B., Yang, Y., Chen, H., Li, C.: BERT-INT: a BERT-based interaction model for knowledge graph alignment (2020)
32. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. *arXiv* (2017)
33. Wang, Z., Lv, Q., Lan, X., Zhang, Y.: Cross-lingual knowledge graph alignment via graph convolutional networks. In: EMNLP (2018)
34. Wu, Y., Liu, X., Feng, Y., Wang, Z., Yan, R., Zhao, D.: Relation-aware entity alignment for heterogeneous knowledge graphs. *arXiv* (2019)
35. Xu, K., et al.: Cross-lingual knowledge graph alignment via graph matching neural network. *arXiv* (2019)
36. Zeng, K., Li, C., Hou, L., Li, J., Feng, L.: A comprehensive survey of entity alignment for knowledge graphs. *AI Open* **2**, 1–13 (2021)
37. Zhu, Q., Zhou, X., Wu, J., Tan, J., Guo, L.: Neighborhood-aware attentional representation for multilingual knowledge graphs. In: *IJCAI* (2019)



# Incorporating Network Structure with Node Information for Semi-supervised Anomaly Detection on Attributed Graphs

Bofeng Chen<sup>1</sup>, Jingdong Li<sup>1</sup>, Xingjian Lu<sup>1</sup>(✉), Chaofeng Sha<sup>2</sup>(✉),  
Xiaoling Wang<sup>1</sup>, and Ji Zhang<sup>3</sup>

<sup>1</sup> School of Computer Science and Technology, East China Normal University,  
Shanghai, China

{51194501030,jdl}@stu.ecnu.edu.cn, {xjlu,xlwang}@cs.ecnu.edu.cn

<sup>2</sup> School of Computer Science, Fudan University, Shanghai, China  
cfsha@fudan.edu.cn

<sup>3</sup> Zhejiang Lab, Hangzhou, China  
Ji.Zhang@zhejianglab.com

**Abstract.** Anomaly detection on attributed graphs has attracted lots of research attention recently. A great deal of existing work focuses on unsupervised anomaly detection. However, in practical applications, we can obtain some labeled instances by experts, and it remains unexplored how to utilize limited labeled instances for improving the accuracy of anomaly detection. In this paper, we propose a semi-supervised anomaly detection method by considering both structure anomalies and attribute anomalies. Firstly, based on graph convolutional networks (GCNs), we learn a hypersphere that encloses normal nodes and forces anomalous nodes to lie outside, we detect structure anomalies by calculating the distances between the node embeddings and the hypersphere center. Since the trained GCNs always fail to learn better representations for anomaly detection due to noise edges are mixed into neighborhood aggregation, we use deep neural networks (DNNs) to detect attribute anomalies. Besides, to make full use of the labeled data, we incorporate the semi-supervised learning into anomaly detection, which can propagate limited label information to a large number of unlabeled instances and learn accurate nodes embeddings. Extensive experiments on five real-world datasets validate the superiority of our method and the significance of each module.

**Keywords:** Anomaly detection · Graph convolutional networks · Semi-supervised learning

## 1 Introduction

A widely accepted definition of anomaly detection (outlier detection) is to find rare instances that significantly deviate from other instances in datasets. Anomaly detection has many applications [12, 14, 23] in real-world. As more and

more scenes in the real world can be abstracted into a graph, anomaly detection on graphs is a vital research problem because of its significant implications in a wide range of applications, such as fraud detection [4] and malware detection [10].

In the past years, there have been a series of techniques for unsupervised anomaly detection due to the difficulty of accessing the ground truth anomalies. One family of methods concentrates on anomaly detection on Euclidean data (e.g., images, tables, and texts). For example, [16, 18, 19] consider attributes only and reduce the anomaly detection problem to one-class classification problem. Another family of graph-structured anomaly detection methods [1–3] that consider both structure and attributes have gradually emerged in recent years. However, the objective of these methods is to learn low-dimension node embeddings, which does not target anomaly detection directly.

In addition to many unlabeled instances, we can access a small number of labeled instances in practical applications, such as a subset of instances verified through a domain expert. So there is a huge need to develop a semi-supervised anomaly detection method that can utilize severely limited labeled instances to improve the performance of anomaly detection.

Recently, SADAG [9] has been proposed for semi-supervised anomaly detection on attributed graphs. Inspired by the idea of SVDD [19] that detects anomalies in a hypersphere, SADAG uses GCNs to learn a hypersphere that includes all normal nodes inside and force the anomalous nodes to lie outside. However, the homophily hypothesis of GCNs is untenable as the existence of noisy edge, and the structural irrelevant attribute will hurts the anomaly detection performance. The encoder might cause the over-smoothing of node representations and make anomalous nodes less distinguishable from the neighbors. Therefore, this method is hard to learn a good hypersphere for the subsequent anomaly detection task.

To alleviate the problem mentioned above, we propose to combine GCNs and DNNs with limited labels to learn node embeddings for the subsequent anomaly detection task. Specially, we adapt GCNs to exploit the complex interactions of structure and attributes, so as to learn a hypersphere that encloses normal nodes and forces anomalous nodes to lie outside. After that, structure anomalies are detected by calculating the distances between the node embeddings and the hypersphere center. The motivation of this is that anomalous instances behave differently with others, and theoretical proof about using the distance as anomaly score can be found from [16]. Due to noise edges of aggregation and the homophily hypothesis of GCNs, we learn another hypersphere based on DNNs to detect attribute anomalies. Notably, learning a hypersphere that encloses normal nodes will force the network to extract the common factors of various data, which has been applied to many anomaly detection works [16, 17]. It must be emphasized that the combination of GCNs and DNNs can help to alleviate the influence of noisy edges and the over-smoothing of node representations. **Besides**, to make full use of the limited labeled data, we incorporate semi-supervised learning into the anomaly detection task to learn good node embeddings. The anomalies of nodes are determined by both structure anomalies and attribute anomalies. The main contributions are summarized below:

- 1) We systematically analyze the limitations of existing anomaly detection methods and propose a new method that takes both structure anomalies and attribute anomalies into account for anomaly detection on attributed graphs. We combine GCNs and DNNs to detect nodes anomalies from both network structure and node attribute perspective respectively.
- 2) In order to make full use of the labeled data, we incorporate the semi-supervised learning into the anomaly detection task. It can transfer limited label information through aggregation to produce accurate node embeddings, which facilitates the anomaly detection task in turn.
- 3) We conduct experiments on five popularly used attributed graph datasets, and the experimental results demonstrate our method outperforms various existing state-of-the-art anomaly detection methods.

The remainder of the paper is organized as follows. Section 2 introduces the problem definition and related work. In Sect. 3, we present an overview of our model in detail. In Sect. 4, we introduce experiment settings and give comprehensive explanations of the results. In Sect. 5, we conclude this paper.

## 2 Background

### 2.1 Problem Definition

Following the commonly used notations, we use calligraphic fonts to denote sets (e.g.,  $\mathcal{V}$ ), bold lowercase letters (e.g.,  $\mathbf{x}$ ) to denote vectors and bold uppercase letters for matrices (e.g.,  $\mathbf{X}$ ). Given an undirected attributed graph  $\mathcal{G} = (\mathcal{V}, \mathbf{A}, \mathbf{X})$ ,  $\mathcal{V} = \{v_1, v_2, \dots, v_n\}$  is the set of nodes,  $\mathbf{A} \in R^{N \times N}$  represents topological structure of attributed graph  $\mathcal{G}$  where  $A_{i,j} = 1$  if there is a link from  $v_i$  to  $v_j$ , otherwise  $A_{i,j} = 0$ .  $\mathbf{X} \in R^{N \times D}$  represents feature matrix of attributed network  $\mathcal{G}$ , the  $i$ -th row of a matrix  $\mathbf{X}$  is denoted by  $\mathbf{x}_i$  and represents the feature of  $v_i$ . The more details about notations used in this paper is shown in Table 1. We define our problem as follows formally:

**Semi-supervised Anomaly Detection on Attributed Graphs.** Given the attribute graph  $\mathcal{G}$ , topological structure  $\mathbf{A}$ , feature matrix  $\mathbf{X}$ , a small ratio of labeled anomalous nodes set  $\mathcal{A}$  and normal nodes set  $\mathcal{N}$  with  $|\mathcal{A}| \ll |\mathcal{N}|$ . The problem of semi-supervised anomaly detection on attributed graphs is to learn a scoring model, so that the anomaly score of anomalous nodes is higher than the normal nodes.

### 2.2 Related Work

**Anomaly Detection on Euclidean Data.** Anomaly detection on Euclidean data has been a hot research topic in the last few years [13, 16, 18, 19]. For example, OC-SVM [18] finds a boundary to put most of the instances on one side and uses this boundary to predict the anomalous score of each instance. Based on this, SVDD [19] proposes hypersphere learning, which aims at learning a smallest hypersphere to include all data and detect anomalies by computing distance

**Table 1.** Notations.

Notation	Description
$\mathcal{G}$	Attributed graph
$\mathcal{V}$	A set of nodes in graph
$\mathbf{A}$	Adjacency matrix of a graph
$\mathbf{X}$	Attribute matrix of all nodes
$\mathcal{A}$	A set of anomalous nodes in graph
$\mathcal{N}$	A set of normal nodes in graph
$\mathbf{z}_s, \mathbf{Z}_s$	Embedding vector/matrix learned from struct
$\mathbf{z}_a, \mathbf{Z}_a$	Embedding vector/matrix learned from attribute
$\mathbf{c}_s$	Hypersphere center for structure anomalies
$\mathbf{c}_a$	Hypersphere center for attribute anomalies

between the center of the hypersphere and instance. An instance that falls away from the hypersphere center is defined as an outlier. DeepSVDD [16] uses neural networks to overcome the difficulty of the designed feature of SVDD. However, these methods consider attributes only and can not be well applied for anomaly detection on attributed graph because the complex interactions between different nodes. It is more challenging to detect anomalies on attributed graphs because both attributes and structure should be taken into consideration.

**Unsupervised Anomaly Detection on Attributed Graphs.** There are a growing amount of methods for unsupervised anomaly detection on attributed graphs [1–3, 5]. For example, ONE [1] and AdONE [2] compute anomaly score based on residual analysis. With the advantage of graph neural networks, Dominant [3] uses graph autoencoder to reconstruct structure and attribute, it treats reconstruction loss as anomaly score and achieves outstanding results compared to other deep methods. However, these methods do not utilize any labeled data and detect anomalies in an unsupervised setting. Nevertheless, we can obtain a small ratio of labels through expert annotations. Our proposed method leverages these limited labeled data to improve the performance of anomaly detection.

**Semi-supervised Anomaly Detection on Attributed Graphs.** Semi-supervised classification on attributed graphs has been extensively studied and achieved state-of-the-art results [8, 21]. Anomaly detection is a special case of traditional classification problem with imbalanced data, thus these classification methods are not appropriate for anomaly detection task. Although SADAGn [9] considers class imbalance for anomaly detection, when task irrelevant noise edges are mixed into neighborhood aggregation, the trained GCNs fails to learn good node representations and make anomalous nodes less distinguishable from the majority, and it becomes difficult to learn a good hypersphere for subsequent anomaly detection task. To solve this problem, in addition to use GCNs to detect structure anomalies, we employ another DNNs to detect attribute

anomalies, which can relieve the influence of noise edges and over-smoothing of GCNs.

### 3 Proposed Method

In this section, we will introduce our proposed method which incorporates network structure with node information for semi-supervised anomaly detection on attributed graphs in detail.

#### 3.1 Framework

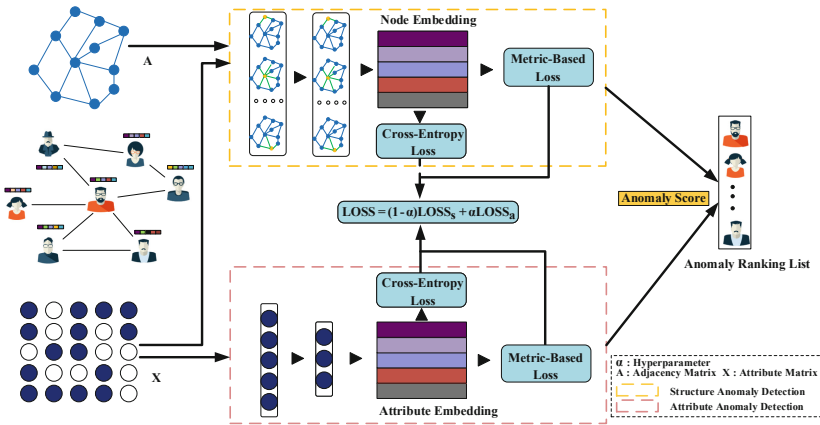


Fig. 1. The proposed framework for anomaly detection

The framework of our method, as shown in Fig. 1 is consists of two main parts: structure anomaly detection and attribute anomaly detection. The structure anomaly detection part first uses GCNs to capture the low-dimensional feature representations of the graph based on the semi-supervised learning and hyper-sphere learning, and it computes the structure anomaly score of nodes in the low-dimensional metric space based on the learned hypersphere. Attribute anomaly detection part uses DNNs to captures the low-dimensional feature representations of attributes and compute attribute anomaly scores in low-dimensional space. The final anomaly score  $S(v_i)$  of node  $v_i$  is determined by the weighted sum of structure anomaly score  $S_s(v_i)$  and attribute anomaly score  $S_a(v_i)$  as follows:

$$S(v_i) = (1 - \alpha)S_s(v_i) + \alpha S_a(v_i) \tag{1}$$

where  $\alpha$  is a parameter to control the trade off between structure anomalies and attribute anomalies. We will describe how to compute structure anomaly score and attribute anomaly score in Sect. 3.2 and Sect. 3.3 in detail.

### 3.2 Structure Anomaly Detection

Structure anomaly detection is composed of two modules: a graph-based semi-supervised learning module to capture the nonlinearity of graph while transferring labeled information between nodes. Moreover, an anomaly detection module learns a hypersphere in metric space so that normal nodes are included in the hypersphere while anomalous nodes fall outside the hypersphere.

Traditional DNNs are difficult to capture the complex relationship between attribute and structure. Inspired by GCNs [8] power of capturing the complex interactions between graph structure and attribute, a two-layer GCN is employed to learn low-dimensional representations. Mathematically,

$$\mathbf{H}^{(l+1)} = \sigma \left( \tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \mathbf{H}^{(l)} \mathbf{W}^{(l)} \right) \quad (2)$$

where  $H^{(l)}$  is the input for the  $l$ -th convolution layer,  $H^{(l+1)}$  is the output after the  $l$ -th convolution layer, the input of the first GCN layer is  $\mathbf{H}^0 = \mathbf{X}$ .  $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$  where  $\mathbf{A}$  is the adjacency matrix of the graph  $\mathcal{G}$  and  $\mathbf{I}$  is identity matrix.  $\tilde{\mathbf{D}}$  is a diagonal matrix where  $\tilde{\mathbf{D}}_{ii}$  is the degree of node  $v_i$  in self-looped graph, i.e. the summation of the  $i$ th row of  $\tilde{\mathbf{A}}$ .  $\mathbf{W}^{(l)}$  is the  $l$ -th layer trainable weight matrix and shared by all nodes, and  $\sigma(\cdot)$  is an activation function such as the ReLU or tanh. Following [16], we use ReLU as our activation function. We adapt GCNs as our encoder to make our work more easy to be followed. We can replace GCNs with complicated models like GAT [20] to learn node embeddings.

Assuming the node embeddings matrix learned by GCNs is  $\mathbf{Z}_s$ , we use the cross-entropy loss to optimize model parameters and learn feature representations as Eq. (3), where  $D_s$  is a neural network and output the structure anomaly probability of a node. We use the cross-entropy loss to optimize model parameters because it is simple and effective. We can also replace it with other loss functions like focal loss [11]. It is worth mentioning that the semi-supervised learning module can spread more limited labeled information to unlabeled data and promote the embeddings matrix of nodes learned by GCNs more meaningful.

$$\mathcal{L}_{D_s} = \frac{1}{|\mathcal{A}| + |\mathcal{N}|} \left( \sum_{o \in \mathcal{A}} \log(D_s(\mathbf{z}_s^o)) + \sum_{m \in \mathcal{N}} \log(1 - D_s(\mathbf{z}_s^m)) \right) \quad (3)$$

To bridge the gap between model architecture and the downstream anomaly detection task, we incorporate the loss of anomaly detection with the semi-supervised learning loss. Inspired by prior anomaly detection works [16, 19], we try to detect anomalies in metric space based on the node embeddings learned from semi-supervised learning module. Firstly, we force the labeled normal nodes close to the center  $\mathbf{c}_s$  by minimizing Eq. (4). Besides, to make full use of the labeled anomalous data, we employ a differential approximation of the AUC [6] to control the relative relationship between the normal nodes and anomalous nodes. As Eq. (5) shows,  $f(\cdot)$  is a sigmoid function, maximizing Eq. (5) encourages the distance from anomalous nodes to the hypersphere center always is higher than the distance from normal nodes to the hypersphere center. Equation (5)



is essential for our anomaly detection task and we will show its significance in Sect. 4.5.

$$\mathcal{L}_{\text{nor.s}} = \frac{1}{|\mathcal{N}|} \sum_{m \in \mathcal{N}} \|\mathbf{z}_s^m - \mathbf{c}_s\|^2 \quad (4)$$

$$\mathcal{R}_{\text{auc.s}} = \frac{1}{|\mathcal{A}||\mathcal{N}|} \sum_{o \in \mathcal{A}} \sum_{m \in \mathcal{N}} f(\|\mathbf{z}_s^o - \mathbf{c}_s\|^2 - \|\mathbf{z}_s^m - \mathbf{c}_s\|^2). \quad (5)$$

The final objective function of the proposed method for structure anomaly detection is a weighted sum of three loss as follow:

$$LOSS_s = \beta \mathcal{L}_{D_s} + \mathcal{L}_{\text{nor.s}} - \lambda \mathcal{R}_{\text{auc.s}} \quad (6)$$

where  $\beta$  is a hyperparameter to controls the importance of the semi-supervised learning module, and  $\lambda$  is the hyperparameter that controls the influence of the differentiable AUC loss. Please note that the ablation experiments to demonstrate the importance of the semi-supervised learning module is included when we explore the sensitivity of  $\beta$ , we will discuss these in Sect. 4.5.

Note that, the objective function of structure anomaly detection motivates the learned network to extract the shared features of the nodes and force nodes mapped to the center of the hypersphere. As a result, normal data are closely mapped to the center  $\mathbf{c}_s$ , whereas anomalous data are mapped away from the hypersphere center because anomalous nodes are rare instances and they have less shared features with the majority. Therefore, the distance from the center of the hypersphere is reasonable to be used as a measurement of anomaly.

After we have trained the model parameters, the structure anomaly score of unlabeled instance on the attributed graph can be obtained as follow:

$$S_s(v_i) = \|\mathbf{z}_s^i - \mathbf{c}_s\|^2 \quad (7)$$

where  $\mathbf{z}_s^i$  is node  $i$ 's embeddings learned from GCNs,  $\mathbf{c}_s$  is a predefined hypersphere center and we set  $\mathbf{c}_s$  as a mean of normal nodes embeddings obtained by a forward pass GCNs. It is noteworthy that we have tried to utilize a pre-trained autoencoder to get well-defined hypersphere center, but it does not produce any positive effect on performance. We do not use  $D_s$  because the classification problem always performs worse in imbalanced data. Specifically, instances with larger scores than others are more likely to be considered as structure anomalies.

Due to the smoothing operations of GCNs and task-irrelevant noise edges are mixed into neighborhood aggregation, GCNs always fail to learn good node representations for anomaly detection. For example, if an anomalous central node is connected to many normal nodes, the feature of the central anomalous node will behave similarly to normal nodes. It is hard to detect these outliers and this motivates us to detect anomalies from an attribute perspective.

### 3.3 Attribute Anomaly Detection

Refer to the model architecture of structure anomaly detection (yellow rectangle part in Fig. 1), attributes anomaly detection part also is composed of two

modules: a deep-based semi-supervised learning module to learn a non-linear mapping of each instance, an anomaly detection module that learns a hypersphere in a low-dimensional space to detect attribute anomalies.

A two-layer deep neural networks is employed to learn low-dimensional representations and the formula of the DNNs can be described formally as follows:

$$\mathbf{H}^{l+1} = \sigma(\mathbf{W}\mathbf{H}^l) \quad (8)$$

where  $\mathbf{H}^{(l)}$  is the input for the  $l$ -th layer, and  $\mathbf{H}^{(l+1)}$  is the output after the network layer, the input of the first layer  $\mathbf{H}^0 = \mathbf{X}$  and  $\sigma()$  is an activation function.

Assuming that the node embeddings learned by the DNNs is  $\mathbf{Z}_a$ , we can optimize model parameters based on cross entropy loss as follow:

$$\mathcal{L}_{D_a} = \frac{1}{|\mathcal{A}| + |\mathcal{N}|} \left( \sum_{o \in \mathcal{A}} \log(D_a(\mathbf{z}_a^o)) + \sum_{m \in \mathcal{N}} \log(1 - D_a(\mathbf{z}_a^m)) \right) \quad (9)$$

where  $D_a$  is a classifier and output the attribute anomaly probability of a node.

Based on the similar motivation from structure anomaly detection, we incorporate the anomaly detection module with the semi-supervised learning module. We use the low dimensional semantic attribute embeddings obtained through DNNs to detect attribute anomalies. Minimizing Eq. (10) pushes the embeddings of normal nodes close to the hypersphere center  $\mathbf{c}_a$ . Since our problem of anomaly detection is to compute the ranking of anomalies so that anomalous nodes score higher than normal nodes, so maximizing Eq. (11) indeed controls the relative anomaly degree of normal nodes and anomalous nodes.

$$\mathcal{L}_{\text{nor.a}} = \frac{1}{|\mathcal{N}|} \sum_{m \in \mathcal{N}} \|\mathbf{z}_a^m - \mathbf{c}_a\|^2 \quad (10)$$

$$\mathcal{R}_{\text{auc.a}} = \frac{1}{|\mathcal{A}||\mathcal{N}|} \sum_{o \in \mathcal{A}} \sum_{m \in \mathcal{N}} f(\|\mathbf{z}_a^o - \mathbf{c}_a\|^2 - \|\mathbf{z}_a^m - \mathbf{c}_a\|^2). \quad (11)$$

The final objective function of the proposed method for attribute anomaly detection can be described by a weighted sum of three loss as Eq. (12):

$$LOSS_a = \beta \mathcal{L}_{D_a} + \mathcal{L}_{\text{nor.a}} - \lambda \mathcal{R}_{\text{auc.a}} \quad (12)$$

As described in structure anomaly detection, the objective function of Eq. (12) motivates the network extracts the shared features of nodes from attribute perspective only, and map anomalous nodes far away from the center of the hypersphere  $\mathbf{c}_a$ . In order to reduce the hyperparameter search space of the model and make the model more easy to be followed,  $\beta$  and  $\lambda$  used in the structure anomaly detection part are the same as the attribute anomaly detection part. After the training of the model, the attribute anomaly score of an unlabeled node can be obtained as follow:

$$S_a(v_i) = \|\mathbf{z}_a^i - \mathbf{c}_a\|^2 \quad (13)$$

where  $\mathbf{c}_a$  is the predefined hypersphere center as  $\mathbf{c}_s$ ,  $\mathbf{z}_a^i$  is node  $i$ 's embedding learned from DNNs. Specifically, instances with larger score than others are more likely to be considered as attribute anomalies. Attribute anomalies part can help to alleviate the influence of noisy edges and the over-smoothing of node representation during the structure anomaly detection, and improve the performance of anomaly detection task.

### 3.4 Loss Function

The objective function of our proposed anomaly detection method is to minimize both structure anomalies and attribute anomalies as Eq. (14):

$$LOSS = (1 - \alpha)LOSS_a + \alpha LOSS_s \quad (14)$$

We use  $\alpha$  to balance the importance between structure anomalies and attribute anomalies. It's noted that our method only detect the structure anomalies when  $\alpha = 0$  while merely detect the attributes anomalies when  $\alpha = 1$ , we will discuss these ablation experiments in Sect. 4.5. We can employ any gradient-based optimization methods like Adam [7] to optimize Eq. (14).

## 4 Experiment

### 4.1 Datasets

As far as we know, there is no public dataset with ground truth for the graph anomaly detection task. Since anomaly detection is a special case of traditional classification problem with imbalanced data, following the previous works [9, 16, 17, 22], we simulate the anomaly by regarding the smallest class as anomalous and the remaining classes as normal. Particularly, we use Cora, Cite-seer (Cite), Pubmed (Pub), Amazon-Photo (Photo), and Amazon-Computers (Comp) datasets with real labels as our experimental datasets, the smallest class is treated as an anomalous class. The number of outliers in five datasets are 180, 264, 4103, 331, 2291. Cora, Cite and Pub are scientific citation datasets where edges represent citations between papers, and the 1/0 value in the attribute represents whether the word appears in bag-of-words. Photo and Comp are Amazon's common purchase datasets, the edges represent frequent purchases between item nodes. The attribute value of 0/1 represents whether the word appears in the corresponding comment. The details of the datasets are shown in Table 2.

**Table 2.** The statistics of datasets.

Data	Nodes	Edges	Attributes	Classes	Anomalous nodes	Anomaly ratio
Cora	2708	5278	1433	7	180	6.6%
Citeseer	3327	4732	3703	6	264	7.9%
Pub	19717	44338	500	3	4103	20.8%
Photo	7650	119043	745	8	331	4.3%
Comp	13381	245778	767	10	2291	14.5%

## 4.2 Baselines and Evaluation Metrics

We evaluated three groups of the anomaly detection method to prove the effectiveness of our method: namely, “Attributes-only”, “Structure-only” and “Attributes+Structure”. “Attributes-only” methods (OC-SVM [18], Deep-SVDD [16], DSAD [17]) leverage node attribute only to detect anomalies. “Structure-only” methods (DeepWalk [15]) consider structure only while ignoring attribute. “Attributes+Structure” methods (Dominant [3], SLGCN [8] and SADAG [9]) capture both attribute and structure simultaneously to detect anomalies. A detailed description of these baselines is illustrated as follows:

- **OC-SVM** [18] is the one-class support vector machine, which tries to find a hyperplane that put all normal instances on one side. It only takes attributes into consideration.
- **Deep-SVDD** [16] trains a neural network to minimize the volume of a hypersphere that encloses the node embeddings. It is an unsupervised anomaly detection method that uses attributes only.
- **DSAD** [17] is a semi supervised anomaly detection methods that extends from Deep-SVDD, which uses both labeled and unlabeled instances to learn a hypersphere. DSAD uses attribute only and ignore structure .
- **NN** is the semi-supervised feed forward neural network classifier with attributes only. The parameters of NN are trained by minimizing the cross-entropy loss. In the experiments, we set the number of hidden layers = 2.
- **DW** [15] is the representative an unsupervised node embedding method that learns embeddings using structural information only. We use logistic regression as classifiers to detect anomalies after learning node embeddings.
- **Dominant** [3] is an unsupervised anomaly detection method for attributed graph. It treats reconstruction loss as anomaly score.
- **SLGCN** [8] is a semi-supervised learning method which encodes attribute and structure with GCNs and use cross entropy loss to train model. It regards anomaly detection as classification problem.
- **SADAG** [9] is a semi-supervised anomaly detection method. The most difference between our method and SADAG is we not only incorporate the semi-supervised learning module with the anomaly detection but also combine structure anomalies and attribute anomalies together to detect anomalies.

Since the class-imbalance of data, the evaluation metrics such as accuracy can not well evaluate the performance of the anomaly detection task. Following prior work [9], we adapt AUC (Area Under Curve) as our evaluation metric to measure the probability that ranks anomaly instances before normal instances.

### 4.3 Experimental Settings

Following experimental settings [9], we set embedding dimension of all methods as 32, we use the Adam [7] with a learning rate of 0.001 and randomly select 10% of all instances as the validation set to avoid over-fitting. The maximum number of epochs we train for small datasets (Cora and Cite) is 500, and 1000 for large datasets (Pub, Photo and Comp). We use the source codes published by authors and tune them to their best performance. Since it is challenging to obtain labels in real anomaly detection scenes, we set the limited ratio of labeled instance  $\mu$  as 2.5%, 5%, 10% to evaluate performance. In our model, we use a two layer GCNs with the hidden size of 64 and ReLU activation for structure anomaly detection, a two layer DNNs with the hidden size of 64 and ReLU activation for attribute anomaly detection. The dimension of the last encoder layer based on GCNs or DNNs is set to 32 for different datasets. Dropout is set to 0.5 for all layers. We tune  $\alpha$  from 0 to 1 with a step of 0.1,  $\beta$  from  $\{0,0.001,0.005,0.01,0.05,0.1,0.5,1\}$  and  $\gamma$  from  $\{0,0.1,1,10,100,1000\}$  to achieve better performance. To make results more stable, we run ten times for each case and record the average and standard deviation of AUCs as the experimental results.

### 4.4 Performance Evaluation

Table 3, 4 and 5 shows the performance for each dataset with different ratio of labeled data, and our proposed method achieves the best results in all datasets compared with baselines. In addition to this, we have some other findings. **1)** Compared with methods that only use structure (DW) or methods that only use attribute (OCSVM, OC-SVM, and Deep-SVDD), these methods (SLGCN, SADAG, and Ours) that use both structure and attribute can achieve better results, which verifies the importance of performing anomaly detection on attributed graphs. The experimental results show an interesting phenomenon that NN performs better than SADAG in some settings, the reason may be that using GCNs to learn nodes representations will lead to anomalous nodes behave similarly to normal nodes as noise edges between them, which affect the anomaly detection performance. This motivates us to detect anomalies from two perspectives: structure and attribute. **2)** Trough the comparison of SADAG, the performance of our method validates the importance of detecting attribute anomalies and structure anomalies simultaneously. **3)** Our method shows a stronger ability to detect anomalies according to the results of AUCs in all cases. Experimental results show that our method has about 2% to 3% improvement of almost dataset with different ratio of labeled instances. Notably, the AUCs of Comp can have an improvement even though the performance of SADAG achieve 99%.

**Table 3.** Average and standard deviation of test AUCs [%] when 2.5% of all instances are labeled (the best results are bold).

Data	Method								
	Attributes-only				Structure-only	Attributes+Structure			
	OC-SVM	Deep-SVDD	DSAD	NN	DW	Dominant	SLGCN	SADAG	Ours
Cora	50.1(0.2)	57.9(5.1)	67.2(5.7)	73.3(3.0)	51.3(2.0)	49.9(6.0)	84.5(5.3)	89.6(5.5)	<b>94.0(1.9)</b>
Cite	50.1(0.1)	58.3(3.0)	53.6(6.2)	64.7(5.9)	50.6(2.0)	53.4(1.9)	60.8(5.4)	66.9(3.5)	<b>71.9(3.0)</b>
Pub	69.1(1.2)	92.9(0.9)	95.3(0.4)	96.2(0.3)	49.8(0.4)	63.1(3.1)	96.0(0.3)	95.0(0.3)	<b>96.5(0.3)</b>
Photo	53.7(1.5)	64.2(4.3)	63.1(4.5)	72.9(7.0)	48.8(2.0)	49.5(5.0)	91.7(1.8)	93.0(4.9)	<b>97.1(0.8)</b>
Comp	47.5(0.7)	60.6(6.1)	63.9(3.8)	79.3(8.1)	49.7(1.9)	53.0(2.2)	98.2(0.6)	98.7(0.6)	<b>99.4(0.2)</b>

**Table 4.** Average and standard deviation of test AUCs [%] when 5% of all instances are labeled (the best results are bold).

Data	Method								
	Attributes-only				Structure-only	Attributes+Structure			
	OC-SVM	Deep-SVDD	DSAD	NN	DW	Dominant	SLGCN	SADAG	Ours
Cora	50.3(0.3)	61.5(4.6)	55.6(6.7)	83.0(1.8)	51.5(2.3)	46.4(5.0)	92.3(5.7)	93.0(2.5)	<b>96.3(1.0)</b>
Cite	50.8(0.5)	59.6(4.0)	74.5(4.2)	68.9(2.0)	51.1(1.8)	53.4(1.9)	67.6(3.2)	71.0(2.1)	<b>74.6(2.8)</b>
Pub	71.0(1.1)	94.4(0.5)	96.0(0.2)	96.9(0.2)	50.0(0.3)	63.8(1.1)	96.4(0.1)	95.6(0.2)	<b>97.1(0.2)</b>
Photo	54.7(1.0)	66.8(4.0)	66.3(4.9)	82.4(4.7)	50.3(1.5)	51.7(4.4)	93.9(1.4)	93.0(7.3)	<b>97.8(0.5)</b>
Comp	49.3(1.8)	66.8(5.8)	71.1(4.2)	89.0(5.2)	50.7(1.5)	53.5(3.2)	98.6(0.3)	99.0(0.3)	<b>99.6(0.1)</b>

**Table 5.** Average and standard deviation of test AUCs [%] when 10% of all instances are labeled (the best results are bold).

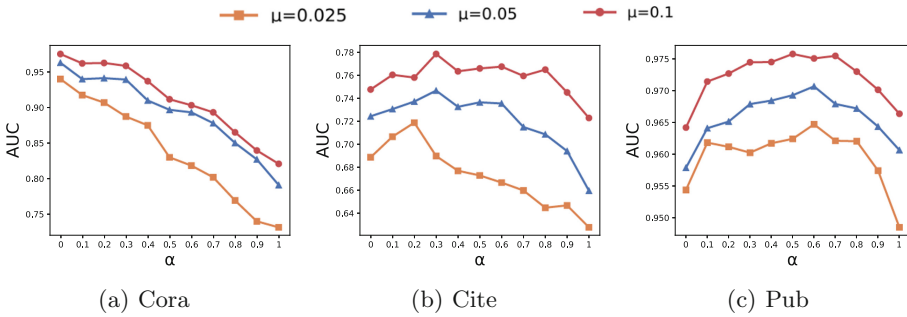
Data	Method								
	Attributes-only				Structure-only	Attributes+Structure			
	OC-SVM	Deep-SVDD	DSAD	NN	DW	Dominant	SLGCN	SADAG	Ours
Cora	51.5(1.7)	64.4(4.9)	55.7(9.1)	86.0(1.3)	52.9(3.0)	49.5(5.8)	96.0(1.4)	95.4(1.8)	<b>97.5(0.5)</b>
Cite	51.4(0.8)	62.1(5.7)	83.2(3.8)	73.3(2.5)	52.2(2.7)	54.5(2.1)	76.6(2.9)	70.0(3.2)	<b>77.8(1.7)</b>
Pub	73.1(0.8)	95.7(0.3)	96.7(0.1)	97.3(0.1)	50.1(0.3)	62.3(0.7)	96.6(0.1)	96.1(0.2)	<b>97.6(0.1)</b>
Photo	55.0(1.2)	71.1(5.5)	70.9(3.9)	89.9(2.6)	52.5(2.2)	51.5(5.2)	94.0(1.0)	94.0(6.1)	<b>98.3(0.2)</b>
Comp	47.9(1.0)	72.6(4.3)	74.8(2.9)	94.2(3.6)	51.2(1.5)	54.0(4.2)	98.6(0.3)	99.4(0.2)	<b>99.7(0.1)</b>

#### 4.5 Parameter Sensitivity Analysis

In this part, we conduct more experiments on three popular used datasets (Cora, Cite, Pub) [1, 2, 8, 9] to explore the effect of three parameters in our loss function. We keep other parameters fixed to evaluate the influence. We also analyze the influence of the regularization parameter  $\lambda$  and the embedding dimension  $K$  on experimental performance.

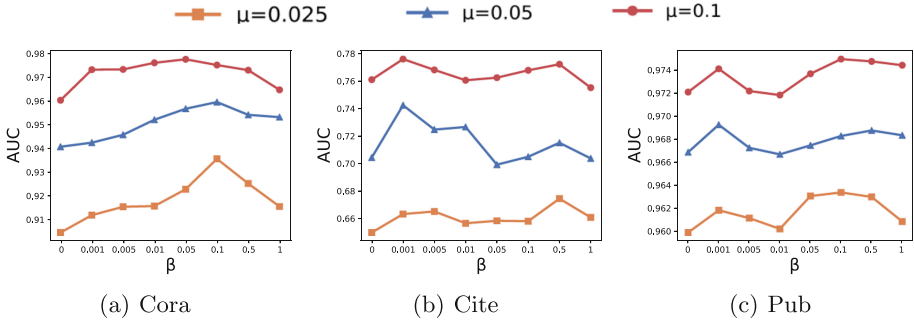
**Effect of the Combination of Structure Anomaly Detection and Attribute Anomaly Detection.** Since  $\alpha$  balances the importance between structure anomalies and attribute anomalies, in two extreme cases, our method will degenerate into a structure anomaly detection part when  $\alpha = 0$  or degenerate into attribute anomaly detection part when  $\alpha = 1$ . So the **ablation experiments** to demonstrate the importance of structure anomaly detection part and attribute anomaly detection part are included when we explore the effect of  $\alpha$ .

We report the average AUCs with different ratio of labeled instances when  $\alpha$  changed from 0 to 1 with a step of 0.1. As Fig. 2 shows, the trend of the curves is very similar—performance increases when  $\alpha$  gets larger at first and decreases when  $\alpha$  is larger than a specific value. The ablation experiments when  $\alpha = 0$  or 1 always perform worse than our model, which demonstrates the importance of the combination of structure anomaly detection and attribute anomaly detection. **Moreover**, we find another observation: our method considers structure anomalies only performs best in Cora. This may be because Cora has relatively denser connections inside the anomalous class than outside, and many papers with similar topics have links, so taking attribute anomalies into account will not improve the performance of anomaly detection. The other datasets have relatively few links inside anomalous class. In this regard, taking attribute anomalies and structure anomalies into account can indeed improve the performance of anomaly detection. Thus, above observations give us some guidance about how to select alpha by analyzing the statistics of datasets. **Finally**, although best  $\alpha$  varies with different dataset, the trend of curve is similar in a particular dataset with different ratio of labeled instances. This phenomenon can provide helpful suggestions for future  $\alpha$  selection when we can get additional labeled instances (e.g. best  $\alpha$  is close when labeled instances changed from  $\mu = 2.5\%$  to  $5\%$ ).

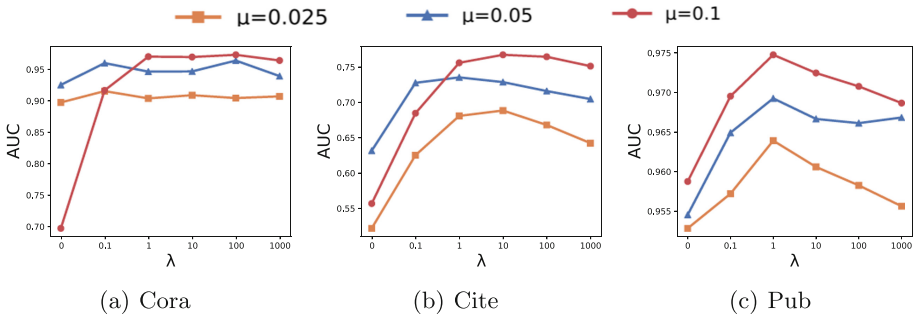


**Fig. 2.** Average test AUCs of each dataset with different ratio of labeled instances when  $\alpha$  was changed.

**Effect of the Semi-supervised Learning Module.** Since  $\beta$  indicates the importance of semi-supervised learning module, we analyze the influence of the  $\beta$  on the results.  $\beta = 0$  means that we remove the semi-supervised learning module, so the ablation experiments to demonstrate the importance of semi-supervised learning module are included when we explore the effect of  $\beta$ . Figure 3 shows the experimental performance of each dataset with different ratio of labeled instances when changing  $\beta$ . Obviously, the performance of the method removing the semi-supervised learning module always performs worse than other methods with the semi-supervised learning module. This phenomena shows that the use of a small number of labeled instances to learn embedding under semi-supervised



**Fig. 3.** Average test AUCs of each dataset with different ratio of labeled instances when  $\beta$  was changed.



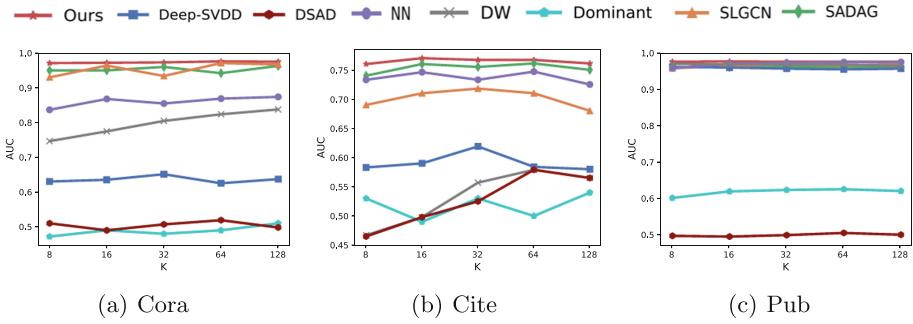
**Fig. 4.** Average test AUCs of each dataset with different ratio of labeled instances when  $\lambda$  was changed.

learning module indeed can spread label information to more unlabeled instances and help to produce accurate nodes representations. The integration of a simple semi-supervised learning module can actually boost the performance of anomaly detection in metric space.

**Effect of the Regularization.** We analyze the influence of  $\lambda$  on the experimental results. The experimental results of the average test AUCs of each dataset by changing  $\lambda$  with different ratio of labeled instances is shown in Fig. 4. The best  $\lambda$  of our method differs across datasets, but the performance always reaches the best value at  $\lambda = 1$  or  $\lambda = 10$ .

**Effect of the Embeddings Dimension.** We compare the influence of the dimension of embeddings  $K$  on the experimental results with seven methods (Baselines without OC-SVM). Due to the space limitation, we only show experimental results with 10% labeled instances. The experimental result in Fig. 5 shows that our method is not sensitive to dimension  $K$  and consistently performs better than other baselines, which shows that our method is more robust to  $K$ .





**Fig. 5.** Average test AUCs of each dataset with 10% of labeled instances when  $K$  was changed.

## 5 Conclusion

In this paper, we develop a semi-supervised anomaly detection method on attributed graphs, which incorporates network structure with node information for anomaly detection. Our method take both attribute anomalies and structure anomalies into account to detect anomalies. Another key factor of our method is to incorporate the semi-supervised learning into anomaly detection, which is a effective idea that can be applied to boost the performance of anomaly detection. We conducted experiments and demonstrated the effectiveness.

**Acknowledgments.** This work was supported by NSFC grants (No. 61972155), Natural Science Foundation of Shanghai (No. 21ZR1419900), Open Research Fund of KLATASDS-MOE (KLATASDS2104), and Fundamental Research Funds for the Central Universities.

## References

1. Bandyopadhyay, S., Lokesh, N., Murty, M.N.: Outlier aware network embedding for attributed networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 12–19 (2019)
2. Bandyopadhyay, S., Vivek, S.V., Murty, M.: Outlier resistant unsupervised deep architectures for attributed network embedding. In: Proceedings of the 13th International Conference on Web Search and Data Mining, pp. 25–33 (2020)
3. Ding, K., Li, J., Bhanushali, R., Liu, H.: Deep anomaly detection on attributed networks. In: Proceedings of the 2019 SIAM International Conference on Data Mining, pp. 594–602. SIAM (2019)
4. Dorronsoró, J.R., Ginel, F., Sgnchez, C., Cruz, C.S.: Neural fraud detection in credit card operations. *IEEE Trans. Neural Netw.* **8**(4), 827–834 (1997)
5. Fan, H., Zhang, F., Li, Z.: AnomalyDAE: dual autoencoder for anomaly detection on attributed networks. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), ICASSP 2020, pp. 5685–5689. IEEE (2020)
6. Iwata, T., Yamanaka, Y.: Supervised anomaly detection based on deep autoregressive density estimators. arXiv preprint [arXiv:1904.06034](https://arxiv.org/abs/1904.06034) (2019)

7. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)
8. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint [arXiv:1609.02907](https://arxiv.org/abs/1609.02907) (2016)
9. Kumagai, A., Iwata, T., Fujiwara, Y.: Semi-supervised anomaly detection on attributed graphs. arXiv preprint [arXiv:2002.12011](https://arxiv.org/abs/2002.12011) (2020)
10. Li, A., Qin, Z., Liu, R., Yang, Y., Li, D.: Spam review detection with graph convolutional networks. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pp. 2703–2711 (2019)
11. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2980–2988 (2017)
12. Liu, B., et al.: Co-detection of crowdturfing microblogs and spammers in online social networks. *World Wide Web* **23**(1), 573–607 (2020). <https://doi.org/10.1007/s11280-019-00727-4>
13. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation forest. In: 2008 Eighth IEEE International Conference on Data Mining, pp. 413–422. IEEE (2008)
14. Pawar, K., Attar, V.: Deep learning approaches for video-based anomalous activity detection. *World Wide Web* **22**(2), 571–601 (2018). <https://doi.org/10.1007/s11280-018-0582-1>
15. Perozzi, B., Al-Rfou, R., Skiena, S.: DeepWalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 701–710 (2014)
16. Ruff, L., et al.: Deep one-class classification. In: International Conference on Machine Learning, pp. 4393–4402 (2018)
17. Ruff, L., et al.: Deep semi-supervised anomaly detection. arXiv preprint [arXiv:1906.02694](https://arxiv.org/abs/1906.02694) (2019)
18. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. *Neural Comput.* **13**(7), 1443–1471 (2001)
19. Tax, D.M., Duin, R.P.: Support vector data description. *Mach. Learn.* **54**(1), 45–66 (2004)
20. 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)
21. Wu, F., Zhang, T., Souza Jr., A.H.D., Fifty, C., Yu, T., Weinberger, K.Q.: Simplifying graph convolutional networks. arXiv preprint [arXiv:1902.07153](https://arxiv.org/abs/1902.07153) (2019)
22. Wu, J., He, J., Liu, Y.: ImVerde: vertex-diminished random walk for learning imbalanced network representation. In: 2018 IEEE International Conference on Big Data (Big Data), pp. 871–880. IEEE (2018)
23. Xiao, D., Song, L., Wang, R., Han, X., Cai, Y., Shi, C.: Embedding geographic information for anomalous trajectory detection. *World Wide Web* **23**(5), 2789–2809 (2020). <https://doi.org/10.1007/s11280-020-00812-z>



# OntoSP: Ontology-Based Semantic-Aware Partitioning on RDF Graphs

Sizhuo Li<sup>1</sup>, Weixue Chen<sup>1</sup>, Baozhu Liu<sup>1</sup>, Pengkai Liu<sup>1</sup>, Xin Wang<sup>1,2(✉)</sup>,  
and Yuan-Fang Li<sup>3</sup>

<sup>1</sup> College of Intelligence and Computing, Tianjin University, Tianjin, China  
{[lszskye](mailto:lszskye@tju.edu.cn), [chenweixue](mailto:chenweixue@tju.edu.cn), [liubaozhu](mailto:liubaozhu@tju.edu.cn), [liupengkai](mailto:liupengkai@tju.edu.cn), [wangx](mailto:wangx@tju.edu.cn)}@tju.edu.cn

<sup>2</sup> Tianjin Key Laboratory of Cognitive Computing and Application, Tianjin, China

<sup>3</sup> Monash University, Melbourne, Australia

[yuanfang.li@monash.edu](mailto:yuanfang.li@monash.edu)

**Abstract.** With the development of the Semantic Web, more and more fields construct and publish their domain knowledge in RDF format. Facing the challenge of managing massive and ever-growing RDF graph data in a distributed environment, it is necessary to consider a graph partitioning algorithm to distribute graph data onto multiple machines. However, the existing RDF graph partitioning algorithms either fail to leverage the semantic information or leave the edge-cut number to be further improved. In this paper, we aim to provide a better solution that fits the need for RDF graph partitioning in a distributed system. Inspired by the ontology hierarchy, the definition of semantic coherence is proposed to assess the partitioning performance. Then, we propose an ONTOlogy-based Semantic-aware Partitioning algorithm ONTOSP for RDF graphs, which can exploit the hierarchical structure of ontology in RDF graphs as heuristic information for better partitioning. The extensive experiments on real-world datasets show that ONTOSP outperforms the state-of-the-art methods by a large margin.

**Keywords:** RDF graph · Graph partitioning · Ontology-based

## 1 Introduction

Knowledge graphs have become the cornerstone of artificial intelligence. With the applications of artificial intelligence, an increasing number of large-scale knowledge graphs have been published by both academic and industrial communities. As a W3C recommendation, Resource Description Framework (RDF) is one of the most commonly used knowledge representation forms. An RDF dataset is usually represented as a collection of *subject-predicate-object* triples, where *subject* and *object* are entities or concepts and *predicate* is the relationship connecting them.

Due to the simplicity and flexibility of RDF, RDF graphs have been widely used and the size of the available RDF knowledge repositories built by human

experts or extracted from large text corpora reaches an unprecedented scale. Currently, it is common that RDF graphs have millions of vertices and billions of edges. Many RDF graphs in the LOD (Linked Open Data) cloud diagram have more than 1 billion triples, e.g., the number of triples of the latest version of the DBpedia [1] dataset has reached 13 billion.

On one hand, the continuous explosion of RDF data leads to new innovations in big data and Semantic Web initiatives. On the other hand, it easily overwhelms the memory and computation resources on servers, and causes performance bottlenecks in many existing RDF stores [2]. Since many scientific and commercial online services have need for answering queries over big RDF data in near real time, careful partitioning and distribution of big RDF data across a cluster of servers are required [3].

The previous graph partitioning algorithms, however, ignore the semantic information of RDF graphs and fail to leverage the hierarchical structure of ontology. Meanwhile, the existing metrics of RDF graph partitioning lacks the consideration of the semantic dimension.

Our research aims to provide a better solution that fits the need for RDF graph partitioning in a distributed system. The example shown in Fig. 1 explains the difference between the traditional partitioning algorithm and our ONTOSP algorithm. The traditional partitioning method based on edge-cut number only produces one edge across two partitions, as shown in Fig. 1(a). Promising as it seems to be, this method does not consider the inherent semantic information of the given RDF graph, vertices of *Philosopher* type, e.g., *Aristotle* and *Ludwig\_Wittgenstein* are assigned to different partitions.

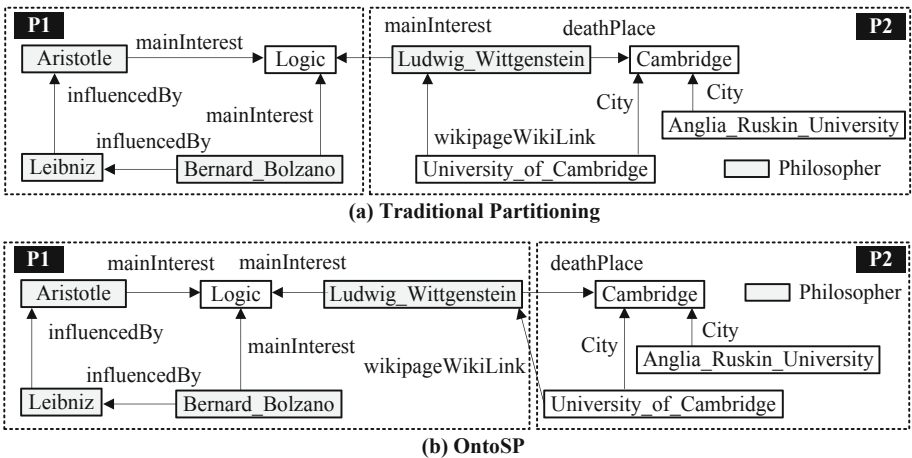


Fig. 1. Different partitioning results

In this paper, we propose the definition of semantic coherence, which is a novel criterion for RDF graph partitioning evaluation. Moreover, the semantic information is introduced to optimize partitioning and facilitate query as heuristic information. The workflow of our approach can be summarized as follows:

(1) ontology is extracted to obtain the hierarchical structure information in the RDF graph; (2) the adaptive semantic similarity between vertices, which is composed of the distance-based similarity, the information-based similarity, and the feature-based similarity, can be calculated; (3) the vertices are reassigned on the basis of the initial partitioning results to gather vertices with the similar structure in the same partition and maximize the semantic coherence of the whole RDF graph.

Our main contributions can be summarized as follows:

- 1) We propose the definition of semantic coherence, which is a novel evaluation dimension of RDF graph partitioning. To the best of our knowledge, this is the first time that ontology information has been adopted to evaluate partitioning performance.
- 2) An elaborate data partitioning algorithm, ONTOSP, is devised for RDF graphs, which is capable of reducing the edge-cut number while maintaining the maximum value of the semantic coherence.
- 3) The extensive experiments on the real-world datasets have been conducted to verify the effectiveness and efficiency of our approach. The experimental results show that our algorithm outperforms the state-of-the-art methods.

The rest of the paper is organized as follows: Sect. 2 introduces the preliminary definitions and briefly reviews the related work. The definition of semantic coherence and the RDF graph partitioning algorithm, ONTOSP, are presented in Sect. 3. The experimental results on real-world datasets are presented in Sect. 4. Finally, we conclude in Sect. 5.

## 2 Preliminaries

In this section, we introduce several background definitions, which are used in our ONTOSP algorithms.

### 2.1 RDF and Graph Partitioning

**Definition 1 (RDF Graph).** *Let  $U$ ,  $B$ , and  $L$  be three infinite disjoint sets of URIs, blank nodes, and literals, respectively. A triple  $(s, p, o) \in (U \cup B) \times U \times (U \cup B \cup L)$  is called an RDF triple, where  $s$  is the subject,  $p$  is the predicate, and  $o$  is the object. A finite set of RDF triples is called an RDF graph.*

Since an RDF triple can be regarded as a directed edge (predicate) connecting two vertices (from subject to object). An RDF graph can be alternatively viewed as a directed graph  $G = (V, E)$ , where  $V = \{s \mid (s, p, o) \in G\} \cup \{o \mid (s, p, o) \in G\}$ , and  $E = \{p \mid (s, p, o) \in G\}$ .

**Definition 2 (Balanced Graph partitioning).** *Given an graph  $G$ , a  $k$ -way partition  $\mathcal{P}$  of  $G$  is a set of graphs  $\mathcal{P} = \{P_1, \dots, P_k\}$  such that (1) each  $P_i$  is a*

subgraph of  $G$ ; (2)  $P_i \cap P_j = \emptyset$ ; and (3)  $\bigcup_{i=1}^k P_i = G$  ( $i, j \in [1, k]$  and  $i \neq j$ ). A balanced graph partitioning needs to satisfy (1) to achieve load balancing,

$$(1 - \theta) \frac{|V|}{k} < |V_i| < (1 + \theta) \frac{|V|}{k}, 0 \leq \theta \leq 1 \quad (1)$$

where  $\theta$  is the balance factor, and  $|V|$ ,  $|V_i|$  represents the number of vertices in graph  $G$  and the number of vertices in partition  $P_i$ , respectively.

## 2.2 Related Work

Graph partitioning is an important problem with applications in many areas, hence a large variety of graph partitioning algorithms have been proposed in several communities for decades. The existing approaches can be roughly divided into the following three categories.

**Multilevel Graph Partitioning Approaches.** The multilevel graph partitioning algorithms first transform graph into hundreds of vertices, partition the smaller graph and then project the result back to the original graph. The whole process can be divided into three stages: the contraction phase, the partitioning phase, and the uncoarsening phase. METIS [4], as one of the most common multilevel graph partitioning algorithms, is often used as a comparison standard for other algorithms due to its fast speed and high partition quality. However, the main weaknesses of existing multilevel graph partitioning algorithms are the high overhead of loading the RDF data and the poor scalability to large datasets [5].

**Streaming Graph Partitioning Approaches.** The streaming graph partitioning algorithms load serial graph data as a sequence containing vertices or edges and assign each element to the corresponding partition. Since the location of each element is decided as it is loaded, streaming graph partitioning algorithms avoid multiple iteration calculations and scale well on large-scale graphs. As a representative streaming graph partitioning, hash partitioning [6], which divides an RDF graph into smaller and similar sized partitions by hashing over the subject, predicate, or object of RDF triples, is one of the dominating approaches in the existing distributed systems. Concretely, Virtuoso, YARS2 [7], and CumulusRDF [8] are distributed RDF systems which adopt hash partitioning as their triple partitioning strategy. Hash partitioning, though simple, falls short in minimizing the amount of inter-partition coordination and data transfer involved in distributed query processing.

**Distributed Graph Partitioning Approaches.** Several distributed graph partitioning models have been proposed for efficient graph processing on a cluster of commodity servers. Concretely, Pregel [9] and GraphChi [10] are known for their iterative graph computation techniques that can speed up certain types of graph computations. The definition of vertex blocks is proposed in [11] for semantic partitioning of large graphs. A vertex block grouping algorithm is devised to place those vertex blocks that have high correlation in graph structure into the same partition.

More recently, several semantic-aware partitioning algorithms are presented for RDF graphs, e.g., [5] and [12]. However, these algorithms also suffer from the issue of ignoring the ontology structure of RDF graphs. In contrast, this is the first work, to the best of our knowledge, which presents an ontology-based semantic-aware partitioning method.

### 3 Ontology-Based Semantic-Aware Partitioning

In this section, the definitions of semantic coherence and adaptive semantic similarity are proposed. Then, we propose an ontology-based semantic-aware partitioning algorithm ONTOSP for RDF graphs, which can exploit the hierarchical structure of ontology in RDF graphs as heuristic information for better partitioning.

#### 3.1 Semantic Coherence

Given an RDF graph  $G = (V, E)$ , for each vertex  $v_i \in V$ , we use  $N(v_i)$  to denote the set of neighbors of  $v_i$ .  $Sim(v_i, v_j)$  is used to denote the adaptive semantic similarity between vertex  $v_i$  and vertex  $v_j$  and is described in detail in Sect. 3.2. The formal definition of semantic coherence is given as follows.

**Definition 3 (Semantic coherence).** *Given a partition  $\mathcal{P}$ , the semantic coherence of a vertex  $v_i$  in the partition  $P_t$  is the ratio of the sum of the adaptive semantic similarity between  $v_i$  and its neighbors in  $P_t$  to the sum of the adaptive semantic similarity between  $v_i$  and all its neighbors.*

$$SC(v_i, P_t) = \frac{\sum_{v_j \in N(v_i), v_j \in P_t} Sim(v_i, v_j)}{\sum_{v_k \in N(v_i), v_k \in V} Sim(v_i, v_k)} \tag{2}$$

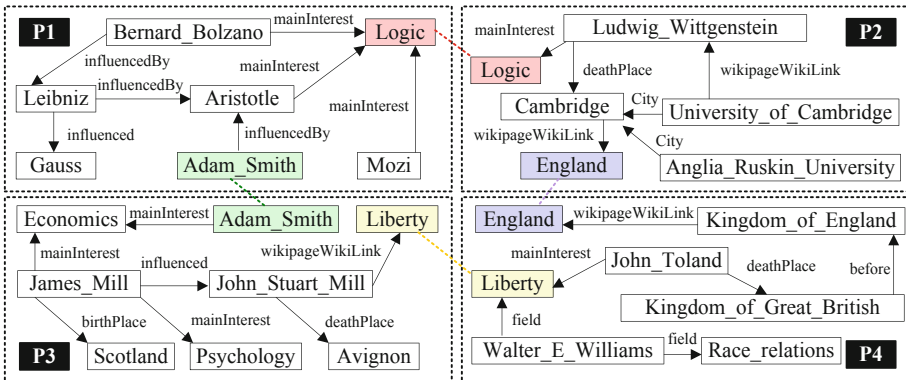


Fig. 2. An example partition of the given RDF graph.

**Example 1.** Given the partition  $\mathcal{P}$  in Fig. 2, the semantic coherence of  $SC(\text{Logic}, P_1)$  is the ratio of the sum of  $Sim(\text{Logic}, \text{Bernard\_Bolzano})$ ,  $Sim(\text{Logic}, \text{Aristotle})$ , and  $Sim(\text{Logic}, \text{Mozi})$  to the sum of  $Sim(\text{Logic}, \text{Bernard\_Bolzano})$ ,  $Sim(\text{Logic}, \text{Aristotle})$ ,  $Sim(\text{Logic}, \text{Mozi})$ , and  $Sim(\text{Logic}, \text{Ludwig\_Wittgenstein})$ .

The definition of semantic coherence measures the correlation between a vertex at the boundary (represented by colored box) and its possible candidate partition. The vertex will eventually be moved from the initial partition to the candidate partition with the highest value of semantic coherence. Note that only the vertices at the boundary can be moved, for the semantic coherence value of the non-boundary vertices is always 1.

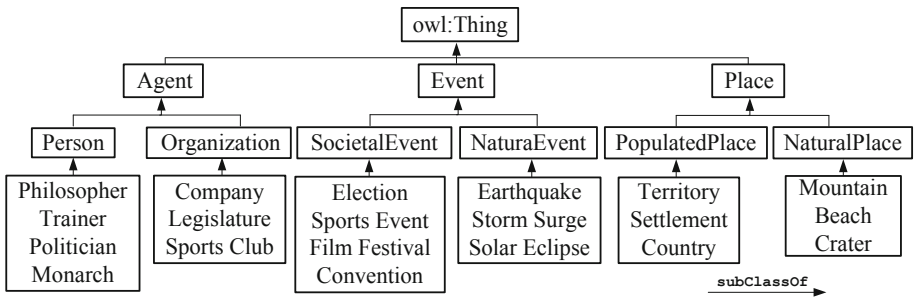


Fig. 3. Ontology hierarchical structure

**Definition 4 (Semantic-aware partitioning).** Given a graph  $G$ , a semantic-aware partitioning is a balanced graph partitioning while maximizing the semantic coherence of  $G$ , the objective function is as follows:

$$O(\mathcal{P}) = \arg \max \frac{1}{|V|} \sum_{v_i \in V_t, P_t \in \mathcal{P}} SC(v_i, P_t) \tag{3}$$

### 3.2 Adaptive Semantic Similarity

Ontology is a structured description of the existing knowledge. It can be described in terms of a tree, where the nodes represent classes, and the edges represent the relationships between classes. An example ontology hierarchical structure, which is formed by transitivity of the property `rdfs:subClassOf`, is extracted from DBpedia, as shown in Fig. 3.

On top of ontology structure, distance-based semantic similarity, information-based similarity, and featured-based semantic similarity are leveraged as three similarity measurements to present the definition of the adaptive semantic similarity.

**Distance-Based Semantic Similarity.** Inspired by [13], the distance-based semantic similarity of two classes is positively related to the depth of their least common superclass and inversely related to their respective depths. The distance-based semantic similarity between vertex  $u$  and  $v$  can be defined as follows,



$$Sim_D(u, v) = \frac{2H(LCS(x, y))}{H(x) + H(y) + 2H(LCS(x, y))} \quad (4)$$

where  $x$  represents the class of  $u$ ,  $y$  represents the class of  $v$ ,  $LCS(x, y)$  represents the least common superclass of  $x$  and  $y$ ,  $H(x)$  and  $H(y)$  refer to the path length from  $x$  and  $y$  to  $LCS(x, y)$ , respectively, and  $H(LCS(x, y))$  is the path length from  $LCS(x, y)$  to the root node. If the least common superclass of two classes is the root node, the distance-based semantic similarity should be zero.

**Information-Based Semantic Similarity.** Since each class is a reification of its superclass in ontology structure, the amount of information contained in the class increases as its depth in the tree structure increases. Motivated by [14], the amount of information contained in the class  $x$  can be defined as follows,

$$I(x) = -\log(p(x)) \quad (5)$$

where  $p(x)$  represents the ratio of the number of the subclasses of class  $x$  to the number of all classes in the ontology structure. Therefore, the information-based semantic similarity between vertex  $u$  and  $v$  can be defined as follows.

$$Sim_I(u, v) = \frac{2I(LCS(x, y))}{I(x) + I(y)} \quad (6)$$

**Feature-Based Semantic Similarity.** Inspired by [15], the feature-based approach leverages the number of common properties of classes to calculate the semantic similarity. The feature-based semantic similarity between vertex  $u$  and  $v$  is given as follows,

$$p = \frac{\min(H(x), H(y))}{H(x) + H(y)} \quad (7)$$

$$Sim_F(u, v) = \frac{|\psi(x) \cap \psi(y)|}{|\psi(x) \cap \psi(y)| + p|\psi(x)/\psi(y)| + (1-p)|\psi(y)/\psi(x)|} \quad (8)$$

where  $p$  is a function of the depth of  $x$  and  $y$ , and  $\psi(x)$ ,  $\psi(y)$  corresponds to the size of properties set contained in class  $x$  and  $y$ , respectively.

**Adaptive Semantic Similarity.** The definition of adaptive semantic similarity has three coefficients  $\alpha$  ( $0 \leq \alpha \leq 1$ ),  $\beta$  ( $0 \leq \beta \leq 1$ ), and  $\gamma$  ( $0 \leq \gamma \leq 1$ ) that satisfy  $\alpha + \beta + \gamma = 1$ . The adaptive semantic similarity of two vertices  $u$  and  $v$  can be calculated as follows.

$$Sim(u, v) = \alpha Sim_D(u, v) + \beta Sim_I(u, v) + \gamma Sim_F(u, v) \quad (9)$$

### 3.3 Initial Partitioning

In order to balance data distribution, two streaming graph partitioning strategies are adopted as our initial partitioning algorithms. Since hash partitioning is probably the most common method supporting load balancing for distributed RDF graph stores, a consistent hashing function is used to assign vertices with unique identifiers to a physical partition index  $V \rightarrow \mathbb{N}$  uniformly at random. The pseudocode of hash partitioning is described in Algorithm 1.

**Algorithm 1: HASHPARTITIONING****Input:** An RDF graph  $G = (V, E)$ , the number of partitions  $k$ **Output:** A partition  $\mathcal{P} = \{P_1, \dots, P_k\}$  of  $G$ 


---

```

1 foreach  $v_i \in V$  do
2   if  $v_i$  has not been assigned then
3      $num \leftarrow getSerialNumber(v_i)$ 
4      $t \leftarrow hash(num) \bmod k$ 
5      $P_t \leftarrow P_t \cup \{v_i\}$                                 /* Assign  $v_i$  to the  $P_t$  */
6 return  $\mathcal{P} = \{P_1, \dots, P_k\}$ 

```

---

Since the hash partitioning traverses each vertex in RDF graph  $G$ , the time complexity of Algorithm 1 is  $O(|V|)$ . Hash partitioning is quite simple and requires no prior knowledge of graph structure, which significantly broadens its applicability to large-scale graphs. However, the randomness of hash partitioning may hinder the subsequent semantic partitioning.

**Algorithm 2: TYPEPARTITIONING****Input:** An RDF graph  $G$ , the number of partitions  $k$ , the set of type sets  $\mathcal{T}$ **Output:** A partition  $\mathcal{P} = \{P_1, \dots, P_k\}$  of  $G$ 


---

```

1  $\mathcal{T} = \emptyset$ 
2 foreach  $v_i \in V$  do
3    $T \leftarrow getType(v_i)$                                 /* Obtain the type of  $v_i$  */
4   if  $T \in \mathcal{T}$  then
5      $\mathcal{T} \leftarrow \mathcal{T} \cup \{v_i\}$                             /* Add  $v_i$  to the set of type  $T$  */
6   else
7      $\mathcal{T} \leftarrow \mathcal{T} \cup \{v_i\}$ 
8      $\mathcal{T} \leftarrow \mathcal{T} \cup \{T\}$                             /* Add type  $T$  to the set of type sets */
9 for  $j \leftarrow 1$  to  $k$  do
10  foreach  $T \in \mathcal{T}$  do
11    if  $|V_j| + |T| \leq \frac{|V|}{k}$  then                            /*  $P_j$  has enough capacity */
12       $P_j \leftarrow P_j \cup T$                                 /* Assign vertices of type  $T$  to  $P_j$  */
13       $\mathcal{T} \leftarrow \mathcal{T} \setminus \{T\}$ 
14    else
15       $cap \leftarrow \frac{|V|}{k} - |V_j|$                             /* Calculate remaining capacity */
16       $T' \leftarrow getVertices(T, cap)$ 
17       $T \leftarrow T \setminus T'$ 
18       $P_j \leftarrow P_j \cup T'$                                 /* Assign the subset of  $T$  to  $P_j$  */
19      break
20 return  $\mathcal{P} = \{P_1, \dots, P_k\}$ 

```

---

Therefore, we introduce type partitioning algorithm, gathering vertices of the same type in the same partition as much as possible. The pseudocode of type partitioning is shown in Algorithm 2. The input of algorithm is an RDF graph  $G$ , the number of partitions  $k$ , and the set of type sets  $\mathcal{T} = \{T_1, \dots, T_m\}$ , where  $T_i$  is a type set of vertices of type  $T_i$ .

Algorithm 2 consists of two stages: (1) construct the set of type sets  $\mathcal{T} = \{T_1, \dots, T_m\}$  and assign each vertex  $v_i$  in graph  $G$  to its type set  $T$  (line 2–8); (2) assign each type set  $T$  to a appropriate partition while enforcing constraint to satisfy balanced graph partitioning (line 9–13), if the size of type set  $T$  exceeds the remaining capacity in the current partition  $P_j$ , the additional vertices in  $T$  will be assigned to the next partition (line 14–19).

Since one of the basic pursuits of semantic-aware partitioning is to maximize the semantic coherence of the given RDF graph, the type partitioning algorithm is supposed to perform better than hash partitioning through gathering semantic-related vertices in the same partition. In fact, this hypothesis is validated in the subsequent experiments.

### 3.4 The OntoSP Algorithm

An elaborate data partitioning algorithm, ONTOSP, which is capable of leveraging the inherent semantics and maximizing the semantic coherence, is presented. The pseudocode of our ONTOSP scheme, which can be divided into the computing phases and the repartitioning phase, is demonstrated in Algorithm 3.

In the computing phase, the destination partition of each vertex in the given RDF graph  $G$  is obtained. For each  $v_i$  in RDF graph  $G$ , *getPartition* function is invoked to retrieve its initial partition  $P_s$  (Line 3), and the semantic coherence of  $v_i$  in current partition  $SC(v_i, P_s)$  can be calculated through Eq. (2) (Line 4).  $d_i$  is used to denote the destination partition index of  $v_i$  and is initialized to  $s$  (Line 5). Then the set of the candidate partitions of  $v_i$  is generated by calling *getCandidatePartition* function (Line 6). Finally, the semantic coherence of  $v_i$  and each its candidate partition  $P_j$  is calculated to find the partition  $P_{d_i}$  where the maximum of  $SC(v_i, P_j)$  is obtained. Meanwhile, the constrains are enforced to satisfy *balanced graph partitioning* (Line 7–9).

In the repartitioning phase, vertices will be reassigned to their destination partitions according to the results in the computing phase to guarantee the maximum semantic coherence of the whole RDF graph and achieve *semantic-aware partitioning*. Concretely, some vertices will be moved from partitions with lower index to partitions with higher index in odd-numbered iterations (Line 10–15) while the other vertices will be moved from partitions with higher index to partitions with lower index in even-numbered iterations (Line 16–21).

**Algorithm 3: ONTOSP**


---

**Input:** An RDF graph  $G$ , the initial partition  $\mathcal{P} = \{P_1, \dots, P_k\}$  of  $G$ , the number of iterations  $\delta$ , the balance factor  $\theta$

**Output:** The semantic-aware partition  $\mathcal{P}$

```

1 for iter  $\leftarrow$  1 to  $\delta$  do
2   foreach  $v_i \in V$  do
3      $P_s \leftarrow \text{getPartition}(v_i)$ 
4      $w_i \leftarrow SC(v_i, P_s)$  /* Calculate the semantic coherence of  $v_i$  in the
       current partition */
5      $d_i \leftarrow s$  /*  $d_i$  is the destination partition of  $v_i$  */
6     foreach  $P_j \in \text{getCandidatePartition}(v_i)$  do
7       if  $SC(v_i, P_j) > w_i$  and  $|V_s| > (1 - \theta)\frac{|V|}{k}$  and  $|V_j| < (1 + \theta)\frac{|V|}{k}$  then
8          $w_i \leftarrow SC(v_i, P_j)$ 
9          $d_i \leftarrow j$ 
10    if iter mod 2 = 1 then
11      foreach  $v_i \in V$  do
12         $P_s \leftarrow \text{getPartition}(v_i)$ 
13        if  $d_i > s$  then /* Move vertex to destination partition with
           higher index */
14           $P_{d_i} \leftarrow P_{d_i} \cup \{v_i\}$ 
15           $P_s \leftarrow P_s \setminus \{v_i\}$ 
16      else
17        foreach  $v_i \in V$  do
18           $P_s \leftarrow \text{getPartition}(v_i)$ 
19          if  $d_i < s$  then
20             $P_{d_i} \leftarrow P_{d_i} \cup \{v_i\}$ 
21             $P_s \leftarrow P_s \setminus \{v_i\}$ 
22 return  $\mathcal{P}$ 

```

---

**Theorem 1.** *Given an RDF graph  $G$ , and its initial partition  $\mathcal{P} = \{P_1, \dots, P_k\}$ . The time complexity of Algorithm 3 is bounded by  $O(k\delta|V|)$ , where  $\delta$  is the number of iterations,  $k$  is the number partitions, and  $|V|$  is the number of vertices in RDF graph  $G$ .*

*Proof.* (Sketch) In Algorithm 3, the time complexity of each iteration consists of two parts: (1) in the worst case, the time complexity of the computing phase is  $(k-1)|V|$ , that is, neighbors of each vertex are distributed in all partitions; (2) the time complexity of the repartitioning phase is  $|V|$ . Therefore, the time complexity of Algorithm 3 is bounded by  $O(k\delta|V|)$   $\square$

## 4 Experiments

In this section, extensive experiments were conducted to evaluate the performance of ONTOSP algorithm. Experiments were carried out on a machine which has 4-core, Intel i5-9300H CPU system, with 8GB of memory, running 64-bit Ubuntu 16.04.

We evaluated two versions of our ONTOSP algorithm, i.e., ONTOSP with hash partitioning as the initial partitioning and ONTOSP algorithm with type partitioning as the initial partitioning against METIS and hash partitioning.

### 4.1 Datasets

To evaluate the effectiveness and efficiency of our algorithm, we adopted the RDF dataset DBpedia [1] in our experiments. DBpedia is a real-world dataset extracted from Wikipedia. The characteristics of the datasets in our experiments are listed in Table 1. Due to the absence of query templates on DBpedia, we designed three queries, including one star query ( $Q1$ ) and two complex queries ( $Q2$ ,  $Q3$ ). The chosen queries are listed in Appendix.

**Table 1.** Characteristics of datasets

Dataset	$ V $	$ E $
DBpedia1	1,528,604	3,718,220
DBpedia2	2,859,191	7,598,934
DBpedia3	4,256,892	11,018,249

### 4.2 Parameter Settings

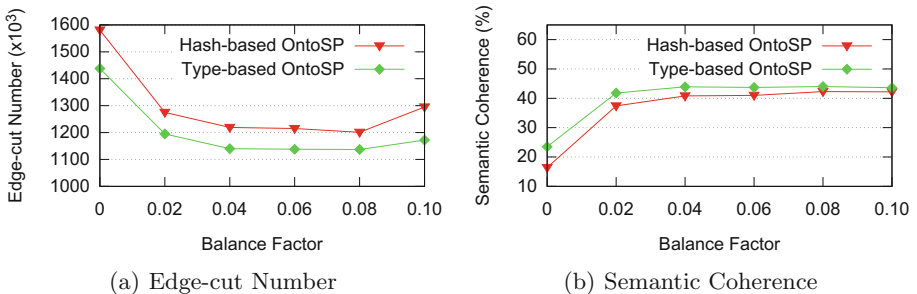
**Performance with Different Semantic Coefficients.** We conducted the hash-based ONTOSP algorithm and the type-based ONTOSP algorithm with different parameter settings. Experiments show that a rapid decline in partitioning performance is occurred while the coefficients have significant difference. Therefore, the experiments were carried out on DBpedia1 by setting the number of iterations  $\delta$  to be 4 and the balance factor  $\theta$  to be 0.04, varying three coefficients of the adaptive semantic similarity from 0.2 to 0.4, as shown in Table 2. It is obvious that slight different semantic coefficients have significant impact on the partitioning results. Finally, we set up  $\alpha$ ,  $\beta$ , and  $\gamma$  as 0.4, 0.3, and 0.3 respectively to achieve lower edge-cut number and higher semantic coherence in the subsequent experiments.

**Table 2.** The results of changing similarity parameters

$\alpha$	$\beta$	$\gamma$	Hash-based OntoSP		Type-based OntoSP	
			Edge-cut	Semantic coherence	Edge-cut	Semantic coherence
0.2	0.4	0.4	1,218,702	0.422	1,128,819	0.442
0.3	0.3	0.4	1,191,357	0.379	1,123,217	0.446
0.3	0.4	0.3	1,229,644	0.369	1,143,537	0.444
0.4	0.2	0.4	1,218,665	0.419	1,127,257	0.443
0.4	0.3	0.3	1,174,954	0.419	1,123,411	0.449
0.4	0.4	0.2	1,182,325	0.424	1,113,952	0.420

**Performance with Different Balance Factors.** The experimental results of different balance factors are shown in Fig. 4. Note that the data distribution is completely balanced in the extreme case, i.e.,  $\theta = 0$ , meaning that vertices cannot be moved in the the repartitioning phase. For balance factor larger than 0.04, the difference of edge-cut number (resp. semantic coherence) is limited to 7.83% (resp. 1.30%). Therefore, the value of balance factor is set to be 0.04 to maintain load balance.

In addition, Fig. 4 shows the effects of different initial partitioning techniques in term of the edge-cut number and the semantic coherence. Typed-based ONTOSP undoubtedly achieves higher semantic coherence and lower edge-cut number than hash-based ONTOSP for all balance factor, which is not counterintuitive.

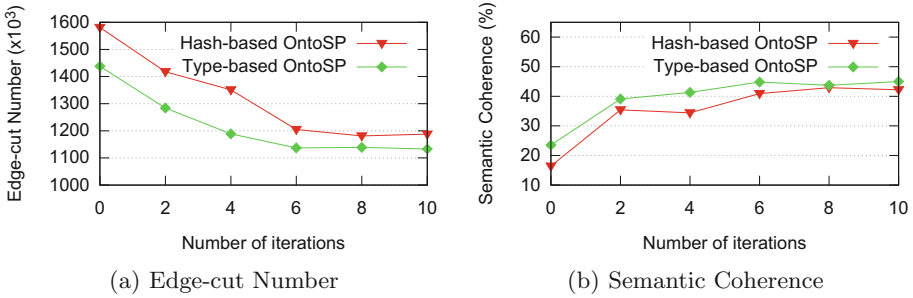


**Fig. 4.** (a) The edge-cut number and (b) the semantic coherence of hash-based ONTOSP and type-based ONTOSP with the increase of balance factor.

**Performance with Different Iteration Numbers.** The experimental results of different iteration numbers are demonstrated in Fig. 5. When the number of iterations  $\delta$  is 0, only the initial partitioning algorithm is performed. For iteration number larger than 6, the difference of edge-cut number (resp. semantic coherence) is limited to 1.99% (resp. 2.00%). According to the experimental results, we suggest to set the number of iterations to be 6 for less execution time.

Similarly, the typed-based ONTOSP outperforms hash-based ONTOSP by a large margin for all iteration numbers. Especially in the extreme case, i.e.,  $\delta =$

0, hash-based ONTOSP suffers from high edge-cut number and extremely low semantic coherence.



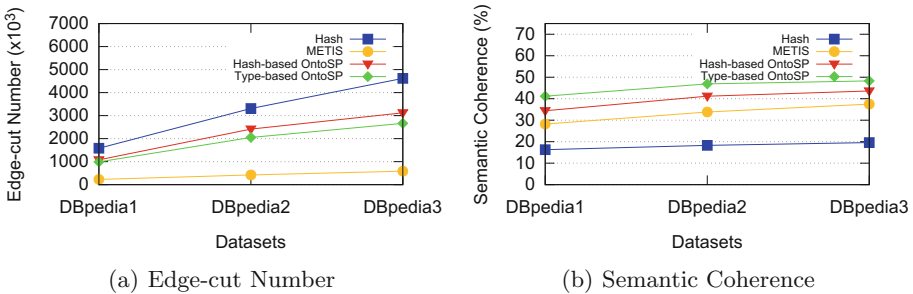
**Fig. 5.** (a) The edge-cut number and (b) the semantic coherence of hash-based ONTOSP and type-based ONTOSP with the increase of iteration number.

### 4.3 Scalability

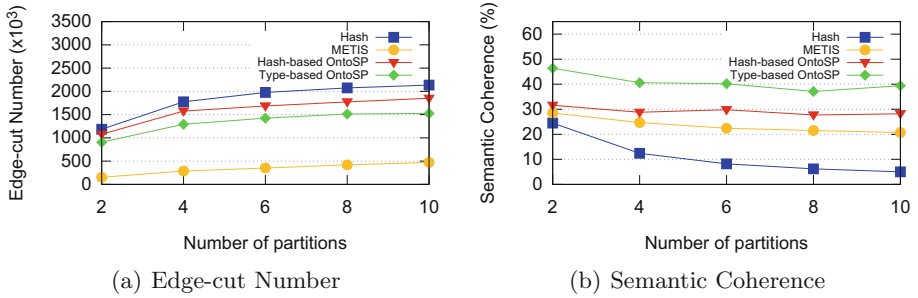
In this section, we evaluated the scalability of our partitioning approach by varying dataset sizes and the number of partitions  $k$ .

**Vary Size of Datasets.** We conducted experiments on three DBpedia datasets with  $\alpha = 0.4$ ,  $\beta = 0.3$ ,  $\gamma = 0.3$ ,  $\theta = 0.04$ , the number of partitions  $k = 3$ , and the number of iterations  $\delta = 6$ . When changing the size of datasets from DBpedia1 to DBpedia3, the edge-cut numbers of all four methods increase, as shown in Fig. 6(a). Compared with the hash partitioning, the edge-cut number of type-based ONTOSP is reduced from 37.44% to 42.29%. The reason why our algorithms worse than METIS is that we sacrifice part of the edge-cut number in exchange for higher semantic coherence.

The semantic coherence of ONTOSP is 2 times higher than that of the hash partitioning, as shown in Fig. 6(b). Since the semantic information is leveraged in the repartitioning phase, type-based ONTOSP achieves better performance than hash-based ONTOSP.



**Fig. 6.** (a) The edge-cut number and (b) the semantic coherence of Hash, METIS, hash-based ONTOSP, and type-based ONTOSP with several datasets.



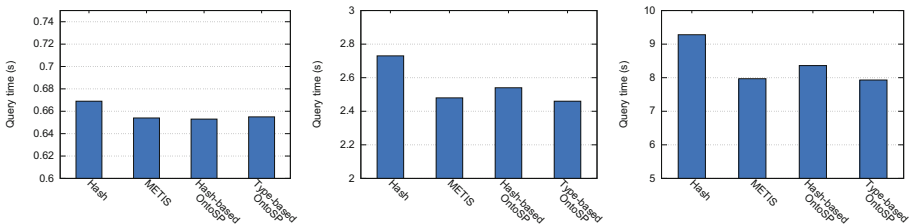
**Fig. 7.** (a) The edge-cut number and (b) the semantic coherence of Hash, METIS, hash-based ONTOSP, and type-based ONTOSP with the increase of partition numbers.

**Vary Number of Partitions.** The sensitivity to the number of partitions is verified on DBpedia1 by varying  $k$  from 2 to 10, as shown in Fig. 7(a). The experimental results verify our intuition that the edge-cut numbers of all these four methods increase as the number of partitions increases. The edge-cut number of the type-based ONTOSP is about 26.89% less than the hash partitioning.

The results of semantic coherence is shown in Fig. 7(b), while hash partitioning and METIS performs poorly for a high number of partitions, type-based ONTOSP is always the best one. In addition, the semantic coherence of hash partitioning significantly decreases with the increase of the number of partitions, while that of the type-based ONTOSP changes slightly.

### 4.4 Efficiency

We conducted experiments on DBpedia3 dataset over three queries, including one linear query  $Q1$  and two complex queries  $Q2$  and  $Q3$ . As shown in Fig. 8, the query time of various algorithms over linear query is almost the same. However, type-based ONTOSP demonstrates the best query efficiency with the increase of the query complexity. The growth rate of query time in hashing partitioning is much higher than that of METIS and our ONTOSP.



**Fig. 8.** The experimental results over  $Q1$ ,  $Q2$ , and  $Q3$  on DBpedia dataset.



## 5 Conclusion

In this paper, we first present the definition of semantic coherence, which is a novel metrics for graph partitioning. Moreover, the ONTOSP algorithm is proposed to achieve balanced data distribution and maximize the semantic coherence of RDF graphs. Finally, extensive experimental results show the scalability and efficiency of our method, which outperforms the state-of-the-art RDF graph partitioning methods by a large margin.

**Acknowledgments.** This work is supported by the National Key Research and Development Program of China (2019YFE0198600) and National Natural Science Foundation of China (61972275).

## 6 Appendix

### 6.1 Queries for DBpedia

```

Q1: SELECT ?X ?Y ?Z WHERE {
    ?X <http://www.w3.org/2020/01/rdf-schema#seeAlso> ?Y.
    ?X <http://dbpedia.org/ontology/deathPlace> ?Z.};
Q2: SELECT ?X1 ?X2 ?Y1 ?Y2 ?Z WHERE {
    ?X1 <http://dbpedia.org/ontology/philosophicalSchool> ?Y1.
    ?Y1 <http://www.w3.org/2000/01/rdf-schema#seeAlso> ?Z.
    ?X1 <http://dbpedia.org/ontology/era> ?Y2.
    ?X2 <http://dbpedia.org/ontology/mainInterest> ?Z.};
Q3: SELECT ?X1 ?X2 ?Y1 ?Y2 ?Z WHERE {
    ?X1 <http://dbpedia.org/ontology/philosophicalSchool> ?Y1.
    ?Y1 <http://www.w3.org/2000/01/rdf-schema#seeAlso> ?Z.
    ?X2 <http://dbpedia.org/ontology/mainInterest> ?Z.
    ?X1 <http://dbpedia.org/ontology/mainInterest> ?Y2.
    ?Y2 <http://dbpedia.org/ontology/field> ?Z.};

```

## References

1. Lehmann, J., et al.: DBpedia-a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web* **6**(2), 167–195 (2015)
2. Wang, X., et al.: Efficient subgraph matching on large RDF graphs using MapReduce. *Data Sci. Eng.* **4**, 1–20 (2019). <https://doi.org/10.1007/s41019-019-0090-z>
3. Zeng, K., Yang, J., Wang, H., Shao, B., Wang, Z.: A distributed graph engine for web scale rdf data. *Proc. VLDB Endow.* **6**(4), 265–276 (2013)
4. Karypis, G., Kumar, V.: Metis-unstructured graph partitioning and sparse matrix ordering system, version 2.0 (1995)
5. Lee, K., Liu, L.: Scaling queries over big RDF graphs with semantic hash partitioning. *Proc. VLDB Endow.* **6**(14), 1894–1905 (2013)

6. Stanton, I., Kliot, G.: Streaming graph partitioning for large distributed graphs. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1222–1230 (2012)
7. Harth, A., Umbrich, J., Hogan, A., Decker, S., et al.: YARS2: a federated repository for querying graph structured data from the web. In: Aberer, K. (ed.) ASWC/ISWC -2007. LNCS, vol. 4825, pp. 211–224. Springer, Heidelberg (2007). [https://doi.org/10.1007/978-3-540-76298-0\\_16](https://doi.org/10.1007/978-3-540-76298-0_16)
8. Ladwig, G., Harth, A.: CumulusRDF: linked data management on nested key-value stores. In: The 7th International Workshop on Scalable Semantic Web Knowledge Base Systems (2011)
9. Malewicz, G., et al.: Pregel: a system for large-scale graph processing. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 135–146. ACM (2010)
10. Kyrola, A., Btleloch, G.E., Guestrin, C.: GraphChi: large-scale graph computation on just a PC. In: 10th USENIX Symposium on Operating Systems Design and Implementation, pp. 31–46. USENIX Association (2012)
11. Lee, K., Liu, L.: Efficient data partitioning model for heterogeneous graphs in the cloud. In: International Conference for High Performance Computing, Networking, Storage and Analysis. ACM (2013)
12. Xu, Q., Wang, X., Wang, J., Yang, Y., Feng, Z.: Semantic-aware partitioning on RDF graphs. In: Chen, L., Jensen, C.S., Shahabi, C., Yang, X., Lian, X. (eds.) APWeb-WAIM 2017, Part I. LNCS, vol. 10366, pp. 149–157. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-63579-8\\_12](https://doi.org/10.1007/978-3-319-63579-8_12)
13. Wu, Z., Palmer, M.: Verbs semantics and lexical selection. In: Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics, pp. 133–138. Association for Computational Linguistics (1994)
14. Lin, D.: An information-theoretic definition of similarity. In: Proceedings of the Fifteenth International Conference on Machine Learning, pp. 296–304. Morgan Kaufmann Publishers Inc. (1998)
15. Tversky, A.: Features of similarity. *Psychol. Rev.* **84**(4), 327 (1977)



# Optimal Subgraph Matching Queries over Distributed Knowledge Graphs Based on Partial Evaluation

Jiao Xing<sup>1</sup>, Baozhu Liu<sup>1</sup>, Jianxin Li<sup>2</sup>, Farhana Murtaza Choudhury<sup>3</sup>,  
and Xin Wang<sup>1</sup>✉

<sup>1</sup> Tianjin International Engineering Institute, Tianjin University, Tianjin, China  
{jiaoxing, liubaozhu, wangx}@tju.edu.cn

<sup>2</sup> School of Information Technology, Deakin University, Melbourne, Australia  
jianxin.li@deakin.edu.au

<sup>3</sup> School of Computing and Information Systems, The University of Melbourne,  
Melbourne, Australia  
farhana.choudhury@unimelb.edu.au

**Abstract.** The *partial evaluation and assembly* framework has recently been applied for processing subgraph matching queries over large-scale knowledge graphs in the distributed environment. The framework is implemented on the master-slave architecture, endowed with outstanding scalability. However, there exists a drawback of partial evaluation, that if the volume of intermediate results is large, assembly computation which is handled by the master would be a bottleneck. In this paper, we propose an efficient assembly algorithm by exploring the characteristics of intermediate results generated during the partial matching stage in each fragment. (1) The partial matching index (PM-Index) structure are constructed by utilizing the *incoming* and *outgoing* vertices in parallel to improve the searching efficiency of the assembling phase. (2) The time and space complexity of PM-Index construction are analysed. (3) The experimental results over benchmark datasets show that our approach outperforms the state-of-the-art methods.

**Keywords:** Partial evaluation · Subgraph matching · RDF graph

## 1 Introduction

In the Semantic Web community, the *Resource Description Framework* (RDF) becomes a de facto standard format for knowledge graphs and has been extensively applied [16]. An RDF dataset consists of sets of triples  $(s, p, o)$  and can be transformed into a graph where the resources denoted by  $s$  and  $o$  are vertices, and the attributes denoted by  $p$  are labeled edges. SPARQL [5] is the standard graph query language on RDF graphs. A SPARQL query can be regarded as the

---

J. Xing and B. Liu—These authors contributed equally to this work and should be considered co-first authors.

subgraph homomorphism problem [10], which is recognized as an NP-complete problem [1, 11].

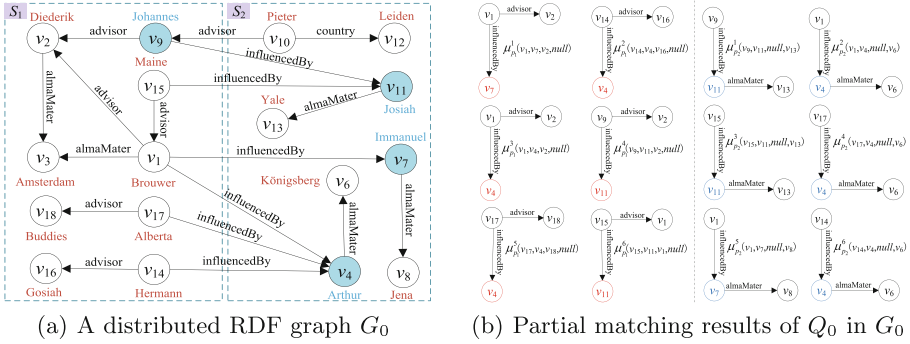
The efficient processing of subgraph matching queries over large-scale RDF graphs in a distributed setting is a widely-known challenging problem. Recently, the *partial evaluation* technique [7] has been applied to solve the problem of regular path queries on distributed environment [3, 13, 15]. The queries  $Q$  are partially evaluated in parallel to obtain partial results on each fragment of data  $F_i$  on each site  $S_i$ , then send all the partial results to a coordinator site. Finally, assemble these partial results to get the final results to  $Q$ .

Based on *partial evaluation* technique, the *partial evaluation and assembly* framework has been proposed to answer SPARQL queries [9]. However, a large number of partial results can be generated during the local computation phase, making assemble phase a computation bottleneck. To improve the efficiency of assemble phase, Peng et al. [8] proposed the LEC feature-based optimization strategy to prune some unpromising intermediate results. However, existing works do not establish an effective index structure during the assembly phase to speed up the search process. The following example demonstrates the drawback of computing partial matching results by the method in [8].

*Example 1.* As shown in Fig. 1(a), given a distributed RDF graph  $G_0$ , and a query  $Q_0 = (?a, \text{influencedBy}, ?b) \wedge (?a, \text{advisor}, ?d) \wedge (?b, \text{almaMater}, ?e)$ , all the partial matching results in Fig. 1(b) can be obtained by fragments. In fragment  $F_i$ , a partial matching result is devoted as  $\mu_{p_i}^j$ , where the superscript  $j$  is used to distinguish among multiple partial matching results from the same site. According to the grouping rules mentioned in [8], in the master, the partial matching results can be partitioned into two groups, i.e.,  $Gr_1 = \{\mu_{p_1}^1, \mu_{p_1}^2, \mu_{p_1}^3, \mu_{p_1}^4, \mu_{p_1}^5, \mu_{p_1}^6\}$ ,  $Gr_2 = \{\mu_{p_2}^1, \mu_{p_2}^2, \mu_{p_2}^3, \mu_{p_2}^4, \mu_{p_2}^5, \mu_{p_2}^6\}$ . The search process is performed  $6 \times 6$  times.

We propose an effective optimization strategy to tackle the computing bottleneck problem on assemble phase, which utilizes the *outgoing* vertex and *incoming* vertex (the formal definition will be explained in detail in Sect. 3) to construct a partial matching index (PM-Index) structure in parallel, speeding up the searching efficiency of partial matching results. PM-Index is essentially a key-value map, where each key  $v$  is either an *incoming* or an *outgoing* vertex, and the value  $PMs$  is the set of partial matching results including  $v$ . Each fragment  $F_i$  maintains a such key-value map for its necessary partial matching results. Specifically, the map is denoted as  $I_i^{in}$  ( $I_i^{out}$ ) if  $v$  is an *incoming* (*outgoing*) vertex of Fragment  $F_i$ . The value  $PMs$  of the mapping  $(v, PMs) \in I_i^{in}$  (or  $I_i^{out}$ ) is the set of partial matching results including  $v$ .

For the example shown in Fig. 1, we can get the following mappings by PM-Index:  $I_1^{out} = \{(v_4, \langle \mu_{p_1}^2, \mu_{p_1}^3, \mu_{p_1}^5 \rangle), (v_7, \langle \mu_{p_1}^1 \rangle), (v_{11}, \langle \mu_{p_1}^4, \mu_{p_1}^6 \rangle)\}$ ,  $I_2^{in} = \{(v_4, \langle \mu_{p_2}^2, \mu_{p_2}^4, \mu_{p_2}^6 \rangle), (v_7, \langle \mu_{p_2}^5 \rangle), (v_{11}, \langle \mu_{p_2}^1, \mu_{p_2}^3 \rangle)\}$ . The search space consists of two parts, (a) for each  $v$  in the key part of  $I_2^{in}$ , it needs to find the same vertex key in  $I_1^{out}$ , and (b) the partial matching results including  $v$  will be assembled. Therefore, the time cost is  $3\log(3) + 14$ , which is much smaller than 36 (the cost incurred by the method mentioned in [8]).



**Fig. 1.** An example of RDF graph and partial matching results

We summarize the contributions of the paper with the following three aspects:

- We propose an effective partial matching index (PM-Index) based on the *partial evaluation and assembly* framework, which can significantly speed up the assembling phase.
- To prove PM-Index is both space-efficient and time-efficient, the time and space complexity of the proposed method is analyzed.
- Extensive experiments on benchmark datasets have been conducted to verify the efficiency and scalability of our method. The experimental results show that our method outperforms the state-of-the-art method.

The rest of this paper is organised as follows. Section 2 reviews the related work. In Sect. 3, we present the preliminaries and problem definition. In Sect. 4, we propose the PM-Index and the index-based query processing algorithm. Section 5 shows the experimental results. Finally, we conclude in Sect. 6.

## 2 Related Work

Due to the factors of performance, confidentiality and security, the cluster-based distributed data management architecture has become the inevitable research trend to deal with the knowledge graph. In this section, we will review several distributed subgraph matching research on large-scale RDF graphs, which can be classified into two categories, including MapReduce-based graph systems, and specialized RDF systems.

### 2.1 MapReduce-Based Graph Systems

SHARD [12], a MapReduce-based triple store for RDF graphs, is able to process SPARQL queries, which decomposes the query graph into a set of triples

(a triplet containing variables) and binds variables to the vertices of the data graph by iterating on the triple pattern. Meanwhile, it is necessary to satisfy all the constraints in the query. Each round of the MapReduce operation adds only one query clause through the join operation. Likewise, The smallest decomposition unit of the query graph in HadoopRDF [6] is also the triple pattern, and it also utilizes the MapReduce to divide the RDF triples into multiple small files based on the predicate. However, aforementioned two methods both ignore the structural information of the query graph, require multiple MapReduce iterations, and a significant number of join transactions are required, leading to high query costs.

## 2.2 Specialized RDF Systems

Trinity.RDF [17], a distributed in-memory key-value store, stores RDF data in the native form, with vertex identifiers as keys and adjacent lists of vertices as values. Trinity.RDF will find the optimal exploration plan and reduce the number of intermediate results using the graph *exploration* instead of join operations, while the final results need to be obtained using a single thread on the master node. In addition, Peng et al. [9] proposed a distributed framework of local computation and assembly to execute SPARQL queries based on gStore [18]. In the local computation stage, each node in the cluster will process the complete query in parallel, and then in the assembly stage, a large number of local matching results are sent to the master node to obtain the final results by join operation. When the number of intermediate results obtained in the local computation phrase is extensive, aforementioned two methods may arise a performance bottleneck in the assembly stage. In this paper, we improve the efficiency of the assembly phase by constructing the structure called PM-index, which will significantly tackle the problem of computing bottlenecks.

## 3 Preliminaries and Problem Statement

Let  $U$  and  $L$  be the disjoint infinite sets of URIs and literals, respectively. A tuple in the form of  $(s, p, o) \in U \times U \times (U \cup L)$  is called an RDF triple, where  $s$  is the subject,  $p$  the predicate, and  $o$  the object, Given an RDF dataset as a finite set of triples, it can be converted to its corresponding RDF graph. In this paper, we focus on the problem of subgraph matching query over distributed RDF graph. In this section, we will present some preliminaries for distributed RDF graphs and subgraph matching queries.

**Definition 1 (RDF Graph).** *Given an RDF dataset as a finite set of triples in the form  $(s, p, o)$ , its corresponding RDF graph is  $G = \{V, E, \Sigma\}$ , where the set of vertices  $V$  is the union of all  $s$  and  $o$ . For each  $(s, p, o)$ , there is a directed edge  $e \in E$  from the vertex  $s$  to the vertex  $o$ , where  $p$  is the label of that edge  $e$ . Here,  $\Sigma$  is the set of all labels, i.e.,  $\Sigma = \{p \mid (s, p, o) \in G\}$ .*

**Definition 2 (Distributed RDF Graph).** *RDF graph  $G$  is partitioned into  $n$  disjoint ‘entity sets’  $\{\mathcal{E}_1, \dots, \mathcal{E}_n\}$ , where each  $\mathcal{E}_i = \{V_i, E_i, \Sigma_i\}$ . Here, (1) For each  $i \in \{1, \dots, n\}$ ,  $\mathcal{E}_i$  is a subset of  $G$ , where  $V_i \subseteq V$ ,  $E_i \subseteq E$ , and  $\Sigma_i \subseteq \Sigma$ ; (2) for each  $i, j \in \{1, \dots, n\} \wedge i \neq j$ , there is  $\mathcal{E}_i \cap \mathcal{E}_j = \emptyset$ , and (3)  $\bigcup_{i=1}^n \mathcal{E}_i = \mathcal{E}(G)$ , i.e.,  $G$ .*

In order to ensure the integrity and consistency of RDF graph when divided in a distributed system, each computing node need to store some copies of edges that are crossing between different entity sets. Let the copy of the associated edges with other partition is denoted as  $N_i^c$ .

**Definition 3 (Fragment).** *Graph  $G$  is partitioned into  $n$  fragments  $\mathcal{F} = \{F_1, \dots, F_n\}$ , such that  $\mathcal{F}_i = \mathcal{E}_i \cup N_i^c$ . In other words,  $G$  can be considered as a distributed RDF graph w.r.t.  $\mathcal{F}$ , such that:*

- 1) For each  $\mathcal{E}_i = \{V_i, E_i, \Sigma_i\}$ ,  $V_i$ ,  $E_i$ , and  $\Sigma_i$  represent the set of internal vertices, the set of edges and the set of labels in  $F_i$  respectively. Formally,  $V_i = \{s | (s, p, o) \in \mathcal{E}_i\} \cup \{o | (s, p, o) \in \mathcal{E}_i\}$ ,  $E_i \subseteq V_i \times V_i$ , and  $\Sigma_i = \{p | (s, p, o) \in \mathcal{E}_i\}$ ;
- 2)  $N_i^c = \{V_i^e, E_i^c, \Sigma_i^c\}$ , where  $E_i^c$  is the set of crossing edges between  $F_i$  and other fragments. If an internal vertex of  $F_i$  has a direct edge with any vertex  $v$  in  $F_j$ , where  $i \neq j$ , then  $v \in V_i^e$ . Formally,  $E_i^c \subseteq V_i \times V_j$ ,  $\Sigma_i^c = \{p | (s, p, o) \in E_i^c\}$ , the set of extended vertices between  $F_i$  and  $F_j$  is  $V_i^e = \{s | (s, p, o) \in E_i^c \wedge s \in V_j\} \cup \{o | (s, p, o) \in E_i^c \wedge o \in V_j\}$ ,  $i, j = 1, 2, \dots, n \wedge i \neq j$ ;

Let  $\mathcal{S} = \{S_0, S_1, \dots, S_n\}$  be a set of  $n + 1$  computing nodes, i.e., *sites*, in a cluster. Without loss of generality, each fragment  $F_i$  is stored at slave site  $S_i$  for  $i \in \{1, \dots, n\}$ . If there is an edge  $e \in E_i^c$  where the direction of  $e$  is from a vertex in  $F_j$  to a vertex  $v$  in  $F_i$ , where  $i \neq j$  then all such  $v$  in  $F_i$  are the *incoming* vertices of  $F_i$  and *outgoing* vertices of  $F_j$ .

*Example 2.* As shown in Fig. 2, given a distributed RDF graph  $G_1$  extracted from the DBpedia dataset,  $G_1$  will be divided into four parts  $\mathcal{F} = \{F_1, F_2, F_3, F_4\}$ , which are respectively stored on the corresponding sites  $\{S_1, S_2, S_3, S_4\}$  in the cluster. For a fragment  $F_2 = \mathcal{E}_2 \cup N_2^c$ , the partition  $\mathcal{E}_2 = \{V_2, E_2, \Sigma_2\}$ , and  $V_2 = \{v_4, v_6, v_7, v_8, v_{10}, v_{11}, v_{12}, v_{13}, v_{16}\}$ ,  $E_2 = \{(v_4, v_6), (v_7, v_8), (v_{10}, v_{12}), (v_{11}, v_{13})\}$ . The copy between  $F_2$  and other fragments  $N_2^c = \{V_2^e, E_2^c, \Sigma_2^c\}$ , where  $V_2^e = \{v_1, v_9, v_{14}, v_{18}, v_{24}\}$ ,  $E_2^c = \{(v_1, v_4), (v_1, v_7), (v_9, v_{10}), (v_9, v_{11}), (v_{14}, v_{16}), (v_{16}, v_{18}), (v_{16}, v_{24})\}$ . In particular, we colored the *incoming* and *outgoing* vertices of each fragment in blue.

Given an RDF graph  $G$  and a query graph  $Q$  as a set of triple patterns, a subgraph matching problem is to find the subgraphs over  $G$  that satisfy all the triple patterns in  $Q$ . Such subgraph matching problem is actually a conjunctive query (CQ) on  $G$ , which is the focus of this paper. In the following we formally present the query graphs and the other necessary definitions that adapted from [14].

A query graph includes  $m$  triple patterns  $(s_r, p_r, o_r)$ , where the value of each  $s_r$ ,  $o_r$  can either be a member of  $V$ , or ‘not labeled’. If a  $s_r$  or  $o_r$  is ‘not labeled’,  $s_r$  or  $o_r$  belongs to a special set  $Var$ , and the name of each element in  $Var$  starts

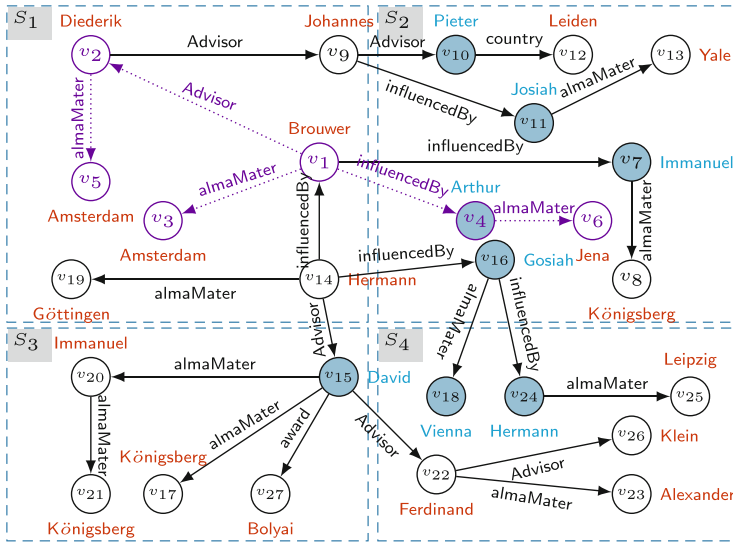


Fig. 2. A distributed RDF graph  $G_1$

with the character ‘?’. Similarly, the value of each  $p_r$  can either be a member of  $\Sigma$ , or  $Var$ .

**Definition 4 (Query Graph).** Given an RDF graph  $G$ , a CQ  $Q$  over  $G$  is defined as:  $Q(z_1, \dots, z_t) \leftarrow \bigwedge_{1 \leq r \leq m} tp_r$ , where  $tp_r = (s_r, p_r, o_r)$  is a triple pattern.  $s_r, o_r \in V \cup Var$ ,  $z_i$  is a variable and  $z_i \in \{s_r | 1 \leq r \leq m\} \cup \{o_r | 1 \leq r \leq m\}$ . A CQ  $Q$  is also referred to as a query graph.

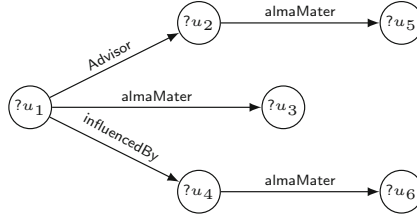
Before defining subgraph matching, we recapitulate certain definitions of the mapping. For a mapping  $\mu$ ,  $dom(\mu)$  is its domain. Two mappings  $\mu_1$  and  $\mu_2$  are compatible, i.e.,  $\mu_1 \sim \mu_2$ , iff for every element  $v \in dom(\mu_1) \cap dom(\mu_2)$ , it holds that  $\mu_1(v) = \mu_2(v)$ . Furthermore, the set-union of two compatible mappings, i.e.,  $\mu_1 \cup \mu_2$ , is also a mapping.

**Definition 5 (Subgraph Matching).** The semantics of a CQ  $Q$  over an RDF graph  $G$  is defined as:

- 1)  $\mu$  is a mapping from the vertices in  $Q$  to the vertices in  $V$ , i.e., mapping from  $\bar{s} = \{s_1, \dots, s_m\}$  and  $\bar{o} = \{o_1, \dots, o_m\}$  to the vertices in  $V$ ;
- 2)  $(G, \mu) \models Q$  iff  $(\mu(s_r), \mu(p_r), \mu(o_r)) \in E$  and the labels of  $s_r, p_r$  and  $o_r$  are the same as that of  $\mu(s_r), \mu(p_r)$  and  $\mu(o_r)$ , respectively, if  $s_r, p_r, o_r \notin Var$ ;
- 3)  $P_Q$  is the set of all results, where each result satisfy the subgraph matching query  $Q$  over  $G$ .

**Problem Statement.** Consider a distributed RDF graph  $G$ , w.r.t., a fragmentation  $\mathcal{F} = \{F_1, \dots, F_n\}$ , and let  $F_i$  store in  $S_i$  in the cluster  $\mathcal{S} = \{S_0, S_1, \dots, S_n\}$ . Given a query graph  $Q$ , the problem is to find all subgraph matching results  $P_Q$  of  $Q$  in  $G$ .





**Fig. 3.** A query graph  $Q_1$

*Example 3.* Given a  $CQ$ ,  $Q_1 = (?u_1, \text{doctoralAdvisor}, ?u_2) \wedge (?u_1, \text{almaMater}, ?u_3) \wedge (?u_1, \text{influencedBy}, ?u_4) \wedge (?u_2, \text{almaMater}, ?u_5) \wedge (?u_4, \text{almaMater}, ?u_6)$ .  $Q_1$  consists of six query vertices and its semantic is to find the schools attended by a person, his advisor, and his influencer. The corresponding query graph is shown in Fig. 3, with one of the query results being highlighted in purple in Fig. 2.

## 4 Partial Matching Index Based Algorithm

Given a distributed RDF graph  $G$ , each site  $S_i$  receives a complete query graph  $Q$ . To answer query  $Q$ , each site  $S_i$  computes a partial matching result, based on a known input  $F_i$ . In this section, we first introduce the partial matching result and joinable results. Then we define the structure of the partial matching index, present the index construction algorithm, and give the complexity analysis of the proposed method.

### 4.1 Partial Matching Index

The definitions of partial matching results and joinable results based on *partial evaluation and assembly* framework were proposed in [9]. Here, we first define partial matching result and joinable results based on the *mapping*. Then we analyze the performance problems of the existing work while assembling large-scale intermediate results and put forward the optimization strategy of partial matching index.

#### Definition 6 (Partial Matching Result).

The semantics of a  $CQ$   $Q$  over a fragment  $F_i$  ( $1 \leq i \leq n$ ) is defined as:

- 1)  $\mu$  is a mapping from vertices in  $\bar{s}$  and  $\bar{o}$  to vertices in  $V_i \cup \{\text{null}\}$ , where  $\bar{s} = (s_1, \dots, s_m)$ ,  $\bar{o} = (o_1, \dots, o_m)$ ;
- 2)  $(\mu(s_r), \mu(p_r), \mu(o_r)) \in E_i$  and the labels of  $s_r$ ,  $p_r$ , and  $o_r$  are the same as that of  $\mu(s_i)$ ,  $\mu(p_i)$ , and  $\mu(o_i)$ , respectively, if  $s_r, p_r, o_r \notin \text{Var}$ ;
- 3)  $\mu(s_r)$  and  $\mu(o_r)$  can also be null, if  $\mu(o_r) = \text{null}$ , then  $\mu(s_r)$  is an extended vertex or  $\mu(s_r)$  is null, w.r.t., there should exist  $\mu(s_r) \in (V_i^e \cup \text{null})$ ;

- 4) *There should exist one crossing edge in  $\mu$  at least, w.r.t.,  $\exists(\mu(s_r), \mu(p_r), \mu(o_r)) \in E_i^c$ . It means that  $\mu(s_r)$  (or  $\mu(o_r)$ ) is an extended vertex and  $\mu(o_r)$  (or  $\mu(s_r)$ ) is an internal vertex, w.r.t.,  $\mu(s_r) \in V_i^e \wedge \mu(o_r) \in V_i$  (or  $\mu(s_r) \in V_i \wedge \mu(o_r) \in V_i^e$ );*
- 5) *If  $\mu(s_r)$  is an internal vertex in  $F_i$ , then  $\mu(o_r)$  cannot be null, w.r.t., if  $\mu(s_r) \in V_i$  (or  $\mu(o_r) \in V_i$ ), there should exist  $\mu(o_r) \neq \text{null}$  (or  $\mu(s_r) \neq \text{null}$ );*
- 6) *Any two vertices  $s \in \bar{s}$  and  $o \in \bar{o}$  (in query  $Q$ ), where  $\mu(s) \in V_i$  and  $\mu(o) \in V_i$ , are weakly connected in  $Q$ .*
- 7) *Then,  $P_{Q,i}$  is the set of the partial matching results of  $Q$  over fragment  $F_i$ .*

In order to distinguish the partial matching result from the complete query result, we denote a partial matching result as  $\mu_p$ . The partial matching results should be joined to obtain the final results. Specifically, the joinable results are defined as follows:

**Definition 7 (Joinable Results).** *Given a query  $Q$ , two fragments  $F_i$  and  $F_j$  ( $i \neq j$ ), let  $P_{Q,i}$  and  $P_{Q,j}$  represent the set of partial matching results in  $F_i$  and  $F_j$ , respectively. For each  $\mu_p \in P_{Q,i}$  and  $\mu'_p \in P_{Q,j}$ ,  $\mu_p$  and  $\mu'_p$  are joinable if the following conditions are satisfied:*

- 1) *In  $\mu_p$  and  $\mu'_p$ , each query vertex should match to the same vertex of data graph, w.r.t., for query vertex  $u$ , if  $\mu_p(u)$  is not null and  $u \in \text{dom}(\mu_p) \cap \text{dom}(\mu'_p)$  satisfies  $\mu_p(u) = \mu'_p(u)$ ;*
- 2) *There exists at least one crossing edge  $(\mu_p(s_r), \mu_p(o_r))$  and  $(\mu'_p(s_r), \mu'_p(o_r))$  such that  $\mu_p(s_r)$  is an internal vertex and  $\mu_p(o_r)$  is an extended vertex in  $F_i$ , while  $\mu'_p(s_r)$  is an extended vertex and  $\mu'_p(o_r)$  is an internal vertex in  $F_j$ . Furthermore,  $\mu_p(s_r) = \mu'_p(s_r)$  and  $\mu_p(o_r) = \mu'_p(o_r)$ ;*
- 3) *Let  $\mu_p \sim \mu'_p$  denote two joinable partial matching results. The joined result  $\mu''_p = \mu_p \cup \mu'_p$ .  $\mu''_p = \bigcup_{r=1}^t ((\mu_p(s_r), \mu_p(o_r)) \sim (\mu'_p(s_r), \mu'_p(o_r)))$ . If  $\mu_p(s_r) = \text{null}$ ,  $\mu_p(s_r) \sim \mu'_p(s_r) = \mu'_p(s_r)$ , otherwise,  $\mu_p(s_r) \sim \mu'_p(s_r) = \mu_p(s_r)$ , and  $\mu_p(o_r) \sim \mu'_p(o_r)$  follows the same rule.*

The correctness of the framework with the above definitions is analysed in [9]. The join methods proposed in [9] and [8] are both partition-based algorithms. In [8], it has been proved that the LEC-based grouping is superior to the grouping method of [9]. Furthermore, it proposes an optimized technique based on the LEC features of the partial matching results. Given a set  $\mathcal{G} = \{Gr_1, \dots, Gr_n\}$  of LEC-based partial matching results, joining costs  $\prod_{i=1}^{i=n} |Gr_i|$ , where  $n$  is the number of groups and  $n$  is  $2^{|Q|}$  at most. Since  $|Q|$  is usually small in practice, grouping has limited improvement when the scale of intermediate results is large. Thus, if each group contains a large number of intermediate results, the efficiency of the assembly algorithm would not be ideal. In order to solve the bottleneck problem, we propose PM-Index to improve the search efficiency in the process of merging and avoid unnecessary consumption cost.

*Example 4.* As shown in Fig. 4, there are several partial matching results in fragmentation  $\mathcal{F}$ , i.e.,  $\mu_{p_1}^3 = (v_{14}, v_{15}, v_{19}, v_1, \text{null}, v_3)$ , and  $\mu_{p_3}^1 = (v_{14}, v_{15}, \text{null},$

$null, v_{17}, null$ ), where  $\mu_{p_1}^3 \sim \mu_{p_3}^1$  and  $\mu_{p_1}^3 \cup \mu_{p_3}^1 = (v_{14}, v_{15}, v_{19}, v_1, v_{17}, v_3)$ . Meanwhile, the outgoing vertices and incoming vertices in each partial matching result are colored red and blue, respectively. In the same fragment  $F_i$ , there may be multiple partial matching results matching to the same *outgoing* vertex in  $F_1$ , such as  $\mu_{p_1}^3$  and  $\mu_{p_1}^4$  are matched to the same outgoing vertex  $v_{15}$ .

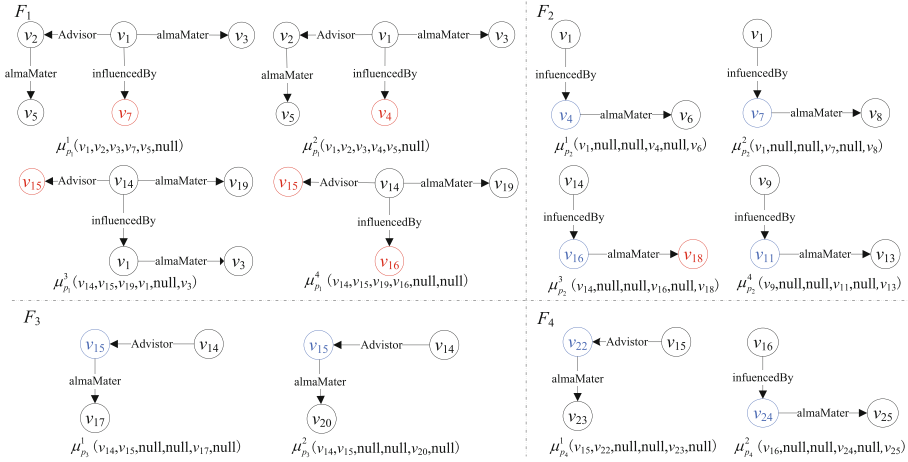


Fig. 4. Partial matching results of  $Q_1$  in  $G_1$

In fragment  $F_i$ , the incoming vertices and outgoing vertices are the subset of the extended vertices (according to Definition 3 in Sect. 3). Therefore, for each partial matching result, there exists an incoming vertex or outgoing vertex  $v \in \mu_p$  at least. To improve the search efficiency, an optimization strategy of partial matching index is proposed, which utilizes the characteristic of the partial matching results that can be joinable. Let  $V_i^I$  denotes the set of incoming vertices and  $V_i^O$  denotes the set of outgoing vertices in  $F_i$ . Formally, we define the partial matching index as follows:

**Definition 8 (Partial Matching Index).** Given a set of partial matching results  $P_{Q,i}$  of  $Q$  in  $F_i$ , the partial matching index, i.e., PM-Index of  $P_{Q,i}$  is a key-value map  $I$  where

- 1) for any tuple  $(v, PMs) \in I$ ,
  - the key is a vertex  $v \in (V_i^I \cup V_i^O)$ .
  - the value  $PMs$  is the set of partial matching results including  $v$ , w.r.t., for each  $\mu_p \in PMs$ , there exists  $v \in \mu_p$ .
- 2) for any vertex  $v \in (V_i^I \cup V_i^O)$ , there exists a unique tuple in  $I$  with  $v$  as the key and  $\mu_p$  including  $v$  in the value.
- 3) if  $v \in V_i^I$  (or  $v \in V_i^O$ ), the map is denoted as  $I_i^{in}$  (or  $I_i^{out}$ ).

*Example 5.* As shown in Fig. 4, consider the partial matching results in  $F_2$ , we have two partial matching indexes  $I_2^{in} = \{(v_4, \langle \mu_{p_2}^1 \rangle), (v_7, \langle \mu_{p_2}^2 \rangle), (v_{16}, \langle \mu_{p_2}^3 \rangle), (v_{11}, \langle \mu_{p_2}^4 \rangle)\}$  and  $I_2^{out} = \{(v_{13}, \langle \mu_{p_2}^3 \rangle), (v_{18}, \langle \mu_{p_2}^4 \rangle)\}$ .

In fragment  $F_i$ , each triple  $(s, p, o)$  is stored in  $S_i$  as a string. To improve space efficiency of PM-Index, we adopt the strategy of dictionary encoding, such that each vertex is encoded into a integer by hash operation. Especially, in each partial matching result, *null* is represented by the integer 0.

---

**Algorithm 1:** PM-Index construction
 

---

**Input:** The set of partial matching results in fragment  $F_i$ , denoted as  $P_{Q,i}$ .  
**Output:** The set of all PM-Index in  $F_i$ , denoted as  $I_i^{out}$  and  $I_i^{in}$ .

```

1 set  $I_i^{out} \leftarrow \emptyset, I_i^{in} \leftarrow \emptyset;$ 
2 foreach  $\mu_p \in P_{Q,i}$  do
3   foreach  $(\mu_p(s_r), \mu_p(o_r)) \in \mu_p$  do
4     if  $\mu_p(s_r)$  (or  $\mu_p(o_r)$ )  $\in V_i^I$  then
5       // incoming vertices
6        $I_i^{in}.put(\mu_p(s_r), \mu_p)$  (or  $I_i^{in}.put(\mu_p(o_r), \mu_p)$ );
7     else if  $\mu_p(s_r)$  (or  $\mu_p(o_r)$ )  $\in V_i^O$  then
8       // outgoing vertices
9        $I_i^{out}.put(\mu_p(s_r), \mu_p)$  (or  $I_i^{out}.put(\mu_p(o_r), \mu_p)$ );
10  return  $I_i^{out}, I_i^{in};$ 
    
```

---

The construction approach of PM-Index is shown in Algorithm 1. First, the PM-Index is initialized by the fragment identifier  $F_i$  (line 1). Then, for each mapping  $(\mu_p(s_r), \mu_p(o_r))$ , if  $\mu_p(s_r)$  (or  $\mu_p(o_r)$ ) is an *incoming* vertex (or an *outgoing* vertex),  $(\mu_p(s_r)$  (or  $\mu_p(o_r)$ ),  $\mu_p$ ) will be input into  $I_i^{in}$  (or  $I_i^{out}$ ) in  $F_i$  (line 2–7). Algorithm 1 will iterate over each partial matching result until there is no partial matching result left.

Intuitively, the PM-Index groups the partial matching results that share the same *incoming* vertex or *outgoing* vertex together.

**Theorem 1.** Given a RDF graph  $G$ , a query graph  $Q$ ,  $I_i^{in}$  in  $F_i$  and  $I_j^{out}$  in  $F_j$ , where  $j = \{1, \dots, n\}$  and  $i \neq j$ , for any  $(v, PMs) \in I_i^{in}$ , if  $v \notin I_j^{out}$ , none of the partial matching results in  $I_j^{out}$  corresponding to PMs can be joinable, vice versa.

*Example 6.* For example,  $I_1^{out} = \{v_{15}, \langle \mu_{p_1}^3 \rangle\}$ ,  $I_3^{in} = \{v_{15}, \langle \mu_{p_3}^1 \rangle\}$ .  $I_2^{out} = \{v_{18}, \langle \mu_{p_2}^3 \rangle\}$ , there is no vertex  $v_{18} \in \bigcup_{j=1, j \neq 2}^4 I_j^{in}$ . Therefore, there are no partial matching results joinable with  $v_{18}$  in  $I_2^{out}$ .

## 4.2 PM-Index Based Assembly Algorithm

To utilize the PM-Index, we propose a assemble algorithm, as shown in Algorithm 2. The key idea of Algorithm 2 is to scan the keys in  $v \in I_i^{in}$  during the search process, and find the same  $v \in I_j^{out}$  to assemble its values, where  $j \in \{1, \dots, n\} \wedge j \neq i$ . Algorithm 2 describes the optimization of assembling partial matching results. The coordinator site receives all PM-indexes from slave sites. Then, all the *incoming* vertex  $v \in I_i^{in}$  are traversed, according to theorem 1, if  $v \notin I_j^{out}$ , none of the value in  $I_i^{in}$  with  $v$  as the key can be assembled. We can remove it from  $I_i^{in}$ . As for the join result  $\mu_p''$ , we need to check whether it still contains any other *incoming* vertices or *outgoing* vertices except  $v$ , and if it does, insert  $\mu_p''$  into the corresponding set of key-value indexes, i.e.,  $I_i^{in}$  and  $I_j^{out}$  until there are no joinable partial matching results in  $I_i^{in}$ ,  $i, j \in \{1, \dots, n\}, i \neq j$ .

---

### Algorithm 2: PM-Index based assembly algorithm

---

**Input:** The PM-Index from all fragments,  $\bigcup_{i=1}^n I_i^{out}$  and  $\bigcup_{i=1}^n I_i^{in}$

**Output:** The set of subgraph matching results  $P_Q$

```

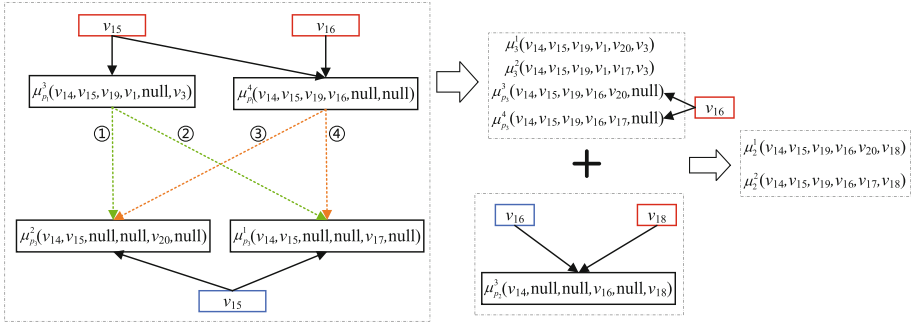
1 set  $P_Q \leftarrow \emptyset$ ;
2 foreach  $v \in I_i^{in}.key, i \in (1, \dots, n)$  do
3   while  $\exists v \in I_j^{out}.key, (j \in (1, \dots, n) \wedge j \neq i)$  do
4     foreach  $\mu_p \in I_i^{in}.valueOf(v)$  do
5       foreach  $\mu'_p \in I_j^{out}.valueOf(v)$  do
6         if  $\mu_p$  and  $\mu'_p$  can be joinable then
7            $\mu_p'' = \mu_p \cup \mu'_p$ ;
           // join the two mappings according to the joinable
           conditions
8           if  $\mu_p''$  is a complete result then
9              $P_Q \leftarrow \mu_p''$ ;
             // insert it into the set of final results.
10          else if  $\mu_p''$  is a partial matching result then
11            if  $\exists v' \in \mu_p'' \wedge v' \in (V_i^I \cup V_j^O)$  then
12              insert  $\mu_p''$  into  $I_i^{in}$  and  $I_j^{out}$  according to
13              the rule of computing PM-Index;
14            else
15              remove  $\mu_p''$ ;
              // a partial matching result with no other
              incoming or outgoing vertices
16          remove  $v$  and its value in  $I_i^{in}$ ;
          // according to theorem 1
17 return  $P_Q$ ;

```

---

*Example 7.* As shown in Fig. 5,  $I_1^{out} = \{(v_4, \langle \mu_{p_1}^2 \rangle), (v_7, \langle \mu_{p_1}^1 \rangle), (v_{15}, \langle \mu_{p_1}^3, \mu_{p_1}^4 \rangle), (v_{16}, \langle \mu_{p_1}^4 \rangle)\}$ ,  $I_3^{in} = \{v_{15}, \langle \mu_{p_3}^1, \mu_{p_3}^2 \rangle\}$ . For the key  $v_{15}$  in  $I_3^{in}$ , the same keys in  $I_1^{out}$  are found and the values are assembled. The final results  $\mu_3^1$  and  $\mu_3^2$  are included into the result set, while the partial results  $\mu_{p_3}^3$  and  $\mu_{p_3}^4$  are included into the set of keys corresponding to  $v_{16}$ .  $I_1^{out} = \{(v_4, \langle \mu_{p_1}^2 \rangle), (v_7, \langle \mu_{p_1}^1 \rangle), (v_{15}, \langle \mu_{p_1}^3, \mu_{p_1}^4 \rangle), (v_{16}, \langle \mu_{p_3}^3, \mu_{p_3}^4 \rangle)\}$ .  $I_2^{in} = \{v_{16}, \langle \mu_{p_2}^3 \rangle, \mu_{p_3}^3, \mu_{p_3}^4$  and  $\mu_{p_2}^1$  can be assembled according to the key  $v_{16}$  respectively. Thus, the final results of assemble process will be  $\mu_2^1$  and  $\mu_2^2$ .

**Space Complexity of PM-Index.** For each fragment  $F_i$ , the number of the PM-Index is  $O(|V_i^I| + |V_i^O|)$  at most. Except the size of partial matching results, the extra space is bounded with  $O(|V_i^I| + |V_i^O|)$ .



**Fig. 5.** An example of partial matching index based assembly

**Time Complexity of PM-Index Based Assembly.** For the partial matching results in  $F_i$ , the time complexity of the search process of the assembly stage based on PM-index is  $O(|V_i^I| \cdot \sum_{j=1, i \neq j}^J \log(|V_j^O|))$ .

## 5 Experiments

In order to verify the effectiveness and efficiency of the PM-Index method under the partial evaluation and assembly framework, a comparative experiment with gStoreD [8, 9] was conducted over several benchmark RDF datasets. The proposed algorithm is implemented on the top of HAWQ [2], which are deployed on a 4-node cluster, of which each node has an 2-core CPU, with 16GB of memory, running 64-bit CentOS 7.7 operating system.

### 5.1 Datasets and Queries

Different scale of LUBM [4] synthetic benchmark datasets are used in the experiments. The statistics of the datasets is shown in Table 1. We need to compare

the query efficiency of PM-Index-based method and the original partial evaluation and assembly framework with different scale of intermediate results. It is necessary to gradually change the number of intermediate results that the query satisfies while limiting the basic structure of the query, so we choose the benchmark datasets rather than real-world datasets to maintain a positive correlation between the increase in the size of the data set and the increase in the number of intermediate results. In addition, to eliminate the impact of the partition strategy on query processing, we use a random partitioning method to divide each dataset into 4 fragments.

**Table 1.** Datasets

Dateset	#edges	#vertices
LUBM10	1,316,700	314,852
LUBM20	2,782,126	663,647
LUBM30	3,890,992	841,108

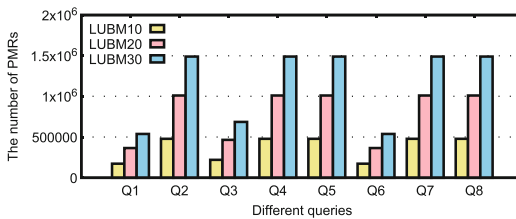
**Table 2.** Queries

#Query vertices	Queries
5	$Q_3$
6	$Q_4, Q_8$
7	$Q_1, Q_2, Q_5$
8	$Q_6, Q_7$

As shown in Table 2, 8 queries with different number of vertices on LUBM datasets are proposed, i.e.,  $Q_1 \sim Q_8$ , which are all complex queries. Since the PM-Index optimization method we proposed is aiming to improve the efficiency of assembling stage when the intermediate results are enormous, the proposed queries will generate intermediate results on a scale between tens of thousands and hundreds of thousands during the partial matching process.

## 5.2 Experimental Results

To verify the effectiveness of the PM-Index based assembly algorithm, extensive experiments were conducted.

**Fig. 6.** Query evaluation on LUBM datasets

**Exp 1. Number of Partial Matching Results.** To make it more intuitive to observe and evaluate the performance of the assembly algorithm with different queries and datasets, we record the number of partial matching results of  $Q_1 \sim Q_8$  over LUBM10, LUBM20 and LUBM30. It is the sum of all partial matching

results received from the computing sites in the coordinator site, which is the size of the intermediate results to be joined during the assembly phase. As shown in Fig. 6, for all queries, the number of partial matches increases approximately linearly with the size of the datasets.

**Exp 2. The Constructing Time and Space Size of PM-Index.** The longest time of constructing PM-Index among all computing sites with different queries and datasets is shown in Fig. 7(a). It can be seen that the time of constructing PM-Index is positively correlated with the size of the intermediate results. Figure 7(b) shows the maximum size of the space occupied by the PM-Index among all computing sites with different queries and datasets. Apart from the space taken up by partial matching results, the space required for the key of index is small, which guarantees the time and space complexity of the proposed method.

**Exp 3. Efficiency and Scalability.** A measurement of the statistical consequences of assembly time from  $Q_1$  to  $Q_8$  over different datasets is shown in Fig. 8. Among distributed RDF systems, gStoreD uses the partial evaluation and assembly framework to achieve better query efficiency [8], while the PM-Index method can further improve the query efficiency. It can be observed that PM-Index based assembly method outperforms that of gStoreD for all queries, which can be improved by 1.53 times in the best case. In addition, as the scale of the datasets increasing, the proportion of time reduction also rises. For queries such as  $Q_2$ ,  $Q_4$  and  $Q_7$ , although the number of vertices involved are different, their scales of intermediate results are near, so that the assembly performances are similar. The reasons are as follows: (1) when dealing with large-scale intermediate results, the efficiency improvement of the LEC-based grouping method is limited. Since the search space of the grouping approach is still large, restricting the search efficiency; (2) when more intermediate results involved, the bottleneck problem in the assembly phase would be more severe for gStoreD; (3) when there are few intermediate results, PM-Index-based method will not reduce the query efficiency of partial evaluation and assembly framework.

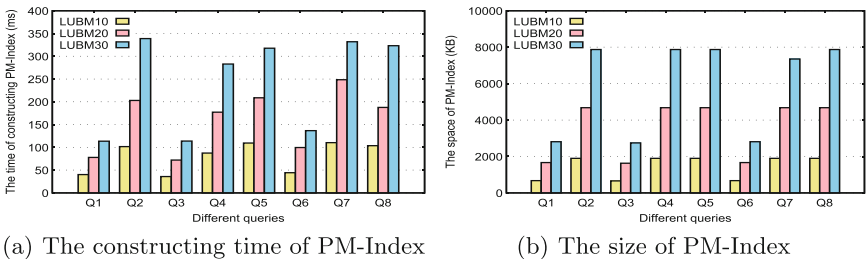


Fig. 7. The constructing time and size of PM-Index



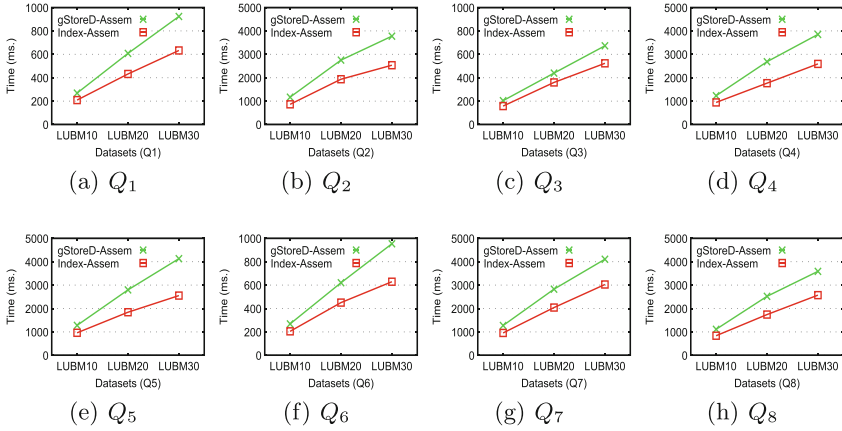


Fig. 8. Query evaluation on LUBM datasets

## 6 Conclusion

In this paper, we proposed a PM-Index-based method to improve the assembly efficiency of the subgraph matching queries in distributed settings based on partial evaluation. Moreover, we also proved that the PM-Index is both time-efficient and space-effective. The extensive experimental results on benchmark datasets verified the efficiency and scalability of the proposed method, which clearly outperforms gStoreD when large-scale intermediate results need to be processed.

**Acknowledgments.** This work is supported by the National Key Research and Development Program of China (2019YFE0198600) and National Natural Science Foundation of China (61972275), partially supported by Australian Research Council Linkage Project (LP180100750).

## References

1. Chandra, A.K., Merlin, P.M.: Optimal implementation of conjunctive queries in relational data bases. In: Proceedings of the Ninth Annual ACM Symposium on Theory of Computing, STOC 1977, pp. 77–90. Association for Computing Machinery, New York (1977)
2. Chang, L., et al.: Hawq: a massively parallel processing sql engine in hadoop. In: Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, pp. 1223–1234 (2014)
3. Fan, W., Wang, X., Wu, Y.: Performance guarantees for distributed reachability queries. VLDB **5**(11), 1304–1316 (2012)
4. Guo, Y., Pan, Z., Heflin, J.: LUBM: a benchmark for OWL knowledge base systems. Web Semant. Sci. Serv. Agents World Wide Web **3**, 158–182 (2005)
5. Harris, S., Seaborne, A.: SPARQL 1.1 query language. W3C recommendation 21 (2013)

6. Husain, M., McGlothlin, J., Masud, M.M., Khan, L., Thuraisingham, B.M.: Heuristics-based query processing for large RDF graphs using cloud computing. *IEEE Trans. Knowl. Data Eng.* **23**(9), 1312–1327 (2011)
7. Jones, N.D.: An introduction to partial evaluation. *ACM Comput. Surv.* **28**(3), 480–503 (1996)
8. Peng, P., Zou, L., Guan, R.: Accelerating partial evaluation in distributed SPARQL query evaluation. In: 35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8–11, 2019. pp. 112–123. IEEE (2019)
9. Peng, P., Zou, L., Özsu, M.T., Chen, L., Zhao, D.: Processing SPARQL queries over distributed RDF graphs. *VLDB J.* **25**(2), 243–268 (2016). <https://doi.org/10.1007/s00778-015-0415-0>
10. Pérez, J., Arenas, M., Gutierrez, C., et al.: Semantics and complexity of SPARQL. In: Cruz, I. (ed.) *ISWC 2006*. LNCS, vol. 4273, pp. 30–43. Springer, Heidelberg (2006). [https://doi.org/10.1007/11926078\\_3](https://doi.org/10.1007/11926078_3)
11. Ren, X., Wang, J., Han, W.S., Yu, J.X.: Fast and robust distributed subgraph enumeration. *Proc. VLDB Endow.* **12**(11), 1344–1356 (2019)
12. Rohloff, K., Schantz, R.: Clause-iteration with mapreduce to scalably query data-graphs in the shard graph-store. In: *DIDC 2011* (2011)
13. Wang, X., Wang, S., Xin, Y., Yang, Y., Wang, X.: Distributed pregel-based provenance-aware regular path query processing on RDF knowledge graphs. *World Wide Web* **23**(3), 1465–1496 (2020)
14. Wang, X., et al.: Efficient subgraph matching on large RDF graphs using MapReduce. *Data Sci. Eng.* **4**(1), 24–43 (2019). <https://doi.org/10.1007/s41019-019-0090-z>
15. Wang, X., Wang, J., Zhang, X.: Efficient distributed regular path queries on RDF graphs using partial evaluation. In: *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, CIKM 2016*, pp. 1933–1936. Association for Computing Machinery, New York (2016)
16. Wang, X., Zou, L., Wang, C., Peng, P., Feng, Z.Y.: Research on knowledge graph data management: a survey. *Ruan Jian Xue Bao* **30**(07), 2139–2174 (2019)
17. Zeng, K., Yang, J., Wang, H., Shao, B., Wang, Z.: A distributed graph engine for web scale RDF data. *Proc. VLDB Endow.* **6**(4), 265–276 (2013)
18. Zou, L., Özsu, M.T., Chen, L., Shen, X., Huang, R., Zhao, D.: gStore: a graph-based SPARQL query engine. *VLDB J.* **23**(4), 565–590 (2014)



# Enhancing both Local and Global Entity Linking Models with Attention

Jinliang Li<sup>1</sup>, Haoyu Liu<sup>2</sup>, Yulong Zhang<sup>1</sup>, Li Zhang<sup>1</sup>, Qiang Yang<sup>3</sup>,  
Jianfeng Qu<sup>1</sup>(✉), and Zhixu Li<sup>1</sup>

<sup>1</sup> School of Computer Science and Technology, Soochow University, Suzhou, China  
{jlli1997,ylzhang1998,lzhang1997}@stu.suda.edu.cn

{jfq, zhixuli}@suda.edu.cn

<sup>2</sup> IFLYTEK Research, Suzhou, China

<sup>3</sup> King Abdullah University of Science and Technology, Thuwal, Saudi Arabia  
qiang.yang@kaust.edu.sa

**Abstract.** Entity linking aims at mapping the mentions in a document to their corresponding entities in a given knowledge base, which involves two continuous steps, i.e., local step which focuses on modeling the semantic meaning of the context around the mention, and global step which optimizes the refereed entities coherence in the document. Upon the existing great efforts on both steps, this paper would like to enhance both local and global entity linking models with several attention mechanisms respectively. Particularly, we propose to leverage self-attention mechanism and LSTM-based attention mechanism to better capture the inter-dependencies between tokens in the mention context for the local entity linking models. We also adopt a hierarchical attention network with a multi-head attention layer to better represent documents with one or multiple topics for the global entity linking models, which could help alleviate the side effect of error accumulation. Extensive empirical study on standard benchmarks proves the effectiveness of the proposed models.

**Keywords:** Entity linking · Knowledge graph · Attention

## 1 Introduction

Entity linking involves linking entity mentions in texts to their corresponding unambiguous entities in a knowledge base. As a crucial step to both text understanding and knowledge base construction, this task is challenging given the ambiguity of mentions and the various forms of entities. To tackle this problem, recent researches on entity linking can be seen as combination of two continuous steps, i.e., local step and global step [4, 5, 11, 13, 15, 17]. While the local step works on modeling the context of each mention in the given document, the global step focuses on optimizing the coherence among the refereed entities in the document.

To model the mention context in a given document, early local methods adopt CNN [3] or LSTM [7] to learn the representation of mention context. Some recent work turn to use additional information such as Wikipedia entity categories [1] or predicted types of mentions [17] for improving the local model. Despite their success, the local methods relying on extra information may introduce noises in capturing important aspects of the context given that the extra information is not always reliable. Given that the tokens in the context do not contribute equally to the meaning of the sequence, attention mechanisms are adopted to make the model attend more focus on tokens that are more informative [5]. However, the inter-dependencies between tokens, especially long-term dependencies, are usually ignored, which are actually important for representing the mention context.

Global methods are designed to compute the sum over all pairwise scores between entities based on the assumption that the referred entities have dependencies with each other [11, 17]. For instance, [5] defines a joint probability distribution over all candidate entities and maximizes it by utilizing loopy belief propagation algorithm to ensure that the selected entities are highly related. However, all the global methods may suffer from error accumulation in the entity linking process, that is, entities that have been incorrectly linked may have negative impact on entities that have not yet been linked. We also find out that the existing global methods never consider the case that there could be multiple topics in the same document, and the referred entities may have different relations with different topics in the document.

To help overcome the drawbacks with the existing local and global models, this paper proposes several attention mechanisms to enhance local and global entity linking models respectively. (1) For the local step, in order to capturing the inter-dependencies between tokens in the mention context, two attention mechanisms are applied. Particularly, a self-attention mechanism is applied to capture the inter-dependencies between tokens, while a LSTM-based attention mechanism is also utilised to capture long-term dependencies between tokens. In this way, we could obtain more fine-grained interactions between tokens in the mention context. (2) For the global step, to alleviate the side effect of error accumulation, a hierarchical attention network is applied which could denote the document in different levels. As a result, the later entity linking decision is not only affected by the previous probability, but also the coherence between the entity and the document. Besides, we also hypothesize that some documents may have several topics. Thus a multi-head attention layer is added at the top of the hierarchical attention model to embed document into multiple vector space, such that the model could pay attend to different aspects of the documents.

To summarize, our contributions are as follows.

- We propose to leverage self-attention mechanism and LSTM-based attention mechanism to better capture the inter-dependencies between tokens in the mention context for the local entity linking models.

- We propose to adopt a hierarchical attention network with a multi-head attention layer to better represent documents with one or multiple topics for the global entity linking models.
- Extensive experiments are conducted on benchmarks, which show that our local attention model outperforms previous methods in most datasets, and we achieve competitive results compared with model that leveraging reinforcement learning in the global step.

The rest of the paper is organized as follows. We cover the related work in Sect. 2, and then define our problem formally in Sect. 3. After presenting our model in Sect. 4, we report our empirical study in Sect. 5. We finally conclude the paper in Sect. 6.

## 2 Related Work

Previous entity linking models can be roughly divided into two categories, i.e., local models or global models. In this section, we mainly cover the prior work that are relevant to our contributions.

### 2.1 Local Models

The core task of the local models for entity linking is to measure the similarity between the local context of a mention and its corresponding candidate entities. Traditional local models mainly rely on feature engineering to compare cosine similarities between the local context of a mention and the Wikipedia title of its corresponding candidate entities [2, 15]. However, the feature extracted by hand-crafted rules are too expensive and sparse to provide enough information for local model. Recent work mostly concentrates on representation learning for the local models of entity linking. [9] first introduces stacked denoising auto-encoders to learn the representation of mentions and entities, and then let the model to predict the desired output through hidden layer. Later, [3] uses CNN to extract different granularity of semantic information from multiple sources and combines these features with logistic regression to obtain the representation of the mention. [5] also proposes a novel attention mechanism which selects words that are informative by comparing the cosine similarity between words in the mention context and the embeddings of its corresponding candidate entities, based on the assumption that a few words related to an entity can represent the mention. [11] concatenates the word embedding of context and feeds it to a single layer feed-forward network to obtain the mention embedding. However, using CNN or selecting important words with TF-IDF may not be sufficient to represent the context of mentions. Our model uses attention mechanism that can incorporate different positions in text to capture interactions and dependency between words which leads to better mention context representations.

## 2.2 Global Models

The goal of global model is to maximize the entity coherence between all refereed entities based on the assumption that mentions in the document are strongly related. [8] uses graph model to calculate the coherence among mentions. [5] defines a joint probability distribution over all candidate entities. Each time a new candidate is selected, a pairwise conditional random field is applied to measure the coherence of the entities where the score is high if entities are semantically related. Since using binary conditional random field is NP-hard, approximation methods become another choice to achieve a trade-off between linking precision and computation cost. For instance, [11] proposes to add multiple relation to each entity instead of entity co-occurrence. They apply max-product loopy belief propagation to estimate the max-marginal probability which achieves competitive results but with less cost. Besides, since the global model can be seen as a sequential decision problem where later entity selection more or less depends on the former linking decisions, previous error linking may affect the latter linking decision. [17] proposes to use reinforcement learning to deal with the accumulated errors which uses reward signal to encourage the correct linking decision and maximum that reward.

However, we try another approach to alleviate the affection that previous wrong linked entities may influence the afterwards decision. More specifically, we leverage document level embedding when come across the new candidate entity to let the later decision not only affected by the previous probability but also the coherence between entity and document. Furthermore, our hypothesis is that one document may contain more than one topic, so we use multi-head attention mechanism which could let the model focus on different aspects of the document.

## 3 Problem Definition

Given a document  $D$  containing many entity mentions  $\mathbf{M} = \{m_1, m_2, \dots, m_T\}$ , where each mention  $m_i$  ( $1 \leq i \leq T$ ) corresponds to a set of candidate entities denoted by  $\mathbf{E}_i = \{e_i^1, e_i^2, \dots, e_i^{N_i}\}$  ( $N_i \geq 1$ ), the task of entity linking aims to map each mention  $m_i$  ( $1 \leq i \leq T$ ) to its corresponding entity  $e_i^* \in \mathbf{E}_i$ .

Usually, the entity linking task can be taken as a ranking problem, where the ranking score is combined by a local ranking score and a global ranking score with a learnable parameter. While the local ranking score is obtained by the employed local model, the global ranking score is obtained by the employed global model.

Let  $c_i$  denote the context of a mention  $m_i$  ( $1 \leq i \leq T$ ) in document  $D$ , the local ranking score of  $m_i$  linking to one of its candidate entity  $e_i^j$  ( $1 \leq j \leq N_i$ ) can be calculated as:

$$\Psi(m_i, e_i^j) = e_i^{j\top} \mathbf{B} f(c_i) \quad (1)$$

where  $e_i^{j\top}$  is the candidate entity embedding,  $\mathbf{B}$  is a diagonal matrix, and  $f(c_i)$  is a function to encode  $c_i$  into a vector representation.

The global ranking score of  $m_i$  linking to one of its candidate entity  $e_i^j$  ( $1 \leq j \leq N_i$ ) can be calculated as:

$$\Phi(e_{t+1}^j, S_t) = \sum_{\hat{e}_i \in S_t} e_{t+1}^{jT} \cdot R \cdot \hat{e}_i \tag{2}$$

where  $R$  is also a diagonal matrix,  $e_{t+1}^j$  is the candidate entity embedding and  $S_t$  is the previous linked entities set.

We then have the final score of  $m_i$  linking to one of its candidate entity  $e_i^j$  ( $1 \leq j \leq N_i$ ) as:

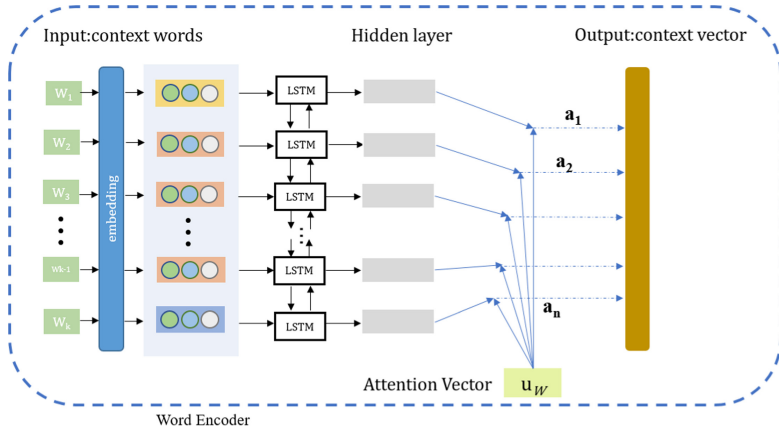
$$\mathbf{score}(m_i, e_i^j) = \lambda \Psi(m_i, e_i^j) + (1 - \lambda) \Phi(e_i^j, S_{t-1}) \tag{3}$$

where  $\lambda$  is a learnable parameter for the combination.

### 4 Model

Our model consists of two parts: local model and global model. While the local model encodes the context of mention with attention mechanism to improve the representation of the mention context, the global model adopts a document-entity coherence to alleviate the error accumulation problem.

#### 4.1 Attention Based Local Model



**Fig. 1.** Improve the representation of the mention context using Bi-LSTM based on attention mechanism.

The goal of local model is to improve the representation of the mention context. To this end, we keep the basic local score function and change the mapping

function  $f$ . Instead of comparing the similarity between context of mention and the candidate entity then select the most similar candidate entity, we use attention mechanism with whole context can learn dependencies words and capture the internal structure of sentence, and lead to better context representation. Thus, we introduce two attention mechanisms to encode the context of mention, the LSTM-based and self-attention-based.

The LSTM-based attention mechanism is shown in Fig. 1. In the first step, for each mention  $m$  we extract its K-neighbors context words  $c = \{w_1, \dots, w_k\}$ . After that, we use standard word embedding approaches [12] to map each word  $w \in c$  to d-dimensional vector space. Next, to capture the dependencies between words, we use Bi-directional LSTM to process the sequence. The words in  $c$  are sent to Bi-directional LSTM to get the hidden output of them with context information.

$$h^t = BiLSTM(x^t) \quad (4)$$

where  $x^t$  is the embedding of the word and  $h^t$  is the hidden state. We then concatenate the forward hidden state with the backward hidden state. When the sequence reaches the end, the hidden states of Bi-directional LSTM are full of contextual information and can be seen as the representation of the sentence. However, the words in sequence are not equally important, therefore we leverage attention mechanism to capture important aspect of the sequence. We use the attention mechanism to redistribute the weight of each word and normalize it by softmax function to get the attention score of them, the vector representation of the sentence is the weighted sum of the hidden output of each word and the attention score.

$$f(c_i) = \sum_{t=1}^K softmax(u^\top h^t) \cdot h^t \quad (5)$$

where  $u$  is weighted vector to reflect the importance of each word that will be learned during training, and the final mention context vector is summarized by its weighted context vector.

Former method uses LSTM to obtain dependency between words, however, LSTM suffers from the vanishing gradient problem which may prevent it from learning the long-term dependencies. The transformer [16] provides an alternative inspect on using matrix operation to learn interaction between distant words. Thus, we introduce another attention mechanism, namely self-attention mechanism, to encode the context of mention. Specifically, we calculate the following three matrices: Query, Key and Value.

$$Q = W^Q * X \quad (6)$$

$$K = W^K * X \quad (7)$$

$$V = W^V * X \quad (8)$$

where  $W^Q, W^K, W^V$  are transforming matrix that map input words to their corresponding representation spaces. We then apply self-attention mechanism



on matrix  $Q, K, V$  in the following way:

$$A = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right) \tag{9}$$

$$S = A \cdot V \tag{10}$$

Hereto, each word in sequence contains contextual information of surrounding words. To convert the output to sentence embeddings, we then apply a pooling layer to the sum of the output embeddings.

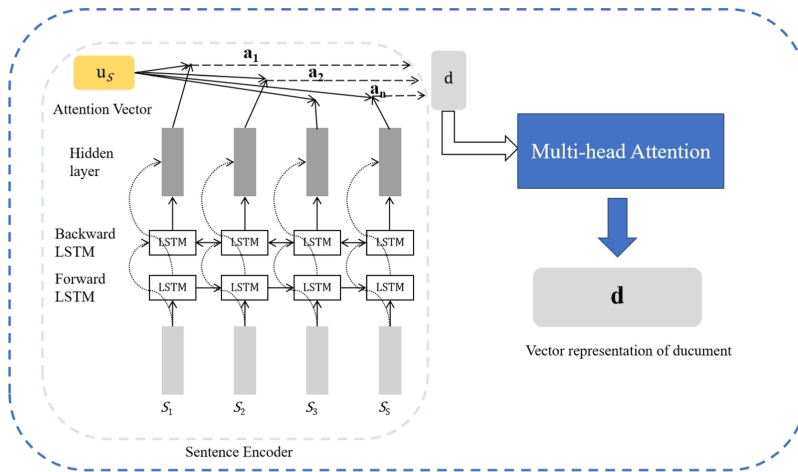
$$f(c_i) = MEAN(S) \tag{11}$$

To improve the performance of the local model, we also add some statistical information and then apply a feedforward neural network to balance these two weights. Finally, we combine the feature (Eq. 11) with the statistical information to calculate the local score as follows:

$$\Psi(e, m, c) = f(\Psi(e, c), \log \tilde{p}(e | m)) \tag{12}$$

where  $\tilde{p}(e | m)$  is mention-entity prior which measures the number of times it has been linked.

### 4.2 Global Model with Document-Entity Coherence



**Fig. 2.** The framework of global model to get the semantic representation of document

The idea of adding an extra document-entity coherence is inspired by that the global model forms a sequential decision problem where former wrong linked

decision may affect the later decision. Thus, we keep in line with Yang (2019) [17] but add a document entity score that measures the coherence between the candidate entity and the document. In this way, we try to minimize the side effect of the accumulated sequential error. Formally, given a document that consists of  $S$  sentence and each sentence that contains  $W$  words. We apply the hierarchical attention mechanism from word level to sentence level to extract main topic of the document. The process is as follows:

$$h^t = BiLSTM(x^t) \quad (13)$$

$$s = \sum_{t=1}^W softmax(u_w^\top h^t) \cdot h^t \quad (14)$$

$$a^t = BiLSTM(s^t) \quad (15)$$

As shown in Figs. 1 and 2, we use Bi-directional LSTM to generate contextual information in both word level and sentence level encoding. After obtaining the hidden representation of each word or sentence, we apply an attention mechanism where  $u_w$  and  $u_s$  are word level and sentence level context vectors that reflect the importance of each word or sentence. Finally, the representation of the whole document is the sum of all weighted sentence vector:

$$d = \sum_{t=1}^S softmax(u_s^\top a^t) \cdot a^t \quad (16)$$

In order to address the scenario when a document has multiple topics, we apply a multi-head attention mechanism that can capture different aspects of the document where each representation subspace focuses on one topic of the document. Finally, the whole document is represented by the following formula:

$$d = \sum_{t=1}^S softmax(U_w^\top a^t) \cdot a^t \quad (17)$$

where  $U_w$  is a  $d_t * d$  matrix which contains the topics that may exist in the whole document,  $d_t$  equals to the topics we expected, and  $d$  equals the LSTM hidden size. And we define document-entity coherence as

$$\Omega(e_i, doc) = e_i * d \quad (18)$$

where  $e_i$  is the embedding of the entity and  $d$  is the vector representation of the document. Our global score then defined as

$$\Phi(e_{t+1}^j, S_t) = \sum e_{t+1}^j{}^\top \cdot R \cdot \hat{e}_i + e_{t+1}^j{}^\top \cdot d \quad (19)$$

where  $e_{t+1}^j$  is the candidate entity embedding,  $S_t$  is previously linked entities,  $R$  is a learnable diagonal matrix and  $d$  is the vector representation of the document.

The final score of the whole model is a linear combination of the local score and global score which is shown below.

$$\mathbf{score}(m_i, e_i^j) = \lambda \Psi(m_i, e_i^j) + (1 - \lambda) \Phi(e_i^j, S_{t-1}) \quad (20)$$

## 5 Experiments

### 5.1 Implements

In local model, since word embeddings and entity embeddings are fixed, we only need to train the parameters in LSTM, matrices  $W^Q, W^K, W^V$  and the feedforward layer. We assume that we have the ground truth  $\{(m, e^*)\}$  where  $m$  is the mention and  $e^*$  is the golden truth. Like previous work, we use max-margin loss to minimize the loss as follows:

$$L(\theta) = \arg \min_{\theta} \sum_{D \in \mathcal{D}} \sum_{m \in D} \sum_{e \in \Gamma(m)} g(e, m) \quad (21)$$

$$g(e, m) = [\gamma - \Psi(e^*, m, c) + \Psi(e, m, c)]_+ \quad (22)$$

where  $\theta$  are the parameters we want to train,  $\gamma$  is the margin parameter and  $\mathcal{D}$  is the document which contains mentions. The loss is expect to rank ground true entities higher than the wrong entities.

In global model, the parameters we try to optimize are parameters in LSTM and feedforward parameters. Like local model, we also define a max-margin loss, which is shown below, to train the model.

$$L(\theta) = \arg \min_{\theta} \sum_{D \in \mathcal{D}} \sum_{m_i \in D} \sum_{e \in \Gamma(m_i)} h(m_i, e) \quad (23)$$

$$h(m_i, e) = [\gamma - \rho_i(e_i^*) + \rho_i(e)]_+ \quad (24)$$

where  $\theta$  is global model parameter and  $\rho$  is the learned function that maps the global feature to each candidate entity.

### 5.2 Datasets

We evaluate three models: local model, global model and the whole combined model (local model plus global model). We train our models on AIDA CoNLL-YAGO dataset [10] which contains 946 training documents, 216 validating documents and 231 testing documents. And the target knowledge base is Wikipedia.

Besides the in-domain dataset, we test our models on five cross-domain datasets: MSNBC, AQUAINT, ACE2004 which includes 106 documents totally [6], WIKI and CEWB which were automatically extracted from ClueWeb and Wikipedia with 640 documents. The statistics of these datasets is shown in Table 1.

### 5.3 Candidate Selection and Entity Embeddings

For each mention in the datasets, we select its candidate entity based on its  $p(e|m)$  and keep the top 30 candidates. For saving model training time, we only select 4 entities with highest  $p(e|m)$  and 3 entities based on context entity score based on Eq. 7. As for entity embeddings, we directly use the embeddings pretrained by [5].

**Table 1.** Statistics of five cross-domain datasets.

Dataset	Number mentions	Number docs	Mentions per doc	Mean length context	Mean number candidate
MSNBC	656	20	32.8	177.5	39
AQUAINT	727	50	14.5	148.6	32.9
ACE2004	257	36	7.1	153.3	47.9
CWEB	11154	320	34.8	183.5	44.2
WIKI	6821	320	21.3	148.7	32.7

## 5.4 Hyper-parameter Setting

We use word2vec [12] to compute the word embedding and the method proposed in [5] to compute the entity embedding. The entity embedding dimension is set to 300 and the ranking margin  $y$  is set to 0.01. We set the batch size to 1 (i.e. each batch only contains 1 document) and use Adam as the optimizer. When F1 score exceeds 92.8% on validation set, we change the learning rate from  $1e-4$  to  $1e-5$ . The training process terminates if the performance (F1 score) does not improve in 20 epochs. In local model, we remove the stop words and rare words that appear less than 3 times in the document, and the window size of the context word around  $m$  is set to 50. We set the hidden size of the Bi-directional LSTM to 300, the size of pattern vector to 300, the dropout probability to 0.1, and the dimension of  $W^Q, W^K, W^V$  to 300. We stack two identical self-attention layers to improve the performance. In global model, the hidden size of Bi-directional GRU is also set to 300, the dropout probability is set to 0.1, and the pattern vectors dimension is set to 300. We set the number of topics of the document to 2 by default in the experiments.

## 5.5 Results

We evaluate our local model, global model and the combined model with several state-of-the-art methods on public datasets separately. That is, we only compare our model with the same type of the existing state-of-the-art methods for fair comparison. And we use micro F1 score to measure the performance of each part.

**Table 2.** Micro F1 score on Cross-domain datasets with local models. Bolded scores are the results that outperform than others.

Local models	MSNBC	AQUAINT	ACE2004	CWEB	WIKI
CNN ([3])	89.05	80.55	87.32	67.97	60.27
Deep-ED [5]	91.97	84.06	86.92	70.07	<b>74.37</b>
LSTM-Based ATTN	<b>92.11</b>	<b>84.61</b>	<b>87.43</b>	<b>70.69</b>	66.63
SELF-ATTN	<b>92.27</b>	<b>85.31</b>	<b>88.58</b>	69.56	64.58

For local model, since most methods combine local and global model together, we only choose Berkeley-CNN [3] which use CNN to capture context representation and the local part of Deep-ED [5] as our baseline models. Table 2 summarizes the results showing that both of our two different attention based local models achieve state-of-the-art results in most datasets. Specifically, the LSTM-based model outperforms 4 methods while the self-attention based model outperforms 3 methods.

**Table 3.** Micro F1 score on Cross-domain datasets with global models. Bolded scores are the results that outperform than others.

Global models	MSNBC	AQUAINT	ACE2004	CWEB	WIKI
ETHZ-ATTN+DCA-SL ([17])	<b>94.57</b>	87.38	89.44	73.47	78.16
ETHZ-ATTN+DCA-RL ([17])	93.80	<b>88.25</b>	90.14	<b>75.59</b>	78.84
Our(global)	94.1	86.85	<b>90.54</b>	73.95	<b>78.94</b>

For global model, we compare our model with two extensive models of DCA [17]: DCA-SL [17] and DCA-RL [17]. For fair comparison, we use the same local model of DCA. As can be seen from Table 3, our model achieve competitive results on public datasets. Specifically, our model outperforms the DCA-SL model and the DCA-RL model on three datasets respectively. In addition, our method achieves state-of-the-art performance on the ACE2004 dataset compared with all existing models.

For the combined model, we compare with several existing methods. We briefly describe these methods below. GLOW [15] uses SVM to combine the local feature and global feature. RI [2] exploits that there are more than one relation between mentions. WNED [6] performs a random graph walks to highlight the importance of global model. Global-RNN [14] uses CNN and RNN to model both local and global features. Deep-ED [5] leverages learned entity embeddings and joint inference stage in global model. Ment-Norm [11] models multiple latent relations between mentions. DCA [17] proposes a context augmentation approach for global model.

The results are shown in Table 4 which indicates that no model based on deep learning can always outperform our model. Our model achieves two state-of-the-art results on MSNBC and ACE2004 of the five datasets. Our model has 0.67% lower than [2] but outperforms the recent deep learning model on AQUAINT. However, our model does not perform well in CWEB and WIKI datasets. This is due to the bad quality of the datasets and we will discuss it in the next section.

**Table 4.** Micro F1 score on Cross-domain datasets using combined models. Bolded scores are the results that outperform than others.

Combined models	MSNBC	AQUAINT	ACE2004	CWEB	WIKI
AIDA ([13])	79	56	80	58.6	63
GLOW ([15])	75	83	82	56.2	67.2
RI ([2])	90	<b>90</b>	86	67.5	73.4
WNED ([6])	92	87	88	77	<b>84.5</b>
Deep-ED ([5])	93.7	88.5	88.5	<b>77.9</b>	77.5
Ment-Norm ([11])	93.9	88.3	89.9	77.5	78.0
Prior( $p(e m)$ ) ([5])	89.3	83.2	84.4	69.8	64.2
ETHZ-Attn + DCA-SL ([17])	94.57	87.38	89.44	73.47	78.16
ETHZ-Attn + DCA-RL ([17])	93.80	88.25	90.14	75.59	78.84
Our model	<b>94.87</b>	89.09	<b>90.94</b>	70.52	71.01

## 5.6 Analysis

**Poor Performance on CWEB and WIKI.** As shown in Table 2, our model do not perform well on CWEB and WIKI datasets, nor do other models. The main reason is that CWEB and WIKI datasets are extracted from online text automatically without human supervision. We observe that some golden entities do not appear in the candidate set such that the model always get wrong answers. Therefore, we may ignore in the candidate selection procedure when some mentions point to their corresponding entities that with very low  $p(e|m)$ .

**Different Attention Mechanism Choice.** In local model, we use two different attention mechanisms to encode the context of mention: the LSTM- based attention and self-attention. They both perform well on public datasets. Theoretically, the performance of LSTM-based attention will drop when the sequence length is too long because of the long-term dependency problem, and the models with self-attention often achieve state-of-the-art performance in most NLP tasks. However, in our model, they do not achieve competitive results against the other methods. There are possibly two reasons: (1) the first one is that the size of our context of mention is no longer than 50, so that the LSTM can extract informative semantic features in short sequence; (2) the second reason is that self-attention may need large training data to achieve the best result, however, the size of our datasets are limited. The LSTM-based methods can work well because they are easier to converge with small training datasets.

**Table 5.** Micro F1 score on different number of topic using global model.

Number of topics	MSNBC	AQUAINT	ACE2004	CWEB	WIKI
1	93.79	86.11	90.14	73.03	75.82
2	<b>94.1</b>	<b>86.85</b>	<b>90.54</b>	<b>73.95</b>	<b>78.94</b>
3	93.49	85.59	88.93	72.93	72.42

**Number of Topics.** In global model, we propose a document entity coherence where we extend it to multi-head attention to make the model can focus on different aspects of the document. Empirically, we assume that one document may contain no more than three topics, so we set top number from 1 to 3 and compare the results, as it shown in Table 5, we can find that the model achieve the best result when topic number is 2. So we hypothesis that the most documents are subject to one topic, some documents may contain two topics and very rare when topic number is greater than 2.

## 6 Conclusions and Future Work

In this paper, we propose to use attention mechanism in both local model and global model for entity linking problem. Different with previous work, we adopt the context of mention to extract semantic features to generate sentence embedding without comparing the context with every candidate entity. Also, we add document entity coherence in global model to alleviate the side effect of previous wrong linked entities and deal with the problem that documents may have multiple topics. Extensive experiments on public datasets show that our model can achieve competitive results compared with many state-of-the-art models. In the future, we would like to enrich the entity embeddings with more structural information by using the graph neural network.

**Acknowledgments.** This work was supported by the National Key R&D Program of China (No. 2018AAA0101900), the Priority Academic Program Development of Jiangsu Higher Education Institutions, National Natural Science Foundation of China (Grant No. 62072323, 61632016, 62102276), Natural Science Foundation of Jiangsu Province (No. BK20191420).

## References

1. Bunescu, R.C., Pasca, M.: Using encyclopedic knowledge for named entity disambiguation. In: EACL 2006, 11st Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference, 3–7 April 2006, Trento, Italy (2006). <https://www.aclweb.org/anthology/E06-1002/>
2. Cheng, X., Roth, D.: Relational inference for wikification. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18–21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL, pp. 1787–1796 (2013). <https://www.aclweb.org/anthology/D13-1184/>

3. Francis-Landau, M., Durrett, G., Klein, D.: Capturing semantic similarity for entity linking with convolutional neural networks. In: NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, 12–17 June 2016, pp. 1256–1261 (2016). <https://www.aclweb.org/anthology/N16-1150/>
4. Ganea, O., Ganea, M., Lucchi, A., Eickhoff, C., Hofmann, T.: Probabilistic bag-of-hyperlinks model for entity linking. In: Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, 11–15 April 2016, pp. 927–938 (2016). <https://doi.org/10.1145/2872427.2882988>
5. Ganea, O., Hofmann, T.: Deep joint entity disambiguation with local neural attention. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, 9–11 September 2017, pp. 2619–2629 (2017). <https://www.aclweb.org/anthology/D17-1277/>
6. Guo, Z., Barbosa, D.: Robust named entity disambiguation with random walks. *Semant. Web* **9**(4), 459–479 (2018). <https://doi.org/10.3233/SW-170273>, <https://doi.org/10.3233/SW-170273>
7. Gupta, N., Singh, S., Roth, D.: Entity linking via joint encoding of types, descriptions, and context. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, 9–11 September 2017, pp. 2681–2690 (2017). <https://www.aclweb.org/anthology/D17-1284/>
8. Han, X., Sun, L.: An entity-topic model for entity linking. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, 12–14 July 2012, Jeju Island, Korea, pp. 105–115 (2012). <https://www.aclweb.org/anthology/D12-1010/>
9. He, Z., Liu, S., Li, M., Zhou, M., Zhang, L., Wang, H.: Learning entity representation for entity disambiguation. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4–9 August 2013, Sofia, Bulgaria, Volume 2: Short Papers, pp. 30–34 (2013). <https://www.aclweb.org/anthology/P13-2006/>
10. Hoffart, J., et al.: Robust disambiguation of named entities in text. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27–31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL, pp. 782–792. ACL (2011). <https://www.aclweb.org/anthology/D11-1072/>
11. Le, P., Titov, I.: Improving entity linking by modeling latent relations between mentions. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, 15–20 July 2018, Volume 1: Long Papers, pp. 1595–1604 (2018). <https://doi.org/10.18653/v1/P18-1148>, <https://www.aclweb.org/anthology/P18-1148/>
12. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held 5–8 December 2013, Lake Tahoe, Nevada, United States, pp. 3111–3119 (2013). <http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality>



13. Nguyen, D.B., Hoffart, J., Theobald, M., Weikum, G.: Aida-light: high-throughput named-entity disambiguation. In: Proceedings of the Workshop on Linked Data on the Web co-located with the 23rd International World Wide Web Conference (WWW 2014), Seoul, Korea, 8 April 2014. [http://ceur-ws.org/Vol-1184/ldow2014\\_paper\\_03.pdf](http://ceur-ws.org/Vol-1184/ldow2014_paper_03.pdf)
14. Nguyen, T.H., Faucegla, N.R., Rodriguez-Muro, M., Hassanzadeh, O., Gliozzo, A.M., Sadoghi, M.: Joint learning of local and global features for entity linking via neural networks. In: COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, 11–16 December 2016, Osaka, Japan, pp. 2310–2320 (2016). <https://www.aclweb.org/anthology/C16-1218/>
15. Ratinov, L., Roth, D., Downey, D., Anderson, M.: Local and global algorithms for disambiguation to wikipedia. In: The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19–24 June 2011, Portland, Oregon, USA, pp. 1375–1384 (2011). <https://www.aclweb.org/anthology/P11-1138/>
16. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems: Annual Conference on Neural Information Processing Systems 2017, 4–9 December 2017, Long Beach, CA, USA, vol. 30, pp. 5998–6008 (2017). <http://papers.nips.cc/paper/7181-attention-is-all-you-need>
17. Yang, X., et al.: Learning dynamic context augmentation for global entity linking. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, 3–7 November 2019, pp. 271–281 (2019). <https://doi.org/10.18653/v1/D19-1026>



# HyperJOIE: Two-View Hyperbolic Knowledge Graph Embedding with Entities and Concepts Jointly

Jing Dong<sup>1</sup>, Binbin Gu<sup>2</sup>, Jianfeng Qu<sup>1</sup>(✉), An Liu<sup>1</sup>, Lei Zhao<sup>1</sup>,  
Zhigang Chen<sup>3,4</sup>, and Zhixu Li<sup>1</sup>

<sup>1</sup> School of Computer Science and Technology, Soochow University, Suzhou, China  
{jdong, jfqu, anliu, zhaol, zhixuli}@stu.suda.edu.cn

<sup>2</sup> University of California, Irvine, USA  
binbing@uci.edu

<sup>3</sup> iFLYTEK Research, Suzhou, China  
zgchen@iflytek.com

<sup>4</sup> State Key Laboratory of Cognitive Intelligence, iFLYTEK, Hefei, China

**Abstract.** Knowledge graphs have two views: an entity graph in the instance view and a concept graph in the ontology view. Recent studies reveal that modeling the two graphs jointly can benefit the understanding to either one. However, the existing work has flaws on both modelling the hierarchical structures in the Euclidean space, and capturing the deep cross-view interaction between an entity and its corresponding concept. In this paper, we propose to explore hyperbolic space for two-view knowledge graph embedding, which provides more effective and efficient embedding learning mechanism, especially for hierarchical structured knowledge. We also propose to capture the deep cross-view interaction between an entity and its corresponding concept through modeling local structure information from intra-view neighbor nodes with hyperbolic attention mechanism. Finally, we propose to maintain the structural correspondence between the concept graph and the entity graph by first encoding two graphs with the same embedding model respectively and then aligning the two graphs with a hyperbolic transformation. Our empirical study conducted on two benchmark data collections proves that our model outperforms several state-of-the-art two-view knowledge graph embedding models.

**Keywords:** Knowledge representation learning · Hyperbolic attention mechanism · Two-view knowledge graph

## 1 Introduction

Knowledge graph embedding (KGE), which encodes knowledge graph (KG) structures into low-dimensional embedding spaces, has attracted a lot of attention in the past decade [2, 15, 23]. As an effective way to capture the latent

semantic relations of entities and concepts, KGE provides an efficient and systematic solution to various knowledge-driven machine learning tasks such as knowledge graph completion [25], entity typing [26], entity alignment [8] and question answering [3].

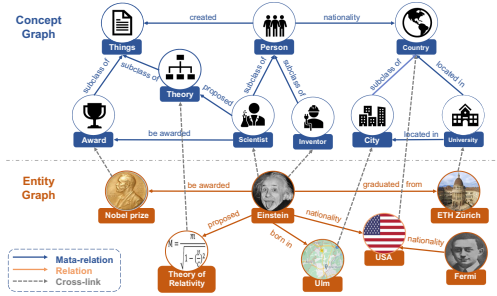


Fig. 1. An example knowledge graph in two views

Normally, a KG can be viewed in two levels as the example shown in Fig. 1, which includes an **entity graph** in the instance level and a **concept graph** in the ontology level. While most traditional KGE models [14, 19] take concepts as special kind of entities, and learn the representations for both concepts and entities in the same way, some other efforts is limited to conduct KGE in a single view, that is, either on the instance-view graph [1, 2] or on the ontology-view graph [7, 9]. Some recent studies [10, 14] reveal that representing a KG from both views provides more comprehensive insights, and thus improves the understanding to both views. A most representative work JOIE [10] proposes to jointly model the two graphs, i.e., entity graph and concept graph, and then characterize the cross-links between concepts and entities with a cross-view association model. While entity embeddings could provide detailed and rich information for their corresponding concepts, a concept embedding also provides a high-level summary of its corresponding entities, which is very helpful to long-tail instances [10]. Some other efforts [14, 22] also works on leveraging concepts as supplementary information to assist entity representation learning.

However, the existing two-view KGE models are flawed in three aspects: (1) It is verified that the characterization of hierarchical structures in the concept meta-relations in the ontology view is crucial for KGE learning [10, 14], where the hierarchical structures include hypernym-hyponym relations such as (*Person*, *sub\_class\_of*, *Scientist*), and composition relations such as (*City*, *part\_of*, *Country*). Nevertheless, the existing work ignores the corresponding hierarchical structures in the instance graph, such as (*BeiJing*, *located\_in*, *China*) and (*The Nobel Prize in Physics (1921)*, *is\_A*, *Nobel Prize*). Thus, the learned entity embeddings are insufficient for the expression of hierarchical relationship. (2) So far, only conventional Euclidean space is employed for doing two-view KGE, but Euclidean space is not suitable for modeling hierarchical patterns, given that

hierarchical patterns requires high-dimensional representation space to preserve different behaviors through different geometric patterns, but the space capacity in Euclidean space expands polynomially [24]. As a result, embedding hierarchical patterns in Euclidean space incurs huge memory costs. (3) The cross-view association model explored by the existing work only characterizes the connections between the entity and its corresponding concept, but ignores the local structure information explored from their intra-view neighbor nodes, which could provide deep cross-view interaction between the entity and its corresponding concept. For instance in Fig. 1, “*Einstein*”’s neighbor node “*Theory of Relativity*” in the entity graph, and “*Scientist*”’s neighbor node “*Theory*” could strengthen the connections between “*Einstein*” and “*Scientist*”.

To address the above flaws, we propose **HyperJOIE**, a novel two-view joint knowledge graph embedding model in a hyperbolic space. Compared to the Euclidean space, the hyperbolic space could offer greater capacity for embedding learning [18], thus is naturally more suitable for expressing hierarchical structures. Thus, the second flaw of the existing models is solved. Besides, modeling KGs in the hyperbolic space usually requires less dimensions than that in the Euclidean space, but can still reach competitive or even better results [5]. Although the hyperbolic space has been employed in single-view KGE [1, 5], it is not applied to two-view KGE yet. To overcome the third flaw of the existing models, we would like to capture the local structure information from intra-view neighbor nodes with a recent proposed attention mechanism in the hyperbolic space [24]. Finally, while concept graph could be regarded as an abstract to the entity graph, the entity graph is an instantiation of the concept graph. In order to maintain the structural correspondence between the two graphs to the greatest extent, we propose to encode the two KGs with the same embedding model and then align the them with a hyperbolic transformation. In such a way, the first flaw of the existing work is handled.

To summarize, our contributions are as follows: (1) This work is the first attempt to explore two-view KGE in hyperbolic space, which provides more effective and efficient embedding learning mechanism, especially for hierarchical structured knowledge. (2) We propose to capture the deep cross-view interaction between the entity and its corresponding concept through modeling local structure information from intra-view neighbor nodes with an attention mechanisms in the hyperbolic space. (3) We propose to maintain the structural correspondence between the concept graph and the entity graph by first encoding the two KGs with the same embedding model respectively and then aligning the two KGs with a hyperbolic transformation.

Our experiments conducted on two datasets verifies the effectiveness of our model comparing to SOTA models.

## 2 Related Work

This section studies both single-view KGE models and two-view joint KGE models, and then covers some recent progress on hyperbolic KGE models.

## 2.1 Single-View KGE Models

Recent years have witnessed plenty of works contributing on the single-view KGE. A survey [11] categorizes KGE into four aspects, including representation space, scoring function, encoding models and auxiliary information. Given the triples  $(h, r, t)$ , where  $r$  is the relation between head entity  $h$  and tail entity  $t$ , the key of KGE is to choose a proper representation space, to devise an effective model encoding triples and to design a plausibility scoring function for optimizing objective. Most models are mainly based on Euclidean space [2, 17, 23], the others basically represent in complex vector space [20, 21], whereas they are all linear embeddings that lack hierarchical expressiveness and spatial capacity. Although linear embedding methods are fairly simple, it is difficult to obtain enough capacity for tree structure data with the same dimensions as hyperbolic embedding. In addition, a few studies focus on embedding concept graph. On2Vec [7] proposes an augment method of the concept graph by replacing fine-grained concepts with coarse-grained concepts in the concept triples.

## 2.2 Two-View Joint KGE Models

Different from single-view KGE, two-view KGE not only trains two graphs respectively but also divides the single-view link prediction task into graph completion and type inference. Recent studies [10, 14, 22] show that the jointly modeling two-view graph can improve KG representation performance in both knowledge graph completion and entity typing. There have been several investigations [22] taking advantage of concepts as additional information to improve KG completion performance. Further, in order to model hierarchical relations, TransC [14] proposes to represent entities and concepts in the same embedding space with cross-view links association. Further, MTransE [8] aims at entity alignment and proposes to learn a transformation across two separate embedding spaces. However, TransC ignores differences in topology and data scale between entity graph and concept graph, and MTransE neglects the internal hierarchy of two graphs.

JOIE [10] is the first and the only joint two-view KGE model with the Euclidean space. This model introduces a hierarchical loss to model hierarchy and a transformation function to capture the correlation between entity graph and concept graph. Nevertheless, JOIE has weak hierarchy capacity since its representation space is the Euclidean space. Besides, owing to encoding two graphs with the uniform parameters, the problem of asynchronous convergence may occur between different modules, which limits the optimization process and is difficult to reach the global optimum.

## 2.3 Hyperbolic KGE Models

The recent progress in hyperbolic graph neural network challenges traditional graph neural network of Euclidean spaces, which fuels a lot of researches on introducing hyperbolic graph neural network into knowledge graph embedding.

**Table 1.** Characteristic properties of Euclidean, and hyperbolic geometries [13].

Geometry	Property	
	Euclidean	Hyperbolic
Curvature $K$	0	$< 0$
Circle length	$2\pi r$	$2\pi \sinh \sqrt{ K }r$
Disk area	$\pi r^2$	$2\pi(\cosh \sqrt{ K }r - 1)$

More recently, several studies on KG completion and alignment reveal that low-dimensional hyperbolic KG embeddings can represent knowledge graph effectively and efficiently. AttH [5], which achieves state-of-the-art results on single KG completion, utilizes hyperbolic geometrical operations (rotation and reflection) to learn KG’s structure and introduce attention mechanism to balance two operations. Moreover, HyperKA [19] introduces hyperbolic translational embeddings within intra-view and train a hyperbolic transformation to capture the cross-link associations. Although these two works demonstrate that embedding knowledge into hyperbolic space can partly improve the performance of KG completion and entity typing, joint KGE learning in hyperbolic space is still an unexplored issue, which is exactly the focus of this paper.

### 3 Problem Formulation and Background

#### 3.1 Problem Formulation

We define the entity graph as  $\mathcal{G}_e = \{\mathcal{E}_e, \mathcal{R}_e, \mathcal{F}_e\}$  and the concept graph as  $\mathcal{G}_c = \{\mathcal{E}_c, \mathcal{R}_c, \mathcal{F}_c\}$ . The entity graph  $\mathcal{G}_e$  and the concept graph  $\mathcal{G}_c$  are two completely independent graphs, where  $\mathcal{E}_e, \mathcal{R}_e$  are the sets of entities and relations and  $\mathcal{E}_c, \mathcal{R}_c$  are the sets of concepts and meta-relations. Correspondingly, we represent the entity triples and concept triples as  $(h_e, r_e, t_e) \in \mathcal{F}_e$  and  $(h_c, r_c, t_c) \in \mathcal{F}_c$ , and the cross-view triples as  $(h_e, t_c) \in \mathcal{F}_{cross}$ . Formally, the task of two-view KGE is to represent the embeddings of entities  $e$ , relations  $r_e$ , concepts  $c$  and meta-relations  $r_c$ , which are represented by the boldfaced  $\mathbf{e}$ ,  $\mathbf{r}_e$ ,  $\mathbf{c}$  and  $\mathbf{r}_c$ , respectively. In order to train and evaluate the model, three triples sets  $(\mathcal{F}_e, \mathcal{F}_c, \mathcal{F}_{cross})$  are split into  $\mathcal{F}^{Train}$ ,  $\mathcal{F}^{Valid}$  and  $\mathcal{F}^{Test}$ , respectively. Embeddings are optimized by the scoring function  $f(\cdot, \cdot)$ , which measures the plausibility of facts.

#### 3.2 Preliminaries on Hyperbolic Geometry

Hyperbolic space is a space with constant negative curvature. As shown in Table 1, non-zero curvature makes hyperbolic geometry different from traditional Euclidean geometry. It can be seen as the calculated metrics of the circle length and the disk area that the hyperbolic geometric space increases exponentially. This property allows hyperbolic space to provide greater spatial capacity

under the same dimensional constraints, which is particularly suitable for forming hierarchies. Further, to facilitate the gradient descent operation, we choose the Poincaré ball model [4] as hyperbolic geometric equivalent model in this paper. Particularly, a  $n$ -dimensional Poincaré ball with negative curve  $-c$  ( $c > 0$ ) is defined as a manifold  $\mathbb{B}_c^n = \{x \in \mathbb{R}^n : c\|x\| < 1\}$ , and we set  $c = 1$  to simplify our work following [18]. Some hyperbolic geometry in this  $n$ -dimensional manifold  $\mathbb{B}_1^n$  are introduced below.

**Möbius Addition and Hyperbolic Distance.** Given two nodes  $\mathbf{u}, \mathbf{v} \in \mathbb{B}_1^n$ , the Möbius addition provides an analogue to Euclidean addition for hyperbolic space, which is:

$$\mathbf{u} \oplus \mathbf{v} = \frac{(1 + 2\langle \mathbf{u}, \mathbf{v} \rangle + \|\mathbf{v}\|^2)\mathbf{u} + (1 - \|\mathbf{u}\|^2)\mathbf{v}}{1 + 2\langle \mathbf{u}, \mathbf{v} \rangle + \|\mathbf{u}\|^2 + \|\mathbf{v}\|^2} \quad (1)$$

Further, the hyperbolic distance on  $\mathbb{B}_1^n$  between  $\mathbf{u}$  and  $\mathbf{v}$  is given by:

$$d_{\mathbb{B}_1^n}(\mathbf{u}, \mathbf{v}) = 2 \tanh^{-1}(\|-\mathbf{u} \oplus \mathbf{v}\|) \quad (2)$$

**Transformation.** The logarithmic map  $\log_0(\cdot)$  maps the manifold  $\mathbb{B}_c^n$  to the tangent space  $\mathcal{T}_0\mathbb{B}_c^n$  at the origin 0, where  $\mathcal{T}_0\mathbb{B}_c^n$  is a  $n$ -dimensional Euclidean space for  $\mathbb{B}_c^n$ , i.e.  $\mathcal{T}_0\mathbb{B}_c^n = \mathbb{R}^n$  [1]. Conversely, the exponential map  $\exp_0(\cdot)$  projects  $\mathcal{T}_0\mathbb{B}_c^n$  onto  $\mathbb{B}_c^n$  at the origin. Specifically, for each point  $\mathbf{u} \in \mathbb{B}_1^n, \mathbf{y} \in \mathcal{T}_0\mathbb{B}_c^n$ , these maps are formulated as:

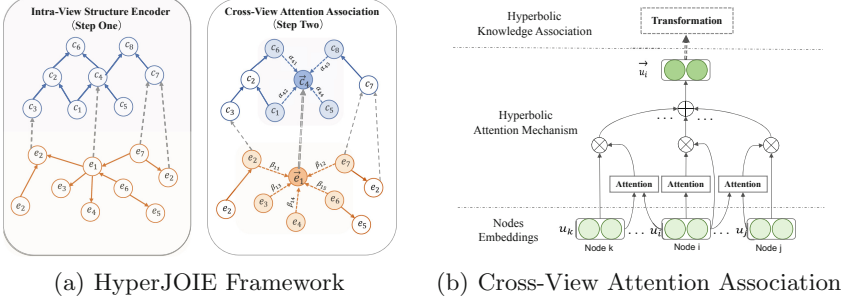
$$\begin{aligned} \log_0(\mathbf{u}) &= \tanh^{-1}(\|\mathbf{u}\|) \frac{\mathbf{u}}{\|\mathbf{u}\|} \\ \exp_0(\mathbf{y}) &= \tanh(\|\mathbf{y}\|) \frac{\mathbf{y}}{\|\mathbf{y}\|} \end{aligned} \quad (3)$$

With such two inverse maps, hyperbolic space can apply analogous Euclidean operations. Specifically, through mapping the hyperbolic vectors to the tangent space with logarithmic map, the Euclidean linear matrix-vector multiplication can be applied to the transformation matrix  $\mathbf{M}$ , and then the transformed vectors can be projected back on another manifold with exponential map. This transformation that projects a vector  $\mathbf{u} \in \mathbb{B}_1^n$  into  $\mathbb{B}_1^n$  [19] is denoted as Möbius matrix-vector multiplication:

$$\mathbf{M} \otimes \mathbf{u} = \exp_0(\mathbf{M} \log_0(\mathbf{u})) \quad (4)$$

## 4 Methodology

In this section, we present our model **HyperJOIE**—a two-view hyperbolic KGE method with jointly learning entities and concepts. Figure 2(a) demonstrates the framework of **HyperJOIE** which consists of two components: intra-view structure encoder and cross-view attention association. In the following, we describe the two modules in details.



**Fig. 2.** (a) is the framework of HyperJOIE which first encodes intra-view KGs and then associates entities and concepts with aggregated hyperbolic attention representation. (b) is the framework of hyperbolic attention associations.

#### 4.1 Intra-view Model

The goal of the intra-view model is to encode two independent KGs into two embedding spaces while preserving original structural information. Since hierarchical structures do not only appear in concept mapping, we use an advanced hyperbolic single-view model AttH [5] to represent the entity and concept graph respectively, so as to represent the hierarchical relations inside both KGs. In the following, we describe the AttH model.

AttH utilizes two kinds of hyperbolic isometries—rotations and reflections, to respectively model compositions and anti-symmetric relations.  $\Theta_r := (\theta_{r,i})_{i \in \{1, \dots, \frac{n}{2}\}}$  are rotation parameters and  $\Phi_r := (\phi_{r,i})_{i \in \{1, \dots, \frac{n}{2}\}}$  are reflection parameters, where  $n$  is the embedded dimension. Through  $2 \times 2$  transformation matrices  $G^\pm(\theta)$ , AttH parameterizes rotation and reflection with the form of a block-diagonal matrices as  $\text{Rot}(\Theta_r)$  and  $\text{Ref}(\Phi_r)$ . Then it applies relation-specific rotations and reflections to the head embedding (Eq. 6), using  $H$  represent the embeddings in the Hyperbolic space.

$$\begin{aligned} \text{Rot}(\Theta_r) &= \text{diag}(G^+(\theta_{r,1}), \dots, G^+(\theta_{r, \frac{n}{2}})), \\ \text{Ref}(\Phi_r) &= \text{diag}(G^-(\phi_{r,1}), \dots, G^-(\phi_{r, \frac{n}{2}})), \\ \text{where } G^\pm(\theta) &:= \begin{bmatrix} \cos(\theta) \mp \sin(\theta) \\ \sin(\theta) \pm \cos(\theta) \end{bmatrix}. \end{aligned} \quad (5)$$

$$\mathbf{q}_{\text{Rot}}^H = \text{Rot}(\Theta_r) \mathbf{u}_h^H, \quad \mathbf{q}_{\text{Ref}}^H = \text{Ref}(\Phi_r) \mathbf{u}_h^H \quad (6)$$

In order to balance the combination of rotations and reflections, AttH takes advantage of the convenience of attention calculation in tangent space. It projects hyperbolic representations to tangent space representations  $\mathbf{u}^E = \log_0(\mathbf{u}^H)$  to gets attention weights  $\alpha_u, \alpha_v$ , and then gets a weighted average combination as:

$$(\alpha_u, \alpha_v) = \text{Softmax}(\mathbf{a}^T \mathbf{u}^E, \mathbf{a}^T \mathbf{v}^E) \quad (7)$$

$$\text{Att}(\mathbf{u}^H, \mathbf{v}^H; \mathbf{a}) = \exp_0(\alpha_u \mathbf{u}^E + \alpha_v \mathbf{v}^E) \quad (8)$$



After combining two hyperbolic geometric representations ( $\mathbf{q}_{\text{Rot}}^H$  and  $\mathbf{q}_{\text{Ref}}^H$ ), AttH utilizes a hyperbolic translation to get the translated head embedding  $Q(h, r)$ . Finally, based on the hyperbolic distance, the score function compares the translated head embeddings to the target tail embeddings (Eq. 10).

$$Q(h, r) = \text{Att}(\mathbf{q}_{\text{Rot}}^H, \mathbf{q}_{\text{Ref}}^H; \mathbf{a}_r) \oplus \mathbf{r}^H \quad (9)$$

$$f(h, r, t) = -d_{\mathbb{B}}(Q(h, r), \mathbf{t}^H)^2 + b_h + b_t \quad (10)$$

where  $(b_u)_{u \in \mathcal{E}}$  are node biases which act as margins in the scoring function [1, 5]. Then, the intra-view component is trained by minimizing the full cross-entropy loss, with negative samples:

$$\mathcal{L}_{\text{intra}} = \sum_{t' \sim \mathcal{U}(\mathcal{E})} \log(1 + \exp(y'_t f(h, r, t'))) \quad (11)$$

where  $y'_t = \begin{cases} -1 & \text{if } t' = t \\ 1 & \text{otherwise} \end{cases}$

## 4.2 Cross-View Model

The cross-view model component aims at modeling the associations between entity graph and concept graph. Figure 2(b) demonstrates the framework of hyperbolic cross-view attention associations. We take the concepts and entities embeddings encoded by the intra-view component as input, and then aggregate the latent information with hyperbolic attention mechanism for concepts and entities respectively. Finally, the hyperbolic knowledge association is adopted to associate the aggregated entities and aggregated concepts. Next, we introduce the hyperbolic attention mechanism and the hyperbolic knowledge association in details.

**Hyperbolic Attention Mechanism.** The attention mechanism is helpful to capture the high-order proximity of nodes based on local neighborhood information. Based on this, we utilize hyperbolic attention mechanism to aggregate the latent representations. Inspired by the recent proposal of HAT [24], we perform the self-attention mechanism on the nodes to capture the deep cross-view interactions. The attention value  $\alpha_{ij}$  indicates the importance of node  $j$  to node  $i$ , which is measured by the hyperbolic distance from node  $i$  to node  $j$ . Given two nodes  $\mathbf{u}_i, \mathbf{u}_j \in \mathbb{B}_1^n$ , the attention weight  $\alpha_{ij}$  is computed as:

$$\begin{aligned} \alpha_{ij} &= -d_{\mathbb{B}}(\mathbf{u}_i, \mathbf{u}_j) \\ &= -\tanh^{-1}(|-\mathbf{u}_i \oplus \mathbf{u}_j|) \end{aligned} \quad (12)$$

According to HAT [24] using hyperbolic distance to measure attention weights can not only preserve the structure transitivity among nodes but also maintain the original characteristics of the nodes. This is because the attention weight of a node to itself is  $\alpha_{ii} = -d(\mathbf{u}_i, \mathbf{u}_i) = 0$  meaning that a node is always

the nearest point to itself among its neighbors. Moreover, the relative attention weights  $w_{ij}$  is calculated by *softmax* function over all the neighbors of node  $i$  (including itself) to get the normalized values:

$$w_{ij} = \frac{\exp(\alpha_{ij})}{\sum_{k \in N_i} \exp(\alpha_{ik})} \quad (13)$$

The aggregated representation  $\vec{\mathbf{u}}_i$  is a linear combination of the normalized weights  $w_{ij}$  and the latent representations of all the nodes  $j \in N_i$ , which is denoted as follows:

$$\vec{\mathbf{u}}_i = \sigma\left(\sum_{j \in N_i}^{\oplus} w_{ij} \otimes \mathbf{u}_j\right) \quad (14)$$

where the  $\sum_{j \in N_i}^{\oplus}$  is the accumulation of Möbius addition. Different from HAT, HyperJOIE adopts TanH as the nonlinearity function  $\sigma$  and calculates the weighted sum by the Möbius scalar multiplication. Formally, given the weight  $w_{ij} \in \mathbb{R}$  of the node  $u_i \in \mathbb{B}_1^n$ , the weighted sum is denoted as:

$$w_{ij} \otimes \mathbf{u}_i = \tanh(w_{ij} \tanh^{-1}(\|\mathbf{u}_i\|)) \frac{\mathbf{u}_i}{\|\mathbf{u}_i\|} \quad (15)$$

In addition, with the help of the logarithmic map, we transform the weighted representation  $w_{ij} \otimes \mathbf{u}_i$  into the tangent space, where we can utilize the linear combination as in the Euclidean spaces, i.e.  $\sum_{j \in N_i} \log_0(w_{ij} \otimes \mathbf{u}_j)$  [6]. After that, we use the exponential map to project the representation back to the hyperbolic space, simplifying the final representation as follows:

$$\vec{\mathbf{u}}_i = \sigma\left(\exp_0\left(\sum_{j \in N_i} \log_0(w_{ij} \otimes \mathbf{u}_j)\right)\right) \quad (16)$$

**Hyperbolic Knowledge Association.** Given a pair of cross-link triples  $(e, c) \in \mathcal{F}_{cross}$ , we firstly utilize the Möbius transformation to project the aggregated entity  $\vec{\mathbf{e}}$  to the concept embedding space. For each cross-link triple, we hope the transformed aggregated entity vector close to the aggregated concept vector  $\vec{\mathbf{c}}$  in concept embedding space, so we choose the hyperbolic distance to design the scoring function. Inspired by [1], we add the biases of entity  $b_e \in \mathbb{R}$  and the biases of concept  $b_c \in \mathbb{R}$  to manifest the margins among instantiated entities of the same concept. Finally, we define the basis score function for the cross-view attention association as follows:

$$f(e, c) = -d_{\mathbb{B}}(\mathbf{M} \otimes \vec{\mathbf{e}}, \vec{\mathbf{c}}) + b_e + b_c \quad (17)$$

where  $\mathbf{M} \in \mathcal{R}_{n \times m}$  transforms the hyperbolic vectors from  $\mathbb{B}_1^n$  to  $\mathbb{B}_1^m$ .

We choose the full cross-entropy loss with uniform negative sampling to train cross-view component, where the negative triples are sampled uniformly from all possible triples by perturbing the tail concept. The cross-view attention association loss is given as:

$$\mathcal{L}_{cross} = \sum_{c' \sim \mathcal{U}(\mathcal{E}_c)} \log(1 + \exp(y_{c'} f(e, c'))),$$

$$\text{where } y'_c = \begin{cases} -1 & \text{if } c' = c \\ 1 & \text{otherwise} \end{cases} \quad (18)$$

### 4.3 Loss Function and Training

The total loss is the combination of the intra-view component and the cross-view component, which is defined as follows:

$$\mathcal{L} = \mathcal{L}_{intra}^{\mathcal{G}_e} + \mathcal{L}_{intra}^{\mathcal{G}_c} + \mathcal{L}_{cross} \quad (19)$$

In order to avoid the challenging optimization in hyperbolic space, we define all parameters in the tangent space at the origin and optimize parameters with the Adam optimizer [12]. The parameters can be recovered to the hyperbolic space with the exponential map [5]. We follow a two-step training procedure: For one epoch, (1) first train our intra-view component  $\mathcal{L}_{intra}^{\mathcal{G}_e}$  and  $\mathcal{L}_{intra}^{\mathcal{G}_c}$  respectively and then (2) optimize the cross-view loss  $\mathcal{L}_{cross}$  in the successive step.

## 5 Experiments

### 5.1 Experimental Setup

**Datasets.** We use two benchmark datasets for evaluation: YAGO26K-906 and DB111K-174 [10], which are extracted from YAGO and DBpedia, respectively. Each dataset includes entity graph, concept graph and cross-view links. Table 2 provides statistics of all datasets used. We use the data splits provided by [10] in order to compare with previous work.

**Evaluation Metrics.** During the test, we use different scoring function to test different tasks. Specifically, we use Eq. 10 for intra-view task and Eq. 17 for cross-view task. Similar to previous work, we compute two ranking-based metrics: (1) mean reciprocal rank (MRR) and (2) hits at  $K$  ( $H@K$ ,  $K \in 1, 3, 10$ ). Following previous standard evaluation protocol [2], we filter out all true triples from the dataset to put penalty on the true triples which are predicted with low rankings.

**Hyper-Parameter Settings.** As for the hyper-parameters, we conduct a grid search to detect the learning rate and batch size, using the validation set to select the best hyper-parameters. We search the learning rate among  $\{0.00005, 0.0001, 0.0002, 0.0005\}$ , and batch size among  $\{200, 1000, 2000, 5000\}$ . The optimal configurations are as follows: On YAGO26K-906, the learning rate  $\alpha = 0.0001$  and batch size  $B = 1000$ ; while on DB111K-174, the learning rate  $\alpha = 0.0001$  and batch size  $B = 5000$ .

**Table 2.** Statistics of datasets.

Datasets		# Nodes	# Rel	# Triples	# Cross-links
YAGO26K-906	Ent	26,078	34	390,738	9,962
	Cpt	906	30	8,962	
DB111K-174	Ent	111,762	305	863,643	99,748
	Cpt	174	20	763	

**Table 3.** KG completion results

Dataset	YAGO26K-906						DB111K-174					
	Entity			Concept			Entity			Concept		
Metrics	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
TransE(base)	0.195	0.141	0.345	0.145	0.123	0.206	0.327	0.223	0.490	0.313	0.232	0.469
TransE(all)	0.189	0.137	0.350	0.189	0.147	0.244	0.318	0.227	0.481	0.539	0.479	0.618
DistMult(base)	0.253	0.229	0.288	0.197	0.177	0.251	0.265	0.256	0.276	0.235	0.152	0.291
DistMult(all)	0.288	0.240	0.312	0.156	0.143	0.165	0.280	0.272	0.297	0.501	0.455	0.647
HolE(base)	0.265	0.259	0.283	0.192	0.187	0.203	0.301	0.292	0.315	0.227	0.189	0.328
HolE(all)	0.252	0.242	0.266	0.138	0.113	0.114	0.290	0.287	0.303	0.432	0.388	0.560
RotatE(base)	0.652	0.563	0.806	0.252	0.185	0.376	0.435	0.372	0.550	0.506	0.422	0.669
RotatE(all)	0.567	0.465	0.753	0.304	0.234	0.436	0.337	0.287	0.427	0.445	0.377	0.552
AttH(base)	0.667	0.589	0.804	0.250	0.199	0.344	0.425	0.348	0.566	0.520	0.429	0.701
AttH(all)	0.610	0.519	0.549	0.262	0.203	0.380	0.372	0.309	0.595	0.475	0.403	0.675
TransC	0.252	0.157	0.378	—	—	—	0.359	0.248	0.493	—	—	—
JOIE	0.327	0.224	0.524	0.263	0.209	0.385	0.473	0.338	0.714	0.622	0.581	0.797
HyperKA	0.314	0.161	0.613	0.269	0.169	0.429	0.302	0.147	0.387	0.241	0.200	0.447
HyperJOIE(no_att)	<u>0.735</u>	<u>0.683</u>	<u>0.826</u>	<u>0.523</u>	<u>0.413</u>	<u>0.739</u>	<u>0.635</u>	<u>0.584</u>	<u>0.729</u>	<u>0.725</u>	<u>0.645</u>	<u>0.870</u>
HyperJOIE(with_att)	<b>0.745</b>	<b>0.697</b>	<b>0.830</b>	<b>0.587</b>	<b>0.497</b>	<b>0.761</b>	<b>0.708</b>	<b>0.656</b>	<b>0.804</b>	<b>0.768</b>	<b>0.667</b>	<b>0.906</b>

## 5.2 KG Completion

**Baselines.** Although lots of methods contribute on single-view KG completion, only a few methods have been applied to the two-view KG completion task with jointly learning entities and concepts. In this work, we compare HyperJOIE to SotA single-view KGE model AttH [5], SotA two-view KGE method JOIE [10], SotA two-view KG alignment method HyperKA [19] and five other baseline methods TransE [2], DistMult [23], HolE [16], TransC [14] and RotatE [20]. For JOIE, we take its best performance for fair comparison. For the other baselines, we set the entity dimension  $n$  to 300 and the concept dimension  $m$  to 50. We also tried to set higher dimensions, but we did not observe further improvements.

**Ablations.** To analyze the benefits of hyperbolic joint framework, we follow the JOIE which deploys two variants of AttH: (1) We train each independent knowledge graph separately without cross-view associations, denoted as (*base*); (2) We train all triples in  $\mathcal{F}_e^{Train}$ ,  $\mathcal{F}_c^{Train}$  and  $\mathcal{F}_{cross}^{Train}$ , denoted as (*all*). Moreover, to evaluate the role of hyperbolic attention in the intra-view component, we report scores for variants of HyperJOIE training cross-view associations

only with hyperbolic transformation, denoted as (*no\_att*). Furthermore, we re-evaluate the intra-view KG completion performance of HyperKA [19] to estimate the intra-view embedding ability of the SotA entity typing method.

**Results and Discussion.** We report the KG completion results in Table 3. As can be seen from Table 3, HyperJOIE outperforms all the other methods. Compared with the linear embedding methods (TransE, DistMult, HolE, TransC and JOIE) in Euclidean space and RotatE [20] in the complex space, HyperJOIE is extremely dominant regarding the performance. For example, the H@1 score of HyperJOIE on entity graph of YAGO26K-906 reaches 0.697, surpassing JOIE by 0.473 and RotatE by 0.134. Also, compared with hyperbolic single-view KGE model, HyperJOIE also achieves better performance on both entity graph and concept graph. Specifically, HyperJOIE improves AttH by 0.108 and 0.238 in H@1 on entity and concept graph completion of YAGO26K-906 respectively. This illustrates that hyperbolic projections for cross-view associations can enhance the embedding representation of the intra-view graphs, especially in concept graph. In addition, we evaluate the performance of HyperJOIE without hyperbolic attention association HyperJOIE(*no\_att*). We find that the version with hyperbolic attention associations HyperJOIE(*with\_att*) shows slight improvement. This indicates that the hyperbolic attention mechanism also contributes to the representation of the intra-view KGs. Moreover, by comparing the performance of AttH and HyperKA on KG completion task, we find that even the SotA entity alignment approach is still lacking in terms of intra-view KGs, which demonstrates the importance of intra-view structure encoder.

### 5.3 Entity Typing

**Baselines.** We compare our method with the SotA entity typing methods (JOIE and HyperKA), entity alignment method MTransE [8], and five other single-view KGE methods, i.e. TransE [2], DistMult [23], HolE [16], TransC [14] and RotatE [20]. The dimension settings are same as Sect. 5.2.

**Ablations.** We deploys two variants of intra-view structure encoder to detect the impact of single-view KGE methods with different learning abilities on cross-view association: (1) We randomly initialize the embedding of entities and concepts without any intra-view structure modeling, denoted as (*basic*); (2) We transpose TransE [2] into hyperbolic space as encoder, denoted as (*simple*). Additionally, to verify the effect of hyperbolic association on cross-view links, we consider an alternative of our method without hyperbolic attention mechanism, denoted as (*no\_att*).

**Results and Discussion.** As shown in Table 4, we observe that HyperJOIE outperforms both HyperKA and JOIE on MRR and H@1 on YAGO26K-906 in entity typing task. However, on the DB111K-174, our method is only 0.007

**Table 4.** Entity typing results

Dataset	YAGO26K-906			DB111K-174		
Metrics	MRR	H@1	H@3	MRR	H@1	H@3
TransE	0.144	0.073	0.353	0.503	0.437	0.608
DistMult	0.411	0.361	0.553	0.551	0.680	0.551
HolE	0.395	0.348	0.548	0.504	0.654	0.504
RotatE	0.479	0.429	0.507	0.382	0.309	0.434
AttH	0.476	0.430	0.502	0.498	0.447	0.535
MTransE	0.689	0.609	0.776	0.672	0.599	0.813
JOIE	0.897	0.856	<b>0.959</b>	<u>0.857</u>	0.756	<b>0.959</b>
HyperKA	<u>0.913</u>	<u>0.871</u>	<u>0.948</u>	<b>0.863</b>	<b>0.789</b>	<u>0.927</u>
HyperJOIE( <i>basic_no_att</i> )	0.534	0.398	0.594	0.470	0.336	0.558
HyperJOIE( <i>basic_with_att</i> )	0.804	0.733	0.853	0.546	0.407	0.637
HyperJOIE( <i>simple_no_att</i> )	0.598	0.450	0.666	0.646	0.507	0.749
HyperJOIE( <i>simple_with_att</i> )	0.836	0.765	0.890	0.651	0.523	0.745
HyperJOIE( <i>no_att</i> )	0.870	0.806	0.922	0.735	0.642	0.887
HyperJOIE( <i>with_att</i> )	<b>0.916</b>	<b>0.880</b>	0.942	0.846	0.782	0.907

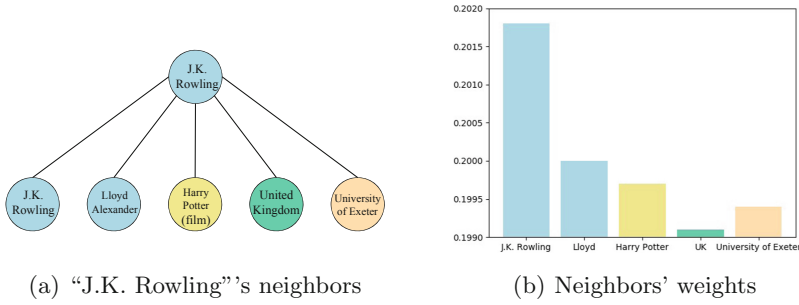
less than the SotA HyperKA on the MRR metric. To estimate the impact of different intra-view structure encoder, we compare HyperJOIE (*basic\_no\_att*), (*simple\_no\_att*) and (*no\_att*), and find that HyperJOIE(*no\_att*) achieves the best performance. This illustrates that high-quality intra-view embedding is beneficial for cross-view associations.

To further evaluate the role of hyperbolic attention and hyperbolic association, we compare HyperJOIE(*with\_att*) with AttH and HyperKA on entity typing task, and find that HyperJOIE is superior to both hyperbolic KGE methods on YAGO26K-906, i.e. the H@1 metric of our methods surpasses HyperKA by 0.009 and surpasses AttH by 0.450 in YAGO26K-906. As for the comparison between (*no\_att*) and (*with\_att*), we can see that hyperbolic attention can enhance entity typing performance.

Further, to figure out the reasons of the poor performance of the hyperbolic attention mechanism on the DB111K-174, we compare the two datasets and find that the concept graph of DB111K-174 is too small and sparse to provide enough latent neighbor information. The concept graph only has 763 triples (8,962 in YAGO26K-906) and an average of 4.38 triples per concept (9.89 in YAGO26K-906). But for a relatively complete concept graph, hyperbolic attention can significantly empower knowledge associations of the two-view KG.

#### 5.4 Case Study

For case study, we visualize the attention weights of “J.K. Rowling”’s neighbors in DB111K-174 dataset as an illustrative example of the hyperbolic attention.



**Fig. 3.** Neighbors of “J.K. Rowling” and its attention weights

As shown in Fig. 3(a), “J.K. Rowling” has 4 neighbors, and the types of nodes are identified by its colors. From Fig. 3(b), we observe that “J.K. Rowling” gets highest weight and “Lloyd” gets the second attention weight. This means the hyperbolic attention mechanism can capture the type information.

## 6 Conclusion and Future Work

We propose a novel hyperbolic two-view knowledge graph embedding method with hyperbolic attention associations, which is the first attempt to explore two-view KGE in hyperbolic space. Our method almost surpasses SotA baselines on both KG completion and entity typing. Our future work will explore the application of a hyperbolic joint learning framework to multi-modal graph representation learning.

**Acknowledgments.** This work was supported by the National Key R&D Program of China (No. 2018AAA0101900), the Priority Academic Program Development of Jiangsu Higher Education Institutions, National Natural Science Foundation of China (Grant No. 62072323, 61632016, 62102276), Natural Science Foundation of Jiangsu Province (No. BK20191420).

## References

1. Balazevic, I., Allen, C., Hospedales, T.: Multi-relational poincaré graph embeddings. In: *Advances in Neural Information Processing Systems*, pp. 4463–4473 (2019)
2. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: *Advances in Neural Information Processing Systems*, pp. 2787–2795 (2013)
3. Bordes, A., Weston, J., Usunier, N.: Open question answering with weakly supervised embedding models. In: Calders, T., Esposito, F., Hüllermeier, E., Meo, R. (eds.) *ECML PKDD 2014. LNCS (LNAI)*, vol. 8724, pp. 165–180. Springer, Heidelberg (2014). [https://doi.org/10.1007/978-3-662-44848-9\\_11](https://doi.org/10.1007/978-3-662-44848-9_11)

4. Cannon, J.W., Floyd, W.J., Kenyon, R., Parry, W.R., et al.: Hyperbolic geometry. *Flavors Geom.* **31**, 59–115 (1997)
5. Chami, I., Wolf, A., Juan, D.C., Sala, F., Ravi, S., Ré, C.: Low-dimensional hyperbolic knowledge graph embeddings. arXiv preprint [arXiv:2005.00545](https://arxiv.org/abs/2005.00545) (2020)
6. Chami, I., Ying, R., Ré, C., Leskovec, J.: Hyperbolic graph convolutional neural networks. *Adv. Neural. Inf. Process. Syst.* **32**, 4869 (2019)
7. Chen, M., Tian, Y., Chen, X., Xue, Z., Zaniolo, C.: On2vec: embedding-based relation prediction for ontology population. In: *Proceedings of the 2018 SIAM International Conference on Data Mining*, pp. 315–323. SIAM (2018)
8. Chen, M., Tian, Y., Yang, M., Zaniolo, C.: Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. arXiv preprint [arXiv:1611.03954](https://arxiv.org/abs/1611.03954) (2016)
9. Guo, S., Wang, Q., Wang, L., Wang, B., Guo, L.: Jointly embedding knowledge graphs and logical rules. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 192–202 (2016)
10. Hao, J., Chen, M., Yu, W., Sun, Y., Wang, W.: Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1709–1719 (2019)
11. Ji, S., Pan, S., Cambria, E., Marttinen, P., Yu, P.S.: A survey on knowledge graphs: representation, acquisition and applications. arXiv preprint [arXiv:2002.00388](https://arxiv.org/abs/2002.00388) (2020)
12. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)
13. Kriukov, D., Papadopoulos, F., Kitsak, M., Vahdat, A., Boguná, M.: Hyperbolic geometry of complex networks. *Phys. Rev. E* **82**(3), 036106 (2010)
14. Lv, X., Hou, L., Li, J., Liu, Z.: Differentiating concepts and instances for knowledge graph embedding. arXiv preprint [arXiv:1811.04588](https://arxiv.org/abs/1811.04588) (2018)
15. Nathani, D., Chauhan, J., Sharma, C., Kaul, M.: Learning attention-based embeddings for relation prediction in knowledge graphs. arXiv preprint [arXiv:1906.01195](https://arxiv.org/abs/1906.01195) (2019)
16. Nickel, M., Rosasco, L., Poggio, T.: Holographic embeddings of knowledge graphs. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30 (2016)
17. Nickel, M., Tresp, V., Kriegel, H.P.: A three-way model for collective learning on multi-relational data. In: *ICML*, vol. 11, pp. 809–816 (2011)
18. Nickel, M., Kiela, D.: Poincaré embeddings for learning hierarchical representations. In: *Advances in Neural Information Processing Systems*, pp. 6338–6347 (2017)
19. Sun, Z., Chen, M., Hu, W., Wang, C., Dai, J., Zhang, W.: Knowledge association with hyperbolic knowledge graph embeddings. arXiv preprint [arXiv:2010.02162](https://arxiv.org/abs/2010.02162) (2020)
20. Sun, Z., Deng, Z.H., Nie, J.Y., Tang, J.: Rotate: knowledge graph embedding by relational rotation in complex space. arXiv preprint [arXiv:1902.10197](https://arxiv.org/abs/1902.10197) (2019)
21. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: *International Conference on Machine Learning (ICML)* (2016)
22. Xie, R., Liu, Z., Sun, M.: Representation learning of knowledge graphs with hierarchical types. In: *IJCAI*, pp. 2965–2971 (2016)
23. Yang, B., Yih, W.t., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. arXiv preprint [arXiv:1412.6575](https://arxiv.org/abs/1412.6575) (2014)



24. Zhang, Y., Wang, X., Jiang, X., Shi, C., Ye, Y.: Hyperbolic graph attention network. arXiv preprint [arXiv:1912.03046](https://arxiv.org/abs/1912.03046) (2019)
25. Zhang, Z., Cai, J., Zhang, Y., Wang, J.: Learning hierarchy-aware knowledge graph embeddings for link prediction. In: AAAI, pp. 3065–3072 (2020)
26. Zhao, Y., Zhang, A., Xie, R., Liu, K., Wang, X.: Connecting embeddings for knowledge graph entity typing. arXiv preprint [arXiv:2007.10873](https://arxiv.org/abs/2007.10873) (2020)



# IOPE: Interactive Ontology Population and Enrichment Guided by Ontological Constraints

Shadi Baghernezhad-Tabasi<sup>1</sup>(✉), Loïc Druette<sup>2</sup>, Fabrice Jouanot<sup>1</sup>,  
Celine Meurger<sup>2</sup>, and Marie-Christine Rousset<sup>1,3</sup>

<sup>1</sup> Université Grenoble Alpes, CNRS, LIG, Grenoble, France  
{shadi.baghernezhad-tabasi,fabrice.jouanot,  
marie-christine.rousset}@univ-grenoble-alpes.fr

<sup>2</sup> Université Claude Bernard Lyon 1, SAMSEI, Lyon, France  
{loic.druette,celine.meurger}@univ-lyon1.fr

<sup>3</sup> Institut Universitaire de France, Paris, France

**Abstract.** Specialized ontologies are constructed to capture the skills of experienced experts, with the goal of sharing them with a larger community of trainees and less experienced experts in the domain. A challenging task in the engineering pipeline of specialized ontologies is *ontology updating*, which encompasses enrichment and population. Domain experts require an understanding of the RDF notation and OWL semantics to update their ontologies, which is not often the case. In this paper, we present IOPE, an interactive framework for the automatic construction of a Graphical User Interface (GUI) to support the controlled update process, by enabling the experts to easily interact with their ontology. We contribute a set of “mapping rules” to transform the ontological constraints into interactive widgets organized in the form of pre-filled Web pages in the GUI, and a set of “binding rules” to transform the expert interactions into RDF graphs, and hence perform the updates. In an extensive set of experiments, we illustrate the efficacy of IOPE in empowering medical experts to update their specialized ontology.

## 1 Introduction

Ontologies are the backbone of many information systems that require access to structured knowledge. By their very nature, real world ontologies are dynamic artifacts that evolve both in their structure (i.e., the data model) and their content (i.e., instances). Keeping them up-to-date is a critical operation for most applications which rely on semantic Web technologies. Ontology updates encompass both *enrichment* and *population*. Ontology enrichment is the task of extending an existing data model of an ontology with additional concepts and semantic relations, while ontology population is the task of adding new instances of concepts to the ontology, through domain documentations. Ontology updates are typically performed in an exploratory and manual fashion, as the non-documented knowledge of the domain expert is required to be taken into

consideration. However, these manual updates put burden on the experts and render the whole ontological ecosystem inefficient. In this paper, we advocate for an alternative and more effective approach, and propose to handle updates automatically through a few interactions with the expert, using a Graphical User Interface (GUI).

The challenges associated to interaction-based automatic updates are two-fold: (i) While ontologies are typically represented in the form of graphs, it is inherently difficult and counterintuitive to provide a graphical graph-based representation of ontologies for the consumption of experts. While there exist several methods to visualize a graph structure [8, 10], the outcome is often hard to digest by domain experts. (ii) It is unclear how experts should perform ontology updates through the interactions, without the prior knowledge of the formal syntax and the semantics of ontology languages.

In this paper, we demonstrate IOPE (Interactive Ontology Population and Enrichment), a framework for the automatic construction of a Graphical User Interface (GUI) using *prefilled Web forms*. We leverage Web forms as a natural interaction means to tackle the challenge of counterintuitive ontology representations. IOPE generates and prefills the Web forms from *ontological constraints*, which support the controlled update process of a given ontology. While IOPE is generic and can be applied to ontologies from a variety of domains, we employ an ontology called ONTOSAMSEI [4] as a use case, whose content helps the domain experts design teaching units for learning skills in simulation-based Medicine. ONTOSAMSEI is a hierarchy of classes and properties enriched by ontological constraints on those classes and properties, that convey the constraints that will have to be fulfilled by their future sub-classes, sub-properties or instances. We show how to exploit such ontological constraints as a source of guidance for (possibly less experienced) educators willing to design their own simulation sessions, hence addressing the challenge of expert noviceship. In [3], we present practical examples of the application of IOPE on ONTOSAMSEI. Moreover, ONTOSAMSEI's IOPE GUI is accessible via the following link (in French): <http://iope.tabasi.info>.

The paper is organized as follows. Section 2 describes the formal background of the ontologies that we consider. Section 3 describes our methodology for the automatic construction of a GUI from an input ontology, and its usage for guiding its update (population and enrichment). Section 4 summarizes the evaluation conducted to assess the added value of the GUI for ontology updating. Section 5 is dedicated to related works while Sect. 6 concludes the paper.

## 2 Formal Background

An ontology is a shared formalization of a domain of interest based on a structured vocabulary made of classes, properties and instances. Ontological constraints are declared on classes and properties to constrain their formal semantics to fit with their actual meaning in the domain of application. Then, factual statements can be added to describe specific entities as instances of classes with

**Table 1.** RDFS and OWL constraints considered in this paper

Type	Shortened syntax	Semantics
Class specialization	$(C \text{ rdfs:subClassOf } D)$	$\forall i ((i \text{ rdf:type } C) \Rightarrow (i \text{ rdf:type } D))$
Property specialization	$(p \text{ rdfs:subPropertyOf } q)$	$\forall i \forall j ((i \text{ p } j) \Rightarrow (i \text{ q } j))$
Domain restriction	$(p \text{ rdfs:domain } C)$	$\forall i \forall j ((i \text{ p } j) \Rightarrow (i \text{ rdf:type } C))$
Range restriction	$(p \text{ rdfs:range } D)$	$\forall i \forall j ((i \text{ p } j) \Rightarrow (j \text{ rdf:type } D))$
Value restriction	$(C \text{ p owl:hasValue } v)$	$\forall i ((i \text{ rdf:type } C) \Rightarrow (i \text{ p } v))$
Alternative values restriction	$(C \text{ p owl:oneOf } [v_1, \dots, v_n])$	$\forall i ((i \text{ rdf:type } C) \Rightarrow \bigvee_{k \in [1..n]} (i \text{ p } v_k))$
Cardinality restriction	$(C \text{ p min } k \text{ D})$	$\forall i ((i \text{ rdf:type } C) \Rightarrow \exists o_1, \dots, o_k (\bigwedge_{i,j \in [1..k]} o_i \neq o_j \wedge \bigwedge_{j \in [1..k]} (o_j \text{ rdf:type } D) \wedge (i \text{ p } o_j)))$

specific values for some properties. The ontological constraints are defined in RDFS<sup>1</sup> and OWL<sup>2</sup>, and described as RDF graphs.

## 2.1 RDF Format

Let  $I$ ,  $L$  and  $B$  be countably infinite pairwise disjoint sets representing respectively *IRIs*, *literals* and *blank nodes*. IRIs (Internationalized Resource Identifiers) are standard identifiers used for denoting any Web resource involved in RDF statements. A literal is a string that represents a specific value for some properties. A blank node represents an anonymous resource (either a literal or an IRI) that can have a local identifier, such as `_:b1`.

An ontology in RDF (a.k.a. a knowledge graph) is a set of (factual or ontological) statements expressed as triples  $(s, p, o) \in (I \cup B) \times I \times (I \cup L \cup B)$  that form a graph whose nodes are IRIs, blank nodes or literals.

## 2.2 Ontological Constraints

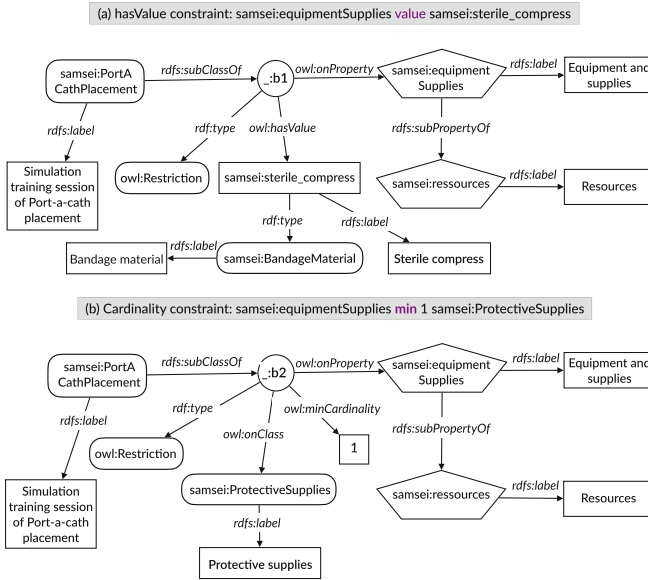
The ontological constraints that we consider are RDFS constraints and some OWL constraints (displayed in Table 1). Figure 1 visualize the RDF graphs associated to two constraints declared in the ONTOSAMSEI ontology on the property `samsei:equipmentSupplies`, for the class `samsei:PortACathPlacement`, which is a particular type of simulation learning unit that trains students to place a port or a catheter. The RDF graph in Fig. 1(a) expresses that `samsei:sterile_compress` (which is an instance of Bandage material) is declared in the ontology as a mandatory value of the property `samsei:equipmentSupplies`. The RDF graph depicted in Fig. 1(b) expresses that at least one equipment of type `samsei:protectiveSupplies` is mandatory for simulating a placement of a port or a catheter.

## 3 Interactive Ontology Update

Our approach consists of transposing the RDF data and the ontological constraints of a given domain ontology into a graphical user interface (GUI) named

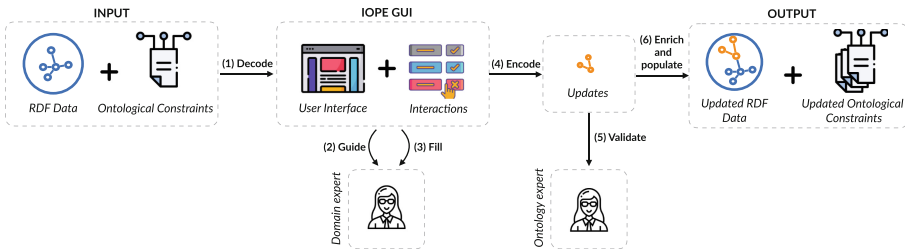
<sup>1</sup> Resource Description Framework Schema (RDFS): <https://www.w3.org/TR/rdf-schema>.

<sup>2</sup> Web Ontology Language (OWL): <https://www.w3.org/TR/owl-features>.



**Fig. 1.** Two constraint graphs (a) and (b) on the property `equipmentSupplies` for the class `PortACathPlacement`

IOPE GUI (step 1 in Fig. 2). It functions as a guidance for domain experts to easily explore the ontology (step 2) and update it (step 3) through interactive graphical widgets. The input entered by domain experts through the IOPE GUI are then transformed into RDF triples (step 4) that must be verified by an ontology engineer (step 5) before being permanently added in the domain ontology (step 6). Figure 2 provides an overview of IOPE’s workflow.



**Fig. 2.** The overview of IOPE’s workflow.

The *Web form templates* on which the IOPE GUI is built are described using a Web form ontology called IOPEWEB that we have developed by adapting RaUL [9].

### 3.1 The IOPEWEB Ontology

The IOPEWEB ontology is shown in Fig. 3. It is organized around 4 main classes, i.e., `IOPE:Page`, `IOPE:PageLayout`, `IOPE:Container` and `IOPE:Widget`. These classes are related by properties for modeling Web pages. The Web pages themselves are structured in the form of containers filled with widgets. Each Web page is also associated to a page layout.

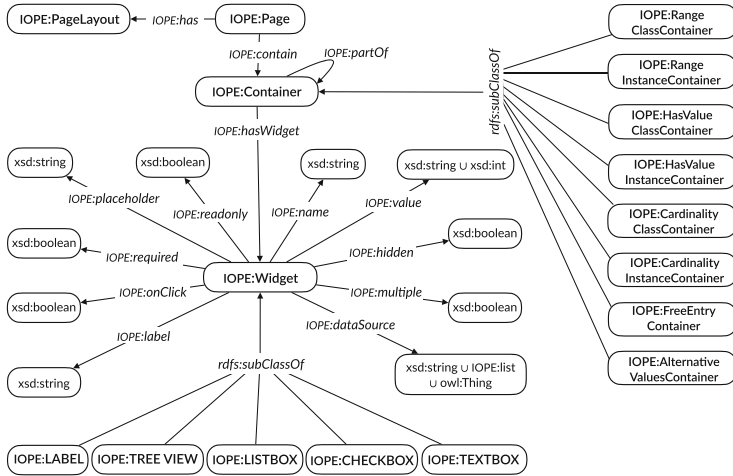


Fig. 3. The IOPE Web form ontology

The widgets are the direct point of user interaction, which are associated to the underlying RDF graph for the input ontology. The visualization and the user interaction are done using several types of widgets, such as label, tree view, list box, text box, and check box. These widgets constitute the subclasses of the main class `IOPE:Widget`, and inherit the standard widget properties described in IOPEWEB.

IOPEWEB describes how the input and output of widgets are modeled. We employ the `IOPE:dataSource` property for the assignment of an input data (from a domain ontology) of type `xsd:string`, simple or nested list `IOPE:list`, or `owl:Thing`, to their corresponding widgets. The `IOPE:value` property is filled by the value, entered by the user through the widget.

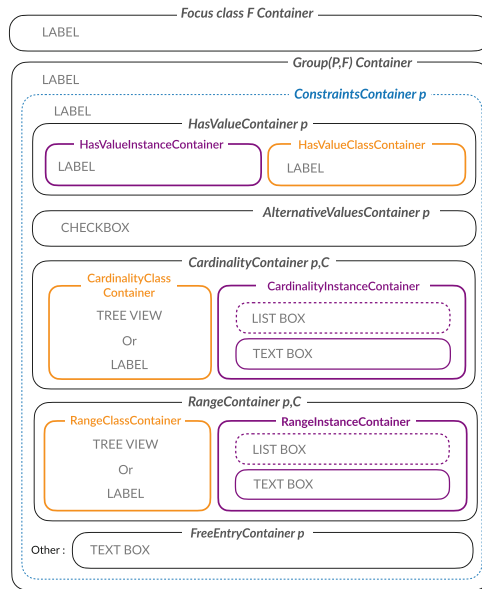
Widgets can be grouped in a Web page within containers. The containers themselves can be nested using the `IOPE:partOf` property. In our setting, different types of specific containers are considered as subclasses of `IOPE:Container` to express that the different types of ontological constraints will be rendered differently in IOPE GUI.

### 3.2 Ontology-Based GUI Construction

In a declarative approach, we employ a set of *mapping rules* to generate automatically pre-filled Web pages, and a set of *binding rules* to generate RDF graphs from the values entered by users through the widgets.

**Input:** The input required for GUI construction is a domain ontology in which the ontological constraints are automatically saturated by a reasoning algorithm as detailed in [5].

**Initialization:** The GUI construction is initiated with the selection of one class of interest in the ontology by the user, called the *focus class F*. The set  $Constraints(F)$  of the ontological constraints associated to  $F$  is decomposed in groups  $Group(P, F)$  of all the constraints involving sub-properties of a given property  $P$ . For the focus class  $F$ , and for each group of properties  $Group(P, F)$ , an instance of a Web page is created with the page layout depicted in Fig. 4. The page layout defines the organization of the Web page with a set of specific containers dedicated to different ontological constraints on sub-properties  $p$  of  $P$  for which there exists constraints in  $Group(P, F)$ .



**Fig. 4.** Web page template prepared for the rendering of constraints of the focus class  $F$  for each property  $p$  which is a specialization of a same property  $P$ .

The following instances of the `IOPE:Container` class are created, with their pre-allocated positions in the empty Web page template shown in Fig. 4.

- “IOPE:FocusClass  $F$  Container” denotes the main container of the created Web page for the focus class  $F$ ;
- “IOPE:Group( $P, F$ ) Container” denotes the container that groups all the other containers corresponding to the constraints holding for the class  $F$  on the sub-properties of  $P$ ;
- “IOPE:ConstraintContainer  $p$ ” denotes the container which contains restrictions of  $F$  on the property  $p$ , where  $p$  is a sub-property of  $P$ ;
- “IOPE:HasValueContainer  $p$ ” denotes the container which contains the HasValue restrictions of  $F$  on the property  $p$ ;
- “IOPE:AlternativeValuesContainer  $p$ ” denotes the container which contains the AlternativeValues restrictions of  $F$  on the property  $p$ ;
- “IOPE:CardinalityContainer  $p, C$ ” denotes the container which contains the cardinality restrictions of  $F$  on the property  $p$  and the class  $C$ ;
- “IOPE:RangeContainer  $p, C$ ” denotes the container which contains the range restrictions of  $F$  on the property  $p$ , where the range of  $p$  is the class  $C$ ;
- “IOPE:FreeEntryContainer  $p$ ” denotes the container which enables the user to add new classes involved in the cardinality restrictions for the property  $p$ .

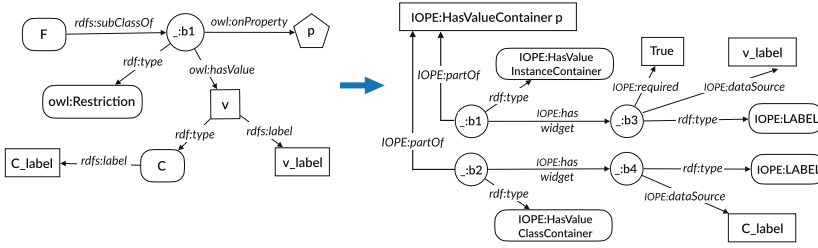
Then, the mapping rules are triggered to map components of each ontological constraint to the widgets inside the aforementioned containers, and fill each Web page guided by the ontology.

**Mapping Rules:** Each mapping rule has a constraint graph pattern in its left-hand side, and an IOPEWEB graph pattern in its right-hand side. The constraint graph pattern expresses a particular ontological constraint on a property and a (focus) class, and the IOPEWEB graph pattern specifies how to pre-fill the corresponding container to render this ontological constraint. Each rule is instantiated by mapping the constraint graph pattern in its left-hand side to the constraints graphs present in the input ontology and involving the chosen focus class. The mapping rules can be triggered in a forward-chaining manner and in any order. The resulting IOPEWEB graph provides the full RDF specification of the pre-filled Web pages that have to be created for the focus class  $F$  chosen by the user. We provide the exhaustive set of mapping rules in [5]. In this paper though, we just give two of them in their instantiated form for clarity purpose.

Figure 5 shows a mapping rule for a value restriction ( $F p$  value  $v$ ). The specific container “IOPE:HasValueContainer  $p$ ” is decomposed into two sub-containers defined as blank nodes, whose types are IOPE:HasValueInstanceContainer and IOPE:HasValueClassContainer. For the two sub-containers, widgets of type IOPE:LABEL are created as blank nodes with the property IOPE:data-Source filled by the corresponding labels of  $v$  and its class  $C$  from the domain ontology. The property IOPE:required is set to True for the first widget, to show that the value  $v$  is mandatory for the property  $p$ .

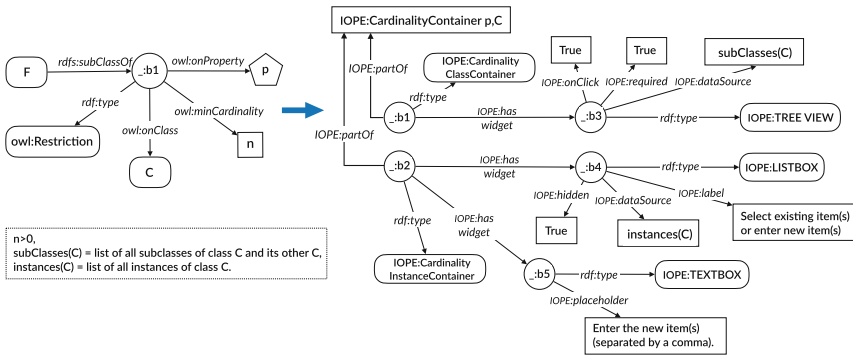
Figure 6 shows a mapping rule for a cardinality restriction ( $F p$  min  $n C$ ) such that  $n > 0$ , where  $C$  has a hierarchy of sub-classes and a list of instances in the domain ontology. The specific container “IOPE:CardinalityContainer  $p, C$ ”





**Fig. 5.** Mapping rule for a value restriction ( $F p$  value  $v$ ) where  $v$   $\text{rdf:type } C$

is decomposed into two sub-containers defined as blank nodes, with types “IOPE:CardinalityClassContainer” and “IOPE:CardinalityInstanceContainer”.



**Fig. 6.** Mapping rule for a Cardinality constraint where  $subClasses(C)$  and  $instances(C)$  are not empty, and  $n > 0$ .

For the first sub-container, a widget of type IOPE:TREEVIEW is created as a blank node with the property IOPE:dataSource filled by the tree view of  $subClasses(C)$ , i.e., the hierarchy of  $C$ 's sub-classes in the domain ontology, enriched with an additional item *Other\_C*. The property IOPE:required and IOPE:onClick are also set to True for this widget to indicate that (i) entering at least one value is mandatory for the property  $p$ , and (ii) this widget supports the interaction with users to display the sub-class hierarchy, interactively.

For the second sub-container, a widget of type IOPE:LISTBOX is created as a blank node with the property IOPE:dataSource filled by the list  $instances(C)$  of instances of the class  $C$ . The IOPE:label property is set to “select existing item(s) or enter new item(s)” and the IOPE:hidden property is set to True to make the widget invisible until the first interaction of the user through the widget of type IOPE:TREEVIEW. A widget of type IOPE:TEXTBOX is also created with the IOPE:placeholder property, whose value is set to “enter the new item(s) (separated by a comma)”, in order to enable the user to enter new instances.

The input entered through user interactions must then be bound to RDF data corresponding to new instances or new constraints submitted to populate or enrich the domain ontology. This binding mechanism is based on a set of *binding rules* which are triggered on the IOPEWEB graph to generate RDF graphs. Next, we will discuss these binding rules.

**Binding Rules:** The role of binding rules is to transform user interactions into RDF graphs. A binding rule has an IOPEWEB graph pattern in its left-hand side, and a RDF graph pattern in its right-hand side. The binding rule is triggered when an input is entered by a user in a IOPEWEB form instantiating the left-hand side of the binding rule. The corresponding instantiation of the right-hand side provides RDF triples that have to be added in the output RDF graph. There are 9 binding rules. In this paper, we describe 2 of them, and provide the rest in [5]. The binding rule shown in Fig.7 is triggered when a focus class  $F$  is chosen by the user. Once triggered, the rule creates an instance  $new\_f$  of the focus class  $F$  in the output RDF graph.

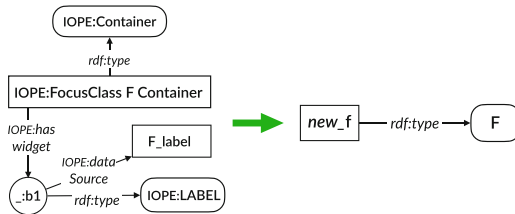


Fig. 7. For creating an instance  $new\_f$  of a focus class  $F$ .

The other binding rules are triggered when the IOPE:value property is filled by an input provided by the user through an interactive widget. Figure 8 shows the binding rule for the IOPE:TEXTBOX widget in the free entry container of a property  $p$  for the focus class  $F$ . Once this binding rule is triggered, a new constraint graph will be generated which expresses a new class and a new cardinality constraint for  $F$  on the property  $p$ .

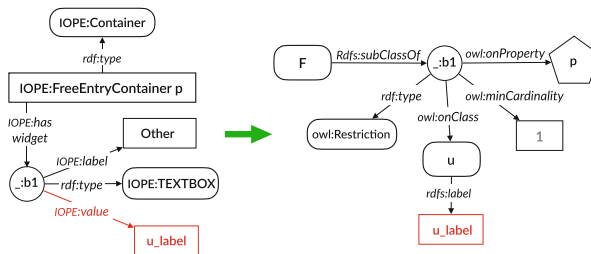


Fig. 8. Binding rule for free entry container on property  $p$  and a focus class  $F$

## 4 Evaluation

The objective of our study is to evaluate the *efficiency*, the *users' satisfaction* and the *effectiveness* of the IOPE interface with the purpose of populating and enriching the ONTOSAMSEI ontology. The users involved in our user study are a subgroup of 22 experts in simulation-based training in Medicine. They are domain experts, but they are not familiar with RDF and OWL. The user study was organized in two steps for each expert. In the first step, the expert logs in the system with her credentials, picks one simulation training session, and begins to observe and update the information in the pre-filled Web pages. In the second step, she will be transferred to a survey form to evaluate some qualitative aspects of IOPE and ONTOSAMSEI ontology and reflect her viewpoint based on her interactions with the IOPE interface.

### 4.1 Evaluation of the IOPE GUI Efficiency

**Time Spent by Users and Number of Interactions.** Each expert spent 163 s (2.72 min) on average, maximum 320 s (5.33 min), minimum 67 s (1.12 min). On average, their number of interactions with IOPE is 5.78, with a maximum of 14 and a minimum 3. The majority of interactions are with CHECK BOX widget (56.15%) followed by TEXT BOX widget (32.30%) and LIST BOX widget (11.53%). Table 2 shows the distribution of experts in two categories of groups. In terms of number of interactions, we have built the groups of “prolific” experts (having more than 6 interactions with IOPE), “active” experts (having between 3 and 6 interactions), and “moderate” experts (with less than 3 interactions). In terms of interaction duration, we have built the groups of experts spending “short-time” (less than 2 min), “medium-time” (between 2 and 4 min), and “long-time” (more than 4 min). Table 3 reports the distribution of time groups for each interaction activity groups. We notice that more interactions do not necessary yield to more time spent to interact. This shows that IOPE helps experts to fulfill their task in a reasonable amount of time, even for prolific experts.

**Table 2.** Distribution of expert groups

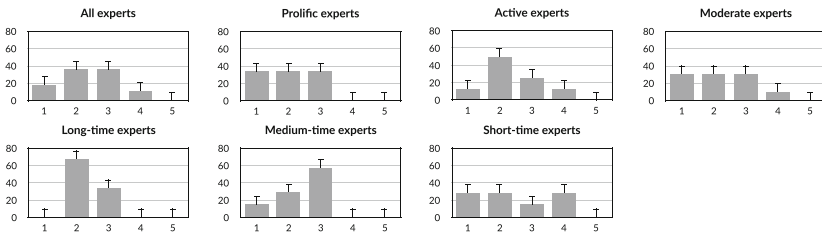
	Moderate	Active	Prolific	Short-time	Medium-time	Long-time
Expert population	22.73%	50%	27.27%	50%	31.82%	18.18%

**Time-to-Insight Users's Evaluation.** After they are done with using the IOPE interface for fulfilling their task, we ask the experts the following question to estimate the time-to-insight for a future interaction with IOPE : “*how much time do you expect to take for setting up a new simulation training session with IOPE?*”. The response is in the form of a Likert scale from 1 to 5 where “1” means “very short time” and “5” means “very long time”.

**Table 3.** Distribution of interaction time groups for interaction volume groups.

		Interaction volume groups		
		Moderate	Active	Prolific
Interaction time groups	Short-time	0.80	0.46	0.33
	Medium-time	0.00	0.27	0.67
	Long-time	0.20	0.27	0.00

Figure 9 shows the results. We observe that the majority of experts chose “short time” and “average time”, i.e., options 2 and 3 in the Likert scale. Moreover, prolific experts and long-time perceive shorter expected time compared to the active and moderate experts. A possible interpretation is that more interactions and more time sent interacting with the system boosts the perception of faster delivery of required information.



**Fig. 9.** Prediction of experts about time-to-insight for their next utilization of the IOPE interface. Dashed bars show the average values of time-to-insights for all the experts.

### Comparative Efficiency of IOPE with a Standard Ontology Editor.

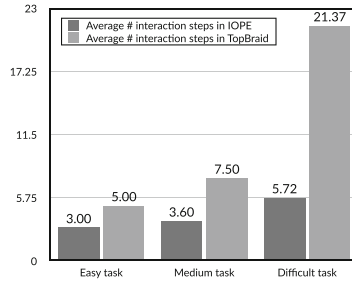
The goal of this experiment was to measure the added-value of IOPE compared to a standard ontology editor such as TOPBRAID [1], in terms of number of interactions required to fulfill edition tasks mentioned in Table 4. The tasks are categorized into three levels of difficulty based on [7].

**Table 4.** Tasks descriptions

Task	Description (Given the simulation training session X ...)
Easy	Fill the number of trainees for X
Medium	Fill the target audience of X
Difficult	Fill the required resources for X

For the fairness of this experiment, since none of the domain experts have ever used TOPBRAID, the different tasks were fulfilled by the five authors of the paper who have a sufficient knowledge about the domain, as well as a sufficient experience using both IOPE and TOPBRAID.

Figure 10 shows the average number of interaction steps to fulfil those tasks in IOPE and TOPBRAID.



**Fig. 10.** Average number of interactions in IOPE and TOPBRAID.

We observe that for both tools, the number of interaction steps increases with the difficulty of the tasks. However, the IOPE’s trend grows from average 3.00 steps for an easy task to average 5.72 steps for a difficult task, while using TOPBRAID grows from average 5.00 steps for an easy task to average 21.00 steps for a difficult task. This shows that IOPE, by weaving relevant information together using constraints, enables the experts to fulfil their tasks more rapidly than by using a standard editor.

#### 4.2 Evaluation of IOPE Users’ Satisfaction

We have measured on a Likert scale in the range 1 to 5 the assessment by users of three aspects of satisfaction, namely *utility*, *usability*, and *adoption*, through the questions of the three first rows of Table 5.

**Table 5.** Measure definitions and corresponding questions asked in the survey.

Measures	Definition	Question asked in the survey
Utility [2, 16]	The usefulness of the method to fulfil a given task	How do you evaluate the utility of IOPE for setting up simulation training sessions?
Usability [2, 15]	The easiness of interactions with the method	To which degree do you find IOPE easy-to-use?
Adoption [16]	The usefulness of the method for future similar tasks	How often will you employ IOPE for setting up and describing a new simulation training session in the future?
Accuracy [14, 15]	The precision of information based on expert’s prior knowledge	How do you evaluate the accuracy of IOPE’s pre-filled information for describing simulation training sessions?
Completeness [15]	The retrieval exhaustiveness of the necessary and required information	How do you evaluate the sufficiency of IOPE’s pre-filled information for describing simulation training sessions?

The aggregated results are shown in the three first column of Fig. 11.

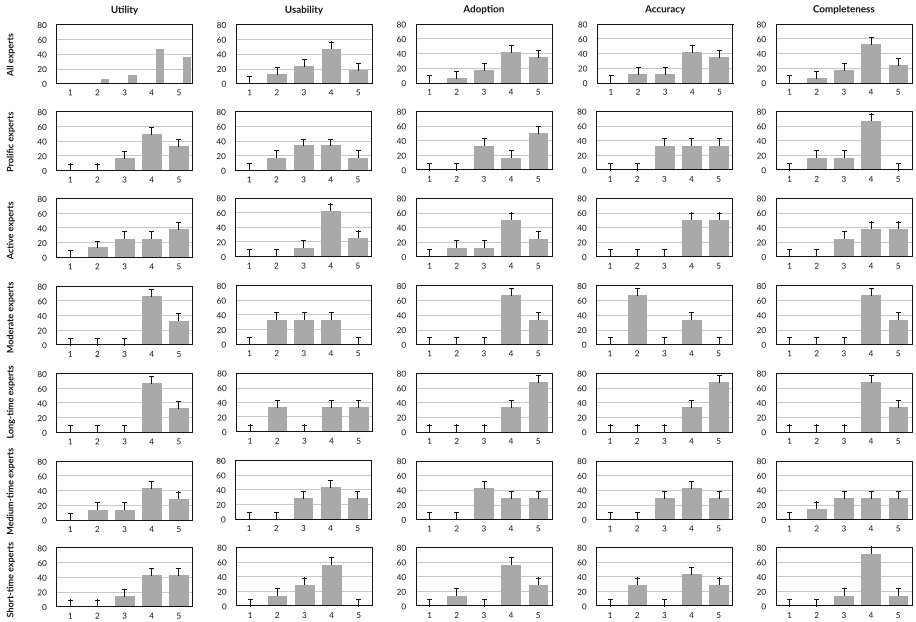


Fig. 11. Satisfaction and effectiveness metrics results.

**Utility.** 82.35% of the participants have a positive view on the utility of IOPE. However, the prolific experts appreciate the utility more than active experts. This shows that more interactions increases the perception of utility, which is also confirmed by long-time experts who are entirely on the positive spectrum.

**Usability.** Overall the experts perceived usability positively. However, there is a vivid contrast between moderate experts versus active and prolific experts, where the former group seems to not enjoy the usability of IOPE. We conjecture that moderate experts got lost early in the process, and abandoned their task. There is also a subset of long-time experts who assessed low usability. They probably spent too much time to fulfil their tasks and got lost in the process also.

**Adoption.** The choice over adoption is from 1 to 5, where 1 means “never” and 5 means “always”. Most of the experts voted to adopt IOPE in the future.

### 4.3 Effectiveness of IOPE for Enriching the ONTOSAMSEI Ontology

In this part of the experiment, we measure the experts’ assessment of *accuracy* and *completeness* of the ONTOSAMSEI ontology through its presentation to the experts by IOPE GUI. We do it by asking the experts the questions in the

two last rows of the Table 5. The aggregated results (on the Likert scale from 1 to 5) are shown in the two last columns of Fig. 11.

**Accuracy.** The majority of the participants are positive on accuracy, while 11.76% are negative. Short-time and moderate experts express more negative votes on accuracy compared to long-time and prolific experts, respectively. This is presumably because fewer investigations in the former groups did not enable them a precise view of the ontology.

**Completeness.** 76.46% of the participants find ONTOSAMSEI complete enough. However, prolific experts appreciate completeness less than the overall population. We found out that they prominently interact with text-boxes, which shows that they use IOPE to effectively enrich the ontology. The entire long-time expert group votes positively, which means that spending more time to go into the details of the simulation training sessions convinces them of their completeness.

## 5 Related Work

In the literature, ontological updates are often performed using ontology editing tools [1, 13]. However, these systems require a basic understanding of the RDF notation and of the OWL semantics to edit the ontology consistently. Graph-based editing approaches alleviate this limitation by leveraging shapes graphs in the form of SHACL standard<sup>3</sup> [17, 19]. While shapes graphs are well adapted for editing complex data, they require the definition of such graphs for each ontology. In contrast, IOPE abstracts all RDF/OWL technicalities and seamlessly enforces the ontological constraints as a strong guidance for the experts to update the ontology, using the pre-filled forms.

WebVOWL [18] is a web application for the interactive graph-based visualization of ontologies which employs the Visual Notation for OWL Ontologies (VOWL) [11]. However, WebVOWL does not visualize the instances but only the OWL part of a (possibly populated) ontology. Also, the graphs displayed by the tool tend to become quickly illegible when their size increases. In IOPE, we employ Web forms as a more widespread medium for visualizing information, and we support the update of instances and of ontological constraints.

Forms are also used in [12] in a nested structure to capture relational aspects of knowledge graphs and update RDF data. However, the nested structure introduces increasing complexity and hence lacks intuitiveness. Moreover, the focus in [12] is solely on the population part and the approach does not extend to OWL constraints. In [6], Web forms are generated from ontologies (using a User Interface ontology, called RaUL) by interpreting ontology assertions as rules. While the approach only incorporates individual assertions (ontology population), IOPE serves both ontology enrichment and population, through interactions with the experts. IOPE stresses on ontological constraints as first-class citizens and renders pre-filled forms to provide a more aggregated view for the experts, which is, to the best of our knowledge, nonexistent in the literature.

<sup>3</sup> *Shapes Constraint Language (SHACL)*: <https://www.w3.org/TR/shacl/>.

## 6 Conclusion

In this paper, we have presented the interactive IOPE framework for enrichment and population of specialized ontologies. Given any input ontology, IOPE exploits the ontological constraints and a set of mapping rules to generate a set of user-friendly Web pages which assist the experts in editing the ontology. Binding rules are then used to derive the RDF graphs corresponding to the updates entered by the experts. We have conducted an extensive set of experiments on the domain of simulation-based medical education, for measuring IOPE's efficiency, effectiveness, as well as the experts' satisfaction in fulfilling their tasks using IOPE. In the future, we plan to improve the *explainability* of IOPE to reduce the number of abandoned editing tasks and increase its usability by domain experts not familiar with ontology engineering.

## References

1. Topquadrant topbraid composer. <https://www.topquadrant.com/products/topbraid-composer/>. Accessed 15 Jan 2021
2. Albert, W., Tullis, T.: Measuring the user experience: collecting, analyzing, and presenting usability metrics. Newnes (2013)
3. Baghernezhad-Tabasi, S., Druette, L., Jouanot, F., Meurger, C., Rousset, M.C.: IOPE: interactive ontology population and enrichment. In: SEMANTiCS (2021)
4. Baghernezhad-Tabasi, S., Druette, L., Jouanot, F., Meurger, C., Rousset, M.C.: OntoSAMSEI: interactive ontology engineering for supporting simulation-based training in medicine. In: WETICE (2021)
5. Baghernezhad-Tabasi, S., Rousset, M.C., Druette, L., Jouanot, F., Meurger, C.: IOPE: interactive Ontology Population and Enrichment guided by ontological constraint. Technical report (2021). <https://hal.archives-ouvertes.fr/hal-03177176>
6. Butt, A., Haller, A., Liu, S., Xie, L.: ActiveRaUL: automatically generated web interfaces for creating RDF data. In: Semantic Web 2013 (2013)
7. Dimara, E., Franconeri, S., Plaisant, C., Bezerianos, A., Dragicevic, P.: A task-based taxonomy of cognitive biases for information visualization. IEEE Trans. Vis. Comput. Graph. **26**, 1413–1432 (2020)
8. Fang, Y., Cheng, R., Luo, S., Hu, J., Huang, K.: C-Explorer: browsing communities in large graphs. In: VLDB (2017)
9. Haller, A., Umbrich, J., Hausenblas, M.: RaUL: RDFa user interface language – a data processing model for web applications. In: Chen, L., Triantafillou, P., Suel, T. (eds.) WISE 2010. LNCS, vol. 6488, pp. 400–410. Springer, Heidelberg (2010). [https://doi.org/10.1007/978-3-642-17616-6\\_36](https://doi.org/10.1007/978-3-642-17616-6_36)
10. Henry, N., Fekete, J., McGuffin, M.J.: Nodetrix: a hybrid visualization of social networks. In: TVCG (2007)
11. Lohmann, S., Negru, S., Haag, F., Ertl, T.: Visualizing ontologies with VOWL. Semant. Web **7**(4), 399–419 (2016)
12. Maillot, P., Ferré, S., Cellier, P., Ducassé, M., Partouche, F.: Nested forms with dynamic suggestions for quality RDF authoring. In: Benslimane, D., Damiani, E., Grosky, W.I., Hameurlain, A., Sheth, A., Wagner, R.R. (eds.) DEXA 2017. LNCS, vol. 10438, pp. 35–45. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-64468-4\\_3](https://doi.org/10.1007/978-3-319-64468-4_3)



13. Noy, N.F., Sintek, M., Decker, S., Crubézy, M., Ferguson, R.W., Musen, M.A.: Creating semantic web contents with protégé-2000. *IEEE Intell. Syst.* **16**(2), 60–71 (2001)
14. Omidvar-Tehrani, B., Amer-Yahia, S.: Data pipelines for user group analytics. In: *SIGMOD Conference*, pp. 2048–2053. ACM (2019)
15. Rahman, P., Jiang, L., Nandi, A.: Evaluating interactive data systems. *VLDB J.* **29**(1), 119–146 (2019). <https://doi.org/10.1007/s00778-019-00589-2>
16. Thomas, J.J.: *Illuminating the path: the research and development agenda for visual analytics*. IEEE Computer Society (2005)
17. Valdestilhas, A., Publio, G., Cimmino Arriaga, A., Riechert, T.: Voceditor an integrated environment to visually edit, validate and versioning RDF vocabularies (2020)
18. Wiens, V., Lohmann, S., Auer, S.: Webvowl editor: device-independent visual ontology modeling. In: *ISWC 2018 Posters & Demonstrations* (2018)
19. Wright, J., Méndez, S.J.R., Haller, A., Taylor, K., Omran, P.G.: Schímatos: a SHACL-based web-form generator for knowledge graph editing. In: *ISWC* (2020)

# **Graph Neural Network**



# Controversy Detection: A Text and Graph Neural Network Based Approach

Samy Benslimane<sup>1</sup>(✉), Jérôme Azé<sup>1</sup>, Sandra Bringay<sup>1,2</sup>,  
Maximilien Servajean<sup>1,2</sup>, and Caroline Mollevi<sup>3,4</sup>

<sup>1</sup> LIRMM UMR 5506, CNRS, University of Montpellier, Montpellier, France  
{samy.benslimane, jerome.aze, sandra.bringay, maximilien.servajean}@lirmm.fr

<sup>2</sup> AMIS, Paul Valéry University, Montpellier, France

<sup>3</sup> Institut du Cancer Montpellier (ICM), Montpellier, France  
caroline.mollevi@icm.unicancer.fr

<sup>4</sup> IDESP, UMR Inserm - Université de Montpellier, Montpellier, France

**Abstract.** Controversial content refers to any content that attracts both positive and negative feedback. Its automatic identification, especially on social media, is a challenging task as it should be done on a large number of continuously evolving posts, covering a large variety of topics. Most of the existing approaches rely on the graph structure of a topic-discussion and/or the content of messages. This paper proposes a controversy detection approach based on both graph structure of a discussion and text features. Our proposed approach relies on Graph Neural Network (GNN) to encode the graph representation (including its texts) in an embedding vector before performing a graph classification task. The latter will classify the post as controversial or not. Two controversy detection strategies are proposed. The first one is based on a hierarchical graph representation learning. Graph user nodes are embedded hierarchically and iteratively to compute the whole graph embedding vector. The second one is based on the attention mechanism, which allows each user node to give more or less importance to its neighbors when computing node embeddings. We conduct experiments to evaluate our approach using different real-world datasets. Conducted experiments show the positive impact of combining textual features and structural information in terms of performance.

**Keywords:** Controversy detection · Graph neural networks · Hierarchical graph representation learning · Attention-based graph embedding

## 1 Introduction

The availability of large amount of data sources and the emergence of various social networks including Twitter and Reddit, increased the social connectivity of people. This allowed them to easily express, propagate, share and dispute opinions and gives us a great opportunity to study and understand social phenomena like controversial topics. Expressed opinions through posts and articles

often trigger fierce and sometimes endless debates, and frequently cause a controversy. A controversial content can simply be defined as any content that attracts both positive and negative feedback [8]. Indeed, polarization stigmatizes user’s behavior in presence of controversial topics [1, 5, 6].

Automatic controversy detection can be helpful. For instance, people can be warned by the existence of a controversy to add some nuance to better understand some issues. Objective information could also be brought to people to prevent misinformation or hateful discussions [6]. Detecting a content as non controversial is also helpful as it shows that people agree on a given issue.

Automatic controversy identification is difficult and constitutes a challenging task as it should be done on a large number of continuously evolving posts covering a wide range of topics. This difficulty is increased by the fact that controversy is sometimes time-aware (what is controversial today was not necessary controversial in the past) and community-aware (what is controversial in a community is not necessary controversial in another community) [10].

Controversy analysis on web pages or articles is usually based on features extracted from the content [11, 16]. However, on social media, the interaction between people is widely used to detect controversy. These interactions include social relation (retweet and follow on Twitter, comment on Reddit) [5, 14] and citation relation on Wikipedia [9].

In this paper, we focus on detecting controversy on the social media Reddit, even if any other social media (Twitter, Facebook, etc.) can also be used after making few adaptations of the graph building stage. The originality of our approach firstly resides on using very recent state-of-the-art GNN methods to embed user nodes in a low d-dimensional space and take into account structural information. Initial text representations of a user are learnt from their messages on a current post and used as input of our GNN-based approach. To the best of our knowledge, only Zhong et al. started to use GNNs, building their post-level controversy detection method by directly exploiting the comment-tree structure [21]. Our work, on the contrary, exploits the user’s interaction graph built from the comment-tree structure and compare multiple state-of-the art (GNNs) to combine both structural and content information.

To detect controversial posts, we propose a GNN-based approach which consists of the following main contributions:

- GNN architecture-based controversy detection. Our controversy detection is based on a graph classification. We propose two strategies to embed the whole graph structure. The first strategy aims to exploit hierarchical structure that may exist in the user graph structure. Graph information are aggregated across the edges iteratively and in a hierarchical way. In our work, we rely on the DIFFPOOL approach which encodes the whole graph by stacking several pooling layers [18]. The second strategy is based on an attention mechanism. It aims to allow each user node to judge which user neighbor is more or less important than the others, during the node embedding process, according to the structure and features of the graph.

- Experimental study. We conduct experiments on real-world datasets to evaluate the proposed GNN-based approach only using structural information. We show that our approach gets good performance compared to our baseline.
- Textual features. We show that incorporating initial textual representation of users can improve the performance of our approach.

The rest of this paper is organized as follows. Section 2 provides some related work. Section 3 presents an overview of our approach to automatically detect controversy on social media. Its different stages are described and formalized. Section 4 presents the performance experiments and discusses the obtained results. Section 5 concludes the paper and highlights some future work.

## 2 Related Work

Works on controversy analysis can be classified in three groups: content-based, structure-based and hybrid approaches.

**Content-Based.** Early methods to detect controversy are mainly based on textual features, and only focus on language semantic, supposing that immediate textual context of concept can be highly indicative [16] or that text-content can be used as a tool for detecting controversial topic/post. Several studies focus on the web controversy, thanks to sources like Wikipedia, where pages can automatically be labelled as controversial, using “edit-wars”<sup>1</sup> and relations/citations between pages. In [16], an approach to measure how controversial a concept is on Wikipedia pages is proposed. Instead of relying on Wikipedia’s metadata, authors argue that immediate textual context of a concept is strongly indicative of controversiality. They represent articles via pre-trained word embeddings methods and define three controversiality estimators based on the nearest neighbors, naive Bayes, and recurrent neural network respectively. In [3], authors were interested in identifying whether a given content on a web page is controversial or not. The collective controversy classification model is based on a nearest neighbor classifier that identifies an article according to the related Wikipedia articles. The idea is that if related Wikipedia articles are controversial, it is likely that the article is also controversial. Other studies focus on probabilistic approaches [9, 11] to combine Wikipedia Controversy meta-data and features like the *MCD* score<sup>2</sup>. Articles, from web media, are also a huge source of information.

**Structure-Based.** Textual messages on social media are usually biased and meanings might be different depending on many factors, such as the culture or language of the communities, and therefore should be treated with precaution. When studying controversy on a user interaction basis, the structural information of those interactions become particularly relevant, especially on social media. Each social media has its own code. For example, Twitter has specific features, such as ‘Retweet’ and ‘Follow’. In [5], a user hybrid graph, combining

<sup>1</sup> MultipleeditorsonaWikipediaconceptsexchangingopposingopinions.

<sup>2</sup> Presence of certain words, ferocity of “edit-wars”, etc.

follow and retweet edges, is built for a topic, which is defined by a set of hashtags. After partitioning the graph on two distinct communities, different methods to measure the controversy are checked, including a random-walk-based controversy measure (RWC). In a similar study [6], they use the same graph to quantify and reduce controversy, by connecting opposing sides and creating bridges between communities for more exposure. In [4], a similar approach is used suggesting that we can level user commitment at their community by looking at their relation. They propose a new method using Biased Random-Walk and adapt a new controversy measure to quantify. A previous research focused on more exposed node boundaries, with statistical polarization measures to evaluate controversy [7]. In [13], the importance of users and named entities involved in a discussion are highlighted. They generate a conditioned graph on named entities partitioned, and quantify controversy using a RWC (Random-Walk Controversy) score. Even if structural features are widely covered and seem to be a strong asset, not covering text features appears to be a huge loss of relevant information.

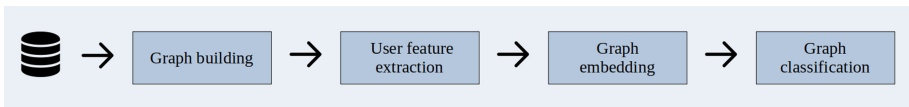
**Hybrid Methods.** Recent studies focus on combining both structural and content information to avoid losing valuable features. On this condition, Social media appears to be the ideal source, with the multiplicity of user interactions. In [19], authors extend the work of Garimella et al. [5] and propose a vocabulary-based controversy detection. Using the partitioned user graph, tweets of the two selected communities are grouped by users, pre-processed, concatenated and labeled by the community name of their corresponding user. They constitute the dataset which is used to train the text representation model FastText. The controversy score is finally computed by using the embedding of the central users. In [8], authors demonstrate that mixing structural features (number of comments, max depth/total comment ratio, average node depth, etc.) of post-comment tree of a Reddit discussion with textual features outputted by language models such as BERT [2] can improve predictive performance of early controversy post-level detection. With the same objective, a Graph Convolutional Network based approach is proposed in [21] by Zhong et al. It aims to integrate information extracted from the comment-tree structure as well as content of post and its comments. A parallel multi-task classifier is added on their model to disentangle topic-related and topic-unrelated features for inter-topic detection. Even with good performance, this approach presents some limits. The comment-tree structure of a post prevents us from exploiting user interactions, and the use of inter-topics relation might interfere too much with the main detection task. However, to the best of our knowledge, this is the first study which focuses on GNN for controversy analysis.

We present in this paper a new hybrid controversy detection approach, based on user graph interaction and state-of-the-art GNNs to combine valuable textual and structure information.

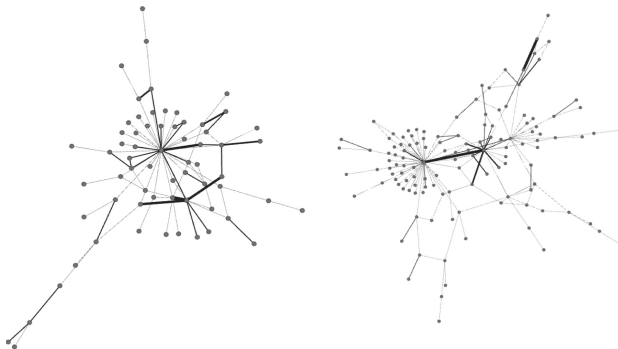
### 3 Graph Neural Network-Based Controversy Detection Approach

This section describes our post-level controversy detection approach. It focuses on the Reddit social media. Any other social media can be used with very few adaptations of the graph building stage. The main idea is to exploit both text content and user interactions by representing the Reddit discussion as a user graph and exploring advanced GNN embedding techniques. Figure 1 presents an overview of our approach. We divide our pipeline into four sequential stages: Graph Building, User Feature Extraction, Graph Embedding, and Graph Classification.

The graph building stage represents data extracted from the Reddit social media as a user graph. We represent the initial comment-post tree as a graph where nodes represent users and edges correspond to the interaction that exists between users. Each node is represented by its own data (user-id, age, location, texts, etc.). The user feature extraction stage enriches graph nodes by adding textual embedded features. These features are computed by using state-of-the-art NLP techniques. This allows to better interpret the texts that users sent out. The graph embedding stage computes the embedding of the whole graph. Different advanced GNN-based graph representation learning techniques are used, namely DIFFPOOL [18], GCN [12], and GAT-GC [20]. Finally, the graph classification stage predicts the binary label associated to the whole graph, that is to classify a post as controversial or not.



**Fig. 1.** Overview of our controversy detection approach.



**Fig. 2.** Left: User graph of a controversial post. Right: User graph of a non-controversial post. The more interaction there is between two nodes, the bolder the link is.

### 3.1 Graph Building

Existing controversy detection methods [8, 21] on Reddit use the classic comment-post tree representation as they mainly focus on the structure of the discussion. However, many research works have established that user interaction can be helpful to extract different features on social media that can improve the controversy detection. In this work, we adopt a graph representation of a discussion to highlight these user interactions. Given a discussion on a post (thread)  $p$  extracted from a subreddit  $s$ , we build an undirected graph where a node  $u_i$  represents a user involved in the discussion. An edge  $(u_i, u_j)$  is created when a user  $u_j$  responds to the post  $p$  or any comment posted by  $u_i$ . Figure 2 shows two user graphs of controversial and non-controversial posts respectively.

More formally, a post  $p$  is represented as a graph  $G = (\mathcal{U}, \mathcal{E}, X)$  where  $\mathcal{U} = \{u_1, u_2, \dots, u_n\}$  denotes the user nodes,  $\mathcal{E} = \{(u_i, u_j)\}_{1 \leq i, j \leq n}$  denotes the edges of the graph, and  $X \in \mathbb{R}^{n \times e}$ ,  $e$  being the feature dimension, denotes the user node features matrix. Each node corresponds to a unique user, and an edge between two nodes exists if there is interaction links between the corresponding users. The computation of the matrix  $X$  is described in Sect. 3.2.

### 3.2 User Feature Extraction

In order to bring valuable information to the graph representation, user features are extracted from the posted texts by using advanced NLP techniques. Recently, different NLP language models pre-trained on a large corpus have been proposed to improve the dynamic text representation, such as BERT [2].

The user features extraction is performed for each user as follows. Each message (post or comment) a user  $u_i$  posts is firstly cleaned (reddit tags and url link removed) and is then embedded in an  $e$ -dimensional vector by using a language model BERT<sup>3</sup>. The embedded vectors obtained from the different messages posted by a user  $u_i$  are aggregated to form the final user features  $x_{u_i}$  as shown in Eq. 1.

$$x_{u_i} = \text{AGGREGATION} \left( [x_{u_i}^0, x_{u_i}^1, \dots, x_{u_i}^m] \right) \quad (1)$$

where  $x_{u_i}^j \in \mathbb{R}^e$  is the individual  $e$ -dimensional embedded vector computed from the  $j^{\text{th}}$  message of user  $u_i$ , and  $m$  is the number of messages a user  $u_i$  posted. In this paper, the aggregation of the embedded text vectors is performed via the Max-pooling function, but any other aggregation function can be used. Features of each user  $u_i$  is stacked on a matrix  $X \in \mathbb{R}^{n \times e}$ .

The user graph  $G = (\mathcal{U}, \mathcal{E}, X)$  is now fully represented and includes node textual features. It will also be referenced by  $(A, X)$  where  $A$  represents its adjacency matrix.

<sup>3</sup> More details on Sect. 4.



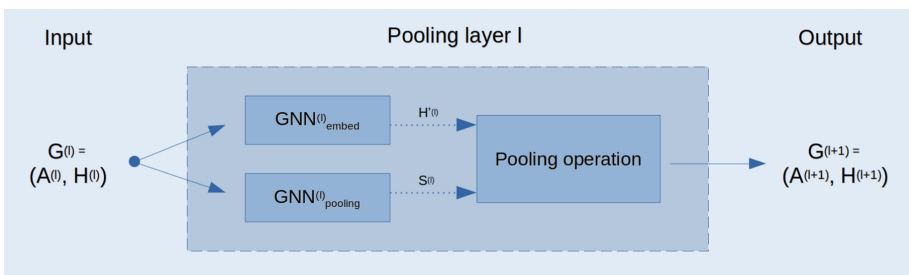
### 3.3 Graph Embedding

The graph embedding stage aims to encode the whole user graph in a low-dimensional vector. This latter will feed the graph classification stage to predict if a post is controversial or not. Recently, different GNN-based approaches were proposed to adapt deep learning architectures to the graph structured data [12, 17]. The main idea is to consider each graph node as a computation node, and to learn classical neural network primitives that compute node embeddings.

This stage relies on GNN architectures with the objective to exploit both user node features computed in the previous stage and the user graph structure of the Reddit discussion. The output is the embedding of the whole graph denoted by  $z_G$ . Learning individual node embeddings denoted by  $z_{u_i}$  is also performed as an intermediate stage. We propose in this paper two main strategies to embed the whole graph for the controversy detection needs. These strategies rely on hierarchical representations of graphs, convolutional network, and attention-based graph representation.

**Hierarchical Graph Representation Learning-Based Strategy.** This strategy exploits hierarchical structure that may exist in the user graph structure. Thus, in the whole graph encoding process, graph information are aggregated across the edges iteratively and in a hierarchical way. We rely on the DIFFPOOL [18] approach which encodes the whole graph by stacking several pooling layers. Each pooling layer is composed of two distinct GNN: one, called  $GNN_{embed}$ , learns user nodes embeddings  $H$ , and the other, called  $GNN_{pooling}$ , learns an assignment matrix  $S$  that indicates which user nodes are assigned to which cluster. The matrix  $S$  is used to coarsen the graph.

As depicted in Fig. 3, the functioning of the pooling layer at level (l) is described as follows:



**Fig. 3.** Diffpool-based Pooling layer architecture.  $A^{(l)}$  and  $H^{(l)}$  represent the adjacency and feature matrices of the input graph at layer ( $l$ ) respectively.  $GNN_{embed}$  and  $GNN_{pooling}$  are the 2 GNN blocks used to respectively compute node embeddings  $H'^{(l)}$  and assignment matrix  $S^{(l)}$ . The pooling operation block converts the input graph  $(A^{(l+1)}, H^{(l+1)})$  into a new coarsened graph  $(A^{(l+1)}, H^{(l+1)})$ .

1. *Node embedding generation.* We first apply the  $\text{GNN}_{embed}^{(l)}$  to the graph obtained at level  $(l)$  represented by its adjacency matrix  $A^{(l)}$ , and its node features matrix  $H^{(l)}$ . As described in Eq. 2, the result is an intermediate node embeddings  $H'^{(l)} \in \mathbb{R}^{m \times d'}$ , with  $m$  number of nodes of the initial graph of the layer, and  $d'$  the new dimensional features.

$$H'^{(l)} = \text{GNN}_{embed}^{(l)}\left(A^{(l)}, H^{(l)}\right) \quad (2)$$

2. *Matrix cluster assignment learning.* We then use the  $\text{GNN}_{pooling}^{(l)}$  to learn a new assignment matrix  $S^{(l)}$  to indicate which nodes of the graph at layer  $(l)$  will be clustered together to form a new coarser node at layer  $(l)$ . The matrix assignment is represented by the Eq. 3

$$S^{(l)} = \text{GNN}_{pooling}^{(l)}\left(A^{(l)}, H^{(l)}\right) \quad (3)$$

3. *Nodes and features pooling.* We finally aggregate nodes belonging to the same cluster and their features from  $H'^{(l)}$  using the assignment matrix  $S^{(l)}$  to output a new coarser graph represented by its adjacency matrix  $A^{(l+1)}$  and features matrix  $H^{(l+1)}$ . This pooling operation is done as follows:

$$A^{(l+1)} = S^{(l)T} A^{(l)} S^{(l)} \quad (4)$$

$$H^{(l+1)} = S^{(l)T} H'^{(l)} \quad (5)$$

We can notice that the number of nodes is decreasing at each new layer  $(l)$ . At the first layer  $(l=0)$ ,  $A^0$  and  $H^0$  correspond to the adjacency matrix  $A$  and the feature matrix  $X$  of the initial user graph respectively. The last layer  $L$  is a single cluster node, and represents the final vector embedding  $z_G$  of the whole graph.

In our work, only one kind of GNN is used: Graph Convolutional Networks (GCN) [12].

**Attention Mechanism-Based User Pooling Strategy.** This strategy is based on attention-based node embedding and allows each user node to judge which user neighbors are more important than the others during the node embedding process, according to the structure and features of the graph. Once node embeddings are generated, they are then aggregated to produce the node embedding of the whole graph. The attention mechanism (GAT) with cardinality preservation [20] is used to differentiate user neighbors by assigning them different scores. This attention mechanism is similar to the Transformers block used in BERT [2] for language modelling.

Let's  $\tilde{\mathcal{N}}_{(u_i)}$  be the multi-set of first-order neighbors of node  $u_i$ , including  $u_i$  itself. This second strategy is described as follows:

1. *Neighbors attention score.* For each node  $u_i \in \mathcal{U}$ , and each user neighbor  $u_j \in \tilde{\mathcal{N}}_{(u_i)}$ , we first compute the attention score  $e_{u_i u_j}$  by using an attention

function  $a$  on transformed features represented by the matrix  $W^{(l)}$  of the current layer ( $l$ ) for both nodes, as described in Eq. 6:

$$e_{u_i u_j}^{(l)} = a\left(\mathbf{W}^{(l)} h_{u_i}^{(l)}, \mathbf{W}^{(l)} h_{u_j}^{(l)}\right) \quad (6)$$

2. *Attention scores normalization.* We then normalize scores using a softmax function to get a probability distribution of each score.

$$\alpha_{u_i u_j}^{(l)} = \text{softmax}(e_{u_i u_j}^{(l)}) = \frac{\exp(e_{u_i u_j}^{(l)})}{\sum_{u_k \in \tilde{\mathcal{N}}(u_i)} \exp(e_{u_i u_k}^{(l)})} \quad (7)$$

3. *User node embedding.* The normalized scores are then used to compute the new user node representation  $h_{u_i}^{(l+1)}$

$$h_{u_i}^{(l+1)} = \sigma\left(\sum_{u_j \in \tilde{\mathcal{N}}(u_i)} \alpha_{u_i u_j}^{(l)} \mathbf{W}^{(l)} h_{u_j}^{(l)}\right) \quad (8)$$

with  $\sigma$  a non-linear activation function,  $h_{u_i}^{(l+1)} \in \mathbb{R}^{n \times d}$  with  $d$  feature dimension of the layer. Note that the cardinality preservation allows in Eq. 8 to scale the result before the use of the activation function  $\sigma$ . The final node representation  $h_{u_i}$  corresponds to the output of the last layer  $h_{u_i}^{(L)}$ .

4. *Graph embedding.* Finally, we compute the final graph embedding  $z_G$  by applying the READOUT function. A simple graph-level pooling function is used: we sum up each node representation at each iteration layer, and then concatenate them as shown in Eq. 9.

$$z_G = \left\| \left\|_{l=0}^{(L)} \left( \text{READOUT}(\{h_{u_i}^{(l)} \mid u_i \in U\}) \right) \right. \right. \quad (9)$$

This attention mechanism presents multiple advantages, in addition to state-of-the-art results in many benchmark graph classification tasks and interpretability. It allows the use of fixed number of parameters, and therefore does not depend on the graph size. It also presents transductive and inductive capabilities.

### 3.4 Graph Classification

The graph classification stage aims to classify the post represented by its graph embedding as controversial or non-controversial. To do so, we simply rely on a classic multi-layer perceptron classifier with the vector  $z_G$  as input. A Softmax activation on the output layer of dimension 2 is used.

## 4 Experimental Evaluation

### 4.1 Dataset

We evaluated the performance of our approach using a real-world Reddit dataset, in English, released by Hessel and Lee [8]. The same dataset is also covered by Zhong et al. [21]. The collected data covers a period from 2007 to February 2014. It contains 6 specific online channels (also called subreddit): *AskMen* (AM), *AskWomen* (AW), *Fitness* (FN), *LifeProTips* (LT), *personalfinance* (PF), *relationships* (RS). On Reddit, each user can comment on a post (threads) which is related to a specific topic (subreddit). Each subreddit contains a set of posts. Metadata and a tree-comment structure are associated to each post. Finally, only posts with a total of at least 30 comments are kept, assuming that less than 30 comments are not enough to build a significant graph. Each post is automatically labelled controversial or not controversial, depending on various post meta-data [8], among them the ratio between up-votes and down-votes<sup>4</sup>. We first separate our data according to the 6 subreddits. For each subreddit  $s$ , we create a set of user graphs  $\mathcal{G}_s$ , one graph per post, each set having at least 1000 posts. We then define  $\mathcal{G}_{s,train}$  and  $\mathcal{G}_{s,val}$  as our train and validation graph set respectively, all equally balanced between controversial and non-controversial posts. Considering all aspects and few more experiments, we only evaluate our approach on the same validation set. The accuracy metric is used to compare the performance of our approach to some existing ones as the dataset is equally balanced. We separately train the NLP model for user texts representation learning and the GNN for information structure learning.

### 4.2 Baseline

We compared our approach with the following representative works on controversy detection using the same Reddit dataset. Note that those methods perform a k-fold to evaluate their performance, using average accuracy as their metric.

- (POST (TEXT+TIME)) [8]. It only focuses on the posts content. It uses language modelling based on BERT [2] and extra-features based on the post timestamp of the post.
- (C-{TEXT\_RATE\_TREE} + POST) [8]. It is based on a simple binary classifier. Textual embeddings of a post are combined with structural features of the comment-tree (average representation of text comments, depth of the tree, etc.) of the post and are used as input of the classifier. We compare post with comment during the first hour and the first three hours.
- (DTPC-GCN) [21]. It is based on a Disentangled Topic-Post-Comment Graph Convolutional Network. Controversial posts are identified by using GCN model and by learning features depending on the respective subreddit post.

---

<sup>4</sup> Up-vote and down-vote indicate agreement and disagreement on the post.

### 4.3 First Experiment: Controversy Detection Based on Structural Information

We implemented our GNN-based controversy detection approach in Pytorch. The hierarchical graph representation learning is based on DIFFPOOL [18] and GCN [12]. We refer to it as **HRL-GCN** (Hierarchical Representation Learning based on GCN). We also test our strategy by using one and two pooling layers, respectively. For each pooling layer, a 3-layer GCN is used. We rely on the same loss and optimizer functions used in DIFFPOOL experimentation [18]. The attention-based node learning is based on GAT [17], using GAT-GC method, with the same hyperparameters used in [20]. We refer to it as **ARL-GAT**. We also test this strategy with two different node aggregators to compute the whole graph embedding, namely MEAN and SUM pooling functions. Both GNN-based strategies are trained with a learning rate at 0.01, a batch size at 32 during 100 epochs. Table 1 shows statistics on the different datasets.

**Table 1.** Statistics on the 6 real-world balanced Reddit datasets.

	AM	AW	FN	LS	PF	RS
Number of posts	3305	2969	3934	1573	1004	2248
Average number of user by post	72	67	76	79	47	48
Average number of comment by post	144	141	159	132	95	98
Average number of words by comment	41	42	34	28	52	61
Ratio of comments with tokens $\geq 256$	2.68	2.64	1.61	1.03	4.1	6.17

First experiments are performed without text representation to underline the importance of structural interaction between users in controversial discussion. Table 2 reports the accuracy results where the first four lines correspond to the baseline and the last four lines correspond to our experiments results.

**Table 2.** Performance comparison of our GNN-based controversy detection with baseline. Performance is evaluated using accuracy of the validation set.

	AM	AW	FN	LS	PF	RS
POST (TEXT+TIME)	68.1	65.4	65.5	66.2	66.5	69.3
DTPC-GCN	67.6					
POST + C- $\{\text{TEXT\_RATE\_TREE}\} < 1 \text{ h}$	71.1	70	68.1	67.9	66.1	65.5
POST + C- $\{\text{TEXT\_RATE\_TREE}\} < 3 \text{ h}$	<b>74.3</b>	72.3	70.5	<b>71.8</b>	<b>69.3</b>	<b>67.8</b>
ARL-GAT (MEAN-aggr)	65.7	69.2	<b>72.4</b>	58.4	53.7	62.9
ARL-GAT (SUM-aggr)	67.5	71	72.2	67	63.7	51.8
HRL-GCN (pool = 2)	69	72.2	71.7	<b>68.3</b>	65.7	63.6
HRL-GCN (pool = 1)	<b>69.6</b>	<b>74.6</b>	72.2	67.9	<b>68.2</b>	<b>66.7</b>

As shown in Table 2, our hierarchical approach (HRL-GCN) gets the best results among our experiments, with a weighted average accuracy at 70.6 using only one pooling layer. Our Attention-based approach ARL-GAT reaches 66.8 and 66.2 with the SUM and MEAN aggregator, respectively. HRL-GCN beats the DPTC-GCN [21] method and the hybrid method proposed by Hessel and Lee [8] with comments of the first hour for almost every dataset. Our proposed method (HRL-GCN, pool = 1) gets around state-of-the-art results on several datasets, even going up to 74.6 accuracy in the AW dataset, beating results in C- $\{\text{TEXT\_RATE\_TREE}\}$  + POST with comments of more than the first three hours. As the AM dataset is the biggest dataset, it could mean that our approach generalizes better when data are abundant, and less when data are sparse. As explained in [8], not enough comments are available for each dataset of the baseline. Indeed, when the subreddit *AskMen* (AM) has in average 10 comments after 45 min, *Relationships* (RS) does not even have those after 3 h. With more available comments, we might have a better embedding representation of our graph, which could lead to better performance.

Table 2 shows that our attention-based approach ARL-GAT, combined with the MEAN-aggregator, performs well in the three first datasets, beating our best baseline method on FN, with an accuracy of 72.4 on the validation set. On the other hand, it underperforms on the other three, falling to 53.7 on PF. PF and RS already have low results on our baseline, which means that the data is difficult to understand. This could also be explained by the fact that those 3 subreddits have the least average number of comments (as shown in Table 1), and therefore each user node has less neighbors. Attention scores are in fact less useful in these cases. In general, higher average degree of nodes could lead to better performance.

#### 4.4 Second Experiment: Textual Content and Structural Information Based on Controversy Detection

We conducted a second experiment to study the impact of adding textual node features to our GNN-based architecture. Instead of considering all options of our GNN-based architecture shown in Table 2, we only considered our hierarchical representations strategy HRL-GCN, with one pooling layer, as it realises the best accuracy scores.

Text features of comments and posts are extracted using different language model based on BERT, and are aggregated by user to be used as the initial features of our user nodes.

We use different models to extract those features:

- PT model. It only uses the pre-trained features to get the message representation. The last layer (768 dimensions) is outputted as our text embeddings.
- FT\_ITSELF model. We fine-tune [15] a BERT model using comments and posts of our train set  $\mathcal{G}_{s,train}$ , with an extra-layer of 64 neurons on top, in addition to the classifier layer. We label each comment with the controversy label of its respective post. Note that each subreddit is fine-tuned separately as we suppose that different communities express themselves differently, and texts can be interpreted differently.

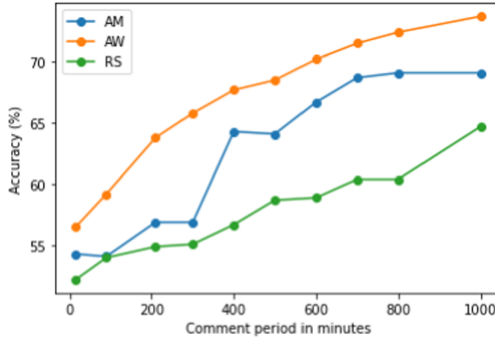
- FT\_SENTIMENT model. We fine-tune a BERT model using sentiment analysis with another Reddit dataset of comments (hosted on [kaggle.com](https://www.kaggle.com)), labeled as negative, positive or neutral. Indeed, we suppose here that sentiments can outline users’ behavior on controversial posts.

In all cases, we use the ‘base-bert-uncased’ version (with its corresponding tokenizer), with 12 transformer layers and 110 millions parameters. For time and memory performance, we only use 256 tokens max per text (instead of 512, as Table 1 shows that in average, less than 3% of the messages are represented by more than 256 tokens). For fine-tuning models, we use the same hyperparameters used in [15]. Table 3 shows the new accuracy scores obtained by incorporating text features in our HRL-GCN strategy.

**Table 3.** Performance of our best GNN approach enriched with different user text embeddings as initial node features.

	AM	AW	FN	LS	PF	RS
HRL-GCN (pool=1)	69.6	<b>74.6</b>	<u>72.2</u>	67.9	68.2	<u>66.7</u>
+ FT_SENTIMENT	69.1	72.9	70.5	<u>68.6</u>	66.7	64
+ FT_ITSELF	67.3	73.9	71.8	68.3	<u>70.6</u>	63.8
+ PT	<u>70.8</u>	73.7	71	65.4	<u>70.6</u>	64.7

For three of the six datasets, adding textual features improve controversy detection results. Our HRL-GCN strategy combined with the pre-trained (PT) BERT features gets better results when using AM and PF datasets, with 70.8 and 70.6 accuracy result respectively. Adding sentiment features from FT\_SENTIMENT allows us to increase accuracy from 67.9 to 68.6 on LS. Even if the content has interesting features for controversy detection, it remains brittle and community specific, which means that textual features can be more impactful in some subreddits than others. For instance, sentiments about controversial topic can be more meaningful in subreddit *LifeProTips* (LT) than in more personal subreddits, like *Relationships* (RS). The complexity of the data and the fact that datasets might be too small compared to the number of features (which goes up to 768 when using PT model) could also explain why our GNN-based models overfit on some datasets, and therefore does not improve accuracy results.



**Fig. 4.** Impact of comments availability on controversy detection performance.

Figure 4 shows the importance of comments availability. It reports accuracy results evolution over time (minutes) of three datasets when using our best HRL-GCN strategy combined with text features from PT. It clearly shows that the more available comments we have, the easier it is to detect controversy.

## 5 Conclusion

We presented an automatic controversy detection method on social media, based on GNN techniques. We considered this detection as a classification task and first exploited the structural information that characterizes user interactions by defining two strategies of graph embedding. The first strategy exploits hierarchical structure that may exist in the user graph. The second strategy allows each user to select its neighbors in the embedding nodes process. We also improved the graph embedding by incorporating textual content features computed from BERT model. Experimental evaluation shows promising results, even beating our baseline in several datasets<sup>5</sup>. However, our current Reddit dataset shows its limits, as a post has usually few comments, which prevents our GNN-based model from getting a better graph representation for controversy detection. The use of a different platform, such as Twitter, which provides more data per topics, or Wikipedia, could be an interesting lead to follow. In terms of future work, we would like to examine the appropriateness of other GNN techniques for controversy detection. For instance, it could be interesting to study the impact, in terms of performance improvements, of using GNN architectures that take into account nodes properties, mixing then structural, textual and user information. Compare results from different social media at the same time could also help to have a better understanding of the subject covered. Quantifying controversy using our approach is also an interesting perspective.

<sup>5</sup> This work was supported by grants from Janssen Horizon endowment fund.



## References

1. Beelen, K., Kanoulas, E., van de Velde, B.: Detecting controversies in online news media. In: 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1069–1072 (2017)
2. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT Conference: Human Language Technologies, vol. 1, pp. 4171–4186 (2019)
3. Dori-Hacohen, S., Jensen, D.D., Allan, J.: Controversy detection in wikipedia using collective classification. In: 39th International ACM SIGIR conference on Research and Development in Information Retrieval, pp. 797–800 (2016)
4. Emamgholizadeh, H., Nourizade, M., Tajbakhsh, M.S., Hashminezhad, M., Esfahani, F.N.: A framework for quantifying controversy of social network debates using attributed networks: biased random walk (BRW). *Soc. Netw. Anal. Mining* **10**(1), 1–20 (2020). <https://doi.org/10.1007/s13278-020-00703-1>
5. Garimella, K., Morales, G.D.F., Gionis, A., Mathioudakis, M.: Quantifying controversy on social media. *ACM Trans. Soc. Comput.* **1**(1), 3:1–3:27 (2018)
6. Garimella, K., Morales, G.D.F., Gionis, A., Mathioudakis, M.: Reducing controversy by connecting opposing views. In: Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI, pp. 5249–5253 (2018)
7. Guerra, P.H.C., Jr., W.M., Cardie, C., Kleinberg, R.: A measure of polarization on social media networks based on community boundaries. In: Seventh International Conference on Weblogs and Social Media, ICWSM. The AAAI Press (2013)
8. Hessel, J., Lee, L.: Something’s brewing! early prediction of controversy-causing posts from discussion features. In: Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, pp. 1648–1659 (2019)
9. Jang, M., Allan, J.: Improving automated controversy detection on the web. In: Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR, pp. 865–868. ACM (2016)
10. Jang, M., Dori-Hacohen, S., Allan, J.: Modeling controversy within populations. In: Proceedings of the SIGIR International Conference on Theory of Information Retrieval, ICTIR, pp. 141–149. ACM (2017)
11. Jang, M., Foley, J., Dori-Hacohen, S., Allan, J.: Probabilistic approaches to controversy detection. In: 25th ACM International Conference on Information and Knowledge Management, CIKM, pp. 2069–2072 (2016)
12. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, ICLR. OpenReview.net (2017)
13. Mendoza, M., Parra, D., Soto, Á.: GENE: graph generation conditioned on named entities for polarity and controversy detection in social media. *Inf. Process. Manag.* **57**(6), 102366 (2020)
14. Morales, A.J., Borondo, J., Losada, J.C., Benito, R.M.: Measuring political polarization: Twitter shows the two sides of venezuela. *CoRR* (2015)
15. Sun, C., Qiu, X., Xu, Y., Huang, X.: How to fine-tune BERT for text classification? In: Sun, M., Huang, X., Ji, H., Liu, Z., Liu, Y. (eds.) CCL 2019. LNCS (LNAI), vol. 11856, pp. 194–206. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-32381-3\\_16](https://doi.org/10.1007/978-3-030-32381-3_16)
16. Sznajder, B., et al.: Controversy in context. *CoRR* (2019)

17. Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: 6th International Conference on Learning Representations, ICLR. OpenReview.net (2018)
18. Ying, Z., You, J., Morris, C., Ren, X., Hamilton, W.L., Leskovec, J.: Hierarchical graph representation learning with differentiable pooling. In: Annual Conference on Neural Information Processing Systems, NeurIPS, pp. 4805–4815 (2018)
19. Ortiz de Zarate, J.M., Feuerstein, E.: Vocabulary-based method for quantifying controversy in social media. In: Alam, M., Braun, T., Yun, B. (eds.) ICCS 2020. LNCS (LNAI), vol. 12277, pp. 161–176. Springer, Cham (2020). [https://doi.org/10.1007/978-3-030-57855-8\\_12](https://doi.org/10.1007/978-3-030-57855-8_12)
20. Zhang, S., Xie, L.: Improving attention mechanism in graph neural networks via cardinality preservation. In: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020, pp. 1395–1402 (2020). [ijcai.org](http://ijcai.org)
21. Zhong, L., Cao, J., Sheng, Q., Guo, J., Wang, Z.: Integrating semantic and structural information with graph convolutional network for controversy detection. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL, pp. 515–526. Association for Computational Linguistics (2020)



# GMGCN: Gated Memory Graph Convolutional Network for Passenger Demand Prediction

Tianyuan Bi<sup>1,2</sup>, Kai Han<sup>1,2</sup>(✉), and Cheng Shen<sup>1,2</sup>

<sup>1</sup> School of Computer Science and Technology, University of Science and Technology of China, Hefei, China

[hankai@ustc.edu.cn](mailto:hankai@ustc.edu.cn)

<sup>2</sup> Suzhou Institute for Advanced Research, No. 188, Ren' Ai Road, Suzhou Industrial Park, Suzhou 215123, Jiangsu, People's Republic of China

**Abstract.** With the rapid proliferation of smart ways to travel, the accurate prediction of passenger demand in urban areas has become an important task. Existing methods mainly focus on the modeling of Euclidean correlation between spatial neighborhoods. However, in regional-level prediction tasks, the spatial Euclidean correlation can no longer satisfy the requirement for accurate prediction due to the inability to effectively capture spatial information. To solve such difficulty, we design a novel gated-memory graph convolutional network (GMGCN) for passenger demand prediction. Specifically, we focus on modeling the non-Euclidean spatial correlation between regions. The non-Euclidean correlation can not only model the spatial distances of each region, but also construct long-range information flow caused by transportation. Moreover, by adding gated temporal convolution and long-term memory modules, GMGCN can better model the relationship between different time periods and avoid information loss in the recurrent neural network, which boosts the GMGCN's ability to capture cyclical changes in passenger demand over time. Besides, the impact of the external features like weather and holidays that change over time are integrated into the model by an embedding module. Extensive experiments on real datasets quantitatively and qualitatively demonstrate the superiority of our approach compared with the state-of-the-art.

**Keywords:** Graph convolutional network · Traffic flow · Demand prediction

## 1 Introduction

The development of “Internet + Transportation” has given birth to the production of traffic big data, which helps to build smart travel systems, alleviates urban traffic congestion, and achieves green travel. Passenger demand prediction is an important application scenario for smart travel. At present, passengers can easily find available vehicles around them through apps such as Uber and Didi.

© Springer Nature Switzerland AG 2021

W. Zhang et al. (Eds.): WISE 2021, LNCS 13080, pp. 355–369, 2021.

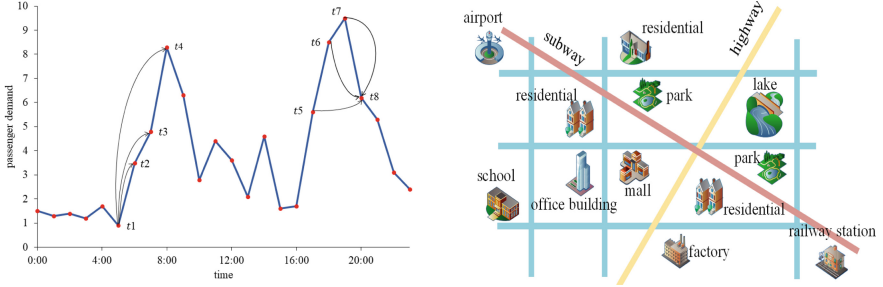
[https://doi.org/10.1007/978-3-030-90888-1\\_27](https://doi.org/10.1007/978-3-030-90888-1_27)

However, these ride-sharing platforms still have the problem of low accuracy in predicting passenger demand. Accurate prediction of the passenger demand, especially the demand for passenger within multiple steps, is the key to dispatch vehicles effectively, which helps to improve vehicle utilization, reduce waiting time, ease traffic congestion, and greatly improve resource scheduling capabilities for the intelligent transportation system.

Previous work [1] on this topic can be roughly divided into two categories: classical statistical based methods and machine learning based models. Statistical models, such as autoregressive integrated moving average [2], are limited to handle low-dimensional time series data. Therefore, with the dramatic increase on the volume of traffic data, most of the recently proposed methods focus on machine learning. Machine learning methods with the capability of capturing complex non-linear relationships, like support vector regression [3], tend to outperform the statistical methods, with respect to handling complex traffic forecasting problems [4]. With the development of deep learning models, methods using neural networks [1] achieve promising results in traffic prediction. The potential of artificial intelligence methods for traffic prediction is gradually exploited. Deep learning models for traffic forecasting, such as deep belief networks [5] and stacked auto-encoders [6], can effectively learn high dimensional features and achieve good forecasting performance. Recurrent neural network (RNN) and its variants [7], including long short-term memory (LSTM) [8] and gated recurrent unit networks [9], also show great potential for solving traffic forecasting problems [4].

However, it is challenging to describe the nonlinear and dynamic spatial-temporal dependence in passenger demand. Future passenger demand in the area is affected by the historical demand in the area, but each time step of passenger demand may have a dynamic time-varying impact on the prediction target. As shown in Fig. 1(a), the importance of  $t_1$  on  $t_2$ ,  $t_3$ ,  $t_4$  varies. Meanwhile, the importance of  $t_5$ ,  $t_6$ ,  $t_7$  on  $t_8$  are vastly different. Passenger demand is also affected by demand from other parts of the city, such as the similar neighborhood condition and transportation condition. As shown in Fig. 1(b), although airports and railway stations are far away, their relationship cannot be measured simply by Euclidean distance because they have similar functions and are connected by a subway. Moreover, under different weather conditions, the passenger demand will also change. For example, more people prefer to take a taxi than ride shared bicycles on rainy days. Therefore, the complex nonlinear spatial-temporal correlation and external environmental influence the make passenger demand prediction a challenging task.

To tackle these challenges, we propose a new in-depth research method, which captures spatial-temporal correlation efficiently. On the one hand, the non-Euclidean correlations is constructed in our network to better capture the global correlation brought by the transportation network. On the other hand, by using the information of different time periods, we capture the periodic time associations in the transportation network. Moreover, attention mechanism is applied in the input embedding module to model the spatial and temporal dependencies holistically. Besides, we extract the hidden features of the external environment



(a) Trends in passenger demand over time (b) Example of non-Euclidean spatial correlation.

**Fig. 1.** The spatial-temporal correlation diagram of passenger demand.

and dynamically adjust the time dependence accordingly to improve prediction accuracy. In this way, our model can capture more complex spatial-temporal correlations and obtain better performance on demand prediction.

Our contributions are summarized as follows:

- (1) We add a lightweight attention module to the input, to improve the model's performance substantially.
- (2) We design a gated memory spatial-temporal convolution module to model the long-term time dependence and non-Euclidean spatial correlation. This module consists of gated temporal convolution, residual spatial graph convolution, and memory unit. The first two are used to capture spatial-temporal correlations, while the latter is used to prevent the information loss.
- (3) Extensive experiments are conducted on two real-world datasets. Compared with other methods, our proposed method achieves better performance on passenger demand prediction, demonstrating the effectiveness of our method.

## 2 Related Work

**Passenger Demand Prediction.** Traditional demand prediction methods use time series models [10], cluster models [11], or hybrid methods [12] to capture correlations. With the prosperity of deep learning, many recent methods adopt convolutional neural networks (CNNs) to model spatial dependencies in traffic forecasting. These methods typically employ RNN and variants such as LSTM to capture temporal correlation [11]. Some methods [13] use CNN to extract spatial relationships from the entire city. Several works [12] propose a hybrid model combining CNN and RNN to extract spatial and temporal correlations at the same time. To better capture the spatial dependency, MGCN [14] proposes to encode the auxiliary information into a graph through GCN, and stacks three GCNs to extract the spatial dependence. There are also works that encode additional features (such as weather, etc.) to capture more detailed relationships. CoST-Net [15] proposes a deep learning model for predicting a variety of traffic

demands, but this model only predicts the traffic demand at the next moment and can not extend to multiple-step prediction tasks. These works are still limited in adaptively modeling dynamic spatial-temporal dependencies, and it is still challenging to get a good performance on multiple-step prediction.

**Graph Convolution Network.** Recently, the graph convolution generalizes the traditional convolution to data with graph structures, to solve the problem that traditional convolution methods can only be applied to standard grid data. The graph convolution method can be divided into two categories: the spatial methods and the spectral methods [16]. The spatial methods directly perform convolution filters on a graph’s nodes and their neighbors. So, the core of this kind of methods is to proper the neighborhood of nodes. Niepert et al. [17] proposed a heuristic linear method to select the neighborhood of every center node, which achieved good results in social network tasks. Li et al. [18] proposed several division strategies, dividing the neighborhood of each node into an equal number of subsets, and introducing graph convolution into human action recognition tasks. In the spectral methods, the locality of the graph convolution is considered by spectral analysis. A general graph convolution framework based on the Graph Laplacian is proposed by Bruna et al. [19], then Defferrard et al. [20] optimized the method by using Chebyshev polynomial approximation to realize eigenvalue decomposition. Graph convolution can process non-Euclidean data, so it is widely used in real-world traffic networks [21], social networks [22], etc. Moreover, graph convolution is widely applied in the field of computer vision, for example, Zhou et al. [23] use graph convolutional network to model the relationships between different semantic parts in image inpainting.

### 3 Methodology

**Definition and Problem.** In fact, the range of passengers’ car-hailing and drivers’ order receiving is in a small area. We divide the city into  $N \times N$  equal size grids, and each grid is defined as an area  $v \in V$ , where  $V = \{v_o, v_1, \dots, v_N\}$  is the set of all disjoint areas in the city and  $N = |V|$  is the number of areas. A graph is represented by  $\mathcal{G} = (V, A)$ .  $A \in \mathbb{R}^{N \times N}$  is the adjacency matrix whose entries represent the connections between areas. At each time step  $t$ ,  $X_t^i \in \mathbb{R}^{d_{in}}$  represents the passenger demand of all regions on node  $v_i$  in time step  $t$ , and  $X_t = (X_t^1, \dots, X_t^N) \in \mathbb{R}^{N \times d_{in}}$  represents the passenger demand of all regions in time step  $t$ . Then, the whole historical traffic data can be denoted by  $X = (X_1, \dots, X_T) \in \mathbb{R}^{N \times d_{in} \times T}$ , where  $T$  is the number of time steps. Another vector  $E_t \in \mathbb{R}^{d_{ex}}$  represents the external features in time step  $t$ , and  $E = (E_1, \dots, E_T) \in \mathbb{R}^{d_{ex} \times T}$  represents the whole historical external features, where  $ex$  is the dimension of external features.

Given a traffic Network  $\mathcal{G}$ , the citywide historical passenger demand sequence  $X$ , and time features  $E$ , the target is to learn a prediction function  $f$  that forecasts the citywide passenger demand sequence in the next  $\tau$  time steps. The

mapping relation is represented as follows

$$[X, E, \mathcal{G}] \xrightarrow{f} \hat{X}_{(t+1):(t+\tau)}, \tag{1}$$

where  $\hat{X}_{(t+1):(t+\tau)} \in \mathbb{R}^{N \times d_{out} \times \tau}$ .

### 3.1 Framework

We present the framework of GMGCN in Fig. 2. It consists of a spatial dependent modeling layer, a long-term temporal attention layer, stacked gated-memory spatial-temporal convolution (GMSTCN) layers, and an output layer. By stacking multiple GMSTCN layers, GMGCN is able to model spatial dependencies at long temporal levels. We will elaborate the specific modules of our GMGCN below.

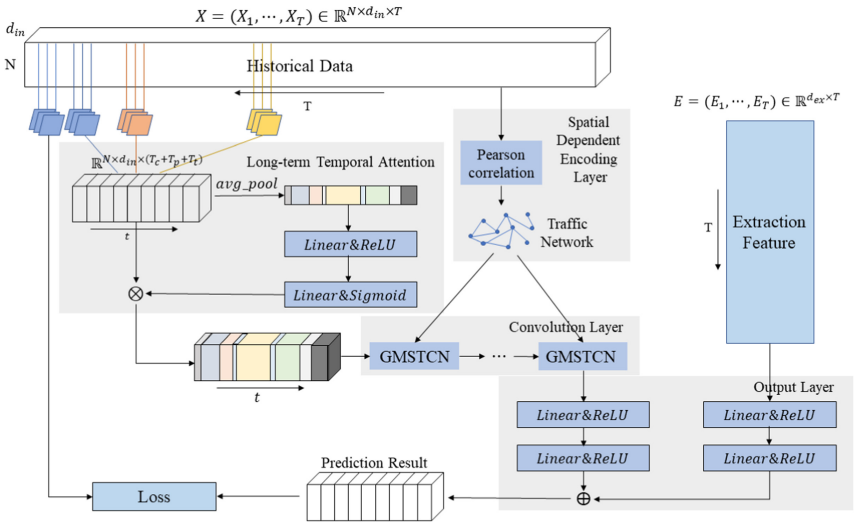


Fig. 2. The framework of GMGCN

### 3.2 Passenger Demand on Graph

First, we model global spatial dependencies across different grids in order to accurately predict passenger demand. We argue that spatial correlation does not exclusively rely on geographic locations. Therefore, we encode the adjacency similarity matrix to capture the spatial dependencies between different grids:

$$A_{i,j} = \begin{cases} 1, & \text{if } Sim(X^i, X^j) > \epsilon \\ 0, & \text{otherwise} \end{cases}, \tag{2}$$

where  $\epsilon$  is the sparsity threshold of adjacency matrix  $A$ . We choose Pearson correlation coefficient [24] to quantify the correlation between different regions:

$$\begin{aligned} Sim(X^i, X^j) &= Pearson(X^i, X^j) \\ &= \frac{Cov(X^i, X^j)}{\sigma_{X^i}\sigma_{X^j}}, \tag{3} \\ &= \frac{E(X^i X^j) - E(X^i)E(X^j)}{\sqrt{E((X^i)^2) - E^2(X^i)}\sqrt{E((X^j)^2) - E^2(X^j)}} \end{aligned}$$

where  $X^i = (X_0^i, \dots, X_T^i) \in \mathbb{R}^{d_{in} \times T}$  represents the whole historical passenger demand on node  $v_i$ .

The adjacent similarity matrix is easy to implement and its computation cost is small. Pearson correlation coefficient can describe the trend how two groups of linear data change and move together. At the same time, using historical data to calculate the adjacency matrix does not require additional data annotation of the dataset. Therefore, the model can be well applied to different datasets.

### 3.3 Long-Term Temporal Attention

To improve the efficiency, we do not use all the historical passenger demands. At the same time, we are interested in capturing the long-term periodic time dependence, so we intercept three time series segments of length  $T_c$ ,  $T_p$  and  $T_t$  along the time axis as the input of the closeness, period and trend component respectively. Specifically, we employ the historical passenger demand series of the proximity time steps before the forecast time period, the same time period of the previous days, and the same time period of the previous weeks as the proximity, period, and trend components, respectively. Details about the three time series segments are as follows:

$$X_c = (X_{t_0-T_c+1}, \dots, X_{t_0}) \in \mathbb{R}^{N \times d_{in} \times T_c}, \tag{4}$$

$$\begin{aligned} X_p = &(X_{t_0-(T_d/T_q)*q+1}, \dots, X_{t_0-(T_d/T_q)*q+T_p}, X_{t_0-(T_d/T_q-1)*q+1}, \dots, \\ &X_{t_0-(T_d/T_q-1)*q+T_p}, \dots, X_{t_0-q+1}, \dots, X_{t_0-q+T_p}) \in \mathbb{R}^{N \times d_{in} \times T_p}, \end{aligned} \tag{5}$$

$$\begin{aligned} X_t = &(X_{t_0-7*(T_t/T_q)*q+1}, \dots, X_{t_0-7*(T_t/T_q)*q+T_p}, X_{t_0-7*(T_t/T_q-1)*q+1}, \dots, \\ &X_{t_0-7*(T_t/T_q-1)*q+T_p}, \dots, X_{t_0-7*q+1}, \dots, X_{t_0-7*q+T_p}) \in \mathbb{R}^{N \times d_{in} \times T_t}, \end{aligned} \tag{6}$$

where  $q$  is the sampling frequency per day, and  $T_q$  is the size of the predicting window.

We concatenate  $X_c, X_p, X_t$  as input  $X$ :

$$X = (X_t, X_p, X_c) \in \mathbb{R}^{N \times d_{in} \times (T_t+T_p+T_c)}. \tag{7}$$

In order to distinguish the correlation between different time slices in different situations, we introduce an attention mechanism to adaptively assign different



importance to the data:

$$z_t = F_{avg\_pool}(X) = \frac{1}{d_{in} \times N} \sum_{i=1}^N \sum_{j=1}^{d_{in}} X_t^i(j), t = 1, \dots, (T_t + T_p + T_c), \quad (8)$$

$$Z = (z_1, \dots, z_{T_t+T_p+T_c}) \in \mathbb{R}^{T_t+T_p+T_c}, \quad (9)$$

$$\bar{X} = Xs, s = \sigma(W_2\delta(W_1Z)), \quad (10)$$

where  $W_1 \in \mathbb{R}^{\frac{T_t+T_p+T_c}{r} \times (T_t+T_p+T_c)}$  and  $W_2 \in \mathbb{R}^{(T_t+T_p+T_c) \times \frac{T_t+T_p+T_c}{r}}$  are learnable weights.

### 3.4 Gated Memory Spatial-Temporal Convolution (GMSTCN) Layer

Figure 3 shows the framework of GMSTCN layer. Each layer includes three main modules, namely the temporal convolution module, the spatial convolution module, and the memory module. We implement temporal and spatial convolution in the same layer of the network to effectively extract spatial-temporal dependencies and employ a memory mechanism to prevent information loss during the recurrent neural network process while introducing as few parameters as possible to prevent a dramatic increase in training difficulty. The detailed mechanism of each module is described in the following subsections.

**Temporal Convolution Module.** Gated mechanisms have a powerful ability to control information flow in the temporal dimension. We use two dilated convolutions to learn different hidden representations in time dimension. Then, two different activation functions are used as output gates to learn different temporal features. Given the input of the  $l$ -th layer  $X^l$ , we can formulate the temporal convolution module as:

$$H^l = \sigma(Dilated\_Conv_1(X^l)) \odot \tanh(Dilated\_Conv_2(X^l)), \quad (11)$$

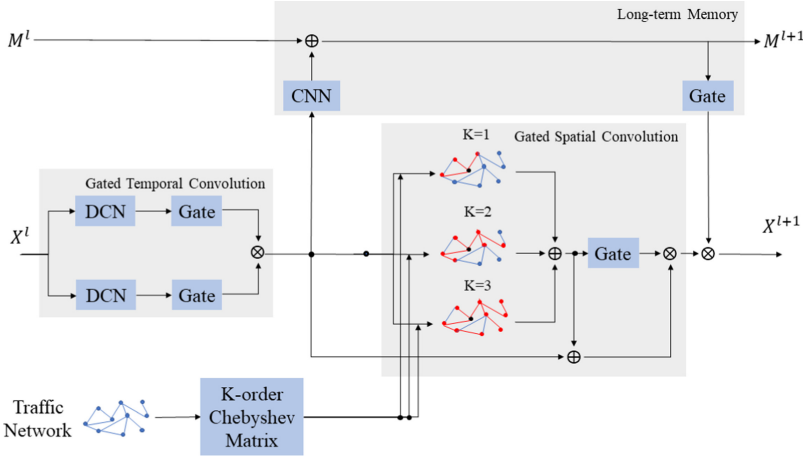
where  $\odot$  is element-wise product,  $\sigma$  is the sigmoid functions,  $\tanh$  is the tangent hyperbolic function. Both nonlinear activation functions determine the different ratio of information passed to the next layer.

We deploy the dilated convolution on the input features, to enlarge the receptive field by stacking dilated convolution layers with dilation factors in an increasing order. Moreover, we adjust the dilation factors and the number of network layers to reduce the dimensionality, thus effectively aggregating the information from the input time steps to the prediction results.

**Memory Module.** In order to improve the sensitivity of spatial-temporal correlation in our model and avoiding information loss, we designed a memory module to take advantage of the long-term dependence.

$$M^{l+1} = o(M^l) + \sigma(Conv(H^l)), \quad (12)$$

where  $M^l$  is the input of the memory module of the  $l$ -th layer and also the output of the  $(l - 1)$ -th layer,  $\sigma$  is a sigmoid function which determines the ratio of information passed to the next layer, and  $o$  is a dimensional control function to change the dimension of  $M^l$  to the dimension of  $\sigma(Conv(H^l))$ .



**Fig. 3.** The framework of GMSTCN layer. It contains a gated temporal convolution module, a gated spatial convolution module, and a long-term memory module. The memory module and the output module interact through a gating mechanism.

**Spatial Convolution Module.** Graph convolution network (GCN) is defined over a graph  $\mathcal{G} = (V, A)$ , where  $V$  is the set of all vertices and  $A \in \mathbb{R}^{|V| \times |V|}$  is the adjacency matrix that represents the connections between vertices. The number of vertices is denoted as  $N = |V|$ . Laplacian matrix of a graph is defined as  $L = D - A$ , and its normalized form is  $L = I_N - D^{-\frac{1}{2}}AD^{-\frac{1}{2}} \in \mathbb{R}^{N \times N}$ , where  $I_N$  is a unit matrix, and the degree matrix  $D \in \mathbb{R}^{|V| \times |V|}$  is a diagonal matrix, consisting of node degrees,  $D_{ii} = \sum_j A_{ij}$ . The eigenvalue decomposition of the Laplacian matrix is  $L = U\Lambda U^T$ , where  $\Lambda = diag([\lambda_0, \dots, \lambda_{N-1}]) \in \mathbb{R}^{N \times N}$  is a diagonal matrix, and  $U$  is Fourier basis. Based on this, the signal  $x$  on the graph  $\mathcal{G}$  is filtered by a kernel  $g_\theta$ :

$$g_{\theta * G} x = g_\theta(L)x = g_\theta(U\Lambda U^T)x = Ug_\theta(\Lambda)U^T x. \tag{13}$$

However, it is expensive to directly perform the eigenvalue decomposition on the Laplacian matrix when the scale of the graph is large. In this paper, we use Chebyshev polynomials [25] to solve this problem approximately but efficiently:

$$g_{\theta * G} x = g_\theta(L)x = \sum_{k=0}^{N-1} \theta_k T_k(\tilde{L})x, \tag{14}$$

where the parameter  $\theta \in \mathbb{R}^N$  is a vector of polynomial coefficients,  $T_k(\tilde{L}) \in \mathbb{R}^{N \times N}$  is the Chebyshev polynomial of order  $k$ ,  $\tilde{L} = 2L/\lambda_{max} - I_n$  is the scaled Laplacian matrix. We further approximate  $\lambda_{max} \approx 2$ , then  $g_{\theta * G} x$  is simplified as:

$$g_{\theta * G} x = \theta'_0 x + \theta'_1 (L - I_N) x = \theta'_0 x - \theta'_1 D^{-\frac{1}{2}} A D^{-\frac{1}{2}} x, \quad (15)$$

where  $\theta'_0$  and  $\theta'_1$  are free parameters. To constrain the number of parameters further to address overfitting and to minimize the number of operations, we let  $\theta'_0 = -\theta'_1 = \theta$ , then the above equation simplifies to:

$$g_{\theta * G} x = \theta (I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}}) x = \theta (D^{-\frac{1}{2}} \hat{A} D^{-\frac{1}{2}}) x, \quad (16)$$

where  $\hat{A} = A + I_N$ .

Moreover, we adopt the adaptive gated mechanism to model the complex non-linearity in passenger demand prediction:

$$X^{l+1} = (g_{\theta * G} H^l + H^l) \odot \sigma_1(g_{\theta * G} H^l) \odot \sigma_2(M^{l+1}), \quad (17)$$

where  $\odot$  is element-wise product,  $H^l$  is the output of the temporal convolution module of the  $l$ -th layer,  $X^{l+1}$  is the output of the  $l$ -th layer and also the input of the  $(l+1)$ -th layer, and  $\sigma$  is sigmoid function that determines the ratio of information passed to the next layer.

### 3.5 Output Layer

Taking external factors into account, such as weather, holidays, etc., in the passenger demand, we add these external factors to the network to obtain more accurate forecasting capabilities.

First, we encode the external features such as week, holiday, and weather into feature matrices by using one-hot encoding respectively, and connect each feature matrix to obtain the external feature matrix  $E = (E_1, \dots, E_T) \in \mathbb{R}^{d_{ex} \times T}$ . In particular, for the time, weekend, and holiday information, we use the labels and order of the time series to make judgments. And for weather conditions, we encode the information provided by the dataset. Then we use different linear transformation for decreasing dimension and ReLU function to process the output of the GMSTCN layers and external features to match the dimensions. Finally, we merge the two outputs and use a tanh activation function to control the transmission of information to obtain the predicted result.

$$\hat{X} = \tanh(E_{out} + M_{out}), \quad (18)$$

where  $E_{out}$  is the external feature processed by the fully connected layer and the relu activation function, and  $M_{out}$  is the output of the last GMSTCN layer processed by the fully connected layer and the relu activation function.

We choose to use mean absolute error (MAE) as the loss function of our model, which is defined by:

$$L(\theta) = \frac{1}{\tau N d_{out}} \sum_{i=1}^{\tau} \sum_{j=1}^N \sum_{k=1}^{d_{out}} |\hat{X}_i^j(k) - X_{T+i}^j(k)|. \quad (19)$$

## 4 Experiment

### 4.1 Datasets

We verify our model on two large real world datasets, BickNYC and TaxiBJ [26]. The public BikeNYC dataset consists of the bike demand and the time meta. The bike demand covers the shared bike hire and returns data of CityBike in New York from 1 Apr 2014 to 30th Sept 2014. Each time step is one hour. The public TaxiBJ dataset contains taxi demand in Beijing from 1 Mar 2015 to 30th Jun 2015. TaxiBJ contains passenger demand, time meta, and meteorological data. Each time step is 30 min. Table 1 presents the details of the two datasets.

**Table 1.** The detailed information of the datasets

Dataset	BickNYC	TaxiBJ
Vehicle	Bike	Taxi
Locations	New York	Beijing
Number of areas	128 ( $16 \times 8$ )	256 ( $16 \times 16$ )
Time interval	60 min	30 min
Time span	01/04/2014–30/09/2014	01/03/2015–30/06/2015
External factors		
Holidays	✓	✓
Weekdays	✓	✓
Weather conditions	\	16 types ( <i>e.g.</i> , Sunny, Rainy)
Temperature/ $^{\circ}$ C	\	[−6.6, 35.7]
Wind speed/mph	\	[0, 48.6]

### 4.2 Baselines

We compare our model with following 5 baseline methods with excellent performance: 1) OLR: Ordinary Linear Regression, which is a traditional model, 2) DeepST [26]: Deep Spatial-Temporal Residual Network, 3) ConvLSTM [27]: Convolution Long Short Term Memory network, 4) ASTGCN [28]: Attention based Spatial-Temporal Graph Convolution Network, 5) STG2Seq [29]: Spatial-temporal Graph to Sequence Model.

These benchmark methods cover both statistical and deep learning methods, using convolutional neural networks, graph convolutional neural networks, long and short-term memory networks, and encoders-decoders, respectively.

In our experiments, we use three most-widely adopted evaluation metrics, Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE) to measure different models' performance. We compare the single time step passenger demand prediction capability of GMGCN with the five baseline methods described above.

### 4.3 Experiment Settings

Before feeding the data into the model, categorical features such as hour of a day, day of a week and holidays are transformed by the one-hot encoding. The passenger demand is normalized by Min-Max normalization [30] for training and re-scaled for evaluating the prediction accuracy. We split the dataset in chronological order with 70% for training, and 20% for testing, and the remaining data for validating. We set the lengths of closeness, period and trend to 6, 6, and 12 respectively. We merge two GMSTCN layers into one block, and set their dilation rate to 1 and 2, respectively. To cover the input sequence length, we stack eight blocks. We train our model using adam optimizer with an initial learning rate of 0.001. For training phase, the batch size is 16 and epochs are 100. We implemented our model in Python with PyTorch 1.8.1 and cu101. Our experiments are conducted under a computer environment with one NVIDIA P100 PCIe GPU ACCELERATOR.

### 4.4 Results

Table 2 shows the results of passenger demand prediction performance at one time step on BickNYC and TaxiBJ datasets.

**Table 2.** The performance of our model and baselines

Method	BikeNYC			TaxiBJ		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE
OLR	4.652	8.502	0.391	14.937	23.921	0.276
DeepST [26]	2.549	6.603	0.242	11.264	18.305	0.157
ConvLSTM [27]	2.435	4.745	0.226	11.461	18.788	0.163
ASTGCN [28]	2.438	4.759	0.220	11.573	19.101	0.167
STG2Seq [29]	2.257	4.513	0.210	10.219	17.241	0.138
<b>GMGCN</b>	<b>1.443</b>	<b>3.502</b>	<b>0.149</b>	<b>7.576</b>	<b>14.078</b>	<b>0.112</b>

Compared to the baselines, the prediction results of GMGCN outperform both non-deep learning-based method OLR, and deep learning-based methods. To verify the performance of GMGCN in multi-step prediction, we compare the model with the DeepST and STG2Seq on BikeNYC dataset, and visualize the comparison of the experimental results in Fig. 4 based on the experiment results. The experimental results show that the performance of our proposed GMGCN is better than the baseline methods at all time steps. At the same time, as the forecast time increases, the error trend of GMGCN is also smaller than that of the baseline methods. All the experiment results above demonstrate the advantages of our model on the passenger demand prediction.

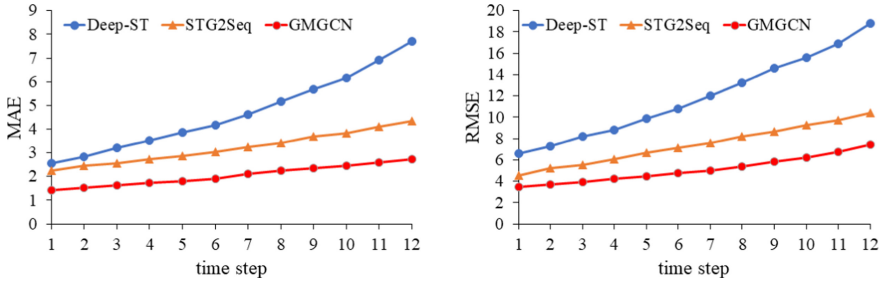


Fig. 4. The performance of our model and baselines on different predicting time steps

### 4.5 Ablation Study

In this section, we investigate the effect of some components in our method, including the temporal attention module, dilation convolution module, memory module, gate mechanism in spatial convolution module, and the external feature.

Table 3 compares the prediction performance with different combinations of components at 3 time steps. The experimental results show that the addition of the long-term module reduces the MAE by 10.4% which is the most influential module. Furthermore, the dilation convolution and gated mechanism reduce the mae by 6.4% and 2.3%, respectively. Besides, the temporal attention mechanism and the addition of external features also both improve the prediction ability of the model to some extent. In summary, the GMGCN can achieve the best results, and each component of our model make sense.

Table 3. The performance of variants of our model on BikeNYC

Att	Dila	Mem	Gate	Ext	MAE	RMSE	MAPE
✓				✓	10.496	19.051	0.182
✓			✓	✓	10.360	18.889	0.172
✓	✓			✓	9.931	18.083	0.160
✓		✓		✓	9.502	17.339	0.145
✓	✓		✓	✓	9.800	17.929	0.156
✓		✓	✓	✓	9.377	17.190	0.144
	✓	✓	✓	✓	9.175	17.007	0.140
✓	✓	✓		✓	8.991	16.458	0.132
✓	✓	✓	✓		8.925	16.546	0.132
✓	✓	✓	✓	✓	<b>8.781</b>	<b>16.317</b>	<b>0.130</b>

## 5 Conclusion

In this paper, we propose a gated memory graph convolutional network for passenger demand prediction. Specifically, we focus on the non-Euclidean correlation between regions and model the global spatial correlation between regions by adjacency similarity matrix. Moreover, by adding gated dilated time convolution and long-term memory modules, GMGCN can better model the relationship between different time periods and avoid information loss in the recurrent neural network, which boosts the GMGCN's ability to capture cyclical changes in passenger demand over time with the addition of a small number of parameters. Besides, the impact of the external features like weather and holidays that change over time are corrected for the prediction results by a fully connected layer and activation function processing. Extensive experiments on real datasets quantitatively and qualitatively demonstrate the superiority of our approach compared with the state-of-the-art.

**Acknowledgments.** This work was supported by the National Key R&D Program of China under Grant No. 2018AAA0101204, the National Natural Science Foundation of China (NSFC) under Grant No. 61772491, and Anhui Initiative in Quantum Information Technologies under Grant No. AHY150300.

## References

1. Park, D., Rilett, L.R.: Forecasting freeway link travel times with a multilayer feed-forward neural network. *Comput.-Aided Civil Infrastruct. Eng.* **14**(5), 357–367 (1999)
2. Hamed, M.M., Al-Masaeid, H.R., Bani Said, Z.M.: Short-term prediction of traffic volume in urban arterials. *J. Transp. Eng.* **121**(3), 249–254 (1995)
3. Smola, A.J., Schölkopf, B.: A tutorial on support vector regression. *Stat. Comput.* **14**(3), 199–222 (2004)
4. Ma, X., Tao, Z., Wang, Y., Haiyang, Yu., Wang, Y.: Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. *Transp. Res. Part C: Emerging Technol.* **54**, 187–197 (2015)
5. Huang, W., Song, G., Hong, H., Xie, K.: Deep architecture for traffic flow prediction: deep belief networks with multitask learning. *IEEE Trans. Intell. Transp. Syst.* **15**(5), 2191–2201 (2014)
6. Lv, Y., Duan, Y., Kang, W., Li, Z., Wang, F.-Y.: Traffic flow prediction with big data: a deep learning approach. *IEEE Trans. Intell. Transp. Syst.* **16**(2), 865–873 (2014)
7. Lv, Z., Xu, J., Zheng, K., Yin, H., Zhao, P., Zhou, X.: LC-RNN: a deep learning model for traffic speed prediction. In: *IJCAI*, pp. 3470–3476 (2018)
8. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
9. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, pp. 1724–1734. Association for Computational Linguistics, October 2014

10. Moreira-Matias, L., Gama, J., Ferreira, M., Mendes-Moreira, J., Damas, L.: Predicting taxi-passenger demand using streaming data. *IEEE Trans. Intell. Transp. Syst.* **14**(3), 1393–1402 (2013)
11. Li, Y., Zheng, Y., Zhang, H., Chen, L.: Traffic prediction in a bike-sharing system. In: *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 1–10 (2015)
12. Zhang, K., Feng, Z., Chen, S., Huang, K., Wang, G.: A framework for passengers demand prediction and recommendation. In: *2016 IEEE International Conference on Services Computing (SCC)*, pp. 340–347. IEEE (2016)
13. Zhang, J., Zheng, Y., Qi, D., Li, R., Yi, X.: DNN-based prediction model for spatio-temporal data. In: *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 1–4 (2016)
14. Geng, X., et al.: Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 3656–3663 (2019)
15. Zhao, L., et al.: T-GCN: a temporal graph convolutional network for traffic prediction. *IEEE Trans. Intell. Transp. Syst.* **21**(9), 3848–3858 (2019)
16. 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)
17. Niepert, M., Ahmed, M., Kutzkov, K.: Learning convolutional neural networks for graphs. In: *International Conference on Machine Learning*, pp. 2014–2023. PMLR (2016)
18. Li, B., Li, X., Zhang, Z., Fei, W.: Spatio-temporal graph routing for skeleton-based action recognition. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 8561–8568 (2019)
19. Bruna, J., Zaremba, W., Szlam, A., LeCun, Y.: Spectral networks and locally connected networks on graphs. In: Bengio, Y., LeCun, Y. (eds.) *2nd International Conference on Learning Representations, ICLR 2014, Conference Track Proceedings*, Banff, AB, Canada, 14–16 April 2014 (2014)
20. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: Lee, D.D., Sugiyama, M., von Luxburg, U., Guyon, I., Garnett, R. (eds.) *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016*, 5–10 December 2016, Barcelona, Spain, pp. 3837–3845 (2016)
21. Yu, B., Yin, H., Zhu, Z.: Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. In: Lang, J. (ed.) *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018*, Stockholm, Sweden, 13–19 July 2018, pp. 3634–3640. ijcai.org (2018)
22. Zhang, M., Chen, Y.: Link prediction based on graph neural networks. In: Bengio, S., Wallach, H.M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018*, Montréal, Canada 3–8 December 2018, pp. 5171–5181 (2018)
23. Zhou, X., Li, J., Wang, Z., He, R., Tan, T.: Image inpainting with contrastive relation network. In: *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 4420–4427. IEEE (2021)
24. Benesty, J., Chen, J., Huang, Y., Cohen, I.: Pearson correlation coefficient. In: *Noise Reduction in Speech Processing*, pp. 1–4. Springer, Cham (2009). [https://doi.org/10.1007/978-3-642-00296-0\\_5](https://doi.org/10.1007/978-3-642-00296-0_5)



25. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, ICLR 2017, Conference Track Proceedings, Toulon, France, 24–26 April 2017. OpenReview.net (2017)
26. Zhang, J., Zheng, Y., Qi, D.: Deep spatio-temporal residual networks for city-wide crowd flows prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31 (2017)
27. Shi, X., Chen, Z., Wang, H., Yeung, D., Wong, W., Woo, W.: Convolutional LSTM network: a machine learning approach for precipitation nowcasting. In: Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, Montreal, Quebec, Canada, 7–12 December 2015, pp. 802–810 (2015)
28. 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)
29. Bai, L., Yao, L., Kanhere, S.S., Wang, X., Sheng, Q.Z.: STG2Seq: spatial-temporal graph to sequence model for multi-step passenger demand forecasting. In: Kraus, S. (ed.) Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, 10–16 August 2019, pp. 1981–1987. ijcai.org (2019)
30. Aksoy, S., Haralick, R.M.: Feature normalization and likelihood-based similarity measures for image retrieval. *Pattern Recogn. Lett.* **22**(5), 563–582 (2001)



# Event Detection in Social Media via Graph Neural Network

Wang Gao<sup>1(✉)</sup>, Yuan Fang<sup>2</sup>, Lin Li<sup>2(✉)</sup>, and Xiaohui Tao<sup>3</sup>

<sup>1</sup> School of Artificial Intelligence, Jiangnan University, Wuhan, China  
gaow@jhun.edu.cn

<sup>2</sup> School of Computer Science and Technology, Wuhan University of Technology,  
Wuhan, China  
cathylilin@whut.edu.cn

<sup>3</sup> School of Sciences, University of Southern Queensland, Toowoomba, Australia

**Abstract.** Recently, Graph Neural Networks (GNN) have been applied to many natural language processing tasks. However, few studies exploit GNN for event detection, especially event detection in social media. In this paper, we proposed a new Event Detection model based on GNN (EDGNN). EDGNN first utilizes a topic model to capture the topical information of the corpus, which is used to help the graph enrich the semantics of short texts. Then, a text-level graph with fewer edges and memory consumption is constructed for each input short text. Furthermore, we incorporate word embeddings trained by Bidirectional Encoder Representations from Transformers (BERT) into EDGNN, which greatly improves the performance of the proposed method. Experimental results on a real-world foodborne disease event dataset demonstrate our model outperforms state-of-the-art baselines.

**Keywords:** Event detection · Graph neural network · Topic model · BERT

## 1 Introduction

Social media platforms such as Facebook and Twitter contain massive and diverse user-generated data, ranging from daily chores to the latest international news. Such a huge amount of data allows many issues to be studied, such as social network analysis and event detection [3, 10, 13, 32]. The main advantage of using social media for event detection is that due to their real-time nature, the public can learn more about what is happening in the real world faster than traditional media. This advantage has aroused great interest in event detection research from the Artificial Intelligence (AI) and Natural Language Processing (NLP) communities.

Events on social media streams can be generally defined as real-world occurrences that take place in similar geographical locations and time periods [9]. For instance, users often share information on social media platforms, such as “I

vomited after eating a hamburger in a fast-food restaurant” or “I got flu and fever last week”. Massive amounts of such textual information could point the way for public health event tracking or influenza surveillance. Wang *et al.* proposed a novel partial differential equation model to predict the evolution trend of influenza using Twitter streaming data [31]. Cui *et al.* proposed an algorithm to detect foodborne disease events from social media, and used the information to improve a restaurant recommendation task [6].

Event detection methods usually employ supervised and unsupervised deep learning approaches to determine whether a post belongs to a certain event. Recently, Graph Neural Networks (GNN) has emerged as a promising deep learning technique that can capture the rich relational structure of graphs and provide significant improvements for a variety of NLP tasks [11, 27, 33, 34]. Defferrard *et al.* applied a Graph Convolutional Neural Network (GCN) to a document classification task for the first time [7]. Experimental results show that their method is superior to Convolution Neural Networks (CNN). Yao *et al.* proposed a novel GCN model for text classification, which learns text graphs based on document-word relationships and word co-occurrence [33].

However, the above GNN-based models usually take the approach of constructing a graph for the entire collection. Because of the large number of edges in the graph, high memory consumption is needed. In addition, since the structure and parameters of the graph depend on the corpus and cannot be changed after training, it is difficult for these methods to support online testing. Furthermore, experimental results show that these models are less effective for short texts that are the main form of information on social media platforms [33]. This may be due to the short length and sparsity, making it difficult for GNN-based models to exploit a large window to obtain high-quality global representations.

To tackle the above problems, we propose a novel GNN-based model for event detection in social media. The main idea comes from the answers to the following two questions: (1) How to build a GNN-based model for event detection consumes less memory and can be tested online? (2) How to alleviate the sparsity of short texts to improve the performance of event detection methods on social media platforms?

Specifically, we design a new Event Detection model based on GNN (EDGNN). Unlike generating a graph for the entire corpus, EDGNN constructs a text-level graph independently for each short text on social media platforms. Therefore, instead of connecting all word nodes in the whole corpus, we only connect word nodes in a fairly small window, which in turn consumes less memory. To alleviate the sparsity problem, our model employs probabilistic topic models to capture topic information of each short text, which is used to help the construction of text-level graphs. In EDGNN, the representations of word nodes and topic nodes are globally shared, and can be updated using a message passing process, that is, each node updates its representation by receiving information from its neighboring nodes. Finally, we input node representations learned by the proposed model into a Bidirectional Gated Recurrent Unit (BiGRU) to detect events in social media. Furthermore, we utilize word embeddings trained by Bidirec-

tional Encoder Representations from Transformers (BERT) [8] to further enrich the semantics of short texts. To measure the performance of EDGNN, we conducted extensive experiments on a real-world foodborne disease event dataset. Experimental results show that the proposed model achieves better performance than state-of-the-art baselines.

The main contributions of this work are summarized as follows:

1. We propose a novel Event Detection model based on GNN (EDGNN) to extract events in social media. Instead of building a single graph for the entire corpus, EDGNN constructs graphs for each short text with global shared representations. The proposed model not only exceedingly reduces the number of edges in the graph and memory consumption, but also captures more local relations between words and topics.
2. EDGNN first employs Conditional Random Field regularized Topic Model (CRFTM) [14] to extract the topic information of short texts. Then, the topic information is used to build text-level graphs to solve the sparsity of short texts. To the best of our knowledge, this is the first work to incorporate topic information based on CRFTM into GNN.
3. The performance of EDGNN is evaluated on a foodborne disease event dataset and compared with classic and state-of-the-art baseline methods. Experimental results illustrate EDGNN can effectively detect events in social media, and outperforms existing models on precision, recall and F1-measure.

## 2 Related Work

In this section, we briefly summarize the related work about event detection in social media and short text classification.

### 2.1 Event Detection

Since social media data contains a large number of real-time events, event detection in social media has always been an important research topic. Cheng *et al.* proposed a Space-Time Scan Statistics (STSS) method for event detection from Twitter data stream [3]. Instead of using tweet content, the approach extracts clusters across time and space from the dataset. Kaleel *et al.* proposed a new approach that can be used to detect events from tweet clusters aggregated using Locality Sensitive Hashing (LSH) [20]. Hua *et al.* proposed a semi-supervised method for identifying spatio-temporal events from social media, which can detect locations from tweets using a multinomial spatial scanning approach [18]. Wang *et al.* proposed a framework that collects and analyzes social media data by using geographic location information, and uses Twitter data to provide information to a traffic alarm system [30]. Their framework also provides a comprehensive analysis of traffic-related tweets along time and space to identify when and where frequent accidents occur.

However, when the above methods are used for event detection in social media, they can only be applied to specific domains and the accuracy is not

satisfactory. On the contrary, the proposed approach can be generalized to deal with short texts in different domains.

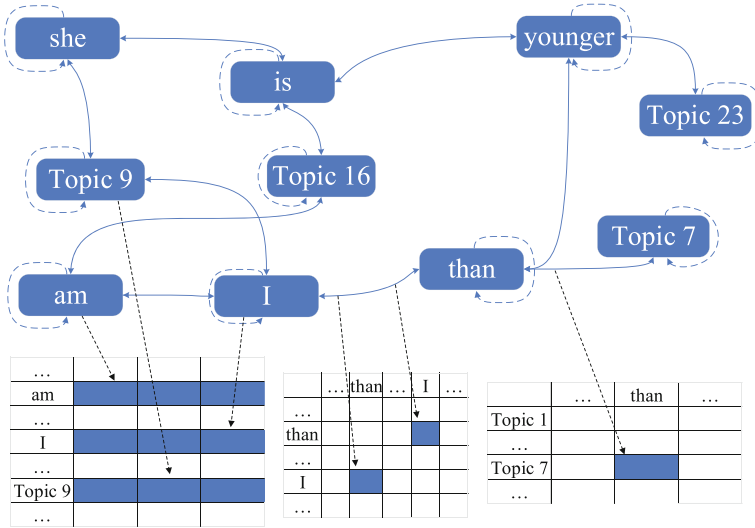
## 2.2 Short Text Classification

As the methods based on deep learning are more intuitive and integrated [16], an increasing number of researchers are applying neural networks to short text classification. Kim proposed a CNN to classify short texts with a layer of convolution over word embeddings [21]. Additionally, they modify the CNN architecture to allow both task-specific and static vectors to be used. Liu *et al.* proposed a new information-sharing mechanism based on Recurrent Neural Network (RNN) to model textual content with shared and task-specific layers [24]. Experimental results on text classification show that their method significantly improves the classification performance. Liu *et al.* proposed a novel model that includes bidirectional Long Short-Term Memory (LSTM), CNN and an attention mechanism for text classification [23]. The model utilizes the attention mechanism to focus differently on the output information of hidden layers.

GNN is able to model non-Euclidean text data and has received widespread attention in recent years [17, 28]. Based on spectral graph theory, Defferrard *et al.* proposed a new form of CNN, which provides numerical methods and mathematical foundations for designing convolution filters on graphs [7]. Experiment results show that their model can utilize graph convolution layers to extract stationary and local features. Kipf *et al.* proposed an extensible model for semi-supervised classification on graph-structured datasets, which is based on a variant of CNN operating on graphs [22]. Their method employs a hierarchical propagation mechanism that allows hidden layer representations to encode node features and the local structure of graphs. Yao *et al.* proposed a new Text Graph Convolutional Network (Text GCN) for document classification, which simultaneously learns the representation of words and documents [33]. However, unlike previous studies, our model uses topic information to alleviate the sparsity problem in short text classification, and builds a graph for each short text.

## 3 Methodology

The proposed method divides the process of event detection in social media into three steps. In the first step, CRFTM is trained on the entire short text dataset, and extracts the topic labels to which each word in each short text belongs. The second step is to construct a text-level graph for each short text, and all parameters of the graph are provided by global sharing matrices. The overall architecture of a text-level graph is illustrated in Fig. 1. Finally, the word and topic representations generated by the graph and BERT are used as input to a BiGRU model for event detection.



**Fig. 1.** The overall architecture of a text-level graph for a single text “she is younger than I am”. For ease of display, in this figure, we set the size of the sliding window to 2. As shown at the bottom of the figure, all parameters of the text-level graph are provided by global sharing matrices.

### 3.1 Text-Level Graph Construction

Let  $S = \{w_1, \dots, w_i, \dots, w_{N_s}\}$  be a short text, where  $N_s$  is the number of words in  $S$  and  $w_i$  is the representation of the  $i_{th}$  word.  $T = \{t_1, \dots, t_i, \dots, t_{N_s}\}$  represents the topic assignments of  $S$ , where  $t_i$  denotes the representation of the  $i_{th}$  topic. Since multiple words in  $S$  may belong to the same topic, there are duplicate topics in  $T$ .  $w_i$  and  $t_i$  are vectors that can be updated during the training process. For a given short text  $S$ , EDGNN treats all words and corresponding topics that occur in  $S$  as nodes of a text-level graph  $\mathcal{G}$ . Each edge between word nodes is constructed based on the word co-occurrence in short texts (*i.e.*, words that appear in a fixed-size sliding window). Another type of edge starts from a word node and ends with its corresponding topic node. The text-level graph  $\mathcal{G}$  of  $S$  is defined as:

$$\begin{aligned} \mathcal{V} &= \{w_i, t_i | i \in [1, N_s]\} \\ \mathcal{E} &= \{e_{w_i, w_j}, e_{w_i, t_i} | i \in [1, N_s], \\ &\quad j \in [i - c + 1, i + c - 1]\}, \end{aligned} \tag{1}$$

where  $\mathcal{V}$  and  $\mathcal{E}$  denote the set of vertices and edges of  $\mathcal{G}$  respectively,  $c$  is the window size. Following [33], the weights between word nodes are initialized using Point-wise Mutual Information (PMI), which is a common way to measure word relevance.

Unlike the previous graph-based methods of building a graph on the whole corpus, the proposed model is able to greatly reduce the nodes and edges of the

graph. Therefore, EDGNN consumes less memory resources and can be applied to larger-scale social media corpora. In addition, these models cannot handle newly emerging short texts. However, our method is capable of solving the problem because the graph of each text is independent and depends only on its own content.

After constructing the text-level graph, we use GCN to train the proposed model. In GCN, convolution can be achieved by either spectral or non-spectral methods. In this work, EDGNN adopts a non-spectral approach proposed by [15] for convolution. Firstly, the message passing process of the approach receives information from each neighboring node. Secondly, the representation of each node is updated according to the original representation and received information. The process is defined as:

$$\begin{aligned} I_n &= \max_{j \in \mathcal{N}_n} e_{nj} r_j \\ r'_n &= (1 - \lambda_n) r_n + \lambda_n I_n, \end{aligned} \quad (2)$$

where  $I_n \in \mathbb{R}^d$  denotes the information that node  $n$  collects from its adjacent nodes,  $\mathcal{N}_n$  denotes all adjacent nodes of  $n$ , and  $max$  represents a reduction function that merges maximum values in each dimension to create a new embedding.  $e_{nj} \in \mathbb{R}$  denotes the edge weight between node  $n$  and  $j$ , which can be initialized with PMI or the topic-word matrix and updated during training.  $r_n \in \mathbb{R}^d$  and  $r'_n$  are the original and updated vector representation of  $n$ , respectively. Trainable trade-off parameter  $\lambda_n$  controls the amount of information that should be retained in  $r_n$ .

The message passing process makes the update of a node representation affected by its neighbors, which enables the representation to gather information from the context. As a result, even for polysemous and homonymous words, the proposed method can capture their accurate meanings by the information from their adjacent words and topics. Furthermore, since all parameters in the graph are obtained from shared matrices, EDGNN introduces global information during the training process, which is similar to the previous graph-based models.

The label of the short text is predicted by feeding the representations of all nodes into a softmax layer:

$$\begin{aligned} O_G &= \sum_{n \in \mathcal{V}} r'_n \\ \hat{y}_G &= \text{softmax}(\text{Relu}(\mathbf{W}O_G + \mathbf{b})), \end{aligned} \quad (3)$$

where  $O_G$  is the output of the graph,  $\mathbf{W}$  and  $\mathbf{b}$  represent weights and bias.

The purpose of training is to minimize the loss function, which is given by:

$$\mathcal{L} = - \sum_i y_{\mathcal{G}_i} \log(\hat{y}_{\mathcal{G}_i}), \quad (4)$$

where  $y_{\mathcal{G}_i}$  denotes the  $i_{th}$  element of the one-hot ground truth label.

### 3.2 Topic Extraction

Topic models such as Latent Dirichlet Allocation (LDA) [1] have been widely used to extract hidden topics from text collections. However, conventional topic modeling algorithms have achieved great success on length documents, but they do not work well on short texts [13, 19]. The main reason is that traditional topic models discover hidden topics by capturing text-level word co-occurrence information, which encounters the data sparsity problem of short texts. To tackle the sparsity problem, we utilize CRFTM to reveal the topic assignments of short texts. CRFTM first integrates short texts into long pseudo-documents, and then makes semantically related words share the same topic labels.

Specifically, CRFTM first leverages Embedding-based Minimum Average Distance (EMAD) to aggregate short texts into regular-sized pseudo-documents [14]. EMAD can discover semantically correlated word pairs in two different short texts that may belong to the same hidden topic. Secondly, for each topic  $k$ , CRFTM draws a word proportion  $\phi_k \sim Dir(\beta)$  and a topic proportion  $\theta \sim Dir(\alpha)$  for the entire corpus, where  $\alpha$  and  $\beta$  represent Dirichlet priors. For each pseudo-document  $m$ , CRFTM draws observed word  $w_{mi} \sim Mult(\phi_{z_{mi}})$  and topic assignments  $\mathbf{z}_m$  can be defined as follows:

$$p(\mathbf{z}_m | \theta_m, \mathbf{x}_m) = \prod_{i=1}^{N_m} p(z_{mi} | \theta_m) \Psi(z_{mi}, x_{mi}), \quad (5)$$

where  $N_m$  is the number of words in  $m$ ,  $x_{mi}$  denotes the contextual words of the  $i_{th}$  word, and  $\Psi$  represents a potential function.

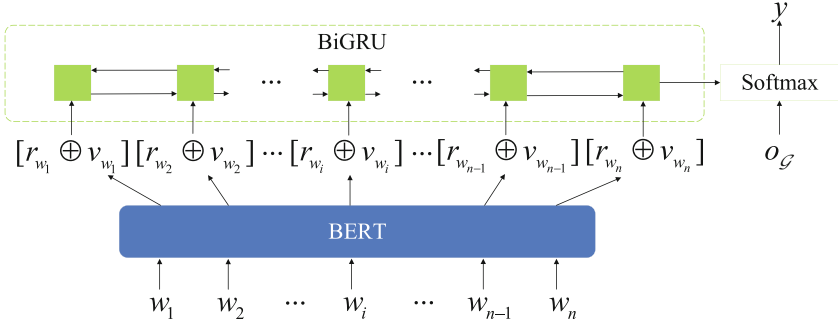
Finally, CRFTM generates a topic-word matrix, which represents the distribution of words in each topic and the topic assignment of each word in the whole collection. In the proposed model, there is an edge connecting each word node and its corresponding topic node, and the initial weight of the edge is obtained from the topic-word matrix.

### 3.3 Event Detection

To further enhance the performance of event detection, we integrate the word embeddings generated by BERT into the proposed method. As a state-of-the-art pre-trained language model, BERT has made great progress in tasks such as text classification and automatic text summarization [2, 26, 29, 35]. In this paper, although the BERT model is used for event detection in social media, the output of BERT is not the final event detection result. The vector representation  $v$  learned by the hidden layer of BERT is capable of representing context-dependent word embeddings [8]. Compared with using text-level graphs or BERT alone, EDGNN combines the representations generated by the text-level graph with the word embeddings learned by BERT to achieve better event detection performance.

Finally, since BiGRU performs better than other classifiers such as CNN, we leverage BiGRU as a classification model to detect events [5]. Figure 2





**Fig. 2.** The overview of event detection using BERT and BiGRU.

illustrates the process of event detection using BERT and BiGRU. Let  $V_g = \{r_{w_1}, \dots, r_{w_i}, \dots, r_{w_n}\}$  be the word node vectors generated by the text-level graph, and  $V_b = \{v_{w_1}, \dots, v_{w_i}, \dots, v_{w_n}\}$  denotes word vectors generated by BERT. The proposed model concatenates  $V_g$  and  $V_b$  to form new word representations, which are used as BiGRU input. Next, as shown in Fig. 2, we feed the BiGRU output representation and  $O_G$  to a softmax layer to detect whether a short text belongs to a specific event. Similar to the training of the graph, EDGNN utilizes cross entropy loss to train BiGRU.

## 4 Experiments

In this section, we conduct experiments to evaluate the performance of EDGNN by comparing it with state-of-the-art baseline models. The performance in terms of precision, recall and F1-measure is reported over a real-world foodborne disease event dataset. The experimental results illustrate the effectiveness of our method.

### 4.1 Datasets

To collect posts related to foodborne disease events on social media, we first derived a series of keywords indicating symptoms of foodborne diseases from the relevant reports of the World Health Organization (WHO). Since these reports are written by experts, there are many medical terms in the extracted keyword list. Nevertheless, when people post their information on social media platforms, they prefer to use colloquial expressions. To solve the challenge, we ask some outpatient doctors to convert medical terms in the keyword list into spoken words.

After that, we crawl about 110,000 posts containing keywords for foodborne illness symptoms on Weibo that is the largest social media platform in China. Although these short texts contain lots of useful data, they also contain a large amount of noise information. The collected short texts are likely to come from

accounts that provide professional health advice. For instance, the keyword list includes “vomiting” and “nausea”, but these accounts may post tweets such as “how to relieve nausea after drinking” and “how to avoid vomiting during pregnancy”. In addition, while some short texts contain keywords in the list, they do not describe foodborne illness events that occur in real-time. A user may post a tweet containing keywords like “I vomited last Friday”, but it does not meet real-time requirements. These tweets are obviously noise information for detecting foodborne disease events on social media. Therefore, we invited five graduate students in the field of public health to manually label the dataset and asked them to provide a tag for whether a given tweet is a foodborne disease case. In the end, we select 8563 labeled short texts, including 5146 non-event tweets and 3417 event-related tweets.

## 4.2 Baseline Methods

Different classification models for short texts are employed in the experiment, and their performance is compared by several evaluation metrics. The baseline methods used in this paper are as follows.

- **CNN** is a deep neural network that automatically learns features by using multiple modules such as convolutional layers and pool layers [21].
- **LSTM** is a modified version of RNN, which is mainly used for sequence prediction and text classification tasks, and has the ability of learning long-term dependence [24].
- **Text GCN** creates a single heterogeneous graph for the entire corpus, and exploits GCN to learn the embeddings of words and documents, which is one of the state-of-the-art text classification models [33].
- **BERT** is a bidirectional multi-layer model based on a transformer architecture, which replaces RNN and CNN with a faster self-attention-based approach [8].
- **EDGNN-B** is a variant of EDGNN. After learning the node representations of a short text, we directly input them into BiGRU to classify whether the tweet is a foodborne disease event.
- **EDGNN-B-G** is based on EDGNN, but it only uses text-level graphs to classify short texts. Specifically, EDGNN-B-G feeds the output of the graph to a softmax layer to obtain short text labels.

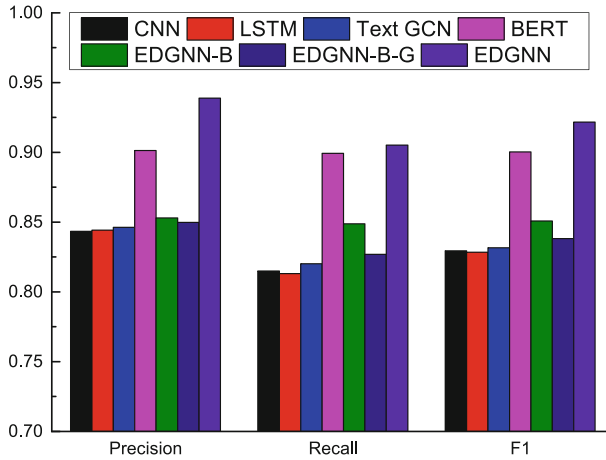
## 4.3 Model Settings

In the text-level graph, we initialize graph node representations with random vectors and set the dimension to 300. The window size  $c$  is set to 3, and the number of topics is set to 30. In the process of graph training, we utilize the adam optimizer, and the dropout rate and the initial learning rate are set to 0.5 and 0.01 respectively. For baseline models, 300-dimensional freely-available word2vec word embeddings<sup>1</sup> are adapted, and we employ default parameter settings based

<sup>1</sup> <https://github.com/Embedding/Chinese-Word-Vectors>.

on their original papers. We randomly divide the dataset into a training set and a test set according to the ratio of 7:3, and randomly select 10% of training data as a validation set. The training process stops when the loss of the validation set is not reduced for 10 consecutive epochs. In the BERT model, we use BERT-base(Chinese)<sup>2</sup> with the hidden size of 768, and the number of headers and layers of 12. Since the topic inference process is a stochastic process, all results reported in the experiment are the average of five runs.

#### 4.4 Classification Results



**Fig. 3.** Classification results of seven models.

Figure 3 shows the classification results of the proposed model against six baseline models. As shown in the figure, compared with baseline models, EDGNN achieves significant improvements in precision, recall and F1-measure.

One observation is that the classification results of graph-based methods are superior to conventional methods such as CNN and LSTM. This may be caused by the structural characteristics of GNN. The number of neighbor nodes in a graph network structure is variable, which can better represent word nodes through a message passing mechanism. Additionally, the weight of edges defines the relationship between words and can be shared globally.

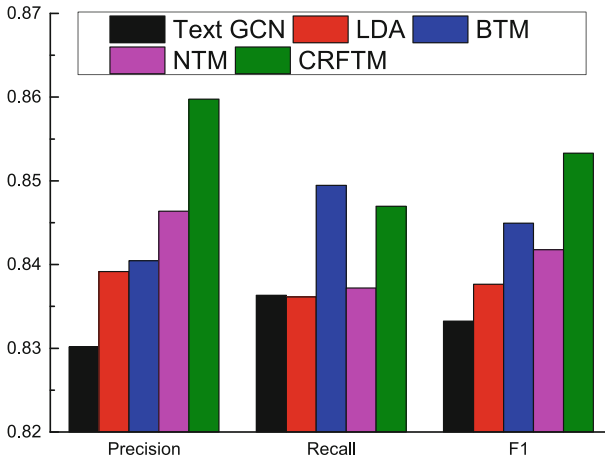
Another observation is that the proposed model achieves better performance than the state-of-the-art graph-based model Text GCN. The reason may be that EDGNN introduces topic information to enrich the semantics of short texts, which alleviates the sparsity problem. Similar to traditional word embedding methods, EDGNN uses the co-occurrence information in a contextual window

<sup>2</sup> <https://github.com/google-research/bert>.

to learn word representations, while Text GCN is trained on the whole corpus. As a result, the proposed model can learn more accurate node representations and achieve better performance.

We also note that EDGNN-B performs better than EDGNN-B-R. This may be because EDGNN-B inputs word node representations and the output of the graph into BiGRU, taking full advantage of all the node information in the text-level graph. The proposed model achieves the best performance because BERT can help EDGNN learn long-range dependencies of word sequences, as discussed in [12]. The global relationship learned by BERT and local features extracted from GNN complement each other, leading to its better results.

#### 4.5 Analysis of Topic Models

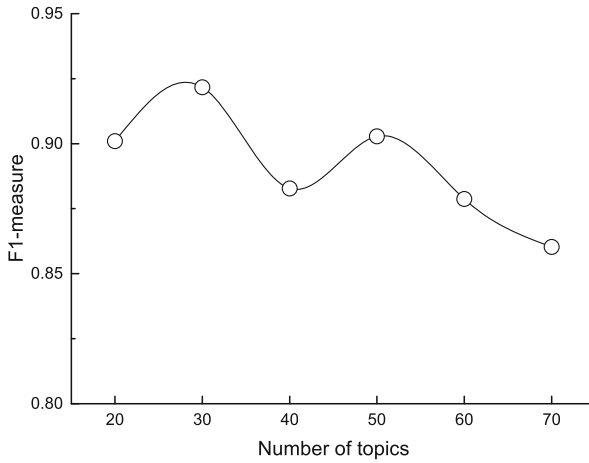


**Fig. 4.** Classification results of different topic models.

In the experiment, we compare four topic models to discover hidden topics from tweets and their impact on event detection. LDA [1], the most widely used topic model, is selected to extract the topical information of tweets, which is used as an extended feature. With the development of deep learning, there are an increasing number of neural topic models, which utilize neural networks to approximate the distribution between words and topics. This kind of topic model not only learns the vector representations of topics, but also can be directly embedded into deep neural networks and trained together. A Neural Topic Model (NTM) proposed by [25] is employed in the experiment, which combines the advantages of both probabilistic topic models and neural networks. Furthermore, a Biterm Topic Model (BTM) extracts topics by directly modeling word co-occurrence information (*i.e.*, biterm) in the corpus, which has proven to be effective for short texts [4].

Figure 4 illustrates the results of Text GCN, EDGNN-B-G with LDA, EDGNN-B-G with NTM, EDGNN-B-G with BTM and EDGNN-B-G with CRFTM. We note that NTM performs better than LDA, indicating the effectiveness of using neural networks to extract latent topics. Furthermore, BTM achieves better results than NTM. The reason may be that the topic discovery mechanism of BTM is more suitable for short texts. CRFTM achieves the best performance, indicating that CRFTM promotes related words to share the same topic, which helps to discover high-quality topics.

#### 4.6 Analysis of the Number of Topics



**Fig. 5.** Experimental results with different settings on number of topics.

In this subsection, we study the impact of the number of topics in EDGNN. A small number of topics usually makes the content of generated topics very broad, since most of these topics contain common words.

Figure 5 shows the F1-measure of EDGNN with the number of topics ranging from 20 to 70. As shown in Fig. 5, we observe that the best performance is obtained when the number of topics is set to 30. However, when the number of topics is further increased, the performance of EDGNN decreases. This is reasonable because an excessive number of topics leads to the topics extracted by the model to be fine-grained and overfit to the corpus.

## 5 Conclusion

This paper focuses on how to use GNN to identify posts related to specific events from social media. The main challenge comes from the high memory consumption

of existing methods, and the sparsity of short texts, which makes them less effective. To tackle the above challenge, we propose a new Event Detection model based on GNN (EDGNN), which divides the event detection process into three steps. Firstly, EDGNN uses a topic model to capture the topic label of each word in each short text. This topical information is helpful to enrich the semantics of short texts. Secondly, the model constructs a text-level graph for each short text, and all parameters of the graph are obtained from global shared matrices. Finally, the output of the graph and the word representations trained by BERT are input to a classifier for event detection.

Experiments on a real-world dataset validate the effectiveness of EDGNN. Furthermore, the proposed method does not take advantage of the unique features of the dataset, which means that our model can extract event-independent features that can be shared between different events. Therefore, EDGNN can be generalized to detect events in different domains on social media platforms. Given the training corpus of different events, our model has the ability to detect different events.

**Acknowledgements.** This work is supported by National Science Foundation of China (NSFC, No. 62106086), Scientific Research Program of Hubei Provincial Department of Education and Doctoral Start-up Fund of Jiangnan University (No. 1028/06060001).

## References

1. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. *J. Mach. Learn. Res. (JMLR)* **3**, 993–1022 (2003)
2. Cai, L., Song, Y., Liu, T., Zhang, K.: A hybrid BERT model that incorporates label semantics via adjustive attention for multi-label text classification. *IEEE Access* **8**, 152183–152192 (2020)
3. Cheng, T., Wicks, T.: Event detection using Twitter: a spatio-temporal approach. *PLoS ONE* **9**(6), 1–10 (2014)
4. Cheng, X., Yan, X., Lan, Y., Guo, J.: BTM: topic modeling over short texts. *IEEE Trans. Knowl. Data Eng. (TKDE)* **26**(12), 2928–2941 (2014)
5. Cho, K., Merriënboer, B.V., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: encoder-decoder approaches. In: *Proceedings of Workshop on Syntax, Semantics and Structure in Statistical Translation (SSST)*, pp. 103–111 (2014)
6. Cui, W., et al.: An algorithm for event detection based on social media data. *Neurocomputing* **254**, 53–58 (2017)
7. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, pp. 3837–3845 (2016)
8. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: *Proceedings of North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 4171–4186 (2019)

9. Dong, X., Mavroudis, D., Calabrese, F., Frossard, P.: Multiscale event detection in social media. *Data Min. Knowl. Discov.* **29**(5), 1374–1405 (2015). <https://doi.org/10.1007/s10618-015-0421-2>
10. Du, J., Michalska, S., Subramani, S., Wang, H., Zhang, Y.: Neural attention with character embeddings for hay fever detection from twitter. *Health Inf. Sci. Syst.* **7**(1), 1–7 (2019). <https://doi.org/10.1007/s13755-019-0084-2>
11. Gao, W., Fang, Y., Zhang, F., Yang, Z.: Representation learning of knowledge graphs using convolutional neural networks. *Neural Netw. World (NNW)* **30**(2), 145–160 (2020)
12. Gao, W., Li, L., Zhu, X., Wang, Y.: Detecting disaster-related tweets via multi-modal adversarial neural network. *IEEE Multimedia* **27**(4), 28–37 (2020)
13. Gao, W., et al.: Generation of topic evolution graphs from short text streams. *Neurocomputing* **383**, 282–294 (2020)
14. Gao, W., Peng, M., Wang, H., Zhang, Y., Xie, Q., Tian, G.: Incorporating word embeddings into topic modeling of short text. *Knowl. Inf. Syst.* **61**(2), 1123–1145 (2018). <https://doi.org/10.1007/s10115-018-1314-7>
15. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: *Proceedings of International Conference on Machine Learning (ICML)*, pp. 1263–1272 (2017)
16. He, J., Rong, J., Sun, L., Wang, H., Zhang, Y., Ma, J.: A framework for cardiac arrhythmia detection from IoT-based ECGs. *World Wide Web* **23**(5), 2835–2850 (2020)
17. Hu, Y., Zhang, Z., Yao, Y., Huan, X., Zhou, X., Lee, W.S.: A bidirectional graph neural network for traveling salesman problems on arbitrary symmetric graphs. *Eng. Appl. Artif. Intell.* **97**, 1–9 (2021)
18. Hua, T., Chen, F., Zhao, L., Lu, C.-T., Ramakrishnan, N.: Automatic targeted-domain spatiotemporal event detection in twitter. *GeoInformatica* **20**(4), 765–795 (2016). <https://doi.org/10.1007/s10707-016-0263-0>
19. Jiang, H., Zhou, R., Zhang, L., Wang, H., Zhang, Y.: Sentence level topic models for associated topics extraction. *World Wide Web* **22**(6), 2545–2560 (2018). <https://doi.org/10.1007/s11280-018-0639-1>
20. Kaleel, S.B., Abhari, A.: Cluster-discovery of twitter messages for event detection and trending. *J. Comput. Sci. (JOCS)* **6**, 47–57 (2015)
21. Kim, Y.: Convolutional neural networks for sentence classification. In: *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1746–1751 (2014)
22. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: *Proceedings of International Conference on Learning Representations (ICLR)*, pp. 1–14 (2017)
23. Liu, G., Guo, J.: Bidirectional LSTM with attention mechanism and convolutional layer for text classification. *Neurocomputing* **337**, 325–338 (2019)
24. Liu, P., Qiu, X., Huang, X.: Recurrent neural network for text classification with multi-task learning. In: *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2873–2879 (2016)
25. Miao, Y., Grefenstette, E., Blunsom, P.: Discovering discrete latent topics with neural variational inference. In: *Proceedings of International Conference on Machine Learning (ICML)*, pp. 2410–2419 (2017)
26. Shen, S., et al.: Q-BERT: hessian based ultra low precision quantization of BERT. In: *Proceedings of Conference on Artificial Intelligence (AAAI)*, pp. 8815–8821 (2020)

27. Shen, Y., Li, H., Yi, S., Chen, D., Wang, X.: Person re-identification with deep similarity-guided graph neural network. In: Proceedings of European Conference on Computer Vision (ECCV), pp. 508–526 (2018)
28. Tang, H., Ji, D., Zhou, Q.: Triple-based graph neural network for encoding event units in graph reasoning problems. *Inf. Sci.* **544**, 168–182 (2021)
29. Tian, X., Wang, J.: Retrieval of scientific documents based on HFS and BERT. *IEEE Access* **9**, 8708–8717 (2021)
30. Wang, D., Al-Rubaie, A., Clarke, S.S., Davies, J.: Real-time traffic event detection from social media. *ACM Trans. Internet Technol. (TOIT)* **18**(1), 1–23 (2017)
31. Wang, Y., Xu, K., Kang, Y., Wang, H., Wang, F., Avram, A.: Regional influenza prediction with sampling twitter data and PDE model. *Int. J. Environ. Res. Public Health* **17**(3), 678–680 (2020)
32. Wu, Z., Yin, W., Cao, J., Xu, G., Cuzzocrea, A.: Community detection in multi-relational social networks. In: Lin, X., Manolopoulos, Y., Srivastava, D., Huang, G. (eds.) WISE 2013. LNCS, vol. 8181, pp. 43–56. Springer, Heidelberg (2013). [https://doi.org/10.1007/978-3-642-41154-0\\_4](https://doi.org/10.1007/978-3-642-41154-0_4)
33. Yao, L., Mao, C., Luo, Y.: Graph convolutional networks for text classification. In: Proceedings of Conference on Artificial Intelligence (AAAI), pp. 7370–7377 (2019)
34. Zhang, M., Chen, Y.: Link prediction based on graph neural networks. In: Proceedings of Advances in Neural Information Processing Systems (NeurIPS), pp. 5171–5181 (2018)
35. Zhou, W., Ge, T., Xu, K., Wei, F., Zhou, M.: Bert-based lexical substitution. In: Proceedings of Annual Meeting of the Association for Computational Linguistics (ACL), pp. 3368–3373 (2019)





# Knowledge-Guided Fraud Detection Using Semi-supervised Graph Neural Network

Yizhuo Rao<sup>1</sup>, Xiaoguang Ren<sup>2(✉)</sup>, Chengyuan Duan<sup>1(✉)</sup>, Xianya Mi<sup>2</sup>,  
Jiajun Cheng<sup>1</sup>, Yu Chen<sup>1</sup>, Hongliang You<sup>1</sup>, Qiang Gao<sup>1</sup>, Zhixian Zeng<sup>3</sup>,  
and Xiao Wei<sup>1</sup>

- <sup>1</sup> Military Science Information Research Center, Academy of Military Sciences,  
Beijing, China
- <sup>2</sup> Artificial Intelligence Research Center, National Innovation Institute of Defense  
Technology, Beijing, China
- <sup>3</sup> The Sixty-third Research Institute, National University of Defense Technology,  
Nanjing, China

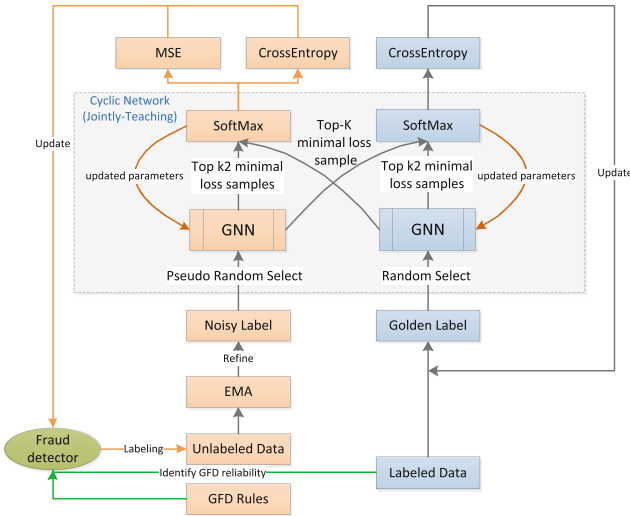
**Abstract.** Fraud detection is about finding unusual behaviors in the data, and it is essential for companies to detect fraudulent users to prevent unpredictable risks. The graph-based approaches model relationships into graphs for capturing the intricate characteristics of complex scenarios to detect fraudsters. However, it still faces the problem of data skew where labeled fraudsters are far fewer than unlabeled examples. Knowledge may help identify these unlabeled data, thus this paper combines domain knowledge with GNN and proposes a **Knowledge-Guided Semi-supervised Graph Neural Network**, namely *KS-GNN*, to address the problem of data skew. We utilize domain experts to design small amount of rules to roughly label unlabeled data as noisy and use a semi supervised method to train fraud detectors. By utilizing only 13 GFD rules conducted by domain experts, the performance of our method yields about 15% improvement over the state-of-the-art fraud detection methods CARE-GNN on banking transaction funds supervision datasets (BTFSD). Moreover, with some modification of the GFD rules on BTFSD, the performance of KS-GNN on other domain datasets such as IEEE-CIS Fraud Detection (<https://www.kaggle.com/c/ieee-fraud-detection/data>) and Yelp-Chi is also improved by about 5% on average compared with the state-of-the-art methods.

**Keywords:** Fraud detection · Semi supervised learning · Graph neural network · Learning with noisy labels

## 1 Introduction

Fraud activities are becoming common under e-commerce scenarios and will seriously damage the rights and interests of users and companies. Therefore, it is crucial to identify fraudulent behaviors and take every precaution to minimize the risk.

Fraud detection can be formulated as a classification problem. Recently, researchers have started applying graph-based method to find out frauds. Fan et al. [2] proposed a semantic constraint on graph, graph functional dependency (GFD), to identify inconsistencies in graph data. Graph neural network (GNN) approaches have also been used to detect fraud behaviors automatically. Wang et al. [12] expand the labeled transaction data and utilize the multi-view graph for fraud detection. CARE-GNN [1] filter neighbors based on label-aware similarity measures with adaptive filtering thresholds to discover the camouflage behaviors of fraudsters. Different from these works, our proposed method concern neighbor similarity by applying expertise.



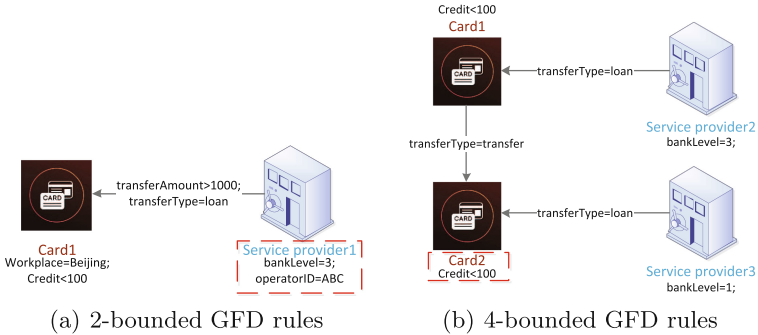
**Fig. 1.** Overall structure of KS-GNN. We use GFD rules to generate noisy labeled samples, and adopt a semi-supervised learning method, Jointly-teaching, to train a classifier with noisy labels with golden labels cooperatively. EMA is to pre-eliminate some unstable pseudo label predicted by GFD and network.

Graph has natural advantages to capture intricate characteristics such as multi-hop transactions (e.g. Fig. 2(b)). However, it is still facing the problem that there are not enough labeled data for learning. Human knowledge may alleviate this problem. By using expertise to roughly label unlabeled data, and train these noisy data and supervised information jointly, it can effectively avoid the problem of lacking of examples. Nevertheless, **how to train GNN with noisy label data and golden label data jointly** is still a great challenge. Since the auto-labeled noisy data are much more than the golden data, the model may be overfitting to noisy data, that is to say GNN only learns to fit artificial rules. It is difficult to train golden label data with these noisy data cooperatively.

To address the challenge above, we propose **Knowledge-Guided Semi supervised Graph Neural Network**, named *KS-GNN*. We adopt Graph Functional

Dependency (GFD) rules [2,3] to express expertise uniformly. *Reliability* is used to identify the noisy level of auto-labeled samples. Before each training epoch, KS-GNN will eliminate some noise data from this round by using Temporal Ensembling model [8]. We proposed a method to train noisy and golden data cooperatively, called *Jointly-Teaching*, which maintains two GNN modules independently. The minimal loss data of one network itself and its peer network are utilized to update parameters to avoid overfitting at each training batch. The overall structure is shown in Fig. 1. We summarize our key contributions as below:

- A novel knowledge guided semi-supervised GNN model, named KS-GNN, is proposed to utilize human knowledge to assist supervised GNN, which can obtain satisfactory result with few labeled data and simple expert knowledge.
- A learning with noisy label method, called *Jointly-Teaching*, is applied to GNN model train noisy knowledge and labeled data on graph comprehensively.
- To refine cleaner data continuously, we combine reliability and Temporal Ensembling model to reduce the impact caused by GFD marking errors.
- Experimental results on financial fraud dataset BTFSD and other real-world open datasets demonstrate that our approach achieves a substantial gain over state-of-the-art semi-supervised methods.



**Fig. 2.** An example of GFD rules of different boundary. The red dotted box represents the fraudulent nodes identified by this GFD rule.

## 2 Methodology

### 2.1 Preliminaries

**Graph Functional Dependency (GFD).** A graph functional dependency is  $Q[\bar{x}](X \rightarrow Y)$ , where  $Q[\bar{x}]$  is a graph pattern and  $X$  and  $Y$  are two sets of literals

of  $\bar{x}$  [3]. For instance, a GFD rule shown in Fig. 2 to identify an abnormal card is  $Q[\bar{x}](X \rightarrow Y)$ . Where  $Q[\bar{x}]$  is shown in Fig. 2,  $X = \{F_a(Card) = (Workplace = 'Beijing'; Credit < 100); F_a(e) = (transferAmount > 1000; transferType = 'loan')\}$ ,  $Y = \{F_a(SP) = (bankLevel = 3; operatorID = ABC; isFraud = True)\}$ . It denotes that a transaction which can match graph pattern  $Q[\bar{x}]$  and GFD constraint  $X$ , then Service Provider (SP) with  $bankLevel = 3$  and  $operatorID = ABC$  is fraud.

## 2.2 The Proposed Model: KS-GNN

**Labeling Unlabeled Data.** We introduce *Reliability* to indicate the applicability of GFD of current task. *Reliability RE* is an indicator to identify the noise level of an data and can be renewed by the noisy network  $f$ . *RE* is only valid for one epoch. The initial *RE* of GFD rules can be calculated as follows: use GFD rules to label golden data. Let  $P(\varphi_i)$  be the precision of each GFD  $\varphi_i$ . Let  $R(\varphi_i)$  be the recall of GFD label. Thus, the reliability of a GFD can be shown as:

$$RE(\varphi_i) = b + \frac{(1 - b) \times (2 \times P \times R)}{(P + R)} \quad (1)$$

where  $b \in [0, 1]$  is a fixed reliability boundary which is to confine the GFD rule into a certain range. Here we set  $b$  to 0.8, thus the range of *RE* is  $[0.8, 1]$ .

Since noisy label might be labeled by more than one GFD rules, their reliability are set the same with the smaller one to protect the knowledge contained in complex rules.

**Noisy Data Refinement.** We conduct preliminary screening out of noise data. Since the network' prediction is likely to be inconsistent on wrongly labeled data over different training iterations [9, 11], we record the outputs of noisy network made on different training epochs and use them to identify the noisy label prediction. After every training epoch, the noisy network's EMA prediction  $\bar{y}_i$  are accumulated into history outputs  $y_{emai}$  by the formula below,

$$y_{emai} \leftarrow \alpha y_{emai} + (1 - \alpha) \bar{y}_i \quad (2)$$

where  $\alpha$  is a momentum term that controls how far the prediction reaches into training history. The network prediction  $y_i$  are then compared with the EMA prediction  $y_{emai}$  by  $L_n = |y_i - y_{emai}|$ , the biggest  $\eta\%$  percentage of noisy label are eliminated from this round of training. That is to say, noise labeling data with unstable predictions were dismissed as useless knowledge.

**Graph Neural Network.** We apply GraphSAGE [6] with mean aggregator as the basic method of our algorithm. The loss of KS-GNN is represented by cross entropy. Golden network  $g$  is a classical supervised graph neural network. Noisy

network  $f$  is a semi-supervised GNN method. We modify the cross entropy loss with reliability and MSE as below,

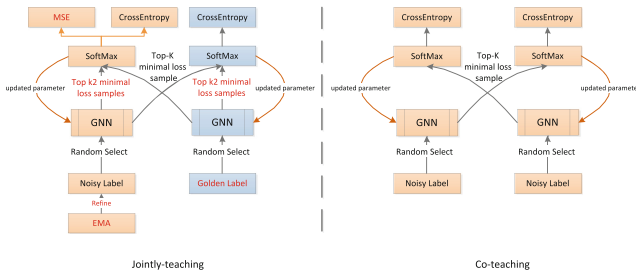
$$\mathcal{L}_f = \frac{1}{N} \sum_{j \in N_{data}} \{-[\log(RE_j) + \log(p_j)] + \omega d_{MSE} \times (y_j, y_{ema_j})\} \quad (3)$$

We add  $\log(RE)$  to the cross entropy, indicating that the higher the reliability is, the smaller the penalty is. Also, we add MSE, Mean Square Error, to measure the label fluctuation between current prediction  $y_j$  and its history prediction  $y_{ema}$ .  $\omega$  is a time-varying coefficients [8] that will gradually release the weight of this term.

The reliability  $RE$  can be updated by  $RE_k = \text{MAX}(b, p_{k-1})$ , where  $b$  is the lower bound on reliability and is set to 0.8.  $RE$  will be updated only after each training epoch.

**Jointly-Teaching.** Han et al. [7] proposed a method to train neural network with noisy labels. Following his idea, we design *Jointly-Teaching* to train a classifier to learn with noisy labels. However, there are many differences between our approach and existing work. The differences between *Jointly-Teaching* and *Co-teaching* are shown in Fig. 3. We explain the differences of the two networks.

1. *Co-teaching* sampling data from the same noisy data which is mixed with golden data, while our method sampling from noisy and golden data respectively. This is because we can tell the clear data from noisy data.
2. *Co-teaching* only selects its small-loss data as the useful knowledge, and transfer these knowledges into its peer network for the further training. While *Jointly-Teaching* retains part of its own data, updates parameters with data of its own, and peer network jointly.
3. Drawing on the idea of temporal ensembling [8], noisy network in *Jointly-Teaching* records its history output to update parameters and pre-cleanses volatile tags to refine noisy data.



**Fig. 3.** Differences between *Jointly-Teaching* GNN (left) and *Co-teaching* (right).

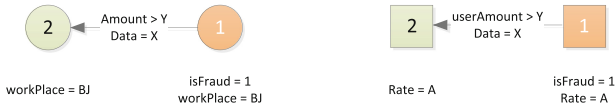
To retain the characteristics of each network own, the network is designed this way. This is because data are sampled from datasets with different noisy level.

The number of minimal loss data is a time-varying coefficients controlled by minimal loss node selection rate  $\sigma_{se}$ , which is similar to Co-teaching.

### 3 Experiments

#### 3.1 Experimental Setup

**Datasets and Graph Construction.** We use a real world dataset, 2017 Banking Transaction Funds Supervision Data (BTFSD), from our cooperator. Also, we apply our model on IEEE-CIS Fraud Detection<sup>1</sup> (CIS-Fraud) and fraud review detection dataset Yelp-Chi [10]. The financial transaction graph of BTFSD looks like Fig. 2 in Sect. 2.1. The graph construction of CIS-Fraud is similar to that of BTFSD. We modeled Yelp-Chi graph the same as CARE-GNN[1].



**Fig. 4.** 2-bounded GFD rules in BTFSD (left) and in Yelp-Chi (right). With slightly modification, GFD rules could easily transfer between various domains.

**GFD Rules Analysis.** GFD rules for BTFSD are made by domain experts. These GFD rules contain 1 1-bounded GFD, 10 2-bounded GFD and 2 3-bounded GFD. We verify these GFD rules by detecting frauds in BTFSD. The GFD rules on BTFSD can accommodate Yelp-Chi with some modification. For example, The meaning of BTFSD GFD rule in Fig. 4 (left) is *user 1 work in BJ, he lends to user 2 over Y amount money at date X may indicate that user 1 is fraud*, while in Yelp-Chi, the GFD rule can be modified as Fig. 4 (right) which means *review 1 and review 2, both rated in A, have over Y amount of users at date X make comment may indicate review 1 is fake*. We modify the BTFSD GFD rules and get 10 2-bounded GFD rules for CIS-Fraud and 6 2-bounded GFD rules for Yelp-Chi.

**Baselines.** To verify the ability of KS-GNN in detecting fraudsters, we compare it with various GNN baselines under the semi-supervised learning setting.

- **GraphSAGE** [6] is a traditional GNN framework.
- **CARE-GNN** [1] is a state-of-the-art GNN model for detecting fraudsters which applies label-aware mechanism.

<sup>1</sup> <https://www.kaggle.com/c/ieee-fraud-detection/data>.

- **KS-noRule** is a reduced version of KS-GNN, which doesn’t use GFD rules to assist. The unlabeled data are randomly marked as fraud, and the reliability of it is set to minimum.
- **KS-Single** is a reduced version of KS-GNN, which only learns noisy network  $f$ . With this implementation, the reliability of golden data is set to 1. Then those golden data are mixed with noisy data for sampling.
- **KS-Co** is a reduced version of KS-GNN, which replaces Jointly-Teaching method with co-teaching.
- **KS-noRefine** is a reduced version of KS-GNN, which removes the refinement process.

**Implementation Details.** Since all datasets are fully labeled, we divide the dataset as follows. (a) Divide the labeled data into two parts by a specific ratio and remove the labels of the noisy part, i.e. 100%, 70%, 50%, 30%, 10% of total fraud are in golden data. (b) Non-fraudulent data are used to supplement those data with removed labels, to keep the ratio of unlabeled data to total data in 5%. Before training, we adopt node2vec [5] to generate features for each node. The parameters setting are the similar to GraphSAGE [6] and Co-teaching [7]. We use the source code provided by authors. Our proposed model are implemented with Pytorch. To avoid the influence of imbalanced classes, we utilize ROC-AUC (AUC) and Recall to evaluate the performance of the models.

**Table 1.** Fraud detection performance (%) on BTFSD under different GLR. GLR means golden label rate.

Datasets	Metric	GLR%	GraphSAGE	CARE-GNN	KS-noRule	KS-single	KS-Co	KS-noRefine	KS-GNN
BTFSD	AUC	100%	57.73	70.28	68.17	<b>84.26</b>	74.98	78.15	77.96
		70%	56.97	70.03	69.20	<b>82.58</b>	78.70	80.92	81.23
		50%	55.24	69.15	69.32	82.24	80.81	83.37	<b>84.75</b>
		30%	53.78	67.89	67.15	80.89	79.12	82.81	<b>84.14</b>
		10%	51.97	64.28	63.81	79.03	78.45	81.69	<b>82.49</b>
	Recall	100%	55.24	59.75	56.12	<b>76.45</b>	69.46	72.97	72.59
		70%	54.52	58.61	57.31	<b>75.38</b>	70.92	74.22	74.56
		50%	53.86	58.69	58.54	73.85	75.38	79.14	<b>80.22</b>
		30%	51.95	60.04	56.89	71.31	74.28	78.78	<b>79.54</b>
		10%	51.31	60.38	54.71	69.83	73.94	78.51	<b>79.17</b>

### 3.2 Performance Evaluation

**Overall Evaluation.** The result of fraud detection performance on BTFSD under different GLR is shown in Table 1. We find that our method and its variants perform better than other methods. Especially, our method outperforms about 15% better than the state-of-the-art fraud detection methods CARE-GNN on BTFSD. Traditional GNN method, GraphSAGE, performs the worst which is because it only aggregate the characteristics from neighbor structures. CARE-GNN adopt label-level attention mechanism to acquire knowledge from labels.

However, our method integrates the domain informations by expertise which is more comprehensive and finally achieves better results than CARE-GNN.

**Golden Label Rate.** Golden label rate (GLR) represents the ratio of labeled nodes assigned to golden labeled set to all labeled data, which mimics the problem of sparse label problem. For example, a 10% Golden label rate indicates that only 10% of the data information is manually filtered, and the other 90% information is hidden. As shown in Table 1, under every GLR, KS-GNN and its variants always achieve the best performance. It demonstrates the validity of combining human knowledge with supervised information.

**KS-GNN Variants.** It is observed in the last five columns of Table 1 that the performance of the noRule implementation is far lower than others. It suggests that expertise could indeed assist the training of GNN. In addition, KS-noRule performs better than traditional baselines, which demonstrate that Jointly-Teaching is able to handle noisy data. With the ascend of GLR, KS-GNN does not perform as well as KS-Single. The reason is that in Jointly-Teaching process, noisy network  $f$  contains too few useful positive data, and the knowledge learned only through parameter transfer of golden network  $g$  is not enough to train a good model. Compared to implementation with Co-teaching, Jointly-Teaching method outperforms it on all GLR. It demonstrates that to retain knowledge from network itself and its peer network is more suitable for training networks with different training data. Our model outperforms well at most of the GLR compare with noRefine. However, our model does not perform as well as KS-noRefine when GLR is elevated. This is because in the case of high GLR, the original data contained less correct information, and the refine process further eliminated part of the correct information (Table 2).

**Table 2.** Fraud detection performance (%) on other domain datasets (Golden labels rate 50%).

Datasets	Metric	GraphSAGE	CARE-GNN	KS-GNN
CIS-Fraud	AUC	53.16	69.12	<b>80.23</b>
	Recall	52.73	63.14	<b>76.01</b>
Yelp-Chi	AUC	54.68	73.71	<b>76.92</b>
	Recall	53.27	69.48	<b>73.46</b>

### 3.3 Generalizations of Our Method

Among most of the datasets, the proposed method performs better than the baselines. When trained on data CIS-Fraud, which is financial domain, our proposed method performs about 11% better than CARE-GNN. While training over fraud review detection dataset Yelp-Chi, the performance of KS-GNN is about 4% over CARE-GNN. That result demonstrates that our approach can be used for other domains other than financial ones.



## 4 Conclusion

In this paper, a knowledge-guided semi-supervised graph neural network is proposed for detecting fraudsters. Human knowledge is used to tackle the problem of labeled data scarcity. We use GFD rules to label unlabeled data. *Reliability* and EMA is used to identify the noise level and refine these noisy data. Additionally, an ingenious model, *Jointly-Teaching*, is designed to train a classifier from labeled data and noisy data cooperatively. Experiments with real-world financial fraud datasets and their corresponding GFDs demonstrate the effectiveness and efficiency of our model. Further experiments on other domain datasets and modified GFDs show that our method is generalizable in many other fields. A feasible idea is to use methods of automatic knowledge mining on knowledge graph, such as GFD mining [3] and GAR [4], to mine domain knowledge automatically to assist KS-GNN.

## References

1. Dou, Y., Liu, Z., Sun, L., Deng, Y., Peng, H., Yu, P.S.: Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In: International Conference on Information and Knowledge Management, Proceedings, pp. 315–324 (2020). <https://doi.org/10.1145/3340531.3411903>
2. Fan, W.: Dependencies for graphs: challenges and opportunities. *J. Data Inf. Qual.* **11**(2), 1–10 (2019). <https://doi.org/10.1145/3310230>
3. Fan, W., Hu, C., Liu, X., Lu, P.: Discovering graph functional dependencies. *ACM Trans. Database Syst.* **45**(3), 1–42 (2020). <https://doi.org/10.1145/3397198>
4. Fan, W., Jin, R., Liu, M., Lu, P., Tian, C., Zhou, J.: Capturing associations in graphs. *Proc. VLDB Endow.* **13**(11), 1863–1876 (2020). <https://doi.org/10.14778/3407790.3407795>
5. 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 (2016)
6. Hamilton, W.L., Ying, R., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems (NIPS) 2017–December, pp. 1025–1035 (2017)
7. Han, B., et al.: Co-teaching: robust training of deep neural networks with extremely noisy labels. *arXiv (NeurIPS)*, pp. 1–11 (2018)
8. Laine, S., Aila, T.: Temporal ensembling for semi-supervised learning. In: ICLR (2015), pp. 1–12 (2017). [https://research.nvidia.com/sites/default/files/publications/laine2017iclr\\_paper.pdf](https://research.nvidia.com/sites/default/files/publications/laine2017iclr_paper.pdf)
9. Nguyen, D.T., Mummadi, C.K., Ngo, T.P.N., Nguyen, T.H.P., Beggel, L., Brox, T.: Self: learning to filter noisy labels with self-ensembling (2019)
10. Rayana, S., Akoglu, L.: Collective opinion spam detection: bridging review networks and metadata, pp. 985–994. Association for Computing Machinery, New York (2015). <https://doi.org/10.1145/2783258.2783370>
11. 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 2017–December, pp. 1196–1205 (2017)
12. Wang, D., et al.: A semi-supervised graph attentive network for financial fraud detection. In: Proceedings - IEEE International Conference on Data Mining, ICDM 2019–November, no. 1, pp. 598–607 (2019). <https://doi.org/10.1109/ICDM.2019.00070>



# MSSF-GCN: Multi-scale Structural and Semantic Information Fusion Graph Convolutional Network for Controversy Detection

Haiyang Wang, Xin Song, Bin Zhou<sup>(✉)</sup>, Ye Wang, Liqun Gao, and Yan Jia

National University of Defense Technology, Changsha, China  
{wanghaiyang19, songxin, binzhou, ye.wang}@nudt.edu.cn,  
jiayanjy@vip.sina.com

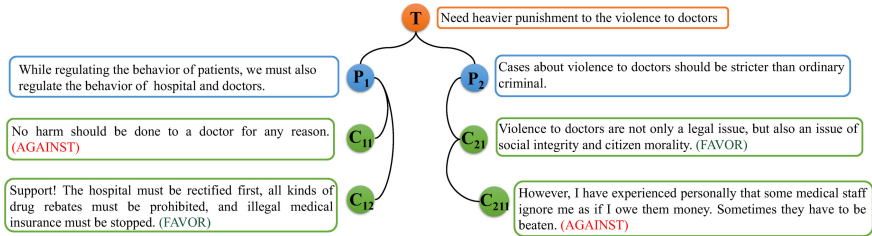
**Abstract.** Detecting controversial posts on the web and social media play an important role in judging the authenticity of web information, measuring the influence of news and alleviating the polarized views. The controversy detection task has attracted widespread attention from researchers in the fields of computer science and social humanities sciences. However, previous works do not achieve: 1) preserve the reply-structure relationship with sentiment information; 2) integrate multi-scale structure and semantic information and provide interpretable results; 3) learn effectively topics and comments information related to the target post. To overcome the first limitation, we construct a **Topic-Post-Comment-Sentiment Graph (TPCS Graph)** for preserving the reply-structure and incorporate the sentiment information. For the second and third limitation, we propose **Multi-scale Structural and Semantic Information Fusion Graph Convolutional Network (MSSF-GCN)** for post-level controversy detection. Moreover, we build a multilingual dataset for controversy detection. We conduct comprehensive experiments on two real-world datasets and the results show that the proposed method exhibits comparable or even superior performance.

**Keywords:** Controversy detection · Information fusion · Graph convolutional networks

## 1 Introduction

With the popularization of the Internet and information retrieval technology, it has become more convenient and faster for people to obtain various opinions and information and publish their opinions. Social media such as Twitter, Weibo are increasingly the place where arguments and discussion are being held. It may lead to polarization of the stance on the same topic or post, which causes controversy. [4] gives a definition of the controversial post: *controversial posts are those that generate strong disagreement among large groups of people*. Controversy detection and analysis play an important role in judging the authenticity

of web information, measuring the influence of news, guaranteeing civil discourse and cultivating critical literacy. Therefore, the computational method of controversy detection and analysis is an area of growing interest in the intersection of several areas of computer science, such as information retrieval, social media analysis, computational social science, natural language processing.



**Fig. 1.** An example of controversial post about whether government need heavier punishment to the violence which aim to doctors. **T** stands for Topic, **P** is Post and **C** are Comments.

In this paper, we focus on detect controversial posts on social media. As we all know, web information is usually classified according to various topics. And under the topics they are interested in, netizen may argue with others on a post or some web information by posting comments. Thus, controversial posts or questions generate strong disagreement. To explain more specifically the controversial phenomenon on the web, we give an example shown in Fig. 1. The topic is whether government need heavier punishment for violence to doctors. There are two target post  $P_1$  and  $P_2$  with some comments. We observe that:

1. Semantic and sentimental information can help us infer the stance of the comment, and then detect the controversy of a post. For example, the stance of  $C_{211}$  is AGAINST.  $C_{211}$  expresses own opinions with strong negative sentiments by using real experiences and metaphors.
2. Global structure information is essential for controversy detection. It refers to traversing all posts and their replies under one topic, and then finding the reply-structure similarity of the controversial post. For example, the controversy of post  $P_1$  occurs between comments and posts, while the phenomenon of blog post  $P_2$  occurred between comments.
3. The fusion of local neighbour information is also very important for detecting controversial posts. For a topic, it can help model the influence between blog posts on the same topic. For example,  $P_1$  and  $P_2$  belong to the same topic. When  $P_2$  is our detection target, if  $P_1$  is a controversial blog post, we think that  $P_2$  has a greater probability of being a controversial post.

Therefore, two key factors are vital to detect the controversy of posts. The first is that modelling the reply-structure relationship and sentiment information. The second is that integrating multi-scale (global and local) structural and

semantic information. Thus, we explore constructing a Topic-Post-Comment-Sentiment Graph (TPCS Graph) which model the reply structure of topics, posts and comments based on the previous work [11]. We propose a **M**ulti-scale **S**tructural and **S**emantic **I**nformation **F**usion Graph Convolutional Network (MSSF-GCN) which can fuse the global structural information, semantic information of content and local neighbour information for post-level controversy detection. In this paper, our key contributions can be summarized as follows:

1. We construct a TPCS Graph to fuse the reply-structure relationships and sentiment information. TPCS Graph leverages sentiment information as a foundation of detecting various opinions and potentially controversy
2. We propose MSSF-GCN for post-level controversy detection. MSSF-GCN can fuse multi-scale information including global structural, semantic and local neighbour to obtain comprehensive feature representation.
3. We build a multilingual dataset for controversy detection which contain German, French and Italian. And we evaluate the proposed model with state-of-the-art models and demonstrate its strength.

## 2 Related Work

Recently, given online social media widespread diffusion, it contains many fierce discussions among users. Thus, controversy detection on social media is a challenge and meaningful and it is usually divided into the topic level and post level. Most models and approaches leverage Twitter-based [8], external knowledge based [8], graph-based [5] and language features [9]. As for the models which take advantage of both semantic and structural features. [11] propose Topic-Post-Comment Graph Convolutional Network (TPC-GCN), which integrates the information from the graph structure and content of topics, posts, and comments for post-level controversy detection.

## 3 Methodology

### 3.1 TPCS Graph Construction

In social media, the goal of a topic (e.g. hashtag) is to group a large amount of information under one theme through keywords. Thus, users can choose the topic in which they are interested and click on hashtags to participate in the discussion. Then, users exchange messages with others through posts, retweets, or comments. Therefore, one of the most common information passing structures in various social media is from the topics to posts, and then from posts to comments. Due to the diversity of social media content and the anonymity of users, people often engage in discussions with different sentiments. Sentiment analysis is seen as a useful tool that can help us deeper insight into the attitudes, opinions, and emotions behind the text which is vital for controversy detection [4]. Considering the above reasons, we formulate the controversy detection task as

a partial node classification problem based on graph. And we propose a TPCS Graph to model the structure of the reply relationship with sentiment information. More formally, we construct TPCS Graph  $G = (V, E)$  where  $V$  and  $E$  denote the set of nodes and edges respectively. The details of nodes and edges are as follows.

**Nodes.** In TPCS Graph  $G$ , there are three different types of nodes which are the topic nodes, post nodes and comment nodes. The topic nodes can be regarded as a hub node to integrate and interchange information. And it is also a root node to connect all post nodes which belongs to this topic. Then, each comment node is connected with the post node or comment node it replies to.

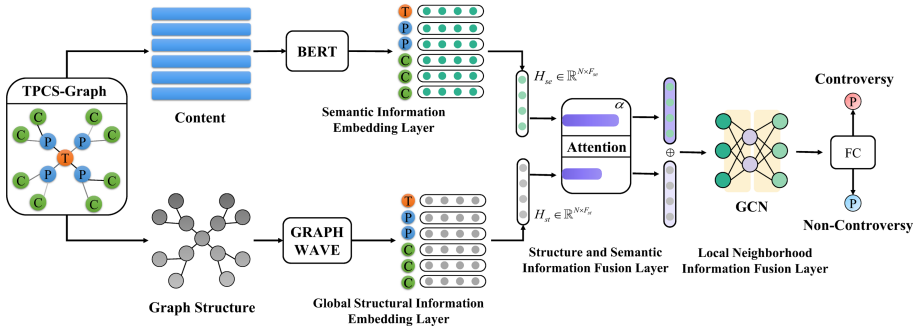
**Edges.** To take sentiment information into account, we leverage PTMs to calculate sentiment scores as edge weights.<sup>1</sup> In detail, we concatenate the text content of the source and target node to calculate the sentiment category  $\{positive, negative\}$  with the score which can be seen as probability. Thus, the edge weight between node  $i$  and node  $j$  is defined as:

$$A[i, j] = Score(i, j), \quad Score(i, j) = \begin{cases} + Positive \\ - Negative \end{cases} \quad (1)$$

where  $A \in \mathbb{R}^{N \times N}$  is the adjacency matrix.  $N$  is the number of nodes.

### 3.2 Multi-scale Structural and Semantic Information Fusion Graph Convolutional Network

Given the TPCS Graph  $G$ , we will introduce Multi-scale Structural and Semantic Information Fusion Graph Convolutional Network (MSSF-GCN) in detail. The framework of MSSF-GCN is shown in Fig. 2. It contains four modules as follows.



**Fig. 2.** System framework of multi-scale structural and semantic information fusion graph convolutional network.

<sup>1</sup> [https://huggingface.co/transformers/task\\_summary.html](https://huggingface.co/transformers/task_summary.html).

**Global Structural Information Embedding Layer.** Structural information is also critical when detecting controversial phenomena on social media. Controversial post nodes usually have more comment nodes and complex neighbour structures. Intuitively, nodes with similar structural roles perform similar functions in the graph. Thus, we need to find the structural similarity of key nodes from a global perspective. With further research on representation learning, the network embedding technique can discover and encode structural properties into a low-dimensional latent space.

In this paper, we employ GRAPHWAVE [3] to learn structural embedding. In our opinion, GRAPHWAVE is an appropriate structural embedding method based TPCS Graph for the following two reasons: 1) it has excellent abilities to capture structural similarity or equivalent of nodes in graphs and provide key insight into the structure of global graphs. 2) it is robust to small perturbations in the local edge structure. In TPCS Graph, We connect the post node to the comment node as long as there is a reply relationship between them. However, some of the comments are perturbations because the content of them is not relevant to the posts. Formally, GRAPHWAVE learns an structural embedding matrix  $H_{st} \in \mathbb{R}^{N \times F_{st}}$ . Each node has  $F_{st}$  dimensional representation  $h_{st} \in \mathbb{R}^{F_{st}}$ .

**Semantic Information Embedding Layer.** With the growth of the internet and the resulting enormous textual content. It is important to develop efficient and effective methods to make use of such an amount of textual information. Recently, PTMs on large unlabelled corpora have shown excellent performance in various tasks in NLP. And PTMs have learned universal language representations and can help researchers avoid training a model from scratch.

In the real world, people often argue with someone because they don't agree with them or they don't like the way they express themselves. Similarly, we think semantic information of text and the way of expression are vary important reasons of the controversial phenomenon in social media. Therefore, semantic information is indispensable to controversy detection models. In order to capture the semantic information of content, we apply BERT [2] to represent each node with an embedding vector  $h_{se} \in \mathbb{R}^{F_{se}}$ . This real-valued vector captures the hidden syntactic and semantic properties of the content. As for all nodes in graph, we can get a semantic embedding matrix  $H_{se} \in \mathbb{R}^{N \times F_{se}}$ .

**Structure and Semantic Information Fusion Layer.** After obtaining the embedding matrix  $H_{st}$  and  $H_{se}$ , we further fuse the two features dynamically. In order to enhance the interpretability, we use the attention mechanism as follows:

$$\mathcal{F}(h_s) = v^T \tanh(W_{\mathcal{F}} h_s + b_{\mathcal{F}}), s \in \{st, se\} \quad (2)$$

$$\alpha_s = \frac{\exp(\mathcal{F}(h_s))}{\sum_{s \in \{st, se\}} \exp(\mathcal{F}(h_s))} \quad (3)$$

$$h_s^{att} = \alpha_s h_s, s \in \{st, se\} \quad (4)$$

where  $W_{\mathcal{F}}$  is the weight matrix,  $b_{\mathcal{F}}$  is the bias,  $v^T$  is a transposed weight.  $\mathcal{F}(\cdot)$  output the score of input vector. Then, we can get the final fusion feature  $h_{fu} = [h_{st}^{att} \oplus h_{se}^{att}]$  by concatenating the global structural and semantic features that are weighted by the attention mechanism.

**Local Neighborhood Information Fusion Layer.** In the past few years, due to the fact of many real-world data can be represented as graphs, GCN [7] have achieved excellent ability to encode both local graph structure and features of the node. GCN can operate directly on a graph and induce node feature vectors from the properties of their neighbourhoods. Besides, it also can involve information from farther neighbours by stacking multiple convolutional layers. Formally, give the TPCS Graph  $G = (V, E)$  with fusion feature matrix  $H_{fu}$ . Let  $H \in \mathbb{R}^{N \times F}$  be a matrix containing features of all  $N$  nodes, where  $F$  is the dimension of the feature vectors. And  $H^{(0)} = H_{fu}$ . Given the adjacency matrix  $A \in \mathbb{R}^{N \times N}$ , GCN updates node feature according to the aggregated information of its neighbor nodes and itself, defined as:

$$H^{(l+1)} = \sigma \left( \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{\frac{1}{2}} H^{(l)} W^{(l)} + B^{(l)} \right) \quad (5)$$

where  $\tilde{A} = A + I_N$  for self-loops.  $\tilde{D}$  is degree matrix and  $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$ .  $\sigma$  is a non-linear activation function. We input fusion feature matrix  $H_{fu}$  to a two layers GCN to obtain representation after message passing, defined as  $H^{(2)} = GCN_s(A, H_{fu})$ .

**Output Layer and Optimization.** we apply a fully connection layer and softmax function to graph feature matrix to get the controversy probability of each **post node**. All model parameters are optimized by using the binary cross entropy loss function and Adam [6] algorithm.

## 4 Experiment

In this section, we want to answer the following key questions: (1) **EQ1:** whether MSFF-GCN can achieve satisfactory detecting results compared to other baseline approaches? (2) **EQ2:** How effective is the fusion of multi-scale structural and semantic features for performance improvement?

### 4.1 Datasets and Baselines

We perform our experiments on two real-world datasets in different languages. The details are as follows:

**Weibo Dataset.** The Weibo dataset<sup>2</sup> released by [11] is a Chinese dataset for controversy detection of social media posts. It contains 49 widely discussed, multi-domain topics. It also contains 1,992 controversial posts and 3,684 non-controversial posts.

<sup>2</sup> <http://mcg.ict.ac.cn/controversy-detection-dataset.html>.

**Smartvote Dataset.** We built a Smartvote dataset for controversy detection based on X-Stance Datasets [10]. X-Stance datasets are large-scale stance detection datasets and it contains 67000 comments, 194 questions and 12 topics about elections in Switzerland and consists of German, French and Italian text. X-Stance is comment-level stance detection datasets. For each comment to the question, X-Stance labelled it favor or against which lays a foundation for us to construct the post(question)-level controversy detection dataset. In detail, we count the distribution of the positions of comments related to each question. As for a question, when the percentage of favor is greater than 66% or less than 33%, we label it as non-controversial, and when the percentage of favor is less than 66% or greater than 33%, we label it as controversial. In total, this dataset contains 56 controversial questions and 138 non-controversial questions which are in line with the distribution in the real world. We randomly divided the dataset into the training set, validation set and test set with a ratio of 3:1:1.

**Table 1.** Performance (%) comparison of different models on two real-world datasets

Model		weibo				Smartvote			
		Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.
GNN	GCN	75.62	75.76	70.70	70.77	79.46	67.22	68.51	77.56
	GAT	63.76	51.12	62.97	67.80	67.80	69.03	62.61	66.67
PTMs	BERT	61.20	58.90	58.63	63.90	54.71	55.85	54.77	56.67
	XLM	63.29	61.46	61.59	65.56	62.97	69.56	56.67	58.97
Fusion	TPC-GCN	78.75	<b>79.86</b>	74.18	76.42	72.87	<b>72.44</b>	67.22	69.87
	MSSF-GCN	<b>83.60</b>	76.35	<b>77.14</b>	<b>80.46</b>	<b>83.43</b>	72.20	<b>70.10</b>	<b>78.59</b>

We implement three representative methods including GNNs, PTMs and fusion methods as baselines. And we compare our model with them to validate the effectiveness of our methods and employ the precision (**Prec.**), recall (**Rec.**), F1-score (**F1**), and accuracy (**Acc.**) as evaluation metrics.

**Graph Convolutional Network (GCN):** We employ the basic GCN model which contains two layers. Nonlinear activation and dropout are used between the two layers.

**Graph Attention Network (GAT):** GAT is a novel neural network architecture that leveraging masked self-attentional layers. We employ the GAT model which contains two layers and nonlinear activation and dropout are used between the two layers.

**BERT:** As our main baseline model, we fine-tune multilingual BERT on controversy detection task which has been pre-trained jointly in 104 languages. We interpret the controversy detection task as sequence pair classification problem.



**XLM:** XLM is a cross-lingual masked language model [1]. In the experiment, we apply the architecture of *xlm-mlm-100-1280* which pre-trained on 100 languages.

**TPC-GCN:** TPC-GCN integrates the information from the graph structure and content of topics, posts, and comments for post-level controversy detection. They extract text features of topics, posts, and comments by BERT and structural by GCN.

## 4.2 Performance Comparison

To answer **EQ1** and **EQ2**, we compare the performance of all methods across the two datasets. Table 1 reports the detection performance of the proposed MSSF-GCN model in comparison to other state-of-the-art approaches. As for the GNN models, we notice that GCN achieves higher scores on both datasets than GAT. This may be due to the stronger ability of GCN to integrate local neighbour information. PTMs reports low scores on the two datasets, indicating that only using the content of a post is insufficient to identify the controversy. Comparing the results of GNN and PTMS, we find that the structure-based approach is superior to the semantic-based approach, which implies the importance of reply-structural information. For example, there is one extreme situation in social media is that when two users with different numbers of followers post the same content. The user with a larger number of followers may get more comments and generate more structural information which is more likely to cause controversy. However, a post by a user with a smaller number of followers may not attract comments from others and thus not cause controversy. The fusion models outperform or are comparable to the other baselines which proves that information fusion of semantic and structure is necessary to improve the performance. And our model outperforms the SOTA model TPC-GCN which indicates the sentiment can help improve the detection performance.

## 5 Conclusion

Controversy detection is an important task for social network and web information analysis. In this paper, we construct a TPCS Graph to model reply-structure relationship and sentiment information. Based on TPCS Graph, this paper presents a novel method MSSF-GCN to fuse the global structural information, semantic information and local neighbour for post-level controversy detection. Different from the existing works, we dig deep into the global structural information to find the structural similarity of the controversial nodes. We also pay attention to the integration of multi-scale structure and semantic information. We build a multilingual dataset for controversy detection that provides support for the follow-up researches. Comprehensive experiments demonstrate the effectiveness of the proposed model on two real-world datasets. In the future, we plan to explore unsupervised controversy detection algorithms. Besides, we are also interested in the multi-task learning model that can be applied for controversy and stance detection.

**Acknowledgements.** This work was supported by the National Key Research and Development Program of China NO. 2018YFC0831703.

## References

1. Conneau, A., Lample, G.: Cross-lingual language model pretraining. In: *Advances in Neural Information Processing Systems*, vol. 32, pp. 7057–7067 (2019)
2. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.N.: BERT: pre-training of deep bidirectional transformers for language understanding. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186 (2018)
3. Donnat, C., Zitnik, M., Hallac, D., Leskovec, J.: Learning structural node embeddings via diffusion wavelets. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1320–1329 (2018)
4. Dori-Hacohen, S.: Controversy analysis and detection (2017)
5. Garimella, K., Morales, G.D.F., Gionis, A., Mathioudakis, M.: Quantifying controversy on social media. *ACM Trans. Soc. Comput.* **1**(1), 3 (2018)
6. Kingma, D.P., Ba, J.L.: Adam: a method for stochastic optimization. In: *ICLR 2015 : International Conference on Learning Representations 2015* (2015)
7. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: *ICLR (Poster)* (2016)
8. Popescu, A.M., Pennacchiotti, M.: Detecting controversial events from twitter. In: *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, pp. 1873–1876 (2010)
9. Rethmeier, N., Hübner, M., Hennig, L.: Learning comment controversy prediction in web discussions using incidentally supervised multi-task CNNs. In: *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pp. 316–321 (2018)
10. Vamvas, J., Sennrich, R.: X-stance: a multilingual multi-target dataset for stance detection. *SwissText/KONVENS* (2020)
11. Zhong, L., Cao, J., Sheng, Q., Guo, J., Wang, Z.: Integrating semantic and structural information with graph convolutional network for controversy detection. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 515–526 (2020)



# A Syntax-Aware Encoder for Authorship Attribution

Jianbo Liu<sup>1</sup>, Zhiqiang Hu<sup>1</sup>, Jiasheng Zhang<sup>1</sup>, Roy Ka-Wei Lee<sup>2</sup>,  
and Jie Shao<sup>1,3</sup>(✉)

<sup>1</sup> University of Electronic Science and Technology of China, Chengdu 611731, China  
{liujianbo,zhiqianghu,zjss12358}@std.uestc.edu.cn

<sup>2</sup> Singapore University of Technology and Design, Singapore 487372, Singapore  
roy\_lee@sutd.edu.sg

<sup>3</sup> Sichuan Artificial Intelligence Research Institute, Yibin 644000, China  
shaojie@uestc.edu.cn

**Abstract.** Authorship Attribution (AA) refers to the task of predicting the author of a given text by learning unique writing styles of different authors. It often involves the extraction of social language characteristics of texts such as writing styles to identify authors. Among the many facets of text styles, we postulate that authors have unique and signature syntactic information entangled in their texts. This paper investigates this hypothesis and proposes a Syntax-Aware Encoder (SAE), a novel graph convolutional network based model augmented with the dependency tree to learn the syntax representation for short texts. Subsequently, we leverage the learned latent representations to perform AA. Extensive experiments are conducted on two social media short-text datasets. The results show that SAE outperforms the state-of-the-art baselines on the AA task.

**Keywords:** Authorship Attribution · Dependency tree · Graph convolutional networks

## 1 Introduction

Authorship Attribution (AA) refers to the task of determining the author for a given document [8]. In the context of social media, cybercrime has become more rampant with growing issues of online falsehood and vicious sock puppets [2, 14]. Therefore, it is pivotal to develop effective AA solutions that could tackle the spectrum of document fraud and cybercrimes.

Existing AA studies have explored many features to perform the task. Nevertheless, very few studies have considered the syntax of texts for AA. We postulate that syntactic information is a vital aspect of an author's writing style, and different authors organize their sentences differently. Thus, the text's syntactic properties could be useful features for AA. A potential solution is to leverage sequence models such as Long Short-Term Memory (LSTM) to capture

syntactic information. Because natural language has the syntactic property of combining words into phrases, some methods extended LSTM to tree structures [12, 15]. The network of Tree-LSTM is constructed based on the syntax tree of the input sentence. Using these methods, the encoded feature vector contains semantic information and syntactic information. However, the syntactic information is mainly used to assist semantic representation, and existing study has shown that explicit syntactic supervision is required to improve the learning of sentence syntactic information [7], so explicit modeling of syntax tree may be helpful for the AA task. Zhang et al. [13] proposed a Syntax-CNN model to encode a sentence’s syntactic information as a constituency tree for the AA task. Syntax-CNN model encodes the constituency tree into a learnable distributed representation. In order to avoid syntactic information loss, this model encodes syntax labels and syntax label positions in the syntax path. It explicitly models the constituency tree and achieves good performance on the AA task, but the results suggest that the depth information of the constituency tree is an important feature that distinguishes writing styles of authors. For short texts, the syntax path of the constituency tree is shorter, which means that it may contain less syntactic information. Short texts will affect the quality of the syntactic information extracted by Syntax-CNN model, so our motivation of this work is to explicitly obtain syntactic information of sentences. As the syntactic information may be affected by the depth of the constituency tree, we would like to use other ways to obtain syntactic information.

This paper aims to address the above-mentioned limitations by proposing Syntax-Aware Encoder (SAE). SAE is a novel Graph Convolutional Network (GCN) based model that learns the syntax representation for the short text based on its dependency tree representation. We summarize our main contributions as follows:

- To the best of our knowledge, we are the first to use GCN with dependency trees to extract the syntactic information of sentences to perform the AA task. Our method avoids the problem of short syntax path of the constituency tree. Meanwhile, it can extract more abundant syntactic information to help solve AA task.
- We conduct extensive experiments on two social media short-text datasets used in previous studies. The experimental results show that SAE consistently outperforms state-of-the-art methods.

## 2 Related Work

With the development of social media, short-text AA becomes more and more important but challenging. A survey [8] provides a comprehensive review of the AA methods that can be applied to the problem of social media forensics. In a n-gram CNN model [11], CNN is used to capture character n-grams for short-text datasets. The n-gram CNN model considers a single short-text post as input to train the model and learn the writing styles of authors. In [9], CNN and various

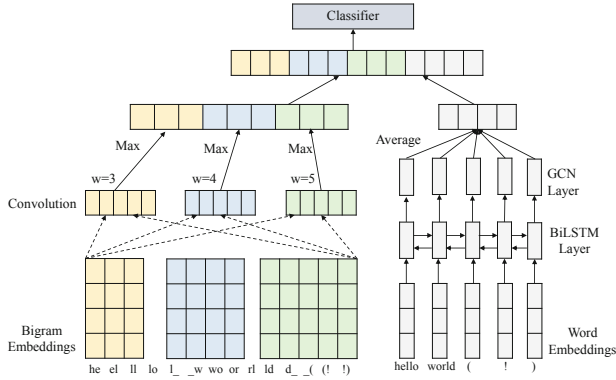


Fig. 1. The structure of SAE.

types of word and character level features are used for AA. DeepStyle [4] and iur [1] are two methods which map author writing style to latent space for the short-text AA task. DeepStyle adopts triplet loss to learn the post embeddings so that the posts belong to the same author should be close to each other in the latent space, while posts from different authors will be far away from each other. The iur model organizes the actions of each author into short sequences arranged in chronological order and learns the mapping from short events of author actions on social media to vector space.

Syntactic information plays an important role in the field of NLP. The syntactic feature engineering extracts certain attributes or statistical information from the syntax tree as syntactic feature. Some methods [12, 15] get encoded feature vectors that contain syntactic information, but the syntactic information is mainly used for auxiliary representation of the semantic representation. Now explicit syntactic supervision is widely used and achieves good performance. In [3], GCN is used to integrate dependency tree information into neural machine translation models and achieve improvement in generating more fluency sentences. For the AA task, Syntax-CNN model [13] explicitly extracts syntactic information from the constituency tree.

### 3 Syntax-Aware Encoder Model

*CNN Module.* N-gram CNN model [11] adopts CNN to extract character n-grams from short text and achieves good performance. CNN can obtain the local context information of input feature vectors. The change of window size can change the information obtained after convolution. In view of previous work [11], we use CNN to extract content representation with character n-grams.

*Structure Module.* We use LSTM to process static word embedding. Let  $W_w = [w_1, w_2, \dots, w_m]$ , which represents the word sequence of the sentence and  $m$  is the number of words in the sentence. LSTM takes  $W_w$  as input and generates the hidden state of each word. However, only encoding the left text will lose some information, and the right context is equally important. Therefore, we use  $LSTM_F$  and  $LSTM_B$  to represent forward pass and backward pass respectively. BiLSTM layer is bi-directional LSTM.

$$h_i = LSTM_F(w_{1:i}) \circ LSTM_B(w_{m:i}), \quad (1)$$

where  $h_i$  is the hidden state of the  $i$ -th word and encodes the word at position  $i$  and its left context, and  $\circ$  is the concatenation operation.

The adjacency matrix of a sentence can be obtained from the dependency relationship between words. However, for short text such as social media text, the adjacency matrix obtained from the dependency tree is always very sparse. The sparse adjacency matrix means that less dependency exists among words, which may lead to the insufficient extraction of sentence structure information. We believe that self-loop and opposite directed dependency arc are also important. For each word node, there are three kinds of information, head-to-dependent, dependent-to-head and self loop, that can be transmitted, which can enrich the sentence structure [5].

Consider a graph  $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ , where  $\mathcal{V}$  is the set of nodes, and  $\mathcal{E}$  is the set of edges. As claimed in [6], the information of the node itself is important, so for each node  $\nu \in \mathcal{V}$ , they add a self-loop edge  $(\nu, \nu) \in \mathcal{E}$ . On this basis, we obtain the matrix  $H \in \mathbb{R}^{m \times l}$  from BiLSTM layer, where each row  $h_\nu \in \mathbb{R}^l$  ( $\nu \in \mathcal{V}$ ) corresponds to a feature for node  $\nu$ , and  $l$  is the dimension of the matrix. GCN extracts the information of each node as follows:

$$q_\nu = \sigma(Ah_\nu W + b), \quad (2)$$

where  $A \in \mathbb{R}^{m \times m}$  is the adjacency matrix corresponding to the graph structure,  $W$  is the weight matrix and  $b$  is the bias which can be learned.  $\sigma(\cdot)$  is a non-linear activation function.  $h_\nu$  is the input feature and  $q_\nu$  is the latent representation which abstracts the information of adjacent nodes.

Finally, we employ GCN to extract the syntax representation from the dependency tree and character n-gram CNN to gather the content representation. The fully connected layer containing a softmax layer takes the concatenated representation of syntax and content as input to predict the author of the given text.

## 4 Experiments

*Datasets.* We evaluate our model on two social media short-text datasets derived from Twitter and Weibo. We follow the same strategy as in [4] to obtain the training and testing sets of each dataset. For each author, 100 posts are used as the training set and the remaining of 20 posts as the testing set.

- **Twitter**: The Twitter dataset [10] consists of tweets from 7,026 authors, in which each author has 120 tweets. The average number of words per text is 17 and the average number of characters is 76.
- **Weibo**: The Weibo dataset consists of Chinese posts from the Weibo website written by 9819 authors. The average number of words per text is 69 and the average number of characters is 172.

*Baselines.* We compare with the following baselines:

- **N-gram CNN**: N-gram CNN [11] uses different window sizes to extract character features and achieves good performance. We train this model on character features (called CNN-1), bigram features (called CNN-2), and word features (called CNN-W).
- **LSTM**: We train an LSTM at the character level (LSTM-1), bigram level (LSTM-2), and word level (LSTM-W).
- **Syntax-CNN** Syntax-CNN [13] model uses character trigram feature and constituency tree information which encodes syntax labels and syntax label positions in the syntax path to learn the writing styles of authors.
- **DeepStyle**: DeepStyle [4] adopts triplet loss to learn the post embeddings so that the posts belong to the same author should be close to each other in the latent space, while posts from different authors will be far away from each other.
- **iur**: iur is the method of employing author embedding. The core of iur is to learn the mapping from short events of author activities on social media to vector space.

*Implementation.* For CNN model, we adopt character bigram with 300 dimension embedding. The CNN layer has three filter windows of 3, 4, 5 and each of which has 500 filter kernels. For the GCN model, the word embedding size is 256 and the output dimension of BiLSTM is 128. The output dimension of the GCN layer is also set to 128. A dropout layer is added before the final output layer. Additionally, we apply Adam with a learning rate of  $1e-4$  to optimize the parameters. We train our model for at most 100 epochs. We use precision at top k (P@k), where a prediction is considered as correct when the top k ranked users predicted by a model for a given test query post contains the ground-truth author.

*Comparison Results.* Table 1 shows the experimental results, where each best result is highlighted in bold. It shows that SAE outperforms all the baseline methods for different k values. Syntax-CNN model which uses the consistency tree path and the position of syntax label achieves the best performance in all baselines. These results suggest that syntactic information extracted from the syntax tree can effectively improve the performance of AA task. Meanwhile, the results of our model are better than the Syntax-CNN model on all P@k measures, e.g., our model achieves 9.2% and 9.9% higher in P@1 for each dataset. The results prove again the importance of syntactic information for AA and

**Table 1.** P@k of various baselines and SAE. We vary k = [1, 10, 50, 100]. SAE gets the best performance.

Method	Twitter				Weibo			
	P@1	P@10	P@50	P@100	P@1	P@10	P@50	P@100
CNN-1 [11]	18.70	30.70	43.10	49.90	22.00	35.20	48.40	55.90
CNN-2 [11]	20.10	33.70	47.60	55.30	17.50	29.70	41.30	51.30
CNN-W [11]	16.40	27.10	37.60	44.00	21.00	36.50	48.60	54.80
LSTM-1 [4]	15.30	25.60	35.90	42.00	19.00	33.60	48.10	55.50
LSTM-2 [4]	15.50	28.00	40.90	48.60	15.80	31.50	43.50	49.80
LSTM-W [4]	11.70	23.70	41.20	48.10	18.70	34.10	48.00	55.20
DeepStyle [4]	21.40	37.10	51.50	59.20	23.90	38.10	51.60	58.90
iur [1]	7.15	16.84	28.61	35.81	3.48	10.49	21.68	28.85
Syntax-CNN [13]	26.2	41.55	55.19	62.47	27.82	40.2	52.38	59.10
SAE	<b>28.62</b>	<b>46.07</b>	<b>60.3</b>	<b>67.33</b>	<b>30.59</b>	<b>44.26</b>	<b>55.96</b>	<b>61.82</b>

**Table 2.** P@1 for SAE and baselines on varying number of users.

Method	Twitter authors				Weibo authors			
	100	500	5000	All	100	500	5000	All
CNN-1 [11]	47.10	35.60	23.10	18.70	52.60	38.80	25.10	22.00
CNN-2 [11]	46.70	36.50	24.60	20.10	42.40	30.40	18.70	17.50
CNN-W [11]	37.10	28.40	17.70	16.40	48.30	36.20	23.80	21.00
LSTM-1 [4]	29.60	27.30	16.10	15.30	32.50	30.90	20.10	19.00
LSTM-2 [4]	34.50	25.20	16.50	15.50	26.40	19.60	16.80	15.80
LSTM-W [4]	22.80	19.30	12.00	11.70	33.40	28.80	19.80	18.70
DeepStyle [4]	51.20	37.90	25.50	21.40	56.60	42.80	27.90	23.90
iur [1]	19.20	12.70	7.78	7.51	19.50	5.7	3.66	3.36
Syntax-CNN [13]	50.35	40.65	26.84	26.20	55.80	43.03	31.87	28.82
Syntax	40.85	33.43	24.05	22.80	41.25	31.03	27.16	21.46
SAE	<b>55.85</b>	<b>44.06</b>	<b>30.12</b>	<b>28.51</b>	<b>56.80</b>	<b>43.62</b>	<b>32.18</b>	<b>30.59</b>

suggest the syntactic information extracted by our SAE is more abundant than Syntax-CNN. SAE applies the dependency tree to obtain the dependency relationship between words in a sentence. Thus, it can capture sentence structure information by using GCN with dependency tree.

*Effect of Varying Number of Authors.* Table 2 shows the P@1 results of our SAE model and various baselines. We change the number of authors from 100 to all authors. As the number of authors increased, the precision of all methods decreased. From the results, DeepStyle performs better than Syntax-CNN on datasets with smaller number of authors. However, Syntax-CNN is better than DeepStyle on datasets with a large number of authors. These results indicate that



**Table 3.** P@1 obtained by SAE with and without structure model.

Dataset	With	Without
Twitter 50 authors	65.9	64.5

**Table 4.** P@1 obtained by SAE with different GCN layers.

Dataset	Layers				
	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$
Twitter 50 authors	65.9	65.0	64.6	65.4	63.4

DeepStyle achieves better performance on small datasets, while the performance of Syntax-CNN is better on large datasets and lower than DeepStyle on small datasets. Our model performs much better than other methods on each author number setting. Even in the worse case of all authors, i.e., over 7,000 and 9,000 Twitter and Weibo authors respectively, SAE is still able to outperform the best baselines. For example, SAE achieves 9% higher than DeepStyle on the Twitter dataset of 100 authors and 8.8% higher than Syntax-CNN on the Twitter dataset of all authors. SAE can achieve good performance on any datasets, especially on large datasets. These results suggest that SAE can solve the problem of AA of large social media datasets. We also can observe that our model achieves 8.8% and 6.1% higher than Syntax-CNN model on Twitter all authors and Weibo all authors. These result suggest that SAE can better capture syntactic information for shorter texts. We also only use structure model to solve the AA task in the experiment and obtain results on the second last row in Table 2. These results indicate that using only syntactic information did not achieve good performance. This conclusion is consistent with [13].

*Ablation Study.* To explore the effect of GCN, which is used in our model to capture the syntactic information on the dependency tree, we conduct an ablation study on the Twitter dataset with 50 authors. We remove the structure model in SAE and demonstrate its effect on the performance of the model. As shown in Table 3, the results of our model with the structure model are better than our model without the structure model and achieves 2.1% higher on accuracy. This demonstrates that the sentence structure plays an important role in the writing styles of authors. We can infer that the syntactic information extracted by SAE is helpful for identifying the styles of authors.

To explore the effect of the number of GCN layers, we conduct experiments with different GCN layers. From the results shown in Table 4, we can observe that our model with one GCN layer gets the best performance. The results of [3] is consistent with our experimental results, that is, stacking multiple GCN layers may not bring any benefits. Since the dependence of short text is not complicated, multi-layer GCN will make the performance worse. These results suggest that the GCN of one layer is suitable for extracting syntactic information of short text.

## 5 Conclusion

In this paper, we propose a novel AA model SAE to capture the syntactic information of text. We employ GCN to extract the syntax representation from the dependency tree of text, and a character n-gram CNN is employed to gather the content representation. The results show that SAE outperforms all the state-of-the-art baselines and has a better capacity to capture syntactic information for short-text AA task. The syntax representation of text can be helpful to overcome the challenges of short-text AA.

**Acknowledgments.** This work was supported by the National Natural Science Foundation of China (No. 61832001) and Sichuan Science and Technology Program (No. 2019YFG0535).

## References

1. Andrews, N., Bishop, M.: Learning invariant representations of social media users. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, EMNLP 2019, pp. 1684–1695 (2019)
2. Ansah, J., Liu, L., Kang, W., Liu, J., Li, J.: Leveraging burst in twitter network communities for event detection. *World Wide Web* **23**(5), 2851–2876 (2020). <https://doi.org/10.1007/s11280-020-00786-y>
3. Bastings, J., Titov, I., Aziz, W., Marcheggiani, D., Sima'an, K.: Graph convolutional encoders for syntax-aware neural machine translation. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, pp. 1957–1967 (2017)
4. Hu, Z., Lee, R.K.-W., Wang, L., Lim, E., Dai, B.: DeepStyle: user style embedding for authorship attribution of short texts. In: Wang, X., Zhang, R., Lee, Y.-K., Sun, L., Moon, Y.-S. (eds.) APWeb-WAIM 2020. LNCS, vol. 12318, pp. 221–229. Springer, Cham (2020). [https://doi.org/10.1007/978-3-030-60290-1\\_17](https://doi.org/10.1007/978-3-030-60290-1_17)
5. Jiang, H., Zhou, R., Zhang, L., Wang, H., Zhang, Y.: Sentence level topic models for associated topics extraction. *World Wide Web* **22**(6), 2545–2560 (2018). <https://doi.org/10.1007/s11280-018-0639-1>
6. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, ICLR 2017, Conference Track Proceedings (2017)
7. Kuncoro, A., Dyer, C., Hale, J., Yogatama, D., Clark, S., Blunsom, P.: LSTMs can learn syntax-sensitive dependencies well, but modeling structure makes them better. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Volume 1: Long Papers, pp. 1426–1436 (2018)
8. Rocha, A., et al.: Authorship attribution for social media forensics. *IEEE Trans. Inf. Forensics Secur.* **12**, 5–33 (2017)
9. Ruder, S., Ghaffari, P., Breslin, J.G.: Character-level and multi-channel convolutional neural networks for large-scale authorship attribution. *CoRR* abs/1609.06686 (2016)
10. Schwartz, R., Tsur, O., Rappoport, A., Koppel, M.: Authorship attribution of micro-messages. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, pp. 1880–1891 (2013)

11. Shrestha, P., Sierra, S., González, F.A., Montes-y-Gómez, M., Rosso, P., Solorio, T.: Convolutional neural networks for authorship attribution of short texts. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Volume 2: Short Papers, pp. 669–674 (2017)
12. Tai, K.S., Socher, R., Manning, C.D.: Improved semantic representations from tree-structured long short-term memory networks. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics, ACL 2015, Volume 1: Long Papers, pp. 1556–1566 (2015)
13. Zhang, R., Hu, Z., Guo, H., Mao, Y.: Syntax encoding with application in authorship attribution. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 2742–2753 (2018)
14. Zhu, J., Ghosh, S., Wu, W.: Robust rumor blocking problem with uncertain rumor sources in social networks. *World Wide Web* **24**(1), 229–247 (2020). <https://doi.org/10.1007/s11280-020-00841-8>
15. Zhu, X., Sobhani, P., Guo, H.: Long short-term memory over recursive structures. In: Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, pp. 1604–1612 (2015)



# Graph Attentive Leaping Connection Network for Chinese Short Text Semantic Classification

Jingdan Zhu<sup>(✉)</sup>

East China Normal University, Shanghai 200062, China

**Abstract.** Chinese short text semantic classification is a ubiquitous task that widely occurs in natural language processing. The existing methods are generally utilized in English, leading to multiple limitations: Unable to capture word-level abundant semantic when the input tokens are character sequence. It would be more vulnerable to data sparsity and the presence of out-of-vocabulary (abbr., OOV) words if utilizing word-based models, and thus more prone to overfitting. The very few approaches that consider both granularities are still limited. To tackle these problems, we propose a novel Graph Neural Network. Our model adopts the word lattice graph to keep multi-granularity information and utilizes the pre-trained model to obtain powerful semantics. Additionally, the attention mechanism and the layer leaping connection enable better structure-aware representation. Experimental results on three Chinese datasets demonstrate that our model achieves state-of-the-art performance in short text classification models.

**Keywords:** Semantic classification · Graph attention network · Leaping connection

## 1 Introduction

Text classification is pushed forward into many target applications [10], e.g., sentiment analysis, question answering, natural language inference, etc. It aims to process different kinds of texts and classify them into pre-defined labelled categories. Short text semantic classification serves the role analogous to sentence pair classification in Chinese context semantic environment. For sentence pair classification tasks, two text sequences will be considered the input for approaches and a label or a scalar value indicating their relation will be received. Numerous tasks, including natural paraphrase identification [14] and answer selection [17] can be seen as specific forms of text matching problems.

As a surge of interest and distinguished work [6, 15] emerge in natural language processing (abbr., NLP) recently, choosing proper methods becomes more practical and challenging. Pre-trained learning models (e.g., BERT or GPT and

their variants) outperform more better than traditional machine learning methods in almost all scenarios.

Of note, deep graph neural networks are preferred utilized in text classification, efficiently capturing semantic connections between words, phrases or sentences, evolving into feasible representation methods. Nevertheless, most of the datasets used for text classification only provide English version. Not only Chinese datasets, but also how to migrate the methods for text classification needs to be measured. Early work utilizes Chinese characters as input to the model, or first segments each sentence into words and then takes these words as input tokens. Word-based models are more susceptible to sparse data and the presence of out-of-vocabulary words will also lead to performance degradation, and thus more prone to overfitting [7]. However, character-based models cannot fully utilize explicit word information, which is not negligible in Chinese semantic classification.

In this paper, we propose a Graph Attention Leaping Connection Network (abbr., GLCN) to consider both semantic information and multi-granularity information, achieving sufficient information aggregation while alleviating over smoothing. Our model needs to build a pair of word lattice graphs. In order to reduce noise and computation, only several segmentation paths are utilized to form the lattice graph during the construction process. Also, we get the initial word representation by aggregating features from the character-level interaction. For nodes updating, we use an attention mechanism to weigh “important” neighbors more. When getting the final representation of each node, we introduce a leaping connection policy for the first time, which considers information from all nodes in the graph and can be generalized to new graphs by Max-Pooling.

There are four main aspects of our contribution:

- 1) Our model makes full use of the multi-granularity information of characters and words.
- 2) Attention mechanism is introduced to better aggregate the information between words and characters.
- 3) Leaping connection constructed by adaptive Max-Pooling achieves node information aggregation without introducing additional learning parameters while avoiding over-smoothing.
- 4) Experiments on three datasets demonstrate that our model outperforms the state-of-the-art model.

## 2 Related Work

**Deep Text Classification.** Recently, pre-trained models (abbr., PTMs) like BERT [5] have shown their powerful ability in learning contextual word embeddings. For Chinese text classification, BERT takes a pair of short texts as input and each character is a separated input token. It has ignored word information. To tackle this problem, some Chinese variants of original BERT have been proposed, e.g. BERT-wwm [4], ERNIE [12] and its update ERNIE2.0 [11]. They

take the word information into consideration based on the whole word masking mechanism during pre-training.

**Graph Neural Networks.** Graph neural networks derive from network embedding, which effectively maps nodes to low-dimensional representations and records the structure of the network. As a typical kind of non-Euclidean data, graph-structure data is playing a crucial role in the field of deep neural networks [16, 18]. These deep neural network architectures are known as Graph Neural Networks (abbr., GNNs), which have been proposed to learn meaningful representations for graph-structure data.

### 3 Graph Attentive Leaping Connection Model

#### 3.1 Problem Definition

For ease of presentation, we define the notations and key data structures used in this paper.

**Definition 1 (Chinese Text Classification).** Given two Chinese short text sequences  $S^a = \{s_1^a, s_2^a, \dots, s_{T_a}^a\}$  and  $S^b = \{s_1^b, s_2^b, \dots, s_{T_b}^b\}$ , the goal of our text classification model  $f(S^a, S^b)$  is to predict weather  $S^a$  and  $S^b$  have the same semantics. Where  $s_i^a$  and  $s_j^b$  represent the  $i$ -th and  $j$ -th Chinese character in two texts respectively, and  $T_a$  and  $T_b$  denote the number of characters.

**Definition 2 (Chinese Lattice Graph).** A Lattice Graph consists of the result of Chinese word segmentation and the original character sequence. Since keeping all possible segmentation paths will lead to excessive computation and noise, we stay several paths by random selection like Fig. 2 to form a word lattice graph  $G = (\mathcal{V}, \mathcal{E})$ . Each word and character represents a node.  $\mathcal{V}$  is the set of nodes.  $\mathcal{N}(v_i)$  denotes the set of all neighbor nodes of node  $v_i$  except itself.

#### 3.2 Model Description

As shown in Fig. 1, our model consists of four components: a lattice embedding module, a neighborhood interaction-based attention module, a leaping connection module and a final semantic classifier.

**Lattice Embedding Module.** For each node  $v_i$  in graph lattice, the initial representation of word  $w_i$  is the aggregation of contextual character representations. We first recombine the two original character-level text sequences to a new one and then feed them to the BERT pre-train model to obtain the contextual representations for each character  $C = \{c^{\text{CLS}}, c_1^a, \dots, c_{T_a}^a, c^{\text{SEP}}, c_1^b, \dots, c_{T_b}^b, c^{\text{SEP}}\}$ .

Next, we define the characters contained in each word  $w_i$  in each graph as  $\{s_i, s_{i+1}, \dots, s_{i+n_i-1}\}$ , which means the node  $v_i$  has  $n_i$  consecutive character tokens and  $s_i$  denotes the index of the first character of  $v_i$  in the text  $S^a$  and  $S^b$ . Then, we calculate a feature-wised score vector  $u_k$ , with a two layers feed forward network (abbr., FFN) for each character  $c_{i+k}$  ( $0 \leq k \leq n_i$ ) in  $w_i$  like [2] and then normalized with a feature-wised softmax as Fig. 2.

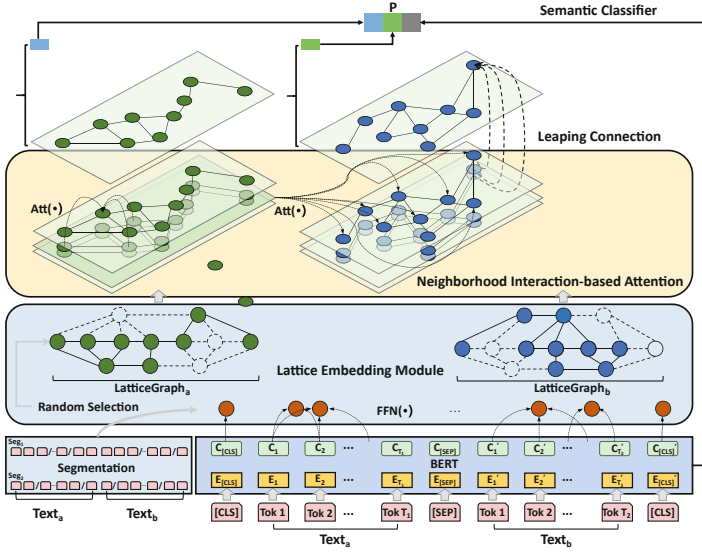


Fig. 1. The framework of GLCN-BERT model.

$$u_{i+k} = \text{softmax}(\text{FFN}(c_{i+k})) \tag{1}$$

The corresponding character embedding  $c_{i+k}$  is weighted with the normalised scores  $u_{i+k}$  to obtain the initial node embedding  $v_i = \sum_{k=0}^{n_i-1} u_{i+k} \odot c_{i+k}$ . where  $\odot$  represents element-wise product of two vectors.

At the end of this module, we get two lattice graph embedding sets  $G^a$  and  $G^b$ , which consist of both character-level and word-level representations.

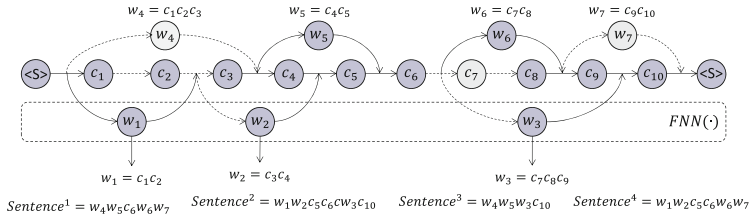


Fig. 2. Contextual word embedding.

**Neighborhood Interaction-Based Attention Module.** Since the utilization of the attention mechanisms allows the learning process to focus on parts of the graph that are more relevant to a specific task. As Fig. 2 shows, the graph attention classification module takes the contextual node embedding  $u_i$  as the initial representation  $h_i^0$  for each node  $v_i$ , then updates its representation from one layer to the next. We simplify the update strategy into two steps:

**(1)Message Propagation.** At  $l$ -th step, each node  $v_i$  in  $G^a$  (the same with  $G^b$ ) will first aggregates messages from its own neighbor nodes and then combine the result with the node representation from the last iteration,

$$\mathbf{h}_i^{self} = \text{GRU} \left( \mathbf{h}_i^{l-1}, \sigma \left( \sum_{v_q \in \mathcal{N}(v_i)} \alpha_{ij} (\mathbf{W}^{self} \mathbf{h}_j^{l-1}) \right) \right) \quad (2)$$

In order to make full use of the information of  $G^b$ , we also aggregate messages from all nodes in graph  $G^b$ ,

$$\mathbf{h}_i^b = \sigma \left( \sum_{v_q \in \mathcal{V}(v_b)} \alpha_{iq} (\mathbf{W}^b \mathbf{h}_q^{l-1}) \right) \quad (3)$$

Here, the  $\sigma$  is a non-linear activation function, e.g. a ReLU. And  $\alpha_{ij}$  and  $\alpha_{iq}$  are attention coefficients [13].

**(2)Representation Updating.** After message propagation, each node  $v_i$  will update its representation from  $\mathbf{h}_i^b$  to  $\mathbf{h}_i^l = \text{GRU}(\mathbf{h}_i^{self}, \mathbf{h}_i^b)$  with a gate recurrent unit (abbr., GRU) [3].

After updating node feature  $L$  steps, we will obtain the graph-aware representation  $\mathbf{h}_i^L$  for each node  $v_i$ .

**Leaping Connection Module.** Without introducing any additional parameters, we selectively adopt max-pooling as the core of the LC module like Fig. 3, which can balance the contradiction between training consumption and over-smoothing. We can get the final representation  $\mathbf{h}_v^{final} = \text{MaxPooling}(\mathbf{h}_v^1, \mathbf{h}_v^2, \dots, \mathbf{h}_v^L)$  through this module. Where  $\{\mathbf{h}_v^1, \mathbf{h}_v^2, \dots, \mathbf{h}_v^L\}$  means the representation of each node at each layer.

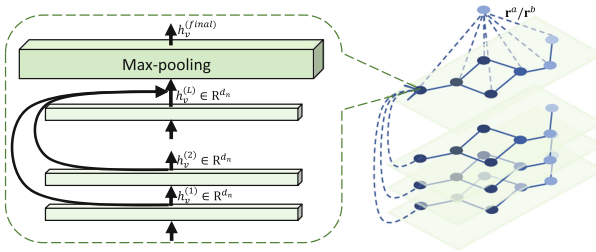


Fig. 3. Leaping connection module.

For each text  $S^a$  or  $S^b$ , the text representation vector  $\mathbf{r}^a$  or  $\mathbf{r}^b$  is obtained by attentive-pooling which can compute the representations of all nodes in each graph.

**Semantic Classifier.** With two text vectors  $\mathbf{r}^a, \mathbf{r}^b$ , and the vector  $\mathbf{c}^{CLS}$  obtained by BERT, our model will predict the similarity of two texts, the training object is to minimize the binary cross-entropy loss.



## 4 Experiment

### 4.1 Experimental Setup

**Dataset.** We conduct experiments on three Chinese datasets for the Chinese short text semantic classification task: LCQMC [8], BQ [1] and ATEC. ATEC is the semantic similarity learning contest data set provided by Ant Financial Services Group. The sample in all datasets contains a pair of texts and a binary label indicating whether the two texts have the same meaning or share the same intention. The statistics of the datasets is shown in Table 1.

**Table 1.** Features of three datasets

Dataset	Size	Pos:Neg	Domain
LCQMC	260068	1.3:1	Open-domain
BQ	120000	1:1	Bank
ATEC	100000	4:1	Finance

**Hyper-parameters.** The number of neighborhood interaction graph updating layers  $L$  is 3 on both datasets. The dimensions of both word representation and hidden size are 128. The model is trained by AdamW with an initial learning rate of 0.0002 and a warmup rate of 0.1. The learning rate of the BERT layer is multiplied by an additional factor of 0.1. As for batch size, we use 32 for all datasets. The dropout was applied after the word and character embedding layers with a keep rate of 0.3. It was also applied before the fully connected layers with a keep rate of 0.5. Moreover, the patience number is 4.

**Environment Settings.** Our model is constructed by python3.7, with the help of the PyTorch framework. All the following experiments are conducted on one CentOS server with two Intel Xeon 2.2 GHz CPUs, 128 G RAM, and one RTX 2080Ti GPU. The input word lattice graphs are produced by the combination of three segmentation tools: jieba<sup>1</sup> and HanNLP<sup>2</sup>.

### 4.2 Evaluation Metrics and Baseline

**Evaluation Metrics.** For each dataset, the accuracy (abbr., ACC.) and F1 score are used as the evaluation metrics. ACC. is the percentage of correctly classified examples. F1 score of matching is the harmonic mean of the precision and recall.

**Baseline.** We compare our model with several BERT-based models pre-trained on large-scale corpora. **Bert-base** [5] is the official Chinese BERT model released

<sup>1</sup> <https://pypi.org/project/jieba/>.

<sup>2</sup> <https://pypi.org/project/hanlp/>.

by Google. It discards the traditional RNN and CNN, and converts the distance of two words at any position to 1 through the attention mechanism. **ERNIE** [12] is designed to learn language representation enhanced by knowledge masking strategies, which include entity-level masking and phrase-level masking. **BERT-wwm** [4] is a Chinese BERT, which was trained on the latest Chinese Wikipedia dump and adapt whole word masking in Chinese text. **BERT-wwm-ext** [4] is a variant of BERT-wwm with more training data and training steps. **ERNIE2.0** [11] is an upgraded version of ERNIE, proposing a mechanism for continual learning. **Roberta** [9] is an enhanced version of BERT that modifies key hyper-parameters, eliminates the pre-training target for the next sentence, and trains with larger mini-batches and learning rates.

### 4.3 Result and Analysis

From Table 2, we find that BERT variants all outperform the original one, which indicates that using word-level information in pre-training is crucial for Chinese text classification. Our model GLCN-BERT performs better than almost all these BERT-based models. It demonstrates that using word-level information and different fusion methods in the fine-tuning stage effectively boosts performance. It can even rival larger models with larger corpus and training time.

**Table 2.** Performance of various models on LCQMC, BQ and ATEC test datasets.

Models	LCQMC		BQ		ATEC	
	ACC.	F1	ACC.	F1	ACC.	F1
BERT [5]	85.7	86.8	84.5	84.0	88.1	88.7
BERT-wwm [4]	86.8	87.8	84.9	84.3	88.3	88.6
BERT-wwm-ext [4]	86.7	87.7	83.9	84.7	88.2	88.5
ERNIE [12]	87.0	87.9	84.7	84.2	88.5	88.9
ERNIE2.0 [11]	<b>87.9</b>	-	85.0	-	89.0	-
Roberta [9]	87.2	-	84.7	-	88.8	-
<b>GLCN-BERT(Our)</b>	<b>87.9</b>	<b>88.7</b>	<b>85.3</b>	<b>85.1</b>	<b>89.2</b>	<b>89.4</b>

In addition, as shown in Fig. 4, using the leaping connection method significantly improves the performance of the model for all three datasets. It indicates that our model can aggregate the information of the node itself and neighbor nodes well. This may be since the short text contains a short sequence of contexts. As the depth increases, the expansion of the node aggregation range leads to each node containing too much global information, which can easily lead to overfitting. Our model takes into account the problem and avoids it effectively.

Finally, Fig. 5 shows the results when we set the early stop value to 3 (training will stop when the best result is not exceeded three times in a row). Thus, we could know that for short sequences of text pairs, a small number of epochs already tend to achieve a good result.

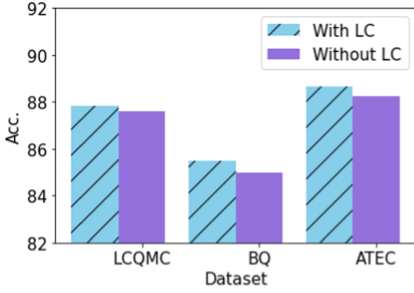


Fig. 4. Test accuracy.

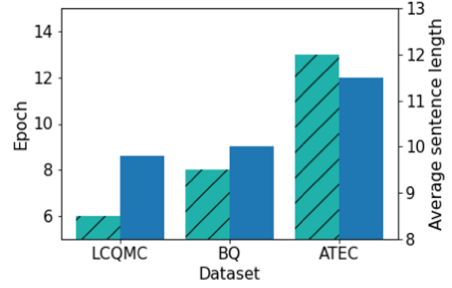


Fig. 5. Early stopping epochs and average text length.

## 5 Conclusion and Future Work

In this work, we propose a Graph Attentive Leaping Connection Network (GLCN-BERT) for Chinese short text classification. Our model takes two word lattice graphs and utilizes a graph attention network structure to obtain information from each layer. Then the leaping connection method is used to aggregate the information flexibly while avoiding overfitting. The proposed approach is evaluated on three Chinese benchmark datasets and achieves the best performance. Extensive experiments also demonstrate that both semantic information and multi-granularity information are essential for text classification modeling.

In the future, we will further investigate the effect of the network depth on text classification and introduce external knowledge, such as paraphrase database, to help learn more accurate and robust text representation.

## References

1. Chen, J.J., et al.: The BQ corpus: a large-scale domain-specific chinese corpus for sentence semantic equivalence identification. In: EMNLP (2018)
2. Chen, L., et al.: Neural graph matching networks for chinese short text matching. In: ACL (2020)
3. Chung, J., et al.: Empirical evaluation of gated recurrent neural networks on sequence modeling. In: ArXiv abs/1412.3555 (2014)
4. Cui, Y., et al.: Pre-training with whole word masking for chinese BERT. In: ArXiv abs/1906.08101 (2019)
5. Devlin, J., et al.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (2019)
6. Jiang, J.-Y., et al.: semantic text matching for long-form documents. In: The World Wide Web Conference (2019)
7. Li, Y., et al.: Enhancing pre-trained chinese character representation with word-aligned attention. In: ArXiv abs/1911.02821 (2020)
8. Liu, X., et al.: LCQMC: a large-scale chinese question matching corpus. In: COLING (2018)

9. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach. In: ArXiv abs/1907.11692 (2019)
10. Minaee, S., et al.: Deep learning-based text classification: a comprehensive review. *ACM Comput. Surv.* **54**(3), 62:1-62:40 (2021). <https://doi.org/10.1145/3439726>
11. Sun, Y., et al.: ERNIE 2.0: a continual pre-training framework for language understanding. In: ArXiv abs/1907.12412 (2020)
12. Sun, Y., et al.: ERNIE: enhanced representation through knowledge integration. In: ArXiv abs/1904.09223 (2019)
13. Velickovic, P., et al.: Graph attention networks. In: ArXiv abs/1710.10903 (2018)
14. Wang, Z., Hamza, W., Florian, R.: Bilateral multi-perspective matching for natural language sentences. In: ArXiv abs/1702.03814 (2017)
15. Wang, Z., et al.: Match<sup>2</sup>: a matching over matching model for similar question identification. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (2020)
16. Xu, K., et al.: How powerful are graph neural networks?. In: ArXivabs/1810.00826 (2019)
17. Yang, Y., Yih, W.-t., Meek, C.: WikiQA: a challenge dataset for open-domain question answering. In: EMNLP (2015)
18. Ying, R., et al.: Hierarchical graph representation learning with differentiable pooling. In: NeurIPS (2018)

# Graph Query



# Graph Ordering: Towards the Optimal by Learning

Kangfei Zhao<sup>1</sup>(✉), Yu Rong<sup>2</sup>, Jeffrey Xu Yu<sup>1</sup>, Wenbing Huang<sup>3</sup>,  
Junzhou Huang<sup>4</sup>, and Hao Zhang<sup>1</sup>

<sup>1</sup> The Chinese University of Hong Kong, Hong Kong S.A.R., China  
{kfzhao,yu,hzhang}@se.cuhk.edu.hk

<sup>2</sup> Tencent AI Lab, Shenzhen, China

<sup>3</sup> Tsinghua University, Shaoyang, China

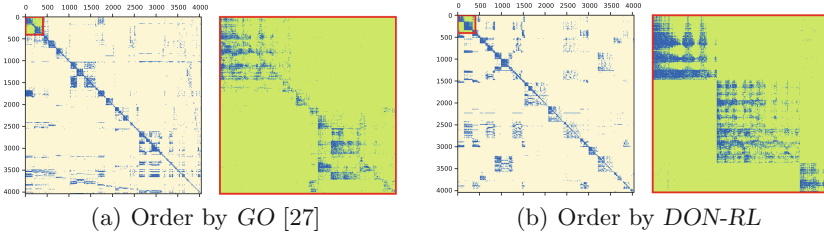
<sup>4</sup> University of Texas at Arlington, Arlington, US  
jzhuang@uta.edu

**Abstract.** Graph ordering concentrates on optimizing graph layouts, which has a wide range of real applications. As an NP-hard problem, traditional approaches solve it via greedy algorithms. To overcome the short-sightedness and inflexibility of the hand-crafted heuristics, we propose a learning-based framework: Deep Ordering Network with Reinforcement Learning (*DON-RL*) to capture the hidden structure from partial vertex order sets over a specific large graph. In *DON-RL*, we propose a permutation invariant neural network *DON* to encode the information from partial vertex order. Furthermore, to alleviate the combinatorial explosion for partial vertex order sets and make the efficient training data sampling, we propose *RL-Sampler*, a reinforcement learning-based sampler to tune the vertex sampling probabilities adaptively during the training phase of *DON*. Comprehensive experiments on both synthetic and real graphs validate that our approach outperforms the state-of-the-art heuristic algorithm consistently. The case study on graph compression demonstrates the potentials of *DON-RL* in real applications.

**Keywords:** Graph ordering · Deep learning · Reinforcement learning

## 1 Introduction

A variety of tasks, such as social network analysis and user recommendation, are derived in the form of graphs—a kind of particular structures that model data points as nodes and the relationships among them as edges. In this paper, we concentrate ourselves on a challenging task: graph ordering. As its name implies, graph ordering aims at permuting a desired order of nodes, in order to facilitate certain down-stream applications, including graph visualization, graph compression, and graph storage optimization. Specifically, [5] gives a comprehensive survey on reordering for graph visualization. Graph compression is to find a compact edge layout by placing a node in a specific way, whose ordering



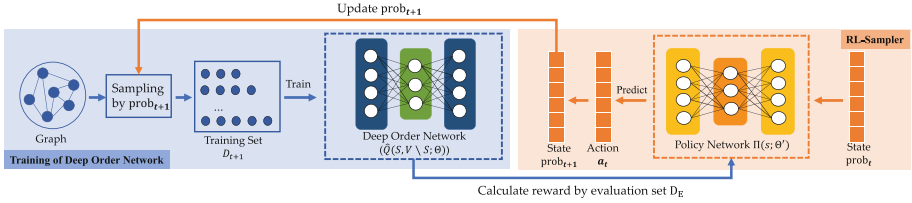
**Fig. 1.** Matrix visualization for the reordered facebook

algorithm can be heuristics based [15], clustering based [8,9] and optimization based [14]. Graph storage optimization is to generate an optimized layout to improve the data locality [3,4]. All of them will be solved more elegantly if we can arrange nodes in a good order beforehand, as illustrated in Fig. 1(b).

Thus far, the state-of-the-art graph ordering approach [27] is based on the greedy algorithm. [27] reassigns every vertex a unique number in  $[1, n]$  ( $n$  is the number of vertices in the graph) by maximizing a cumulated locality score function for vertices in a sliding window of size  $w$  ( $> 1$ ). Since this is an NP-hard combinatorial optimization problem as a general form of the maximum Travel Salesman Problem (TSP) ( $w = 1$ ), [27] proposed to use a hand-crafted heuristic to build a solution greedily. However, it fails to capture the combinatorial complexity and is inflexible to deal with diversified graphs.

In this paper, we approach graph ordering in a novel sense: finding the optimal by learning. Since designing advanced heuristics requires much expert knowledge and hands-on experiences, we attempt to learn a heuristic from one given graph directly. To the best of our knowledge, we are the first to solve the graph ordering problem by making use of learning approaches. Concretely, distinguished from the traditional heuristics-based algorithms, we propose a neural network based model: Deep Order Network (*DON*) to replace the predefined evaluation function to capture the complex implicit locality in graphs. The strongly expressive power of neural networks drives us to learn a more accurate evaluation function against conventional hard-craft ones. Furthermore, to address the combinatorial explosion in the training space of *DON*, we exploit the idea of reinforcement learning and construct *RL-Sampler* to tune the sampling probability for the training set in an adaptive manner. *RL-Sampler* enables *DON* to focus on vertices of more importance in the solution and directs *DON* towards a more effective model. The overall learning framework is called *DON-RL*. We demonstrate it in Fig. 1, which shows the matrix visualization for the Facebook [19] reordered by the *GO* algorithm and our new learning-based approach over graph. We conduct extensive experiments on both synthetic and real-world graphs with different properties. The experimental results show that *DON-RL* consistently outperforms state-of-the-art algorithms. A case study of graph compression is also conducted to demonstrate the potential usage of our approach in real applications<sup>1</sup>.

<sup>1</sup> Our detailed technical report can be found in <https://arxiv.org/abs/2001.06631>.



**Fig. 2.** The overview of *DON-RL*. *DON-RL* contains two components. One is in the left/blue box named *DON* which is the parameterized model to approximate  $\mathcal{Q}^*(S, V \setminus S)$ . The other is in the right/orange box named *RL-Sampler* which updates the sample probability of training sets adaptively. The training of the two networks is performed interactively.

## 2 Graph Ordering

Given a graph  $G = (V, E)$  where  $V(G)$  is a set of vertices and  $E(G)$  is a set of edges of  $G$ , the **graph ordering** problem is to *find the optimal order for all vertices*. In brief, let  $\Phi(\cdot)$  be a permutation function which assigns a vertex a unique number in  $[1..n]$  where  $n = |V(G)|$ . The problem is to find the optimal graph ordering  $\Phi(\cdot)$  by maximizing a cumulated locality score  $F(\cdot)$  (Eq. (1)) over a sliding window of size  $w$ .

$$F(\Phi) = \sum_{0 < \Phi(v) - \Phi(u) \leq w} \xi(u, v) \tag{1}$$

Here,  $u, v$  are two vertices ordered within a window of size  $w$  in the permutation, and  $\xi(u, v)$  is a pair-wise similarity function to measure the closeness of  $u$  and  $v$ . In [27],  $\xi(u, v)$  is defined as  $\xi(u, v) = \xi_s(u, v) + \xi_n(u, v)$ , where  $\xi_s(u, v)$  is the number of times that  $u$  and  $v$  are sibling, and  $\xi_n(u, v)$  is the number of times that  $u$  and  $v$  are neighbors. As proved in [27], maximizing  $F(\Phi)$  is a variant of maximum TSP (for  $w = 1$ ) and thus is NP-hard.

Furthermore, [27] proposed an iterative scheme, as sketched in Algorithm 1, which iteratively extends a partial solution  $S$  by a vertex  $v^*$  according to an evaluation function  $\mathcal{Q}^*(S, V \setminus S)$ . [27] designs a hand-crafted  $\mathcal{Q}(S, w, V \setminus S)$  which considers the information within  $w - 1$  window to approximate  $\mathcal{Q}^*(S, V \setminus S)$ :

$$\mathcal{Q}(S, w, V \setminus S) = k(S_{w-1}, v) = \sum_{j=\max\{1, i-w\}}^{i-1} \xi(v_j, v) \tag{2}$$

where  $k(S_{w-1}, v)$  measures the increase of  $F(\Phi)$  inserting the  $i$ -th vertex in  $S$ .  $S_{w-1}$  is the last  $\min\{i, w - 1\}$  inserted vertices of  $S$ . When applying Eq. (2) in Algorithm 1, we can observe that  $\mathcal{Q}(S, w, V \setminus S)$  only captures the local information within  $w$  window to the ordered vertices. However, since the input space of  $\mathcal{Q}^*(S, V \setminus S)$  is all partial permutations of  $V$ , it is hard to approximate by a simple hand-crafted  $\mathcal{Q}(S, w, V \setminus S)$ . Therefore, to modeling  $\mathcal{Q}^*(S, V \setminus S)$ , we adopt a parameterized model  $\hat{\mathcal{Q}}(S, V \setminus S; \Theta)$  to pursue a better approximation.



---

**Algorithm 1.**  $GO(G = (V, E))$ 

---

```

1:  $S \leftarrow \emptyset$ ;
2: while  $|S| \neq |V|$  do
3:    $v^* \leftarrow \arg \max_{v \in V \setminus S} Q(S, w, V \setminus S)$ ;
4:    $S \leftarrow S \cup \{v^*\}$ ;
5: end while
6: return  $S$ ;

```

---

Like deep Q-learning [20], it is possible to employ sampled partial solutions from graphs to train  $\hat{Q}(S, V \setminus S; \Theta)$ . In this vein, we define the graph ordering as a learning problem:

**Definition 1.** *Given a graph  $G = (V, E)$ , the graph ordering problem is to train a parameterized model  $\hat{Q}(S, V \setminus S; \Theta)$  with sampled partial solution set  $D$  to generate a graph ordering  $\Phi(\cdot)$  by maximizing  $F(\Phi)$  (Eq. (1)) according to Algorithm 1.*

### 3 Methodologies

To attack the graph ordering problem by machine learning, we propose a new framework: Deep Ordering Network with Reinforcement Learning (*DON-RL*) to capture the complex combinatorial structures of a single large graph. *DON-RL* contains two key components: Deep Order Network (*DON*) which aims to make the better approximation of the evaluation function  $Q^*(S, V \setminus S)$  and Reinforcement Learning-based Sampler (*RL-Sampler*) which samples valuable partial solutions of graph order efficiently. Figure 2 demonstrates the overall learning framework.

#### 3.1 Deep Order Network

In this section, we propose a model, Deep Order Network (*DON*), as a parameterized evaluation function  $\hat{Q}(S, V \setminus S; \Theta)$  to replace the evaluation function  $Q(S, w, V \setminus S)$  in Algorithm 1. Concretely, this problem can be formulated as a classification problem:  $\hat{Q}(S, V \setminus S; \Theta)$  takes the a vectorized representation of partial solution  $S$  as input, outputs a vector  $\hat{\mathbf{p}}_S \in \mathbb{R}^N$  where  $\hat{\mathbf{p}}_S(v_i)$  represents the likelihood of adding a vertex  $v_i$  to the permutation, given the current partial solution  $S$ . Hence, the prediction is made by  $v^* = \arg \max \hat{\mathbf{p}}_S$  and  $\hat{Q}(S, V \setminus S; \Theta)$  being learned can replace  $Q(S, w, V \setminus S)$  of Algorithm 1 in a seamlessly way. Since the input of  $\hat{Q}(S, V \setminus S; \Theta)$  is a set, we need guarantee the permutation invariant of the vectorized representation of  $S$ . Therefore, we employ the permutation invariant neural network: *DeepSets* [29] to model  $\hat{Q}(S, V \setminus S; \Theta)$ . Concretely, we take the one-hot representation of vertex in  $S$  as input. The output  $\hat{\mathbf{p}}_S$  can be defined as:

$$\hat{\mathbf{p}}_S = \rho(\text{Agg}_{v_i \in S}(\phi(v_i))), \quad (3)$$

where  $\phi(\cdot) : \mathbb{R}^N \rightarrow \mathbb{R}^K$  and  $\rho(\cdot) : \mathbb{R}^K \rightarrow \mathbb{R}^N$  are two transformation functions. Here we employ deep neural network as the backbone of  $\phi(\cdot)$  and  $\rho(\cdot)$ . Intuitively,  $\phi(\cdot)$  generates the element-wise representation for each  $v_i$ . Then,  $\text{Agg}(\cdot)$  performs a generalization of classic aggregation functions, e.g., SUM, AVG and MAX on the set  $S$ . Finally, the aggregated representation of  $S$  is processed by  $\rho(\cdot)$  to produce the final result  $\hat{\mathbf{p}}_S$ . The model is trained by optimizing the cross-entropy loss over a training set  $D$  as given in Eq. (4), where  $p(v_i|S)$  is the target distribution.

$$\mathcal{L} = \frac{1}{|D|} \sum_{S \in D} \sum_{v_i \in V} -p(v_i|S) \log(\hat{\mathbf{p}}_S(v_i)). \quad (4)$$

Considering the marginal distribution of any partial solution  $S$ , which indicates how probable  $S$  appears in the final solution, we have

$$p(S) \propto F(\Phi(S)) \quad (5)$$

The target distribution can be computed explicitly as Eq. (6):

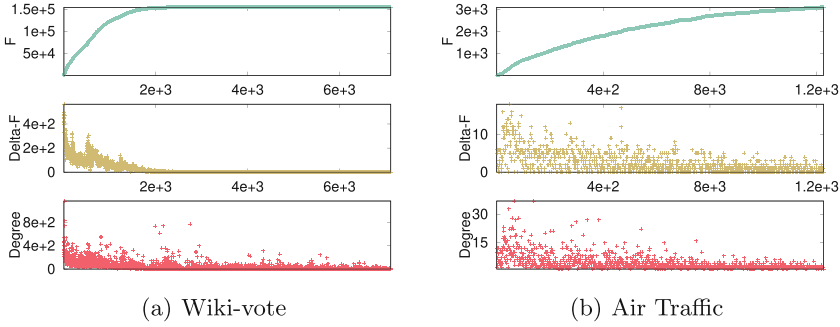
$$p(v_i|S) = \frac{p(v_i, S)}{p(S)} = \frac{F(\Phi(S \cup \{v_i\}))}{\sum_{v_i \in V \setminus S} F(\Phi(S \cup \{v_i\}))} \quad (6)$$

At the first glance, the maximal probability of  $p(v_i|S)$  in Eq. (6) reflects the same ordering preference to the original greedy algorithm. However, for  $S$  within a window,  $F(\Phi(S \cup v)) = F(\Phi(S)) + k(S, v)$ . *DON* models  $F(\Phi(S))$  in Eq. (6) for its input  $S$ , which is regarded as a constant in the greedy algorithm. The term  $F(\Phi(S))$  is a part of the combinational structure that enables generalization for unseen combinations via learning from multiple partial solutions in a parameter-sharing way. Our method can be regarded as an implicit graph representation learning approach where *DON* embeds the combinational structure of  $F$ -score defined in Eq. (6). When applying greedy search over *DON*, a high-quality solution can be constructed by set expansion towards a high-quality solution of in the vector space.

In the training phase, the training set  $D$  is generated by sampling partial solutions of size  $w - 1$ , where each  $S$  is sampled from a biased distribution  $\text{prob} \in \mathbb{R}^N$  on vertex set. This  $\text{prob}$  can be chosen by some heuristics, e.g., the vertex degree distribution. However, due to the complex combinatorial nature of this problem, this greedy and static sampling strategy can not sample the representative partial solutions to train the model. We give the computational complexity. When the pair-wise similarity  $\xi$  is pre-computed offline, the training complexity is  $O(sb_swh|V|)$  and the complexity of solution construction is  $O(wh|V|^2)$ , where  $s$  is the number of training steps,  $b_s$  is the batch size,  $h$  is the number of hidden units of *DON*.

### 3.2 RL-Sampler

In this section, we discuss how we adjust the sampling probability  $\text{prob}$  adaptively. As shown in Sect. 3.1, the feasible input of  $\mathcal{Q}(S, V \setminus S; \Theta)$  is all possible



**Fig. 3.** The  $F$  score, the increment of  $F$  score and the vertex degree of the permutation results of  $GO$  on two real graphs.

combinations of the vertices, which indicates that it is intractable to enumerate all the partial solutions in training  $DON$ . To alleviate the combinatorial explosion issue, we employ the sampling strategy in the training space to reduce the candidate partial solutions.

By intuition, high-degree vertices sharing many common neighbors play a significant role in constructing high-quality feasible solutions. However, since the graph ordering is a discrete optimization problem, the significance of a discrete element that contributes to a solution can be very different from others, and there exists skewness regarding such significance among all elements.

To validate this claim, we report the permutation results of two real graphs: Wiki-vote [19] and Air Traffic [1] in Fig. 3. Here, the horizontal axis is the linear graph ordering. In vertical, we show the  $F$  score, the increment of  $F$ , and the degree of the vertex from top to bottom in three subfigures, respectively. As shown in Fig. 3 we can observe that the patterns of the two solutions are very different. For Wiki-vote, over  $2/3$  vertices in the ordering almost make no contribution to enlarging the  $F$  score, while for air traffic, as vertex adding into the permutation, the  $F$  score increases continuously. Interestingly, it is counterintuitive that the high-degree vertices do not always make a significant contribution to increasing the  $F$  score. This result inspires us that what if adjusting the sampling probability dynamically during the training phase. This further motivates us to use Reinforcement Learning framework to tune sampling probability. Therefore, we propose *RL-Sampler*, a Policy Network to tune the sampling probability. We model this adaptive tuning in an Markov Decision Process as follows:

**States:** a state  $s$  is a vector of sampling probability  $\mathbf{prob} \in (0, 1)^n$ ,  $\mathbf{prob} = \{\mathbf{prob}(v_i)\}_{i=1}^n$  where  $n$  is the number of vertices.  $\mathbf{prob}(v_i)$  is the probability of sampling vertex  $v_i$  for one training instance and  $\sum_{i=1}^n \mathbf{prob}(v_i) = 1$ . We employ the degree distribution as the initialization of  $\mathbf{prob}_0$ .

**Actions:** an action is a vector of 0–1,  $\mathbf{a} = \{a(i)\}_{i=1}^n \in \{0, 1\}^n$ . Each element  $a(i)$  denotes a tuning action imposed on  $\mathbf{prob}(v_i)$ , i.e., increase (0) or decrease (1). There is a tuning rate  $\lambda \in \mathbb{R}$  to control the constant delta changes of each

**Algorithm 2.** Train the Policy Network  $\Pi(s; \Theta')$ 

- 
- 1: **Input:** graph  $G$ , initial state  $\mathbf{prob}_0$ , evaluation set  $D_E$ , learning rate  $\alpha$ , discounting factor  $\gamma$
  - 2: Initialize  $\Pi(s; \Theta')$
  - 3: **for** each RL step **do**
  - 4:   **for**  $t = 0$  **to**  $T$  **do**
  - 5:     Sample  $a_t \propto \pi_{\Theta'}(\mathbf{a}_t | s_t)$ , where  $s_t = \mathbf{prob}_t$
  - 6:     Compute  $\mathbf{prob}_{t+1}$  by  $\mathbf{a}_t$  and normalization
  - 7:     Train and update  $\hat{Q}(S, V \setminus S; \Theta)$  based on  $D_{t+1}$ , where  $D_{t+1}$  is sampled from  $G$  by  $\mathbf{prob}_{t+1}$
  - 8:     Evaluate  $\hat{Q}(S, V \setminus S; \Theta)$ , and receive the reward  $r_t$  computed on  $D_E$
  - 9:   **end for**
  - 10: **for**  $t = 0$  **to**  $T$  **do**
  - 11:    Compute cumulated reward  $R_t = \sum_{i=t}^T \gamma^{i-t} r_i$
  - 12:     $\Theta' \leftarrow \Theta' + \alpha(R_t - b(s_t)) \nabla \log \pi(s_t)$
  - 13: **end for**
  - 14: **end for**
  - 15: **Output:** the Policy Network  $\Pi(s; \Theta')$
- 

element  $\mathbf{prob}(v_i)$  for a given  $a(i)$ . The adjustment is specified in Eq. (7). The final step of an adjustment is the  $L1$  normalization of  $\mathbf{prob}$ .

$$\mathbf{prob}(v_i) = \begin{cases} \mathbf{prob}(v_i) + \lambda, & a(i) = 0 \\ \mathbf{prob}(v_i) - \lambda, & a(i) = 1 \end{cases} \quad (7)$$

**Transition:** given the state-action pair  $(\mathbf{prob}_t, \mathbf{a}_t)$  at time  $t$ , the transition to next state  $\mathbf{prob}_{t+1}$ , is deterministic.

**Reward:** the reward  $r_t = r(\mathbf{prob}_t, \mathbf{a}_t)$  reflects the benefits of updating a state  $\mathbf{prob}_t$  by an action  $\mathbf{a}_t$  at time  $t$ . We use the evaluation results of *DON* after a certain number of steps on an evaluation set  $D_E$  periodicity as the reward. To alleviate the overfitting issue, we fix the sampling probability to  $\mathbf{prob}_0$  to generate  $D_E$ . The cumulative reward is computed by adding future reward discounted by a discounting factor  $\gamma \in [0, 1]$ .

**Policy:** A policy function  $\pi(\mathbf{a}_t | \mathbf{prob}_t)$  specifies a tuning operation  $\mathbf{a}_t$  to perform on a given state  $\mathbf{prob}_t$ . Here we employ a parametric function  $\pi_{\Theta'}(\mathbf{a}_t | \mathbf{prob}_t)$  with parameter  $\Theta'$  to analog output of the policy. This model can be any multi-label classification model [24], e.g., multilayer perceptron (MLP), which takes  $\mathbf{prob}$  as input and outputs the probability of an action  $\mathbf{a}$ . We use  $\pi_{\Theta'}(\mathbf{a}_t | s_t)$  to denote the output probability at time  $t$  and use  $\Pi(s; \Theta')$  to denote the Policy Network.

We deploy a model-free, policy-based RL algorithm, REINFORCE algorithm [28] to train the Policy Network directly. For the REINFORCE algorithm, the objective is to find the optimal policy  $\pi^*$  by maximizing the following expected accumulated and discounted rewards of Eq. (8), which is estimated by sampling state-action trajectory  $\varsigma (s_0, a_0, r_0, s_1, a_1, \dots)$  of  $T$  steps.

$$J(\Theta') = \mathbb{E}[\sum_{t=0}^{T-1} \gamma^t r_t | \pi_{\Theta'}] = \mathbb{E}_{s \sim \pi_{\Theta'}(s)}[\sum_{t=0}^{T-1} \gamma^t r_t] \quad (8)$$

The gradient of the Eq. (8) is formulated in Eq. (9).

$$\nabla J(\Theta') = \mathbb{E}_{s \sim \pi_{\Theta'}(s)}[\nabla \log \pi(s) (\sum_{t=0}^{T-1} \gamma^t r_t - b(s))] \quad (9)$$

Here, as the raw reward of a trajectory,  $\sum_{t=0}^{T-1} \gamma^t r_t$ , is always positive, a baseline function  $b(s)$  is introduced to reduce the variance of the gradients. The formula  $\sum_{t=0}^{T-1} \gamma^t r_t - b(s)$  indicates whether a reward is better or worse than the expected value we should get from state  $s$ . For the baseline function  $b(s)$ , we adopt the widely-used moving average of the historical rewards, i.e., the evaluation result of *DON*. The training algorithm of the Policy Network is given in Algorithm 2. The algorithm takes a graph  $G$ , an initial and experience-based sampling probability  $\mathbf{prob}_0$ , the evaluation set  $D_E$ , and the learning rate of RL as input, and trains Policy Network  $\Pi(s; \Theta')$  by interacting with the training process of Deep Order Network  $\hat{Q}(S, V \setminus S; \Theta)$ . The intuition is, if the reward of an action is positive, it indicates the action is good and the gradients should be applied to make the action even more likely to be chosen in the future. However, if the reward is negative, it indicates the action is bad and the opposite gradients should be applied to make this action less likely in the future.

We explain the algorithm below. First, Policy Network  $\Pi(s; \Theta')$  is initialized (line 2). The training of policy gradient starts at line 3. Before that, the training of *DON* has started several steps to collect enough evaluation results for computing the baseline. In each RL step, the algorithm updates  $\Pi(s; \Theta')$  by one Monte Carlo sampling of a trajectory of length  $T$  (line 4). For each time  $t$ , it repeats the following 4 steps one by one. First, draw a random action  $\mathbf{a}_t$  based on the probability output by Policy Network, i.e.,  $\pi_{\Theta'}(\mathbf{a}_t | \mathbf{prob}_t)$  (line 5). Instead of directly choosing the action with the highest probability, the random action manages a balance between exploring new actions and exploiting the actions which are learned to work well. Second, apply the action  $\mathbf{a}_t$  on the state  $\mathbf{prob}_t$  by imposing an increment/decrement of  $\lambda$  (line 6), to generate the next state  $\mathbf{prob}_t$ . Third, generate the training data set  $D_{t+1}$  of *DON* by sampling  $G$  with probability  $\mathbf{prob}_{t+1}$ , and feed  $D_{t+1}$  to train  $\hat{Q}(S, V \setminus S; \Theta)$  (line 7). Fourth, after training  $\hat{Q}(S, V \setminus S; \Theta)$  in a certain number of steps, we use the evaluation set  $D_E$  to evaluate *DON*, and collect the evaluation result as reward  $r_t$  (line 8). When a trajectory is simulated, we compute the cumulative rewards (line 11) and apply gradient ascent to update the Policy Network  $\Pi(s; \Theta')$  (line 12).

**Evaluation of *DON*-RL.** The evaluation set  $D_E$  is generated by randomly choosing the first vertex with the degree distribution,  $\mathbf{prob}_0$ , and then extending it to a partial solution with size  $w - 1$  by the best neighbor heuristic, and computing the ground truth of the partial solution.  $D_E$  is generated at the beginning of the training and is used during the training phase. The RMSE for the evaluation set is given in Eq. (10).

$$\mathcal{L}(D_E) = \frac{1}{|D_E|} \sum_{S \in D_E} \sqrt{\frac{1}{|V|} \sum_{v_i \in V} (p(v_i | S) - \hat{\mathbf{p}}_S(v_i))^2} \quad (10)$$

**Table 1.** Datasets

<b>Graphs</b>	<b> V </b>	<b> E </b>	<b>Density</b>	<b>Description</b>
Wiki-vote	7,115	103,689	0.0020	vote graph
Facebook	4,039	88,234	0.0108	social graph
p2p	6,301	20,777	0.0005	file sharing graph
Arxiv-HEP	9,877	25,998	0.0003	co-authorship graph
Cora	23,166	91,500	0.0001	citation graph
PPI	21,557	342,353	0.0015	biological graph
PL10K_1.6	10,000	121,922	0.0024	power-law graph
PL10K_1.8	10,000	58,934	0.0011	
PL10K_2.0	10,000	31,894	0.0006	
ER10K_0.02	10,000	1,000,337	0.0200	ER graph
ER10K_0.05	10,000	2,498,836	0.0500	
ER10K_0.1	10,000	5,004,331	0.1000	

**Table 2.** The hyper-parameter configuration

<b>Hyper-parameters</b>		<b>Values</b>
<i>DON</i>	learning rate	$10^{-3} \sim 10^{-4}$
	mini-batch size	64 $\sim$ 512
	# hidden units	32, 64, 128, 256
	global steps	$5 \times 10^3 \sim \times 10^6$
Policy network	learning rate	$10^{-3} \sim 10^{-5}$
	trajectory length $T$	1 $\sim$ 10
	RL steps	50 $\sim$ 300
	discounting factor $\gamma$	0.9, 0.95
	tuning rate $\lambda$	$0.1n \sim 0.2n$
	# hidden units	32, 64, 128, 256
	& evaluation set	2000, 5000

## 4 Experimental Studies

In this section, we present our experimental evaluations. First, we give the specific setting of the testing. Then, we compare the proposed model with the state-of-the-art algorithmic heuristic, conduct an A/B testing to validate the effect of *RL-Sampler*, and observe the performance of models as  $w$  varies. Finally, a case study of compressing graph is presented.

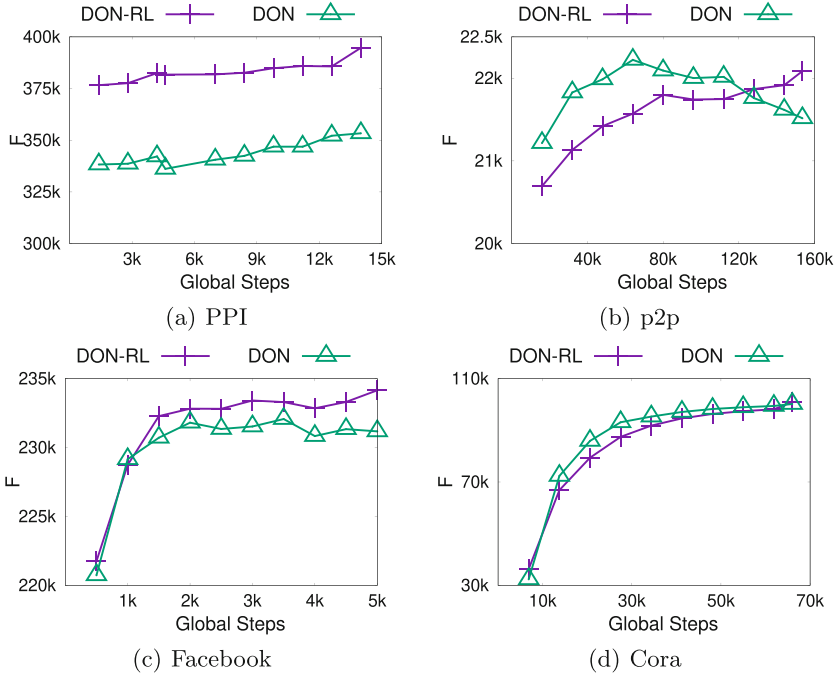
**Table 3.** Result of graph ordering

$F(\Phi)$	$ V' $	$GO$	$DON$	$DON-RL$
Wiki-vote	5,880	145,736	156,871	<b>157,669</b>
Facebook	3,974	230,031	207,511	<b>234,151</b>
p2p	5,624	20,472	21,422	<b>22,086</b>
Arxiv-HEP	9,877	85,958	90,629	<b>91,001</b>
Cora	22,317	98,334	<b>101,063</b>	100,966
PPI	19,041	383,343	347,237	<b>404,728</b>
PL10K_1.6	8,882	166,540	190,021	<b>197,622</b>
PL10K_1.8	8,292	66,272	86,840	<b>89,930</b>
PL10K_2.0	8,484	29,373	37,332	<b>38,071</b>
ER10K_0.02	10,000	136,925	145,084	<b>162,462</b>
ER10K_0.05	10,000	615,706	673,357	<b>724,743</b>
ER10K_0.1	10,000	2,319,250	2,554,032	<b>2,647,686</b>

**Datasets:** We use six real graphs collected from SNAP [19] and KONECT [1] and six synthetic graphs generated by SNAP random graph generator: power-law and Erdős-Rényi. Table 1 summarizes the information of these real and synthetic graphs.

**Model settings:** In  $DON$ , both  $\phi$  and  $\rho$  networks are two-layer MLP. We use SUM as the pooling layer to add up the representations of  $\phi$ . The input of  $\phi(\cdot)$  is the one-hot representation,  $\{0, 1\}^{m \times n}$ , of a partial solution with size  $m$ , and the output,  $\mathbf{p}_S \in (0, 1)^n$ , is generated by a softmax function. All the hidden units are activated by ReLU function. In  $RL-Sampler$ , the Policy Network  $\Pi(s; \Theta')$  is a multi-label classifier of two-layers MLP. The input is the state  $\mathbf{prob}_t \in (0, 1)^n$  and the output layer predicts the probability to perform action  $\mathbf{a}_t$ , i.e., a vector  $(0, 1)^n$  by sigmoid activation. The hidden units are activated by ReLU function. For computing the rewards, we use RMSE as the evaluation metric of the evaluation set. The reward is defined as the opposite number of RMSE. Table 2 shows the hyper-parameters settings in the training.

**Exp-1:Result of Graph Ordering.** Table 3 shows the overall results of the  $F$  score of the permutations generated by  $DON$ , compared with the baseline graph ordering algorithm  $GO$  [27]. The window size  $w$  is fixed to 5, which is commonly used in [27]. As a preprocessing step, vertices whose degree is 1 are merged to one equivalent vertex. In Table 3, the column  $|V'|$  is the compacted vertex number of the graph. We can see that this trick compresses up to 17% vertices for real and power-low graphs but it is invalid for three Erdős-Rényi graphs due to their degree distribution. With regards to maximize  $F$ , the solutions of  $DON$  and  $DON-RL$  surpass that of the greedy algorithm significantly. For the real graphs, all the solutions of  $DON-RL$  outperform that of  $GO$  from 1.8% up to



**Fig. 4.** *DON* vs. *DON-RL*

8.2%. And the solution of *DON*, on the 3 real graphs, Wiki-Vote, p2p and Cora, the improvements of  $F$  are 7.6%, 4.7% and 2.8%, respectively. For the synthetic graphs, both *DON* and *DON-RL* can generate solutions of much higher  $F$  score than *GO*. The *DON-RL* performs best and its solutions surpass that of *GO* up to 35.0% for powerlaw graph, and 18.6% for ER graph. It is easy for *GO* to stuck in forming local dense areas while incurring sparsity in the global scope. This phenomenon agrees with our intuition, that is, the greedy algorithm could neglect better solutions in the future scope.

**Exp-2: Adaptive Training by RL.** We conduct ablation study to observe the effectiveness of *RL-Sampler*, i.e., using the same Base Network hyper-parameters to train a bare *DON* model and a conjuncted *DON-RL* model. For *DON*, we employ the vertex degree distribution as the sampler to sample training data as the partial solutions. During the overall training process, we generate a solution after per 10% of global training steps. We observe the changes of objective function  $F(\Phi)$  as the loss function of the model is not a direct reflection of the objection of this NP problem. Due to the space limitation, we only present the results over 4 real graphs in Fig. 4.

We can observe that the performance of *DON* and *DON-RL* are different among different graphs during training. As shown in Fig. 4(a),4(c), *DON-RL* can achieve better results than *DON* in graphs with high density since the local-



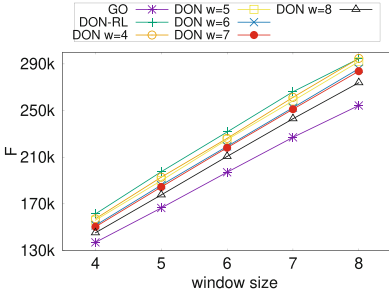


Fig. 5. Varying window size  $w$

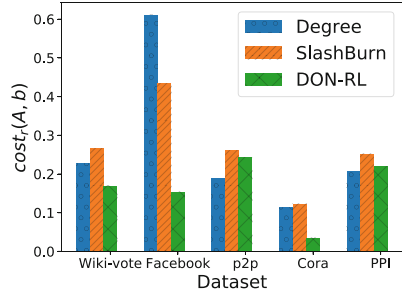


Fig. 6. Compression costs

ity of vertices is significantly skew on the graph. On the other hand, for the graph with low density and flat degree distribution, such as Cora (Fig. 4(d)) and p2p (Fig. 4(b)), using RL does not make significant improvement during the training, since the locality can be well captured by *DON*. However, *DON-RL* can make the training process robust and avoid over-fitting. As shown in Fig. 4(b), *DON* reaches its best effect much earlier than *DON-RL* then it goes worse as over-fitting. The performance of *DON-RL* increases constantly and surpasses *DON* in the end.

Therefore, the *RL-Sampler* is suitable for the underlying data whose features are highly skew.

**Exp-3: Varying the Window Size.** We investigate the effect of the window size  $w$  in Eq. (1). Figure 5 shows for graph PL10K\_1.6, the  $F$  scores of 5 models training with  $w$  is set to  $\{4, 5, 6, 7, 8\}$ . For these *DON* models, they perform stably better than *GO* as  $w$  grows. In Fig. 5, *DON-RL* is the 5 models associated with RL training, which generates permutations of highest  $F$ . Recall that the approximate ratio of *GO* is  $\frac{1}{2w}$ , which means theoretically, the larger the  $w$ , the larger the gap between *GO* and the optimal solution, i.e., the improvement space of *GO*. Therefore, the model-based approaches can better take advantage of the gap to optimize the learned embedding and solution. Figure 5 reflects this point as  $w$  grows, the increment of locality score of model-based approaches grows a little faster than that of *GO*. In addition, another interesting phenomenon is that models trained with smaller  $w$  have better performance than that of larger  $w$ . This also means, we can use pre-trained models with small  $w$  to generate different permutations of larger  $w$ .

**Exp-4: Case Study.** We explore the potential of using graph ordering to deal with *graph compression*. Concretely, given a graph, we want to layout its edges so that its adjacency matrix  $A$  is easy to be compressed. Formally, this problem is equivalent to find a vertex ordering to minimize the storage cost  $\text{cost}_{nz}(A, b)$ , where  $b$  is the block width. Concretely, we divide the matrix  $A$  into  $b \times b$  square matrix blocks and count the number of the nonempty block as  $\text{cost}_{nz}(A, b)$  [15]. Since  $\text{cost}_{nz}(A, b)$  is not comparable for different block size  $b$ . Here we adopt a normalized version  $\text{cost}_r(A, b)$ :

$$\text{cost}_r(A, b) = \frac{\text{cost}_{nz}(A, b)}{\lceil N/b \rceil^2}, \quad (11)$$

where  $\lceil N/b \rceil^2$  is the number of blocks in  $A$ . A good ordering for the compression should have low  $\text{cost}_{nz}(A, b)$  for given block size  $b$ .

Figure 6 demonstrates the compression cost  $\text{cost}_r(A, b)$  for different methods on all datasets,  $b = 32$ . We compare the compression performance with two methods. *Degree* permutes the vertex based on the decreasing degree. *SlashBurn* [15] is the graph ordering algorithm proposed for graph compression. We set  $w = 5$  for our method, *DON-RL*, for all datasets. As shown in Fig. 6, *DON-RL* outperforms the other methods on three datasets (Wikivote, Facebook and Cora) and achieves a comparable results on two datasets (p2p and PPI). For the number of nonempty blocks, *DON-RL* reduces the counts by  $2.97\times$  and  $2.25\times$  compared with the second best orderings on Facebook and Cora. Interestingly, *Degree* is a good heuristic ordering for the graph compression. It achieves the best performance on p2p and PPI. The results validate that our learned vertex order is a good potential criterion for graph compression.

## 5 Related Works

**Graph Representation Learning.** Our approach is related to graph representation learning regarding adopting ML/DL to solve graph problems. Graph representation learning has been extensively studied. The surveys can be found in [10, 13]. Early graph representation learning uses matrix factorization to find a low-rank space for the adjacency matrix [2, 6, 11], aiming to reconstruct the original graph. Structure preserving representation learning techniques are proposed to encode graph proximity. For example, *DeepWalk* [21] and *Node2Vec* [12] encode the neighborhood relationship by exploiting truncated random walks as embedding context. Furthermore, there are graph representations to preserve vertex information at different granularity [23, 30], for incorporating high-order proximity. Unfortunately, these representations cannot be directly applied to our problem. For one thing, they are unsupervised task-independent representation which are expected to perform inference on general down-streaming tasks (e.g., node classification, link prediction, recommendation). Thereby, they do not contain abundant vertex features that are directly related to our graph ordering task. For the other thing, our problem aims to find an optimal combinatorial structure, whereas the graph representation performs inference on individual instances, which only involves local information.

**Neural Networks for Combinatorial Optimization:** Early works that use neural network to solve NP-hard optimization problems are surveyed in [22]. Recently, deep learning has been adopted to solve combinatorial optimization. A new sequence-to-sequence neural architecture, Pointer Network [26], is proposed and is used to solve the planar Travel Salesman Problem (TSP). The vanilla Pointer Network is supervised by a set of problem instances and solutions, and then [7] uses reinforcement learning, the Actor-Critic algorithm [17],

to train Pointer Network. To further improve the performance, [18] proposes an alternative neural network which encodes input nodes with multi-head attention [25]. These approaches [7, 18, 26] concentrate on 2D Euclidean space TSP, and it is non-trivial to extend them to deal with graphs. For graph data, Dai et al. [16] propose a framework to learn a greedy meta-algorithm over graphs. First, a graph is encoded by a graph embedding network. Second, the embedding is fed into another network to estimate an evaluation function. The overall network is trained by deep Q-learning [20]. The framework is demonstrated to solve Minimum Vertex Cover, Maximum Cut and TSP. These studies aim to generalize the process of problem-solving for a distribution of problem instances offline. Due to the restriction on the limited action space and the data distribution, they are not applicable to our scenario. Recently, [32] and [31] adopt graph neural network based model to solve the collapsed k-core problem and subgraph counting, respectively. However, these solutions cannot directly apply to our task neither.

## 6 Conclusion

In this paper, we focus on an NP-hard problem, graph ordering, in a novel, machine learning-based perspective. Distinguished from recent research, the NP-hard problem is over a specific larger graph. We design a new model: Deep Order Network (*DON*) to learn the underlying combinatorial closeness over the vertices of graph, by sampling small vertex sets as local partial solutions. To further improve the sampling effectiveness, we propose a new training data sampler *RL-Sampler* base on reinforcement learning. The whole framework is called *DON-RL*. With extensive experiments, *DON-RL* improves the quality of solution up to 7.6% and 35% for real graphs and synthetic graphs compared with the state-of-the-art, respectively. Moreover, ablation study on training data sampler shows the power of *RL-Sampler* on performance boosting. We also demonstrate the potentials of *DON-RL* in real applications. Our source code is public available in <https://github.com/Kangfei/DON>.

**Acknowledgement.** This work is supported by the Research Grants Council of Hong Kong, China under No. 14203618, No. 14202919 and No. 14205520.

## References

1. KONECT (the koblenz network collection). [konect.cc/networks/](http://konect.cc/networks/)
2. Ahmed, A., Shervashidze, N., Narayanamurthy, S.M., Josifov, V.A., Smola, J.: Distributed large-scale natural graph factorization. In: Proceedings of WWW'13
3. Auruox, L., Burelle, M., Erra, R.: Reordering very large graphs for fun and profit. In: International Symposium on Web Algorithms (2015)
4. Balaji, V., Lucia, B.: When is graph reordering an optimization? studying the effect of lightweight graph reordering across applications and input graphs. In: Proceedings of IEEE IISWC'18
5. Behrisch, M., Bach, B., Riche, T. S., Fekete, J.: Matrix reordering methods for table and network visualization. *Comput. Graph. Forum*, **35**(3) (2016)

6. Belkin, M., Niyogi, P.: Laplacian eigenmaps and spectral techniques for embedding and clustering. In: Proceedings of NIPS'01
7. Bello, I., Pham, H., Le, Q.V., Norouzi, M., Bengio, S.: Neural combinatorial optimization with reinforcement learning. In: Proceedings of ICLR'17, Workshop Track
8. Boldi, P., Rosa, M., Santini, M., Vigna, S.: Layered label propagation: a multiresolution coordinate-free ordering for compressing social networks. In: Srinivasan, S., (eds.) Proceedings of WWW'11
9. Boldi, P., Santini, M., Vigna, S.: Permuting web graphs. In: Avrachenkov, K., Donato, D., Litvak, N., (eds.) Proceedings of WAW'09
10. Cui, P., Wang, X., Pei, J., Zhu, W.: A survey on network embedding. *IEEE Trans. Knowl. Data Eng.* (2018)
11. Y. Fu and Y. Ma. Graph embedding for pattern analysis. Springer Science & Business Media, 2012
12. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks. In: Proceedings of KDD'16
13. Hamilton, W.L., Ying, R., Leskovec, J.: Representation learning on graphs: methods and applications. *IEEE Data Eng. Bull.*, **40**(3) (2017)
14. Harper, L.H.: Optimal assignments of numbers to vertices. *J. Soc. Ind. Appl. Math.* **12**(1), 131–135 (1964)
15. Kang, U., Faloutsos, C.: Beyond 'caveman communities': hubs and spokes for graph compression and mining. In: Proceedings of ICDM'11
16. E. B. Khalil, H. Dai, Y. Zhang, B. Dilkina, and L. Song. Learning combinatorial optimization algorithms over graphs. In Proc. of NIPS'17
17. Konda, V.R., Tsitsiklis, J.N.: Actor-critic algorithms. In: Proceedings of NIPS'99
18. Kool, W., van Hoof, H., Welling, M.: Attention, learn to solve routing problems!. In: Proceedings of ICLR'19
19. Leskovec, J., Krevl, A.: SNAP datasets: stanford large network dataset collection (2014). [snap.stanford.edu/data](http://snap.stanford.edu/data)
20. Mnih, V., et al.: Playing atari with deep reinforcement learning. In: Proceedings of NIPS'13, Deep Learning Workshop
21. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: online learning of social representations. In: Proceedings of KDD'14
22. Smith, K.A.: Neural networks for combinatorial optimization: a review of more than a decade of research. *INFORMS J. Comput.* **11**(1) (1999)
23. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: LINE: large-scale information network embedding. In: Proceedings of WWW'15
24. Tsoumakas, G., Katakis, I.: Multi-label classification: an overview. *IJDWM*, **3**(3) (2007)
25. Vaswani, A., et al.: Attention is all you need. In: Proceedings of NIPS'17
26. Vinyals, O., Fortunato, M., Jaitly, N.: Pointer networks. In: Proceedings of NIPS'15
27. Wei, H., Yu, J.X., Lu, C., Lin, X.: Speedup graph processing by graph ordering. In: Proceedings of SIGMOD'16
28. Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learn.* **8** (1992)
29. Zaheer, M., et al.: Deep sets. In: Proceedings of NIPS'17
30. Zhang, Z., Cui, P., Wang, X., Pei, J., Yao, X., Zhu, W.: Arbitrary-order proximity preserved network embedding. In: Proceedings of KDD'18
31. Zhao, K. Yu, X.J., Zhang, H., Li, Q., Rong, Y.: A learned sketch for subgraph counting. In: Proceedings SIGMOD '21
32. Zhao, K., Zhang, Z., Rong, Y., Yu, J.X., Huang, J.: Finding critical users in social communities via graph convolutions. *IEEE Trans. Knowl. Data Eng.*, pp. 1–1 (2021)



# Fast Approximate All Pairwise CoSimRanks via Random Projection

Renchi Yang<sup>(✉)</sup> and Xiaokui Xiao

National University of Singapore, Singapore, Singapore  
{renchi,xkxiao}@nus.edu.sg

**Abstract.** Given a graph  $G$  with  $n$  nodes, and two nodes  $u, v \in G$ , the *CoSimRank* value  $s(u, v)$  quantifies the similarity between  $u$  and  $v$  based on graph topology. Compared to SimRank, CoSimRank is shown to be more accurate and effective in many real-world applications including synonym expansion, lexicon extraction, and entity relatedness in knowledge graphs. The computation of all pairwise CoSimRanks in  $G$  is highly expensive and challenging. Existing solutions all focus on devising approximate algorithms for the computation of all pairwise CoSimRanks. To attain a desired absolute accuracy guarantee  $\epsilon$ , the state-of-the-art approximate algorithm for computing all pairwise CoSimRanks requires  $O(n^3 \log_2(\ln(\frac{1}{\epsilon})))$  time, which is prohibitively expensive even  $\epsilon$  is large. In this paper, we propose **RPCS**, a fast randomized algorithm for computing all pairwise CoSimRank values. The basic idea of **RPCS** is to approximate the  $n \times n$  matrix multiplications in CoSimRank computation via random projection. Theoretically, **RPCS** runs in  $O(\frac{n^2 \ln(n)}{\epsilon^2} \ln(\frac{1}{\epsilon}))$  time and meanwhile ensures an absolute error of at most  $\epsilon$  in each CoSimRank value in  $G$  with a high probability. Extensive experiments using six real graphs demonstrate that **RPCS** is more than up to orders of magnitude faster than the state of the art. In particular, on a million-edge Twitter graph, **RPCS** answers the  $\epsilon$ -approximate ( $\epsilon = 0.1$ ) all pairwise CoSimRank query within 4 h, using a single commodity server, while existing solutions fail to terminate within a day.

**Keywords:** CoSimRank · Random projection · Approximate algorithm

## 1 Introduction

Measuring node similarity is a fundamental problem in graph analysis and mining. CoSimRank [19] is recently proposed as a powerful similarity measure for quantifying the similarity between two nodes in a graph based on graph topology. CoSimRank finds numerous applications, such as *synonym expansion* and *lexicon extraction* in natural language processing, linguistically-informed statistical tool in *Cistern* project [6], modeling *entity relatedness* in knowledge graphs [18, 33], as well as measuring user similarity in social networks [13].

CoSimRank is closely related to existing similarity measures such as *Personalized PageRank* (PPR) [4, 8] and *SimRank* [7]. Given a graph  $G$  and two nodes  $u, v \in G$ , the PPR value of node  $v$  w.r.t node  $u$  is defined as the probability that a random walk starting from node  $u$  terminates at node  $v$ , which measures the strength of connections between two nodes instead of their neighborhood similarity. In contrast, the basic idea of SimRank is that two nodes are considered to be similar if their neighbors are similar. More specifically, the SimRank value of node pair  $(u, v)$  considers the first meeting time of random walks from  $u$  and  $v$ . Combining the ideas of PPR and SimRank, the CoSimRank value  $s(u, v)$  aggregates all possible meeting times of random walks from  $u$  and  $v$ , resulting in improved effectiveness than PPR and SimRank in many applications, as shown in [19], but incurring tremendous computational overheads.

Existing solutions [19, 31, 32] towards CoSimRank computation all focus on all-pairwise CoSimRank queries. Specifically, given an input graph  $G$  with  $n$  nodes, the all-pairwise CoSimRank query asks for approximate CoSimRank values of all possible node pairs in  $G$ . `PowerMethod` [19] directly applies *power iterations* [17] to approximate CoSimRank values iteratively, which involves expensive matrix multiplications as well as numerous iterations to ensure  $\epsilon$  absolute error guarantee in each CoSimRank value. `F-CoSim` [32] accelerates CoSimRank computation for given query nodes by first constructing the spanning polytree structures through breadth-first search. When the query set is the node-set of  $G$ , `F-CoSim` has the same time complexity as `PowerMethod`. `Co-Simrate` [31] reduces the number of iterations required in `PowerMethod` by reusing the intermediate results from previous iterations to speed up the computation in further iterations. In many practical scenarios [19, 31, 32], high-precision results are not necessary since a relatively large absolute error guarantee  $\epsilon$ , e.g., 0.1, in each CoSimRank value is sufficient for our purpose to identify the similar nodes. Unfortunately, all these methods entail  $O(n^3)$  time for matrix multiplications in each iteration, to attain the desired accuracy  $\epsilon$ , even  $\epsilon$  is large. Though we can harness the power of computing cluster to compute the CoSimRank values w.r.t each node independently in parallel, each thread still suffers from an  $O(n^2)$  time cost, which is prohibitive for large graphs. Thus, the retrieval of all pairwise CoSimRanks remains a highly challenging problem.

Motivated by this, we propose `RPCS` (short for Random Projection-based CoSimRank), an efficient algorithm for approximate all pairwise CoSimRank computation. The basic idea is to perform dimensionality reduction on the  $n \times n$  random walk matrix of the input graph  $G$  to obtain an  $n \times d$  matrix ( $d \ll n$ ), such that we can approximate the matrix multiplications in the  $d$ -dimensional space in an efficient manner, thereby avoiding the  $O(n^3)$  time cost in the course of computing CoSimRanks. Using Johnson–Lindenstrauss transformation [1, 9, 15] for the dimensionality reduction, `RPCS` is able to provide an accuracy guarantee  $\epsilon$  on each CoSimRank value in terms of its absolute error with a high probability. Additionally, `RPCS` needs the same number of iterations as in `PowerMethod` but each iteration consumes  $O(\min\{n^2/\epsilon^2, n^3\})$  time, which is favorable and satisfies our requirements when  $\epsilon < \frac{1}{\sqrt{n}}$ . Extensive experiments using six real graphs

**Table 1.** Frequently used notations.

Notation	Description
$G = (V, E)$	The input graph $G$ with node set $V$ and edge set $E$
$n, m$	The number of nodes and edges in $G$ , respectively
$N(v_i)$	The set of out-neighbors of node $v_i$
$d(v_i)$	The out-degree of node $v_i$ , <i>i.e.</i> , $ N(v_i) $
$c$	The damping factor in CoSimRank
$\epsilon$	The absolute error threshold for CoSimRank values
$\mathbf{S}$	The exact CoSimRank matrix (see Eq. (1))
$\widehat{\mathbf{S}}$	The approximate CoSimRank matrix

demonstrate that RPCS is more than up to orders of magnitude faster than the state of the art. In particular, on a million-edge Twitter graph, RPCS answers the  $\epsilon$ -approximate ( $\epsilon = 0.1$ ) all pairwise CoSimRank query within 4 h, using a single commodity server, while existing solutions fail to terminate within a day.

The rest of the paper is organized as follows. Section 2 provides the necessary background for CoSimRank and the formal problem definition. Related work is reviewed in Sect. 3. In Sect. 4.1, we present our proposed RPCS method and related analysis. Our solution and existing methods are evaluated in Sect. 5. Finally, Sect. 6 concludes the paper.

## 2 Preliminary

### 2.1 Notations and Terminology

Let  $G = (V, E)$  be an unweighted graph with  $n$  nodes and  $m$  edges, where  $V$  and  $E$  denote the node and edge sets, respectively. We denote by  $d(v_i)$  the out-degree of node  $v_i$  and by  $N(v_i)$  the out-neighbors of node  $v_i$ . For simplicity, in the following we assume that  $G$  is directed. For an undirected graph, we simply replace each undirected edge  $(u, v)$  with two directed ones with opposing directions, *i.e.*,  $(u, v)$  and  $(v, u)$ .

We denote matrices in bold uppercase, *e.g.*,  $\mathbf{M}$ . We use  $\mathbf{M}[i]$  to denote the  $i$ -th row vector of  $\mathbf{M}$ , and  $\mathbf{M}[:, j]$  to denote the  $j$ -th column vector of  $\mathbf{M}$ . In addition, we use  $\mathbf{M}[i, j]$  to denote the element at the  $i$ -th row and  $j$ -th column of  $\mathbf{M}$ . Given an index set  $\mathcal{I}$ , we let  $\mathbf{M}[\mathcal{I}]$  (resp.  $\mathbf{M}[:, \mathcal{I}]$ ) be the matrix block of  $\mathbf{M}$  that contains the row (resp. column) vectors of the indices in  $\mathcal{I}$ . Let  $\mathbf{A}$  be the adjacency matrix of the input graph  $G$ , *i.e.*,  $\mathbf{A}[i, j] = 1$  if  $(v_i, v_j) \in E$ , otherwise  $\mathbf{A}[i, j] = 0$ . Let  $\mathbf{D}$  be the diagonal out-degree matrix of  $G$ , *i.e.*,  $\mathbf{D}[i, i] = d(v_i) = \sum_{v_j \in V} \mathbf{A}[i, j]$ . We define the transition matrix (a.k.a. random walk matrix) of  $G$  is defined as  $\mathbf{P} = \mathbf{D}^{-1}\mathbf{A}$ . Accordingly,  $\mathbf{P}^\ell[i, j]$  signifies the probability that a  $\ell$ -step ( $\ell \geq 1$ )

random walk (*i.e.*, the random walk that walks from the current node to the next node along out-going edges) from node  $v_i$  would end at node  $v_j$ . Particularly, Lemma 1 proves an important property of transition matrix  $\mathbf{P}$ . Table 1 lists the frequently-used notations throughout the paper.

**Lemma 1.** *Given any integer  $k \geq 1$ ,  $\sum_{v_j \in V} \mathbf{P}^k[i, j] = 1 \forall v_i \in V$  holds.*

## 2.2 Problem Definition

Definition 1 presents the formal definition of CoSimRank.

**Definition 1 (CoSimRank [19]).** *Given an input graph  $G = (V, E)$  with its transition matrix  $\mathbf{P}$  and a damping factor  $c \in (0, 1)$ , the CoSimRank value of nodes  $v_i$  and  $v_j$  ( $v_i, v_j \in V$ ) is defined as:*

$$\mathbf{S}[i, j] = \sum_{\ell=0}^{\infty} c^\ell \langle \mathbf{P}^\ell[i], \mathbf{P}^\ell[j] \rangle. \quad (1)$$

The matrix form of CoSimRank is defined as

$$\mathbf{S} = \sum_{\ell=0}^{\infty} c^\ell \mathbf{P}^\ell \cdot (\mathbf{P}^\ell)^\top. \quad (2)$$

Since the computation of exact CoSimRank matrix  $\mathbf{S}$  is infeasible due to involving summing up an infinite series, this paper mainly focuses on  $\epsilon$ -approximate all-pairwise CoSimRank query, which is defined as follows.

**Definition 2 ( $\epsilon$ -approximate all pairwise CoSimRank query [31]).** *Given an input graph  $G = (V, E)$  and an absolute error threshold  $\epsilon \in (0, \frac{1}{1-c})$ ,  $\epsilon$ -approximate all-pairwise CoSimRank query returns an  $n \times n$  matrix  $\widehat{\mathbf{S}}$  that*

$$\left| \widehat{\mathbf{S}}[i, j] - \mathbf{S}[i, j] \right| \leq \epsilon, \quad (3)$$

holds for every two nodes  $v_i, v_j \in V$ , where  $\mathbf{S}[i, j]$  is the exact CoSimRank value defined in Eq. (1).

According to Lemma 1, for every two nodes  $v_i, v_j \in V$ , we have  $\mathbf{S}[i, j] \leq \sum_{\ell=0}^{\infty} c^\ell = \frac{1}{1-c}$ . Thus, we require  $0 < \epsilon < \frac{1}{1-c}$ .

## 3 Related Work

In this section, we first review three algorithms for  $\epsilon$ -approximate all-pairwise CoSimRank query, *i.e.*, `PowerMethod`, `Co-SimRank`, and `F-CoSim`, that are most related to our solutions; after that, we simply review other work related to CoSimRank computation.



**Table 2.** Theoretical guarantees of  $\epsilon$ -approximate all-pairwise CoSimRank algorithms.

Name	Accuracy	Time complexity
PowerMethod [19]	$ s(v_i, v_j) - \hat{s}(v_i, v_j)  \leq \epsilon, \forall v_i, v_j \in V$	$O\left(n^3 \ln\left(\frac{1}{\epsilon}\right)\right)$
Co-Simmate [31]	$ s(v_i, v_j) - \hat{s}(v_i, v_j)  \leq \epsilon, \forall v_i, v_j \in V$	$O\left(n^3 \log_2\left(\ln\left(\frac{1}{\epsilon}\right)\right)\right)$
F-CoSim [32]	$ s(v_i, v_j) - \hat{s}(v_i, v_j)  \leq \epsilon, \forall v_i, v_j \in V$	$O\left(n^3 \ln\left(\frac{1}{\epsilon}\right)\right)$
RPCS	$\mathbb{P}\left[ s(v_i, v_j) - \hat{s}(v_i, v_j)  \leq \epsilon, \forall v_i, v_j \in V\right] \geq 1 - \frac{1}{n}$	$O\left(\min\left\{\frac{n^2 \ln(n)}{\epsilon^2} \cdot \ln\left(\frac{1}{\epsilon}\right), n^3 \ln\left(\frac{1}{\epsilon}\right)\right\}\right)$

**PowerMethod.** The **PowerMethod** method is proposed in [19]. It computes a single element of  $\mathbf{S}$  iteratively from an inner product of two  $k$ -step random walk matrices, *i.e.*,  $\mathbf{P}^k$  and  $\mathbf{P}^{k\top}$ . Specifically, **PowerMethod** initializes  $\widehat{\mathbf{S}}^{(0)} = \mathbf{I}$ , and then in  $k$ -th iteration computes

$$\widehat{\mathbf{S}}^{(k)} = c\mathbf{P}\widehat{\mathbf{S}}^{(k-1)}\mathbf{P}^\top + \widehat{\mathbf{S}}^{(0)}. \quad (4)$$

Let  $t = \frac{\ln((1-c)\epsilon)}{\ln(c)} - 1$ . After  $t$  iterations, we have that for every two nodes  $v_i, v_j \in V$ ,

$$\begin{aligned} \left| \widehat{\mathbf{S}}^{(t)}[i, j] - \mathbf{S}[i, j] \right| &= \sum_{k=0}^{\infty} c^k \mathbf{P}^k[i] \cdot \mathbf{P}^k[j] - \sum_{k=0}^t c^k \mathbf{P}^k[i] \cdot \mathbf{P}^k[j] \\ &= \sum_{k=t+1}^{\infty} c^k \mathbf{P}^k[i] \cdot \mathbf{P}^k[j] \\ &\leq \sum_{k=t+1}^{\infty} c^k = \frac{c}{1-c} - \sum_{k=1}^t c^k = \frac{c^{t+1}}{1-c} = \epsilon, \end{aligned}$$

which exactly satisfies Eq. (3). The computation of  $\widehat{\mathbf{S}}$  involves a time complexity of  $O(n^3 \ln(\frac{1}{\epsilon}))$ . This time complexity consists of two parts: the first part is for matrix multiplications in Eq. (4) requires  $O(n^3)$  time, while the second part comes from the  $t$  iterations.

**Co-Simmate.** **Co-Simmate** [31] further reduces the theoretical time complexity of **PowerMethod** to  $O(n^3 \log_2(\ln(\frac{1}{\epsilon})))$  by reorganizing Eq. (4) and reusing the intermediate results from previous iterations to facilitate the computation in further iterations in **PowerMethod**. Initially, **Co-Simmate** sets  $\widehat{\mathbf{S}}^{(0)} = \mathbf{I}$  and  $\mathbf{Q}^{(0)} = \mathbf{P}$ . Subsequently, it iteratively calculates

$$\begin{aligned} \widehat{\mathbf{S}}^{(k)} &= \widehat{\mathbf{S}}^{(k-1)} + c^{2^k} \cdot (\mathbf{Q}^{(k-1)} \widehat{\mathbf{S}}^{(k-1)} \mathbf{Q}^{(k-1)\top}), \\ \mathbf{Q}^{(k)} &= \mathbf{Q}^{(k)2}. \end{aligned} \quad (5)$$

According to [31], applying Eq. (5) with  $t = \max\{0, \log_2(\frac{\ln((1-c)\epsilon)}{\ln(c)} - 1) + 1\}$  iterations is sufficient to produce an approximate CoSimRank matrix  $\widehat{\mathbf{S}}^{(t)}$  satisfying Eq. (3). Therefore, the computational time complexity of **Co-Simmate** is  $O(n^3 \log_2(\ln(\frac{1}{\epsilon})))$ .

**F-CoSim.** F-CoSim [32] is based on the following ideas. First, F-CoSim decomposes  $G$  into  $G = T \oplus (G \ominus T)$ , where  $T$  is a “spanning polytree” and can be viewed as the old graph, while  $G \ominus T$  can be viewed as the graph update. After that, due to the special “polytree” structure of  $T$ , the authors devised a fast algorithm to compute CoSimRank values  $\widehat{\mathbf{S}}_T$  over  $T$ . Finally, F-CoSim computes the changes of  $\widehat{\mathbf{S}}_T$  in response to the graph update  $G \ominus T$ . Given a set of nodes  $\mathcal{I}$  and the number of iterations  $t$  as inputs, F-CoSim returns approximate CoSimRank values  $\widehat{\mathbf{S}}[\cdot, \mathcal{I}]$  in  $O(n^2|\mathcal{I}|t)$  time in the worst case, which leads to time complexity of  $O(n^3t)$  if we let  $\mathcal{I} = V$ . Since F-CoSim also computes  $\widehat{\mathbf{S}}$  in an iterative way,  $t$  is also required to be set as  $\frac{\ln((1-c)\epsilon)}{\ln(c)} - 1$  in order to attain the desired accuracy  $\epsilon$ . Hence, the total time complexity of F-CoSim for  $\epsilon$ -approximate all pairwise CoSimRank query is bounded by  $O(n^3 \ln(\frac{1}{\epsilon}))$ .

In [32], the dynamic scheme, D-CoSim, is proposed for CoSimRank computation over evolving graphs. Second-order CoSimRank is introduced in [13] to effectively measure node similarities in social networks. Dhulipala et al. developed a parallel algorithm for single-source CoSimRank queries based on Graph Based Benchmark Suite (GBBS) [3].

Additionally, observe that the definition of CoSimRank (see Eq. (2)) is closely associated with PPR and SimRank. Due to the rapid advancements in approximate PPR and SimRank computations, a promising idea for the efficient CoSimRank computation might be utilizing these approximate PPR algorithms [5, 14, 21, 25, 26, 29, 30] or recent approximate SimRank algorithms [20, 22, 24, 27, 28]. However, most of these methods are designed for single-source PPR/SimRank queries instead of all pairwise queries and it is non-trivial to have them tailored for CoSimRank computation. Thus, in this paper, we do not discuss how to exploit these methods for CoSimRank computation and leave it as future work.

Table 2 compares the theoretical assurance of our proposed algorithm against that of existing algorithms in terms of accuracy and complexity. Our proposed RPCS answers  $\epsilon$ -approximate all pairwise CoSimRank query successfully with probability  $1 - \frac{1}{n}$  using  $O\left(\min\left\{\frac{n^2 \ln(n)}{\epsilon^2} \cdot \ln(\frac{1}{\epsilon}), n^3 \ln(\frac{1}{\epsilon})\right\}\right)$  time. In the following section, we elaborate details and theoretical analysis of RPCS.

## 4 The RPCS Algorithm

This section presents our randomized algorithm, *i.e.*, RPCS, for answering  $\epsilon$ -approximate all-pairwise CoSimRank queries. Observe that the tremendous overheads incurred by the CoSimRank computation are caused by the  $n \times n$  matrix multiplications between  $\mathbf{P}^k$  and  $\mathbf{P}^{k\top}$ . A fundamental tool to speed up the matrix multiplication is approximating the results in a  $d$ -dimensional ( $d \ll n$ ) space through random projection such that the pairwise distance can be preserved within a certain error. Johnson–Lindenstrauss transformation [1, 9, 15] allows us to reduce the dimension from  $n$  to  $d$  that is independent of  $n$ . In this way, the matrix multiplications can be done in  $O(n^2d)$  time. Although Johnson–Lindenstrauss transformation is mainly devised for preserving the Euclidean

---

**Algorithm 1: RPCS**

---

**Input:** An input graph  $G$ ,  $c, \epsilon, p_f, \delta$ .

**Output:**  $\widehat{\mathbf{S}}$ .

```

1  $t \leftarrow \left\lceil \frac{\ln(1 - \frac{c - (1-c)\epsilon}{c(1-\delta)})}{\ln(c)} \right\rceil$ ;
2  $d \leftarrow \left\lceil \frac{2 \ln(\frac{n^2}{2p_f})}{\delta - \ln(1+\delta)} \right\rceil$ ;
3 if  $d \geq n$  then
4    $\mathbf{Q} \leftarrow \mathbf{P}$ 
5 else
6   Generate  $\mathbf{T} \in \mathbb{R}^{n \times d} \sim \mathcal{N}(0, 1)$ ;  $\triangleright O(nd)$  time
7    $\mathbf{Q} \leftarrow \frac{1}{\sqrt{d}} \cdot \mathbf{P}\mathbf{T}$ ;  $\triangleright O(md)$  time
8  $\mathbf{H}^{(1)} \leftarrow \sqrt{c} \cdot \mathbf{Q}$ ;  $\widehat{\mathbf{S}} \leftarrow \mathbf{I} + \mathbf{H}^{(1)} \cdot \mathbf{H}^{(1)\top}$ ;  $\triangleright O(n^2d)$  time
9 for  $k \leftarrow 2$  to  $t$  do
10   $\mathbf{H}^{(k)} \leftarrow \sqrt{c}\mathbf{P} \cdot \mathbf{H}^{(k-1)}$ ;  $\triangleright O(md)$  time
11   $\widehat{\mathbf{S}} \leftarrow \widehat{\mathbf{S}} + \mathbf{H}^{(k)} \cdot \mathbf{H}^{(k)\top}$ ;  $\triangleright O(n^2d)$  time
12 return  $\widehat{\mathbf{S}}$ ;

```

---

distance between two vectors accurately, it is also able to preserve the inner product of two vectors within a certain error, which will be shown to be sufficient for our purpose. We illustrate our proposed algorithm in Sect. 4.1, followed by a theoretical analysis of RPCS in terms of accuracy and complexity in Sect. 4.2. Section 4.3 discusses the choice of parameter  $\delta$  used in RPCS to achieve the optimal running time in practice.

### 4.1 Main Algorithm

Algorithm 1 shows the pseudo-code of RPCS, which takes an input graph  $G$ , damping factor  $c \in (0, 1)$ , absolute error threshold  $\epsilon$ , failure probability  $p_f$ , and parameter  $\delta$  as inputs. Initially, RPCS calculates  $t = \left\lceil \frac{\ln(1 - \frac{c - (1-c)\epsilon}{c(1-\delta)})}{\ln(c)} \right\rceil$  and  $d = \left\lceil \frac{2 \ln(\frac{n^2}{2p_f})}{\delta - \ln(1+\delta)} \right\rceil$  (Lines 1–2). If  $d \geq n$ , meaning that projecting  $\mathbf{P}$  to an  $n \times d$  matrix leads to an  $O(n^3)$  time or even higher time costs, we can set  $\mathbf{Q} = \mathbf{P}$  instead to ensure that the time complexity incurred in matrix multiplications is bounded by  $O(n^3)$ , and thus, RPCS degrades to PowerMethod method. Otherwise, an  $n \times d$  projection matrix  $\mathbf{T}$  is generated, where each entry is an independent and identically distributed random variable sampled from a Gaussian  $\mathcal{N}(0, 1)$  (Lines 3–7), and RPCS initializes  $\mathbf{Q} = \frac{1}{\sqrt{d}} \cdot \mathbf{P}\mathbf{T}$ . After that, Algorithm 1 sets  $\mathbf{H}^{(1)} = \mathbf{Q}$  and computes  $\widehat{\mathbf{S}} = \mathbf{I} + \mathbf{H}^{(1)} \cdot \mathbf{H}^{(1)\top}$  (Lines 8–9). Subsequently, RPCS iteratively computes approximate CoSimRank matrix  $\widehat{\mathbf{S}}$  with  $t - 1$  iterations. Specifically, in  $k$ -th iteration, RPCS computes  $\mathbf{H}^{(k)} = \sqrt{c}\mathbf{P} \cdot \mathbf{H}^{(k-1)}$  and increase  $\widehat{\mathbf{S}}$  by  $\mathbf{H}^{(k)} \cdot \mathbf{H}^{(k)\top}$

(Lines 10–11). Finally, Algorithm 1 returns  $\widehat{\mathbf{S}}$  as an approximation of CoSimRank matrix  $\mathbf{S}$ .

## 4.2 Analysis

Before analyzing the accuracy guarantee of Algorithm 1, we first introduce the following lemmas.

**Lemma 2 ((Preservation of inner products [10]).** *Let  $\delta, p_f \in (0, 1)$  and  $d \geq \frac{2 \ln(1/p'_f)}{\delta - \ln(1+\delta)}$ . Let  $\mathbf{T}$  be an  $n \times d$  matrix, where each entry is sampled i.i.d. from a Gaussian  $\mathcal{N}(0, 1)$ . Given any two vectors  $\mathbf{z}_i, \mathbf{z}_j \in \mathbb{R}^n$ , we define  $\mathbf{x}_i = \frac{1}{\sqrt{d}} \cdot \mathbf{z}_i \mathbf{T}$ ,  $\mathbf{x}_j = \frac{1}{\sqrt{d}} \cdot \mathbf{z}_j \mathbf{T}$ . Then, we have*

$$\mathbb{P} \left[ \left| \mathbf{x}_i \cdot \mathbf{x}_j^\top - \mathbf{z}_i \cdot \mathbf{z}_j^\top \right| \leq \delta \cdot \|\mathbf{z}_i\| \cdot \|\mathbf{z}_j\| \right] \geq 1 - p'_f. \quad (6)$$

By applying union bound to Lemma 2, we obtain the following corollary:

**Lemma 3.** *Let  $\delta, p_f \in (0, 1)$  and  $d \geq \frac{2 \ln(1/p'_f)}{\delta - \ln(1+\delta)}$ . Let  $\mathbf{T}$  be an  $n \times d$  matrix, where each entry is sampled i.i.d. from a Gaussian  $\mathcal{N}(0, 1)$ . We define  $\mathbf{Q} = \frac{1}{\sqrt{d}} \cdot \mathbf{P} \mathbf{T}$ , where  $\mathbf{P}$  is the transition matrix of an input graph  $G$ . Then, for every two nodes  $v_i, v_j \in V$ , the following inequality holds:*

$$\mathbb{P} \left[ \left| \mathbf{Q}[i] \cdot \mathbf{Q}[j]^\top - \mathbf{P}[i] \cdot \mathbf{P}[j]^\top \right| \leq \frac{\delta}{\sqrt{d(v_i) \cdot d(v_j)}} \right] \geq 1 - \frac{n^2 p'_f}{2}.$$

*Proof.* First, note that for any node  $v_i \in V$ ,

$$\|\mathbf{P}[i]\| = \sqrt{\sum_{v_j \in N(v_i)} \frac{1}{d^2(v_i)}} = \sqrt{\frac{d(v_i)}{d^2(v_i)}} = \sqrt{\frac{1}{d(v_i)}}. \quad (7)$$

According to Lemma 2, for any two nodes  $v_i, v_j \in V$ , we have

$$\mathbb{P} \left[ \left| \mathbf{Q}[i] \cdot \mathbf{Q}[j]^\top - \mathbf{P}[i] \cdot \mathbf{P}[j]^\top \right| > \frac{\delta}{\sqrt{d(v_i) \cdot d(v_j)}} \right] \leq p'_f. \quad (8)$$

Using union bound over all  $\binom{n}{2} = \frac{n(n-1)}{2}$  node pairs, for every two nodes  $v_i, v_j \in V$ , we have

$$\mathbb{P} \left[ \left| \mathbf{Q}[i] \cdot \mathbf{Q}[j]^\top - \mathbf{P}[i] \cdot \mathbf{P}[j]^\top \right| > \frac{\delta}{\sqrt{d(v_i) \cdot d(v_j)}} \right] \leq \frac{n^2 p'_f}{2}, \quad (9)$$

which completes our proof.  $\square$

Based on the above analysis, we establish the accuracy guarantee of RPCS as follows:

**Theorem 1.** Given a damping factor  $c \in (0, 1)$ , failure probability  $p_f$  and an absolute error threshold  $\epsilon$  as inputs to Algorithm 1,  $\widehat{\mathbf{S}}$  is returned. Then, for every two nodes  $v_i, v_j \in V$ ,

$$\left| \widehat{\mathbf{S}}[i, j] - \mathbf{S}[i, j] \right| \leq \epsilon$$

holds with probability at least  $1 - p_f$ .

*Proof.* By Lines 6–9,  $\mathbf{H}^{(k)} = \sqrt{c^k} \mathbf{P}^{k-1} \mathbf{Q}$ .

$$\begin{aligned} \widehat{\mathbf{S}} - \mathbf{S} &= \mathbf{I} + \sum_{k=1}^t \mathbf{H}^{(k)} \mathbf{H}^{(k)\top} - \sum_{k=0}^{\infty} c^k \mathbf{P}^k \mathbf{P}^{k\top} \\ &= \sum_{k=1}^t c^k \mathbf{P}^{k-1} (\mathbf{Q}\mathbf{Q}^\top) (\mathbf{P}^{k-1})^\top - \sum_{k=1}^t c^k \mathbf{P}^k \mathbf{P}^{k\top} - \sum_{k=t+1}^{\infty} c^k \mathbf{P}^k \mathbf{P}^{k\top} \\ &= \sum_{k=1}^t c^k \mathbf{P}^{k-1} (\mathbf{Q}\mathbf{Q}^\top - \mathbf{P}\mathbf{P}^\top) (\mathbf{P}^{k-1})^\top - \sum_{k=t+1}^{\infty} c^k \mathbf{P}^k \mathbf{P}^{k\top} \end{aligned}$$

Let  $\mathbf{E}$  be an  $n \times n$  matrix, in which  $(i, j)$  entry is equal to  $\frac{\delta}{\sqrt{d(v_i) \cdot d(v_j)}}$ . Using

Lemma 3 with  $d = \left\lceil \frac{2 \ln(\frac{n^2}{2p_f})}{\delta - \ln(1+\delta)} \right\rceil$ , we have that with probability at least  $1 - p_f$ , we then obtain

$$\|\widehat{\mathbf{S}} - \mathbf{S}\|_{\max} \leq \left\| \sum_{k=1}^t c^k \mathbf{P}^{k-1} \mathbf{E} (\mathbf{P}^{k-1})^\top + \sum_{k=t+1}^{\infty} c^k \mathbf{P}^k \cdot \mathbf{P}^{k\top} \right\|_{\max}.$$

According to Lemma 1, for every two nodes  $v_i, v_j \in V$ ,

$$\begin{aligned} \|\widehat{\mathbf{S}} - \mathbf{S}\|_{\max} &\leq \sum_{k=1}^t c^k \delta + \sum_{k=t+1}^{\infty} c^k = \frac{c}{1-c} - (1-\delta) \cdot \sum_{k=1}^t c^k \\ &= \frac{c}{1-c} (1 - (1-\delta) \cdot (1-c^t)), \end{aligned}$$

According to Line 1 in Algorithm 1,  $t = \left\lceil \frac{\ln(1 - \frac{c - (1-c)\epsilon}{c(1-\delta)})}{\ln(c)} \right\rceil$ , we have  $\|\widehat{\mathbf{S}} - \mathbf{S}\|_{\max} \leq \epsilon$ ,

i.e.,  $\left| \widehat{\mathbf{S}}[i, j] - \mathbf{S}[i, j] \right| \leq \epsilon \forall v_i, v_j \in V$ , it with probability at least  $1 - p_f$ . The theorem is proved.  $\square$

First, the space complexity is  $O(n^2)$  since we need to store the CoSimRank values for all possible node pairs in  $G$ . Considering the case where  $d \geq n$  (Line 4 in Algorithm 1), RPCS degrades to PowerMethod method, and thus, the time complexity is  $O(n^3 \ln(\frac{1}{\delta}))$ .

Next, we discuss the case where  $d < n$  (Lines 6–7 in Algorithm 1). According to [1], the generation of  $\mathbf{T}$  requires  $O(nd)$  time (Line 6). In  $k$ -th iteration, the

**Algorithm 2: TernarySearch**


---

**Input:**  $c, \epsilon$ .  
**Output:**  $\delta$ .

- 1  $\delta_l \leftarrow 0, \delta_u \leftarrow \frac{1-c}{c} \cdot \epsilon;$
- 2 **while true do**
- 3      $\delta'_l \leftarrow \delta_l + \frac{\delta_u - \delta_l}{3};$
- 4      $\delta'_u \leftarrow \delta_u - \frac{\delta_u - \delta_l}{3};$
- 5     **if**  $\delta'_u \leq \delta'_l$  **or**  $\delta_u - \delta_l \leq \frac{1-c}{1000c} \cdot \epsilon$  **then break;**
- 6     **if**  $f(\delta'_l) < f(\delta'_u)$  **then**
- 7          $\delta_l \leftarrow \delta'_l;$
- 8     **else**
- 9          $\delta_u \leftarrow \delta'_u;$

10  $\delta \leftarrow \frac{\delta_l + \delta_u}{2};$   
11 **return**  $\delta;$

---

computation of  $\mathbf{H}^{(k)}$  costs  $O(md)$  time and  $O(nd)$  space, while the computation of  $\widehat{\mathbf{S}}$  requires  $O(n^2d)$  time and  $O(n^2)$  space. By Lines 1–2 in Algorithm 1, we need to set  $d = \left\lceil \frac{2 \ln(\frac{n^2}{2p_f})}{\delta - \ln(1+\delta)} \right\rceil$  and  $t = \left\lceil \frac{\ln(1 - \frac{c - (1-c)\epsilon}{c(1-\delta)})}{\ln(c)} \right\rceil$  iterations in total. Therefore, the overall time complexity is

$$O(mdt + n^2dt) = O(n^2dt) = O\left(n^2 \ln\left(\frac{n^2}{2p_f}\right) \cdot \frac{\ln\left(\frac{c(1-\delta)}{(1-c)\epsilon - c\delta}\right)}{\delta - \ln(1+\delta)}\right), \quad (10)$$

Since  $\delta \in (0, 1)$  and  $(1-c)\epsilon - c\delta > 0$ , we obtain  $0 < \delta < \frac{(1-c)\epsilon}{c}$ . According to the inequality  $\ln(1+\delta) \leq \delta - \delta^2/2 + \delta^3/3 \forall \delta \geq 0$ , the time complexity in Eq. (10) is bounded by  $O(\frac{n^2 \ln(n)}{\epsilon^2} \ln(\frac{1}{\epsilon}))$  when letting  $\delta = \frac{(1-c)\epsilon}{2c}$ . To sum up, the time complexity of RPCS is

$$O\left(\min\left\{\frac{n^2 \ln(n)}{\epsilon^2} \cdot \ln\left(\frac{1}{\epsilon}\right), n^3 \ln\left(\frac{1}{\epsilon}\right)\right\}\right).$$

In the subsequent subsection, we elaborate how to minimize the practical time cost in Eq. (10) by carefully picking  $\delta$ .

**Remark.** According to [2, 23], setting  $d$  to  $\left\lceil \frac{\ln(\frac{n^2}{2p_f})}{2(\delta - \ln(1+\delta))} \right\rceil$  is good enough for Johnson–Lindenstrauss transformation to achieve the desired accuracy in practice. The gap between the theoretic bound and practical one is due to the union bound used in the proof of Lemma 3.

### 4.3 Choosing $\delta$

In the following, we explain the rationale of Algorithm 2 for choosing  $\delta$ . According to Eq. (10), the total time complexity of RPCS is linear to

$$f(\delta) = \frac{\ln\left(\frac{c(1-\delta)}{(1-c)\epsilon - c\delta}\right)}{\delta - \ln(1+\delta)}. \quad (11)$$

Recall that  $0 < \delta < \frac{(1-c)}{c} \cdot \epsilon$ . Therefore, to achieve the minimum time cost, the aim is then to find a value for  $\delta$  such that  $f(\delta)$  is minimized, *i.e.*,

$$\min_{0 < \delta < \frac{(1-c)}{c} \cdot \epsilon} f(\delta). \quad (12)$$

$f(\delta)$  is a convex function when  $\delta$  is in range  $(0, \frac{(1-c)}{c} \cdot \epsilon)$ . Thus, we can find an approximate minimum value for  $f(\delta)$  by using a ternary search algorithm [12], as presented in Algorithm 2. Specifically, Algorithm 2 takes damping factor  $c$  and error threshold  $\epsilon$  as inputs and then computes the initial lower bound  $\delta_l = 0$  and upper bound  $\delta_u = \frac{1-c}{c} \cdot \epsilon$  for  $\delta$  (Line 1). Subsequently, Algorithm 2 starts iterations to update the lower bound  $\delta_l$  and upper bound  $\delta_u$ . In each iteration, Algorithm 2 first computes  $\delta'_l = \delta_l + \frac{\delta_u - \delta_l}{3}$  and  $\delta'_u = \delta_u - \frac{\delta_u - \delta_l}{3}$  (Lines 3–4). If  $\delta'_u \leq \delta'_l$  or  $\delta_u - \delta_l \leq \frac{1-c}{1000c} \cdot \epsilon$ , the lower bound  $\delta_l$  is very close to the upper bound  $\delta_u$ , and thus, we can terminate the search of the lower and upper bounds (Line 5). When the termination condition is not satisfied, Algorithm 2 updates the lower bound  $\delta_l = \delta'_l$  if  $f(\delta'_l) < f(\delta'_u)$ , otherwise  $\delta_u = \delta'_u$  (Lines 6–9), and proceeds to next iteration. Finally, Algorithm 2 computes  $\delta = \frac{\delta_l + \delta_u}{2}$  and returns it (Lines 10–11). Note that when  $\delta_u - \delta_l \leq \frac{1-c}{1000c} \cdot \epsilon$ , Algorithm 2 finishes searching. Hence, Algorithm 2 requires  $\log_3\left(\frac{(1-c)\cdot\epsilon}{c} \cdot \frac{1000c}{(1-c)\cdot\epsilon}\right) = \log_3(1000)$  iterations in a worst case.

## 5 Experiments

We experimentally evaluate our proposed RPCS against two competitors in terms of efficiency on 6 real datasets. All experiments are conducted on a Linux machine powered by an Intel Xeon(R) Gold 6240@2.60 GHz CPU and 377 GB RAM. Source codes of all methods are implemented in Python and all matrices are represented in a sparse form to avoid unnecessary space overheads.

### 5.1 Experimental Setting

**Datasets.** We experiment with six real-world graphs that are used in previous work [31, 32], which are taken from [11]. Table 3 lists the statistics of the datasets. *as-735 (AS)*<sup>1</sup> is a communication network of autonomous systems extracted

<sup>1</sup> <http://www.cise.ufl.edu/research/sparse/matrices/SNAP/as-735.html>.

**Table 3.** Statistics for datasets.

Name	# Nodes ( $n$ )	# Edges ( $m$ )	Type
<i>Facebook</i>	4,039	88,234	Undirected
<i>as-735</i>	7,716	26,467	Undirected
<i>ca-HepPh</i>	12,008	237,010	Undirected
<i>email-Enron</i>	36,692	183,831	Directed
<i>Twitter</i>	81,306	1,768,149	Directed
<i>Google+</i>	107,614	13,673,453	Directed

from the Border Gateway Protocol logs, where an edge represents a who-talks-to-whom relationship. *ca-HepPh*<sup>2</sup> is a collaboration graph from the arXiv High Energy Physics, where each node is an author and each edge represents a collaboration relationship. *email-Enron*<sup>3</sup> is an email communication network collected from Enron, in which each node signifies an email address and there is an edge between two nodes if at least one email is sent between them. *Facebook*<sup>4</sup>, *Twitter*<sup>5</sup>, and *Google+*<sup>6</sup> are social networks used in [16].

**Parameter Settings.** We compare RPCS against *PowerMethod*, *Co-Simmate*, and *F-CoSim* in terms of efficiency for  $\epsilon$ -approximate all-pairwise CoSimRank queries. Following prior work [31], we set damping factor  $c = 0.8$ , meaning that  $\epsilon$  should be in range  $(0, 5)$ . In our experiments, we vary absolute error threshold  $\epsilon$  in range  $\{1.0, 0.5, 0.2, 0.1, 0.05\}$ . We report the running time (measured in wall-clock time) of each algorithm on each dataset with various  $\epsilon$  settings. Note that the  $y$ -axis is in log-scale and the measurement unit for running time is second (sec). We omit any methods if they can not terminate within two days or run out of memory.

## 5.2 Efficiency Evaluation

Figure 1 plots the running time of RPCS, *PowerMethod* and *Co-Simmate* on six datasets when varying  $\epsilon$  in  $\{1.0, 0.5, 0.2, 0.1, 0.05\}$ . First, observe that *Co-Simmate* runs much slower than *PowerMethod*, which is inconsistent with their theoretical time complexities as introduced in Sect. 3. The reason is that the transition matrix  $\mathbf{P}$  is often very sparse in practice, making the empirical running time of *PowerMethod* far less than its theoretical time. Moreover, *PowerMethod* only requires updating  $\hat{\mathbf{S}}^k$  based on Eq. (4), while *Co-Simmate* needs to compute  $\hat{\mathbf{S}}^k$  and  $\hat{\mathbf{Q}}^{(k)}$  in each iteration, which involves additional matrix multiplications. Another observation we can make from Fig. 1 is that, on small graphs including

<sup>2</sup> <http://snap.stanford.edu/data/ca-HepPh.html>.

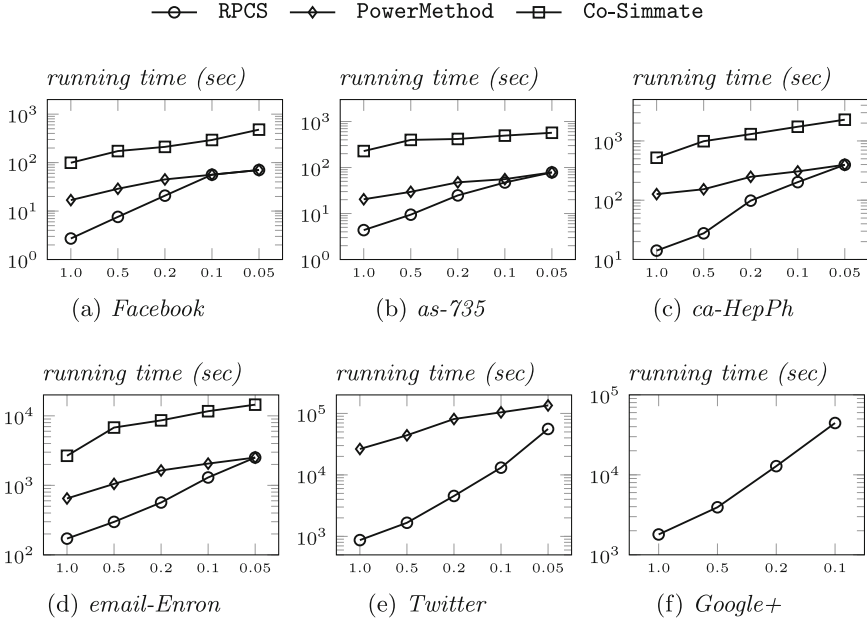
<sup>3</sup> <http://snap.stanford.edu/data/email-Enron.html>.

<sup>4</sup> <http://snap.stanford.edu/data/ego-Facebook.html>.

<sup>5</sup> <http://snap.stanford.edu/data/ego-Twitter.html>.

<sup>6</sup> <http://snap.stanford.edu/data/ego-Gplus.html>.





**Fig. 1.** Running time with varying  $\epsilon$ .

*Facebook*, *as-735* and *ca-HepPh*, RPCS is 2-9 $\times$  faster than PowerMethod when  $\epsilon \geq 0.2$ , and requires almost the same time as PowerMethod when  $\epsilon \leq 0.1$ . This is due to that on small graphs, a small  $\epsilon$  value is likely to lead to  $d \geq n$ , and thus, RPCS degrades to PowerMethod. On the million-edge graph *Twitter*, RPCS consistently outperforms PowerMethod by up to three orders of magnitude. For instance, when  $\epsilon = 0.1$ , RPCS requires 3.7h while PowerMethod costs about 1.2 days on *Twitter* dataset. For the largest *Google+* dataset, our solution RPCS is the only viable solution to obtain approximate all pairwise CoSimRank values when varying  $\epsilon$  from 0.1 to 1.0 on a single server, while both PowerMethod and Co-Simmate run out of memory and fail to return results. This implies that RPCS consumes much fewer space costs in practice compared with PowerMethod and Co-Simmate.

## 6 Conclusion

This paper presents RPCS, a random projection-based method for answering  $\epsilon$ -approximate all pairwise CoSimRank query with  $\epsilon$  worst-case absolute error in each CoSimRank value with a high probability. RPCS requires  $O(\frac{n^2 \ln(n)}{\epsilon^2} \ln(\frac{1}{\epsilon}))$  time to process all node pairs in the graph. In many practical scenarios, a relatively large absolute error guarantee  $\epsilon$  in each CoSimRank value is sufficient for our purpose to identify similar nodes. RPCS is up to orders of magnitude faster than prior work in such scenarios. However, RPCS suffers from an expensive time

complexity of  $O(n^4)$  when high-precision results are desired, *e.g.*,  $\frac{1}{n}$ . In the future work, we will study how to answer  $\epsilon$ -approximate all pairwise CoSimRank query in  $O(n^2 \ln(n) \ln(\frac{1}{\epsilon}))$  time.

## References

1. Achlioptas, D.: Database-friendly random projections: johnson-lindenstrauss with binary coins. *J. Comput. Syst. Sci.*, pp. 671–687 (2003)
2. Chen, L.: Johnson-lindenstrauss transformation and random projection (2015)
3. Dhulipala, L., Shi, J., Tseng, T., Blesl, G.E., Shun, J.: The graph based benchmark suite (gbb). In: *Proceedings of the Joint International Workshop on Graph Data Management Experiences and Systems and Network Data Analytics*, pp. 1–8 (2020)
4. Haveliwala, T.H.: Topic-sensitive pagerank: a context-sensitive ranking algorithm for web search. *IEEE Trans. Knowl. Data Eng.*, pp. 784–796 (2003)
5. Hou, G., Chen, X., Wang, S., Wei, Z.: Massively parallel algorithms for personalized pagerank. *Proc. VLDB Endow.* **14**(9), 1668–1680 (2021)
6. Munich, L.P.: For information of the university cistern (2013). [cistern.cis.lmu.de](http://cistern.cis.lmu.de)
7. Jeh, G., Widom, J.: Simrank: a measure of structural-context similarity. In: *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 538–543 (2002)
8. Jeh, G., Widom, J.: Scaling personalized web search. In: *Proceedings of the International Conference on World Wide Web*, pp. 271–279 (2003)
9. Johnson, W.B., Lindenstrauss, J.: Extensions of lipschitz mappings into a hilbert space. *Contemp. Math.* **26**(189–206), 1 (1984)
10. Kaban, A.: Improved bounds on the dot product under random projection and random sign projection. In: *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 487–496 (2015)
11. Leskovec, J., Krevl, A.: SNAP Datasets: stanford large network dataset collection (2013). [snap.stanford.edu/data](http://snap.stanford.edu/data)
12. Levitin, A.: *Introduction to the design and analysis of algorithms*. Pearson, Boston (2012)
13. Liao, X., Wu, Y., Cao, X.: Second-order cosimrank for similarity measures in social networks. In: *IEEE International Conference on Communications*, pp. 1–6 (2019)
14. Lin, W.: Distributed algorithms for fully personalized pagerank on large graphs. In: *Proceedings of the International Conference on World Wide Web*, pp. 1084–1094 (2019)
15. Matoušek, J.: On variants of the johnson-lindenstrauss lemma. *Random Structures and Algorithms*, pp. 142–156 (2008)
16. McAuley, J.J., Leskovec, J.: Learning to discover social circles in ego networks. In: *Advances in Neural Information Processing Systems*, pp. 548–56 (2012)
17. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: bringing order to the web. *Tech. Rep.*, Stanford InfoLab (1999)
18. Ponzani, M., Ferragina, P., Chakrabarti, S.: A two-stage framework for computing entity relatedness in wikipedia. In: *Proceedings of the ACM International Conference on Information and Knowledge Management*, pp. 1867–1876 (2017)
19. Rothe, S., Schütze, H.: Cosimrank: A flexible and efficient graph-theoretic similarity measure. In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 1392–1402 (2014)

20. Shi, J., Jin, T., Yang, R., Xiao, X., Yang, Y.: Realtime index-free single source simrank processing on web-scale graphs. *Proc. VLDB Endow.* **13**(7), 966–980 (2020)
21. Shi, J., Yang, R., Jin, T., Xiao, X., Yang, Y.: Realtime top-k personalized pagerank over large graphs on gpus. *Proc. VLDB Endow.*, pp. 15–28 (2019)
22. Tian, B., Xiao, X.: Sling: A near-optimal index structure for simrank. In: *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 1859–1874 (2016)
23. Venkatasubramanian, S., Wang, Q.: The johnson-lindenstrauss transform: an empirical study. In: *Proceedings of the Thirteenth Workshop on Algorithm Engineering and Experiments*, pp. 164–173 (2011)
24. Wang, H., Wei, Z., Yuan, Y., Du, X., Wen, J.R.: Exact single-source simrank computation on large graphs. In: *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 653–663 (2020)
25. Wang, S., et al.: Efficient algorithms for approximate single-source personalized pagerank queries. *ACM Trans. Database Syst.*, pp. 1–37 (2019)
26. Wang, S., Yang, R., Xiao, X., Wei, Z., Yang, Y.: Fora: simple and effective approximate single-source personalized pagerank. In: *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 505–514 (2017)
27. Wang, Y., et al.: Disk: a distributed framework for single-source simrank with accuracy guarantee. *Proc. VLDB Endow.* **14**(3), 351–363 (2020)
28. Wei, Z., et al.: Prsim: Sublinear time simrank computation on large power-law graphs. In: *Proceedings of the ACM SIGMOD International Conference on Management of Data*. pp. 1042–1059 (2019)
29. Wu, H., Gan, J., Wei, Z., Zhang, R.: Unifying the global and local approaches: an efficient power iteration with forward push. In: *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 1996–2008 (2021)
30. Yang, R., Shi, J., Xiao, X., Yang, Y., Bhowmick, S.S.: Homogeneous network embedding for massive graphs via reweighted personalized pagerank. *Proc. VLDB Endow.* **13**(5), 670–683 (2020)
31. Yu, W., McCann, J.: Co-simmate: Quick retrieving all pairwise co-simrank scores. In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. pp. 327–333 (2015)
32. Yu, W., Wang, F.: Fast exact cosimrank search on evolving and static graphs. In: *Proceedings of the International Conference on World Wide Web*, pp. 599–608 (2018)
33. Zeng, W., Tang, J., Zhao, X.: Measuring entity relatedness via entity and text joint embedding. *Neural Process. Lett.*, pp. 1861–1875 (2019)



# Critical Nodes Identification in Large Networks: An Inclination-Based Model

Chen Chen<sup>1</sup>, Xijuan Liu<sup>1</sup>, Shuangyan Xu<sup>1</sup>, Mengqi Zhang<sup>1</sup>,  
Xiaoyang Wang<sup>1(✉)</sup>, and Xuemin Lin<sup>2</sup>

<sup>1</sup> Zhejiang Gongshang University, Hangzhou, China  
{chenc, liuxijuan, xshy, xiaoyangw}@zjgsu.edu.cn  
<sup>2</sup> University of New South Wales, Kensington, Australia  
lxue@cse.unsw.edu.au

**Abstract.** In social networks, the departure of some users can lead to the drop-out of others in cascade. Therefore, the engagement of critical users can significantly influence the stability of a network. In the literature, the anchored  $k$ -core problem is proposed, which aims to enlarge the community by anchoring  $b$  nodes. While, in real social networks, nodes are usually associated with different preferences, i.e., inclination, such as close or conflict interest. Intuitively, a community will be more stable if more nodes have close interest and fewer of them carry conflict interest. However, most existing researches simply treat all users equally, and the inclination property is neglected. To fill the gap, in this paper, we propose and investigate the inclined anchored  $k$ -core problem, which aims to anchor  $b$  nodes, such that more close nodes and fewer conflict nodes will join the community. We show that this problem is NP-hard. To facilitate the computation, a layer-based searching framework is adopted. In addition, an upper bound based technique is developed to enable early termination in iterations. Comprehensive experiments and case studies are conducted on 9 networks to demonstrate the effectiveness and efficiency of the proposed methods.

**Keywords:** Inclined anchored  $k$ -core · Graph analysis · NP-hard

## 1 Introduction

In social network analysis, modeling user engagement for user behavior has attracted great interest from researchers [1, 6–8, 11, 20, 21]. The  $k$ -core model is widely used in the study of network stability or engagement [1]. Given a graph  $G$ ,  $k$ -core is the maximal subgraph, in which each node has at least  $k$  neighbors. The size of  $k$ -core plays an important role in graph analysis. To enlarge the size of  $k$ -core, [1] proposes and investigates the problem of anchored  $k$ -core. By giving some important users incentives, we ask them to stay in the community (i.e.,  $k$ -core) regardless of their neighbor size. These nodes are called anchored nodes. The anchored  $k$ -core problem aims to maximize the size of the resulting  $k$ -core by anchoring  $b$  nodes. [1] proves the problem is NP-hard. In [21], efficient

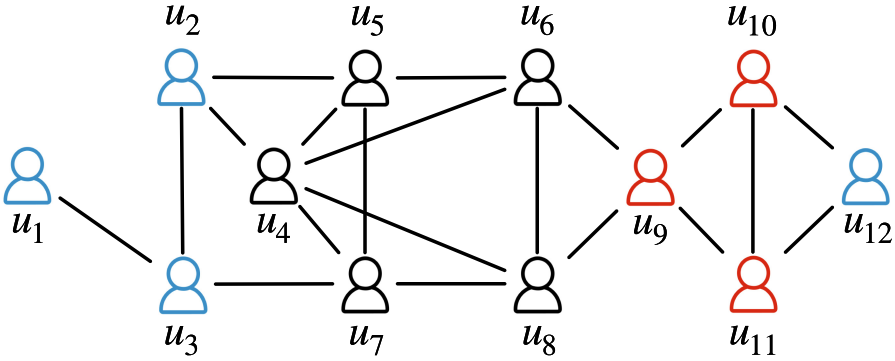


Fig. 1. Motivation example

algorithms are further developed to speedup the computation. However, existing studies treat all the nodes equally. In real social networks, users usually preserve different properties, such as close and conflict interests. Thus, the engagement of a community is often related to the number of users that have close interest inside, and simply enlarging the community size, i.e.,  $k$ -core, cannot reflect the propensity and stability of the network.

Intuitively, a community is more stable if more nodes inside have close interest [2]. It will also assist more new users to join the community if the community is strongly inclined, i.e., more nodes with close interest and fewer nodes with conflict interest. Motivated by this, in this paper, we consider a network with labels, where users are marked as close, conflict or unlabeled (i.e., without preference). When anchoring a node  $u$ , we call the new nodes that join the  $k$ -core as the followers of  $u$ . To better measure the effectiveness of anchoring a node, we propose the concept inclined score, i.e., the difference between the number of close followers and conflict followers. Then, we propose and investigate the inclined anchored  $k$ -core problem, which aims to reinforce the inclined network by giving incentives to  $b$  users to persuade them to stay engaged, so that more close nodes but fewer conflict nodes will join in the  $k$ -core community. As observed, we not only consider the size of resulting  $k$ -core, but also pore over the inclination of the returned community. Through this way, we can better portray the resilience of networks. Following is a motivating example.

*Example 1.* As shown in Fig. 1, there is a social network with 12 users and their corresponding connections. Suppose  $k = 3$ , the close node set  $F = \{u_1, u_2, u_3, u_{12}\}$  (blue nodes), the conflict node set  $\mathcal{E} = \{u_9, u_{10}, u_{11}\}$  (red nodes), and the others are marked as unlabeled. The willingness of a user to engage in the group depends on the number of his neighbors. If a user leaves the group, the willingness of his neighbors will decrease, which may lead to a cascade of others to drop out the community. Given  $k = 3$ , only nodes  $u_4, u_5, \dots, u_8$  belong to the 3-core. To enhance the inclination and stability of the group, we can anchor the critical nodes by giving them incentives. If we anchor  $u_1$ , there will be three close

followers  $\{u_1, u_2, u_3\}$  (i.e., join the anchored  $k$ -core) and no conflict followers. When  $u_{12}$  is anchored, there will be four followers  $\{u_9, u_{10}, u_{11}, u_{12}\}$ , but three of them are from the conflict group. Thus, for  $b = 1$ , we can select the optimal node as  $u_1$  to anchor. It can also be observed, simply enlarging the  $k$ -core size will not guarantee the increase of nodes with close interest.

**Challenge and Contribution.** To the best of our knowledge, we are the first to investigate the inclined anchored  $k$ -core problem. Firstly, we prove the inclined anchored  $k$ -core problem is NP-hard when  $k \geq 3$ , which implies that it is non-trivial to solve it within polynomial time, and we turn to the greedy heuristic in this paper. Also, we show the non-monotonic and non-submodular properties of the objective function, which makes lots of pruning rules in previous studies no longer hold for our problem. Secondly, in real-world social networks, the candidate space is usually quite large, which is time-consuming to conduct the exploration. To enhance the greedy searching framework, a layer-based method is adopted to accelerate the computation. In addition, upper bound based techniques are integrated to speedup the procedure. Finally, we conduct extensive experiments over 9 datasets to evaluate the efficiency and effectiveness of proposed methods.

## 2 Preliminaries

We consider an unweighted and undirected graph  $G = (V, E)$ , where  $V$  and  $E$  represent the sets of nodes and edges, respectively. We denote  $n = |V|$  and  $m = |E|$ . Given a subgraph  $S \subseteq G$ ,  $N(u, S)$  is the set of adjacent nodes of  $u$  in  $S$ .  $\text{deg}(u, S)$  is the degree of  $u$  in  $S$ , i.e., the number of adjacent nodes of  $u$  in  $S$ . Table 1 summarizes the notations frequently used throughout the paper.

**Definition 1 ( $k$ -core).** *Given a graph  $G$ , a subgraph  $S$  is the  $k$ -core of  $G$ , denoted by  $C_k(G)$ , if 1)  $\text{deg}(u, S) \geq k$  for each node  $u \in S$ ; 2)  $S$  is maximal, i.e., any supergraph  $S' \supset S$  is not a  $k$ -core.*

The  $k$ -core of a graph  $G$  can be obtained by recursively removing the node whose degree is less than  $k$ , with a time complexity of  $\mathcal{O}(m)$  [16].

**Definition 2 ( $k$ -peel).** *Given a graph  $G$ , the  $k$ -peel of  $G$ , denoted by  $\mathcal{P}_k(G)$ , is the set of nodes that belong to  $k$ -core but not  $k + 1$ -core, i.e.,  $\mathcal{P}_k(G) = C_k(G) \setminus C_{k+1}(G)$ .*

The  $k$ -core model is widely used to measure the properties of the graph. In order to enlarge the  $k$ -core, we can offer some incentives to a set  $\mathcal{A}$  of nodes, named the anchored nodes, to make them stay in the  $k$ -core community. That is, once a node is anchored, it is always reserved in the  $k$ -core, i.e., with infinite large degree. Through anchoring these nodes, some other nodes will join the  $k$ -core in cascade and finally the  $k$ -core is enlarged.

**Table 1.** Notation table

Notation	Definition
$G$	An unweighted and undirected graph
$V; E$	The node set of $G$ ; the edge set of $G$
$n$	The number of nodes in $G$
$m$	The number of edges in $G$
$S$	A subgraph of $G$
$N(u, S)$	The set of adjacent nodes of $u$ in $S$
$deg(u, S)$	The number of adjacent nodes of $u$ in $S$
$C_k(G); \mathcal{P}_k(G)$	The $k$ -core of $G$ ; the $k$ -peel of $G$
$\mathcal{A}$	A set of anchored nodes
$F; \mathcal{E}$	The close node set; the conflict node set
$\mathcal{F}(u, G)$	The followers of the anchored node $u$
$\mathcal{F}^+(\mathcal{A}, G); \mathcal{F}^-(\mathcal{A}, G)$	The close (resp. conflict) followers of $\mathcal{A}$
$Score(\mathcal{A}, G)$	The difference of $\mathcal{F}^+(\mathcal{A}, G)$ and $\mathcal{F}^-(\mathcal{A}, G)$

**Definition 3 (anchored  $k$ -core).** *Given a graph  $G$  and an anchored node set  $\mathcal{A} \subseteq V$ , the anchored  $k$ -core, denoted by  $C_k(G \oplus \mathcal{A})$ , is the corresponding  $k$ -core of  $G$  with nodes in  $\mathcal{A}$  anchored.*

We use  $\mathcal{F}(u, G)$  to denote the set of nodes that join the  $k$ -core when  $u$  is anchored, i.e.,  $\mathcal{F}(u, G) = C_k(G \oplus u) \setminus C_k(G)$ , and call  $\mathcal{F}(u, G)$  as the followers of  $u$ . As discussed, a community may has its own preferences, such as nodes with close or conflict interests. When selecting anchored nodes, we would like to preserve more users with close interests and less users with conflict interests. Given the set  $F$  (resp.  $\mathcal{E}$ ) of nodes with **close** (resp. **conflict**) interests, we use  $\mathcal{F}^+(\mathcal{A}, G)$  (resp.  $\mathcal{F}^-(\mathcal{A}, G)$ ) to denote the close (resp. conflict) followers of  $\mathcal{A}$ , i.e.,  $\mathcal{F}(\mathcal{A}, G) \cap F$  (resp.  $\mathcal{F}(\mathcal{A}, G) \cap \mathcal{E}$ ). Then, we define the inclined score as follows.

**Definition 4 (inclined score).** *Given a graph  $G$ , an anchored node set  $\mathcal{A}$ , and the close and conflict node sets  $F$  and  $\mathcal{E}$ , the inclined score of  $\mathcal{A}$ , denoted as  $Score(\mathcal{A}, G)$ , is the difference between the number of close followers and that of conflict followers, i.e.,  $Score(\mathcal{A}, G) = \mathcal{F}^+(\mathcal{A}, G) - \mathcal{F}^-(\mathcal{A}, G)$ .*

We propose the concept of inclined score to better judge the effectiveness of anchored nodes by considering the inclination. When the context is clear, we omit the second parameter, i.e.,  $G$ , from the notations.

**Problem Statement.** Given a graph  $G$ , the close and conflict node sets  $F$  and  $\mathcal{E}$ , the degree constraint  $k$  and a budget  $b$ , the inclined anchored  $k$ -core problem aims to anchor a set of  $b$  nodes  $\mathcal{A}^*$  with the largest inclined score, i.e.,

$$\mathcal{A}^* = \arg \max_{\mathcal{A} \subseteq G \wedge |\mathcal{A}|=b} \text{Score}(\mathcal{A})$$

*Example 2.* Figure 1 is a toy network with 12 nodes. Assume  $k = 3$  and  $b = 1$ . The close and conflict node sets are  $F = \{u_1, u_2, u_3, u_{12}\}$  and  $\mathcal{E} = \{u_9, u_{10}, u_{11}\}$ , respectively. By anchoring the node  $u_1$ , we can obtain the anchored  $k$ -core  $\{u_1, u_2, \dots, u_8\}$  with inclined score 3. The inclined score is  $-2$ , if we anchor  $u_{12}$ .

According to Theorems 1 and 2, the inclined anchored  $k$ -core problem is NP-hard, and the objective function is non-monotonic and non-submodular.

**Theorem 1.** *The inclined anchored  $k$ -core problem is NP-hard for  $k \geq 3$ .*

*Proof.* When there are only close nodes in the graph, the inclined anchored  $k$ -core problem can be reduced to the traditional anchored  $k$ -core problem, which is NP-hard for  $k \geq 3$  [1]. Hence, the inclined anchored  $k$ -core problem studied in this paper is also NP-hard.

**Theorem 2.** *The objective function is non-monotonic and non-submodular.*

*Proof. Non-monotonic.* We prove  $\text{Score}(X)$  is non-monotonic by constructing a counter example. Reconsider the graph in Fig. 1. For  $k = 3$ ,  $\mathcal{E} = \{u_9, u_{10}, u_{11}\}$  and  $F = \{u_1, u_2, u_3, u_{12}\}$ , suppose  $\mathcal{A} = \{u_3\}$ . We have  $\text{Score}(\mathcal{A}) = 2$ . By adding  $u_{12}$  to  $\mathcal{A}$ , the inclined score becomes 0. While, the score becomes 3, if we add  $u_1$  to  $\mathcal{A}$ . Thus, the objective function is non-monotonic.

*Non-submodular.* Given two sets  $A$  and  $B$ , we say the function  $\text{Score}(X)$  is submodular if  $\text{Score}(A) + \text{Score}(B) \geq \text{Score}(A \cup B) + \text{Score}(A \cap B)$ . We also show the inequality does not hold by constructing a counter example. In Fig. 1, for  $k = 3$ ,  $\mathcal{E} = \{u_9, u_{10}, u_{11}\}$  and  $F = \{u_1, u_2, u_3, u_{12}\}$ , suppose  $A = \{u_{10}\}$  and  $B = \{u_{12}\}$ . We have  $\text{Score}(A) = -2$ ,  $\text{Score}(B) = -2$ ,  $\text{Score}(A \cap B) = 0$  and  $\text{Score}(A \cup B) = -2$ . Hence, the inequation does not hold. Therefore, the theorem is correct.

### 3 Solution

In this section, a baseline searching framework is firstly proposed. Then, optimized solutions are developed to accelerate the search.

#### 3.1 Baseline Algorithm

For the inclined anchored  $k$ -core problem, an exact solution is to enumerate all possible node sets  $\mathcal{A}$  with size  $b$  and compute the corresponding result, then we can get the optimal solution. However, the exact algorithm is time-consuming. Due to the complexity of the inclined anchored  $k$ -core problem, we resort to the greedy heuristic as the traditional anchored  $k$ -core solution does [21]. The details are shown in Algorithm 1. It iteratively finds the node with the largest score in current  $k$ -core. It is easy to verify that we only need to consider the nodes not in the  $k$ -core as candidates.



---

**Algorithm 1:** Baseline Algorithm

---

**Input** :  $G$ : a graph,  $k$ : degree constraint,  $b$ : the budget  
**Output** :  $\mathcal{A}$ : the set of anchored nodes

```

1  $\mathcal{A} \leftarrow \emptyset$ ;
2 for  $i$  from 1 to  $b$  do
3   for each  $u \in G \setminus \{\mathcal{A} \cup C_k(G)\}$  do
4      $\lfloor$  compute  $Score(u, G)$ ;
5      $u^* \leftarrow$  the node with the maximum score;
6      $\mathcal{A} \leftarrow \mathcal{A} \cup \{u^*\}$ ;
7 return  $\mathcal{A}$ 

```

---

### 3.2 Follower Computation

Because of the non-monotonic property, the pruning techniques developed for the traditional anchored  $k$ -core problem are no longer held for our inclined case. For our problem, an essential task is to compute the followers of an anchored node.

**Lemma 1.** *By anchoring a node, all of its followers are from  $\mathcal{P}_{k-1}$ .*

*Proof.* By anchoring a node, the coreness of a node increases at most 1 [21]. Therefore, if one node  $u$  is not in  $(k - 1)$ -core, then anchoring a node  $x$  can increase the coreness of  $u$  to  $k - 1$  at most, and  $u$  cannot appear in the  $k$ -core. Thus, it cannot be the follower of  $x$ .

**Peel Layer.** Based on the definition of  $(k - 1)$ -peel, we can divide the nodes in  $\mathcal{P}_{k-1}$  into different layers. Motivated by the onion layer structure [21], we recursively remove the node with degree less than  $k$  and organize the nodes in  $\mathcal{P}_{k-1}$  in a peel layer structure  $P$ . Thus, the set of nodes which are deleted in the  $i$ -th batch belong to the  $i$ -layer, denoted by  $P_i$ . Specifically,  $P_1 = \{u \mid deg(u, C_{k-1}(G)) < k \wedge u \in C_{k-1}(G)\}$ . Due to the deletion of  $P_1$ , we can get  $P_2$  in the same manner. Similarly, we recursively get all  $P_i$ . So, the peel layer structure  $P = \bigcup_{i=1}^t P_i$ , where  $t$  represents the recursion times. We use  $p(u)$  to denote the layer index of a node  $u$ , i.e.,  $p(u) = i$  when  $u \in P_i$ .

According to the peel layer structure  $P$  and Lemma 1, we have that the valid candidate anchored nodes with at least one follower must belong to  $P$ , which can significantly reduce the size of candidates. We propose an effective candidate pruning technique based on peel layer structure based on Observation 1. To better explain the pruning rule, we first represent the definition of ladder path.

**Definition 5 (ladder path).** *Given an anchored node  $x$ , there is a ladder path from  $x$  to  $u$ , denoted by  $x \rightsquigarrow u$ , where (i) all nodes are from  $P$ ; (ii)  $p(v) < p(w)$  for every two consecutive nodes  $v$  and  $w$  along this path.*

**Algorithm 2:** Radiate Search

---

**Input** :  $x$ : the candidate anchor;  $P$ : layer structure  
**Output** :  $\mathcal{F}$ : the followers of  $x$

- 1 the status of  $x$  is set received;
- 2 **for** each node  $v$  in  $V(G) \setminus \{x\}$  **do**
- 3      $q \leftarrow$  the number of neighbors of  $v$  in  $C_k(G)$ ;
- 4      $dis(v) = k - q$ ;  $v$  is set listening;
- 5  $Q \leftarrow \emptyset$ ;
- 6 **for** each  $w \in N(x) \cup P$  and  $p(w) > p(x)$  **do**
- 7      $Q.push(w)$  ;
- 8      $dis(w) = dis(w) - 1$  ;  $dis(x) = dis(x) - 1$ ;
- 9     **if**  $dis(w) \leq 0$  **then**  $w$  is set received;
- 10 **while**  $Q \neq \emptyset$  **do**
- 11      $u \leftarrow Q.pop()$ ;
- 12     **if**  $dis(u) > a(u) + r(u) + l(u)$  **then**
- 13          $u$  is set closed; SHRINK( $u$ );
- 14     **else**
- 15          $u$  is set activated;
- 16         **for** each listening node  $z$  in  $N(u)$  **do**
- 17             **if**  $z$  is not in  $C_k(G)$  and  $p(z) > p(u)$  **then**
- 18                  $Q.push(z)$ ;
- 19                  $dis(u) = dis(u) - 1$ ;
- 20                  $dis(z) = dis(z) - 1$ ;
- 21                 **if**  $dis(z) \leq 0$  **then**  $z$  is set received;
- 22 **return** received nodes in  $P$

---

**Observation 1.** Given a graph  $G$ , if a candidate anchor  $x$  has at least one follower  $u$ , we have that there exists a ladder path  $x \rightsquigarrow u$ .

Clearly, if there is no one ladder path  $x \rightsquigarrow u$  for a node  $x$ , it cannot be considered as a candidate anchor. Then, we compute the followers for each candidate anchor based on the radiate search after candidate reduction. We give the definition involved before presenting the details.

**Definition 6. (distance).** Give a graph  $G$  and a node  $u$ , the distance of  $u$ , denoted as  $dis(u)$ , is the difference between degree constraint  $k$  and the number of neighbors of  $u$  in  $k$ -core, i.e.,  $dis(u) = k - deg(u, C_k(G))$ . The distance of  $u$  represents that  $u$  still need  $dis(u)$  nodes to satisfy the requirement of degree not less than  $k$ .

**Radiate Search.** We find the followers of anchors by radiate search, which is similar to breadth-first search. During the search procedure, each node has four states. When we start to anchor a node  $u$ , then other nodes will be **listening**  $l(v)$ . Due to the degree of anchored node  $u$  is infinity, which satisfies the degree

**Algorithm 3:** Shrink( $u$ )

---

```

Input   :  $u$ : the node for degree check
1 for each activated neighbor  $v$  of  $u$  do
2    $dis(v) = dis(v) + 1$ ;
3   if  $dis(v) > a(v) + r(v) + l(v)$  then
4      $v$  is set closed; SHRINK( $v$ );
5 for each received neighbor  $v$  of  $u$  do
6    $dis(v) = dis(v) + 1$ ;
7   if  $dis(v) > 0$  then  $v$  is set activated;
8   if  $dis(v) > a(v) + r(v) + l(v)$  then
9      $v$  is set closed; SHRINK( $v$ );

```

---

constraint, i.e.  $dis(u) \leq 0$ ,  $u$  is **received**  $r(u)$ . A node is **activated**  $a(v)$  when its neighbor is received or activated. In addition, if  $p(v) \geq p(u)$ , then the node  $v$  cannot be activated based on the layer structure  $P$ . After anchoring a node  $u$ , the distance of each neighbor  $v$  of  $u$  subtract 1, and we check whether the distance of  $v$  is 0. If the distance is 0, then  $v$  is received, so we continue to check the neighbors of  $v$ . Otherwise, we need to check that whether the neighbors of  $v$  can participate in the  $k$ -core to decrease  $dis(v)$ . Note that, if  $dis(v) > a(v) + r(v) + l(v)$ , then the distance of  $v$  cannot be 0, thus,  $v$  is **closed**  $c(v)$ . The details are described in Algorithm 2.

At first, we set the status of anchored node  $x$  as received (Line 1). Then, we process all other nodes in  $G$  (Lines 2–4). For each node, we compute the degree of  $v$  in  $C_k(G)$  (Line 3) and obtain the distance of  $v$ . Then, we set the status of  $v$  as listening (Line 4). After that, we set queue  $Q$  as empty in Line 5. In Lines 6–9, for each node  $w$  who is the neighbor of  $x$  or belongs to layer structure  $P$  with  $p(w) > p(x)$ , we push it into the queue  $Q$  (Line 7). Then, we update the distance for both  $w$  and  $x$  by subtracting 1 (Line 8). If the distance of  $w$  is no larger than 0, we set the status of it as received (Line 9). In Lines 10–21, we process all nodes in  $Q$  until it becomes empty. In Line 11, we pop a node  $u$  from  $Q$ . If the distance of  $u$  is larger than  $a(u) + r(u) + l(u)$  which means the node  $u$  cannot be the follower, we set the status of  $u$  as closed, and use the SHRINK of Algorithm 3 to update its neighbors' distance and status (Lines 12–13). Otherwise, we set the status of  $u$  as activated (Line 15) and process each listening node  $z$  who is the neighbors of  $u$  (Lines 16–21). If node  $z$  is not in the  $k$ -core and  $p(z) > p(u)$ , we push  $z$  into  $Q$  and update the distance  $u$  and  $z$  by subtracting 1. Then, we judge the distance of  $z$ . If  $dis(z)$  is no larger than 0, we set the status of  $z$  as received. Finally, we return all received nodes in  $P$  until  $Q$  is empty.

*Example 3.* As shown in Fig. 2, given the part of a graph  $G$  and the distance  $d(u)$  for each  $u$ , when we anchor the node  $u_3$ , then  $u_3$  is received and its neighbors are activated and other nodes are listening. The distance of  $u_2$  subtract 1, i.e.,  $d(u_2) = 4$  and the distance of neighbors  $\{u_5, u_7, u_4, u_1\}$  of  $u_2$  subtract 1. We have that  $d(u_5) = 0$ ,  $d(u_2) = 0$ . Thus, the node  $u_2$  and  $u_5$  are received. Similarly,

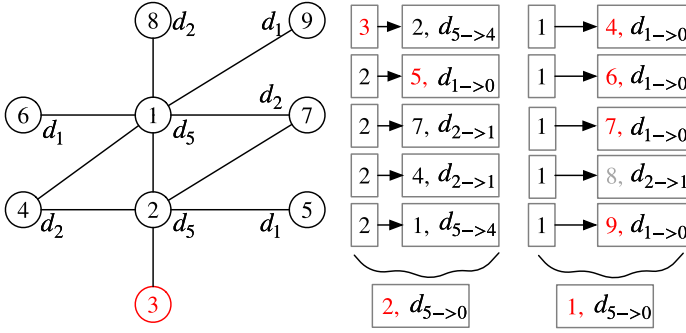


Fig. 2. Example of radiate search

---

#### Algorithm 4: KSM Algorithm

---

**Input** :  $G$ : a graph,  $k$ : degree constraint,  $b$ : the budget

**Output** :  $\mathcal{A}$ : the set of anchored nodes

```

1  $\mathcal{A} \leftarrow \emptyset$ ;
2 Compute upper bounds of inclined score;
3 for  $i$  from 1 to  $b$  do
4    $\alpha \leftarrow -\infty$ ;
5   for each  $u$  in  $P$  with at least one ladder path do
6     if  $u \notin \mathcal{A}$  and  $f^+(u) > \alpha$  then
7       compute  $Score(u, G)$  based on Radiate Search with upper bound;
8       update  $\alpha$  if needed;
9    $u^* \leftarrow$  the nodes with maximum score;
10   $\mathcal{A} \leftarrow \mathcal{A} \cup \{u^*\}$ ;
11  Refine upper bounds;
12 return  $\mathcal{A}$ 

```

---

activating  $u_1$  can activate its neighbors and make their distance subtract 1. Finally, we find that all nodes are received except  $u_8$ .

### 3.3 Search Algorithm

Let  $CC(u)$  be the connected component of  $u$ . The number of close nodes in  $CC(u)$  is denoted by  $f^+(u)$ . Then, we can use  $f^+(u)$  to serve as an upper bound to filter the candidate. It is easy to verify the correctness of Lemma 2.

**Lemma 2.** *Let  $\alpha$  denote the largest marginal score currently after anchoring a candidate node. If  $f^+(u) < \alpha$ , we can remove  $u$  from the candidate.*

By further extending the result in Lemma 2, we can maintain a tighter upper bound when conducting the radiate search. That is, when exploring the layer structure, we can decrease the upper bound value  $f^+(u)$  by if we meet a node

in the conflict set. Then, we can terminate the computation when the updated upper bound violates Lemma 2. By integrating all optimization techniques, we present the KSM algorithm as shown in Algorithm 4. We skip a node  $u$ , if  $f^+(u)$  is no large than  $\alpha$ . Otherwise, we conduct a radiate search that integrates the extended Lemma 2. At the end of current iteration, we have the best anchor  $u^*$  with maximum score and merge it into  $\mathcal{A}$ . Lastly, Algorithm 4 returns the set  $\mathcal{A}$  of  $b$  anchored nodes after  $b$  iterations.

## 4 Experiments

In this section, comprehensive experiments are conducted over 9 datasets to evaluate the efficiency and effectiveness of proposed techniques.

**Table 2.** Statistics of datasets

Dataset	Nodes	Edges	$d_{avg}$	$d_{max}$
Artificial	496	3,971	16	89
Eco-mahindas	1,258	7,513	12	206
Soc-hamsterster	2,426	16,630	14	273
Email	1,005	16,064	32	345
Facebook	4,039	88,234	44	1045
Brightkite	58,228	214,078	7.4	1134
Arxiv	34,546	420,877	24.4	846
Gowalla	196,591	950,327	10	14,730
YouTube	1,134,890	2,987,624	5	28,754

### 4.1 Experiment Setup

**Algorithms.** In the experiments, we implement and compare the following algorithms.

- **Rand:** select  $b$  nodes that are not in the  $k$ -core randomly.
- **Exact:** enumerate all possible combinations with the optimal result.
- **Traditional:** obtain the node set with traditional anchored  $k$ -core model.
- **BL:** the baseline greedy algorithm, i.e., Algorithm 1.
- **KSM:** the optimized algorithm i.e., Algorithm 4.

**Datasets and Workloads.** We conduct experiments on 8 real-world networks, which are public available in Networkrepository<sup>1</sup> and SNAP.<sup>2</sup> We also employ 1 artificial graph, which is generated by GTGraph with 500 nodes and 5000 edges.

<sup>1</sup> <http://networkrepository.com>.

<sup>2</sup> <http://snap.stanford.edu>.

Table 2 shows the statistic details of the datasets. Note that, nodes in the original graphs have no close or conflict properties, so we use a method similar to BFS to assign the labels to each node in the datasets. For each connected component in the original graph, we randomly select an initial node and regard it as a close, and each neighbor of the initial node has a 50% probability of being close, a 20% probability of being conflict, and a 30% probability of being unchanged (i.e., no label). For a unchanged node, each of its neighbor is set as unchanged. Then, we repeat the process for the encountered nodes in BFS. In our experiment, both  $k$  and  $b$  vary from 5 to 25. For each setting, we run the algorithm 20 times and take the average performance as the final result.

**Table 3.** Inclined score ratio of Exact and KSM

Ratio	$b = 1$		$b = 2$		$b = 3$	
	Exact	KSM	Exact	KSM	Exact	KSM
Artificial	100%	100%	100%	100%	100%	100%
Eco-mahindas	100%	100%	100%	100%	100%	100%
Soc-hamsterster	100%	100%	100%	100%	100%	97%

**Table 4.** Response time of Exact and KSM

Running time	$b = 1$		$b = 2$		$b = 3$	
	Exact	KSM	Exact	KSM	Exact	KSM
Artificial	0.0005	0.00005	0.1131	0.0001	18.44	0.0002
Eco-mahindas	0.0097	0.0006	5.8537	0.0011	2510	0.0016
Soc-hamsterster	0.0435	0.0001	50.913	0.0003	42985.6	0.0005

## 4.2 Effectiveness Evaluation

To evaluate the effectiveness of algorithms, we report the ratio of inclined score of KSM and Exact by anchoring  $b$  nodes. This is because KSM only enhances the efficiency compared to BL. Then, we compare the results obtained by the traditional anchored  $k$ -core and our inclined anchored  $k$ -core problems. In addition, case studies are also conducted.

**Compared with Exact** . Table 3 reports the inclined score ratio of KSM and Exact by anchoring  $b$  nodes, where  $k = 10$  and  $b$  varies from 1 to 3. We set the inclined score ratio of Exact as 100% and that of KSM is  $\frac{\text{score returned by KSM}}{\text{score returned by Exact}} \times 100\%$ . Note that, due to the high computational cost of Exact, we only test Exact on small datasets with small  $b$  values. As observed, KSM only slightly drops in one settings. In addition, Table 4 reports the running time of Exact and KSM. We can see KSM achieves significant speedup compared with Exact, which further verifies the effectiveness of the greedy framework.

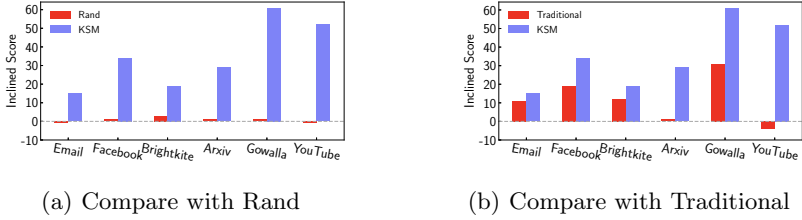


Fig. 3. Effectiveness evaluation compared with Rand and Traditional

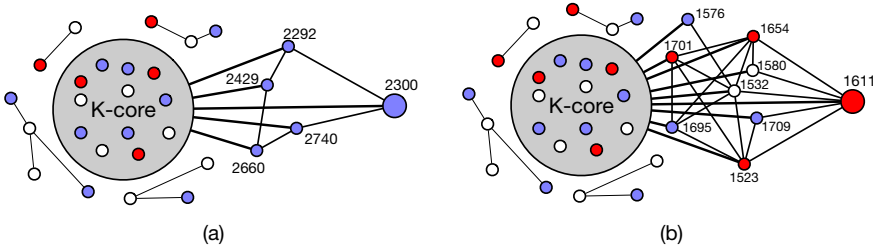


Fig. 4. Case study

**Compared with Rand.** In Fig. 3(a), we report the inclined score compared with Rand on 6 larger datasets, i.e., Email, Facebook, Brightkite, Arxiv, Gowalla, YouTube with  $b = 10$  and  $k = 10$ . The datasets are ordered by their network sizes, i.e., the number of edges. Note that, the larger the inclined score is, the more effective the algorithm will be. As we can see, KSM outperforms Rand by a big margin. This is because, there is usually few followers for anchored nodes that are selected in random.

**Compared with Traditional.** We report the results by compared the inclined model with the traditional model. The corresponding inclined score are shown in Fig. 3(b). Traditional does not consider the properties of different nodes, and only focuses on enlarging the total  $k$ -core. As observed, Traditional may lead to a very small inclined score, even a negative score, such as Youtube. Thus, it is necessary to develop algorithms to handle the inclined case.

**Case Study.** We conduct a case study on Facebook dataset with  $k = 20$  and  $b = 1$ . Figure 4(a) and Fig. 4(b) are the results obtained by KSM and Traditional, respectively, where blue/red nodes denotes the close/conflict nodes, and white nodes are the ones without labels. As shown in Fig. 4(a), the best anchored node is the one with id 2300. It has 5 followers and all of them are close nodes, i.e., the inclined score is 5. As shown in Fig. 4(b), the returned anchor is with id 1611. It has 9 followers, where there are 3 close and 4 conflict nodes, i.e., the inclined score is  $-1$ . Compared with the anchored node 2300, though the node 1611 has more followers, it has smaller inclined score.

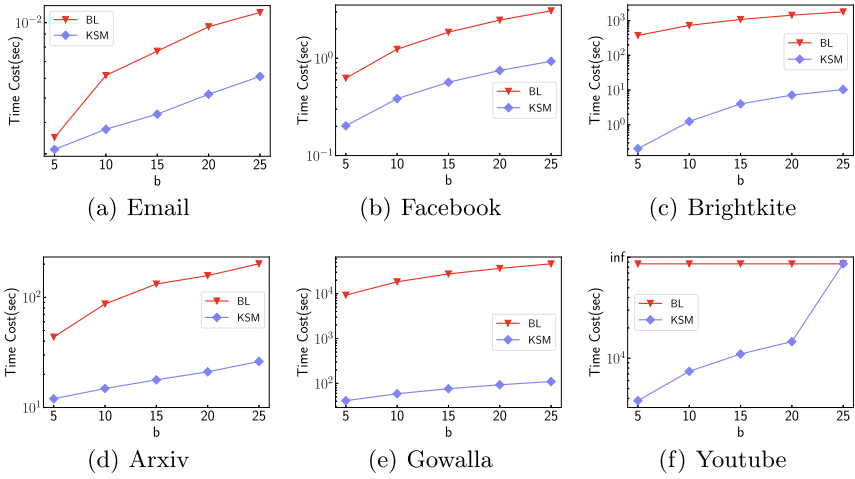


Fig. 5. Efficiency evaluation by varying  $b$

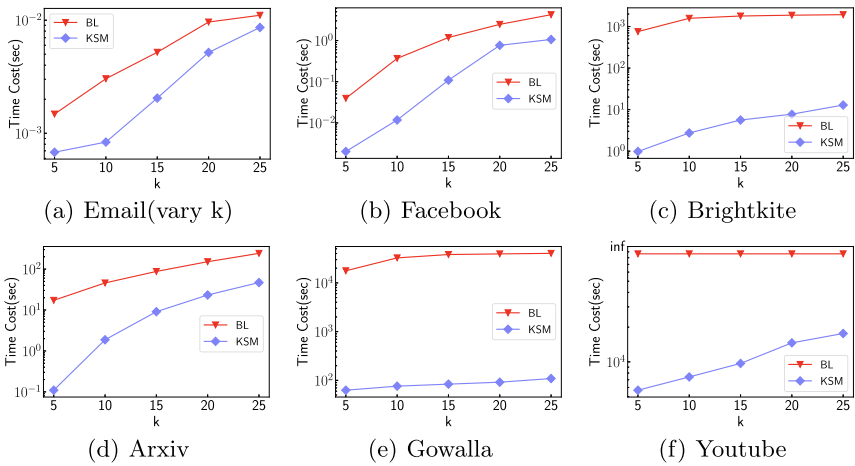


Fig. 6. Efficiency evaluation by varying  $k$

### 4.3 Efficiency Evaluation

To evaluate the efficiency of our algorithms, we report the response time of BL and KSM by varying  $b$  and  $k$  on six larger datasets. The results are shown in Figs. 5 and 6, respectively. It is clear that KSM constantly outperforms BL on all datasets, and can achieve up to 2 orders of magnitude speedup. As observed, the response time of both methods increases when  $b$  increases. This is because more iterations need to be conducted. When  $k$  increases, the response time also grows. This is because when  $k$  is larger, the nodes in the layer structure have



a larger degree, which leads to the need to explore more neighbors and greater time consumption.

## 5 Related Work

In social network analysis, different cohesive subgraph models have been proposed to accommodate different scenarios, such as  $k$ -core [14, 15],  $k$ -truss [9, 23], clique [3, 17], etc. The  $k$ -core model is firstly proposed by Seidman [14], which has been widely adopted for social network analysis with numerous applications, such as protein function prediction [19], social contagion [18], influence study [10], etc. Numerous studies of different topics have been investigated based on the  $k$ -core model. Bhawalkar et al. [1] propose the anchored  $k$ -core problem and prove its hardness, which aims to maximize the size of  $k$ -core by anchoring  $b$  nodes. In [21], Zhang et al. develop an efficient algorithm for the anchored  $k$ -core problem on large-scale graphs. [7] proposes the directed anchored  $k$ -core problem for directed graphs. Zhang et al. [22] study the collapsed  $k$ -core problem by removing  $b$  nodes. In [4, 24, 26], authors aim to find important edges to maximize and minimize the corresponding  $k$ -core by adding and deleting edges from the graph. In [12, 13], authors consider the  $k$ -core minimization problem based on the game theory model. The collapse problems are considered with bipartite settings in [5, 25]. In [27], Zhu et al. investigate the minimization problem based on the  $k$ -truss model. However, none of them considers the different property of nodes, i.e., close or conflict.

## 6 Conclusion

To measure the inclination and stability of social network, in this paper, we propose and study the inclined anchored  $k$ -core problem by considering the node properties, i.e., close or conflict. The problem aims to anchor a set of  $b$  nodes such that the inclined score is maximized, i.e., more close nodes but fewer conflict nodes engage in the  $k$ -core. We prove the problem is NP-hard. Due to the complexity of the problem, a greedy framework is presented. Optimized techniques are further developed to enhance the computation. Finally, comprehensive experiments on real-life networks are conducted to verify the effectiveness and efficiency of the developed techniques.

**Acknowledgments.** Chen Chen, Xijuan Liu and Shuangyan Xu are joint first authors. This work is supported by NSFC 61802345, ZJNSF LQ20F020007, ZJNSF LY21F020012 and Y202045024.

## References

1. Bhawalkar, K., Kleinberg, J., Lewi, K., Roughgarden, T., Sharma, A.: Preventing unraveling in social networks: the anchored  $k$ -core problem. *SIAM J. Discrete Math.* **29**(3), 1452–1475 (2015)

2. Burke, M., Marlow, C., Lento, T.: Feed me: motivating newcomer contribution in social network sites. In: SIGCHI, pp. 945–954 (2009)
3. Chen, C., Wu, Y., Sun, R., Wang, X.: Maximum signed  $\theta$ -clique identification in large signed graphs. TKDE (2021)
4. Chen, C., Zhu, Q., Sun, R., Wang, X., Wu, Y.: Edge manipulation approaches for k-core minimization: metrics and analytics. TKDE (2021)
5. Chen, C., Zhu, Q., Wu, Y., Sun, R., Wang, X., Liu, X.: Efficient critical relationships identification in bipartite networks. WWW J. (2021)
6. Cheng, D., Chen, C., Wang, X., Xiang, S.: Efficient top-k vulnerable nodes detection in uncertain graphs. TKDE (2021)
7. Chitnis, R., Fomin, F.V., Golovach, P.A.: Parameterized complexity of the anchored k-core problem for directed graphs. Inf. Comput. **247**, 11–22 (2016)
8. Chitnis, R.H., Fomin, F.V., Golovach, P.A.: Preventing unraveling in social networks gets harder. In: AAAI (2013)
9. Cohen, J.: Trusses: Cohesive subgraphs for social network analysis. National security agency technical report (2008)
10. Kitsak, M., et al.: Identification of influential spreaders in complex networks. Nat. Phys. **6**(11), 888–893 (2010)
11. Malliaros, F.D., Vazirgiannis, M.: To stay or not to stay: modeling engagement dynamics in social graphs. In: CIKM, pp. 469–478 (2013)
12. Medya, S., Ma, T., Silva, A., Singh, A.: A game theoretic approach for k-core minimization. In: International Conference on Autonomous Agents and MultiAgent Systems (2020)
13. Medya, S., Ma, T., Silva, A., Singh, A.: A game theoretic approach for core resilience. In: IJCAI (2020)
14. Seidman, S.B.: Network structure and minimum degree. Soc. Netw. **5**(3), 269–287 (1983)
15. Sun, R., Chen, C., Wang, X., Wu, Y., Zhang, M., Liu, X.: The art of characterization in large networks: finding the critical attributes. WWW J. (2021)
16. Sun, R., Chen, C., Wang, X., Zhang, Y., Wang, X.: Stable community detection in signed social networks. TKDE (2020)
17. Sun, R., Zhu, Q., Chen, C., Wang, X., Zhang, Y., Wang, X.: Discovering cliques in signed networks based on balance theory. In: Nah, Y., Cui, B., Lee, S.-W., Yu, J.X., Moon, Y.-S., Whang, S.E. (eds.) DASFAA 2020. LNCS, vol. 12113, pp. 666–674. Springer, Cham (2020). [https://doi.org/10.1007/978-3-030-59416-9\\_43](https://doi.org/10.1007/978-3-030-59416-9_43)
18. Ugander, J., Backstrom, L., Marlow, C., Kleinberg, J.: Structural diversity in social contagion. Proc. Natl. Acad. Sci. **109**(16), 5962–5966 (2012)
19. Vazquez, A., Flammini, A., Maritan, A., Vespignani, A.: Global protein function prediction from protein-protein interaction networks. Nat. Biotechnol. **21**(6), 697–700 (2003)
20. Wu, S., Das Sarma, A., Fabrikant, A., Lattanzi, S., Tomkins, A.: Arrival and departure dynamics in social networks. In: International Conference on Web Search and Data Mining, pp. 233–242 (2013)
21. Zhang, F., Zhang, W., Zhang, Y., Qin, L., Lin, X.: OLAK: an efficient algorithm to prevent unraveling in social networks. PVLDB **10**(6), 649–660 (2017)
22. Zhang, F., Zhang, Y., Qin, L., Zhang, W., Lin, X.: Finding critical users for social network engagement: the collapsed k-core problem. In: AAAI (2017)
23. Zhao, J., Sun, R., Zhu, Q., Wang, X., Chen, C.: Community identification in signed networks: a k-truss based model. In: CIKM (2020)
24. Zhou, Z., Zhang, F., Lin, X., Zhang, W., Chen, C.: K-core maximization: an edge addition approach. In: IJCAI (2019)

25. Zhu, Q., Zheng, J., Yang, H., Chen, C., Wang, X., Zhang, Y.: Hurricane in bipartite graphs: the lethal nodes of butterflies. In: SSDBM (2020)
26. Zhu, W., Chen, C., Wang, X., Lin, X.: K-core minimization: an edge manipulation approach. In: CIKM, pp. 1667–1670 (2018)
27. Zhu, W., Zhang, M., Chen, C., Wang, X., Zhang, F., Lin, X.: Pivotal relationship identification: the k-truss minimization problem. In: IJCAI (2019)



# LPMA - An Efficient Data Structure for Dynamic Graph on GPUs

Fan Zhang<sup>(✉)</sup>, Lei Zou, and Yanpeng Yu

Peking University, Beijing 100080, China  
{zhangfanau, zoulei, yuyanpeng}@pku.edu.cn

**Abstract.** There is a growing interest to offload dynamic graph computation to GPU and resort to its high parallel processing ability and larger memory bandwidths compared with CPUs. The existing GPU graph systems usually use compressed sparse row (CSR) as de-facto structure. However, CSR has a critical weakness for dynamic change due to the large overhead of re-balance process after update. GPMA+ is the state-of-art dynamic CSR-oriented structure that uses PMA structure and optimized segment-oriented parallel update procedure to address the dynamic weakness of CSR but still has a bottleneck on the array expansion. In this paper, we propose an optimized dynamic structure LPMA, which is a leveled structure instead of continues array to retain low time complexity and high parallel update and lift the expansion bottleneck of GPMA+. Theoretical analysis and extensive experiments on four real-life large graphs prove the superiority of LPMA.

**Keywords:** Dynamic graph · GPU Parallel · CSR based

## 1 Introduction

To address the great challenges due to high dynamicity and complexity of graph algorithms, one way is to employ hardware assist. There is a growing interest to offload dynamic graph computation to GPU and resort to its high parallel processing ability and larger memory bandwidths compared with CPUs. Thus, in this paper, we propose an efficient GPU-oriented dynamic graph system. In our system, the updates and queries are received and buffered by the CPU part, which sends the batches of updates and queries to the GPU part periodically. The GPU part maintains a continuously updating graph and queries are conducted over the dynamic graphs in GPU.

Generally, there are two categories in designing GPU-based dynamic graph data structures to address the challenges: neighbor list based and compressed sparse row (CSR) based. The neighbor list based structures [5, 14] maintain a nodes array and a neighbor list for each node. The main contribution of the neighbor list based structures is to design a variable array or list to store the dynamic neighbor lists. The neighbor list based structures could be sorted or unsorted. The CSR based structures [9, 12] maintain a variable and sorted edge array to store the dynamic graph. The benefit of CSR based structures is the generality of supporting off-the-shelf GPU graph libraries (such as Gunrock [13]) and algorithms [8, 15], since most of them rely on CSR format and the ordered edge array facilitates converting a snapshot of dynamic graphs into CSR representation.

In this paper, we focus on designing a CSR based dynamic graph data structure on GPU. As we know, CSR is the de-facto graph representation in existing GPU graph systems (e.g., Gunrock [13] and Medusa [16]) due to compact representation, but CSR is a static data structure that stores each node’s neighbors consecutively. Any insertion or deletion will lead to  $O(|E|)$  data movement, where  $|E|$  is the number of edges in  $G$ . Because of the unpredictable, variable and skewed node degree distributions, we can not reserve a continuous space for the neighbor list of each node. The state-of-the-art of CSR based dynamic graph data structure is GPMA+ [12]. The main idea of GPMA+ is to maintain sorted edges in a contiguous fashion by reserving spaces to accommodate updates with a constant bounded gap ratio. Edge insertions are merged and re-distributed into the eligible segment range in a balanced fashion (called re-balance). With the increasing of edge insertions, GPMA+ will expand and re-allocate a double-size continuous memory space and re-balance the whole edge array. However, re-balancing in a large range of segments and expansion lead to up to  $20\times$  latency spikes which reduce the insertion throughput significantly. Deletions are the dual operation of insertions and have similar performance periodic jitter issue. We omit the discussion about deletion to reduce the repetition. Section 2.2 gives more detailed performance analysis of GPMA+.

In order to address the above bottlenecks, we propose a novel leveled data structure to maintain the sorted edge array in this paper, called *leveled packed memory array* (LPMA). Instead of storing all edges in a continuous space in GPMA+, LPMA partitions the whole sorted edge array into different physical levels and stores them in a perfect binary tree which all interior nodes have two children and all leaves have the same depth or same level. Different from GPMA+, LPMA maintains the edge ordering according to in-order traversal over the binary tree and does not requires a physically continuous memory space to accommodate all edges. The leveled structures in LPMA have two benefits. First, during expansion, we only need to allocate a new level and append it to the current LPMA tree rather than re-allocating a double-size continuous space in GPMA+. It alleviates memory allocation cost and memory fragmentation issue in GPMA+. Second, after expansion, LPMA employs a *localized re-balance* strategy that reduces the re-balancing cost significantly. GPMA+ always performs the *global re-balance*, collecting all edges and re-distributing them in the whole expanded space, but LPMA re-balances edges between some local consecutive edge segments. Obviously, our method can reduce the data movement during re-balancing. Theoretical analysis that our method can save more than 90% re-balancing cost (Sect. 3.3) and experiment results also confirm that. To summarize, we made the following contributions.

- We propose a novel leveled data structure LPMA in the context of dynamic graphs. In order to address the bottleneck in existing solutions, we propose an efficient expansion and *Localized Re-balance* strategy.
- We theoretically analyze the benefit of our approach in terms of data movement saved by LPMA. We also study some GPU-friendly designs for LPMA.
- Extensive experiments on several large graphs confirm that LPMA outperforms the state-of-art methods significantly.

## 2 Preliminary

### 2.1 Problem Definition

**Definition 1 (Operation Stream).** An operation stream is a time-evolving sequence of operations  $\{\sigma_1, \sigma_2, \dots, \sigma_x\}$ , where each  $\sigma_i$  specifies an edge insertion or deletion or a query at time  $t_i$ . The snapshot of  $\mathcal{G}$  at  $t_i$  is a graph  $\mathcal{G}_{t_i}$ , which is the updated graph after performing all edge insertions and deletions before time  $t_i$  in the operation stream.

Inserting or deleting  $\vec{uv}$  with weight  $w_i$  at time  $t_i$  is denoted as  $\sigma_i(+/-, \vec{uv}, w_i, t_i)$ , where ‘ $+/-$ ’ denotes an edge insertion or deletion. To support inserting or deleting an isolated vertex, we assume that each vertex  $v$  has an inborn adjacent edge  $\vec{v}, \infty$ , where  $\infty$  is a pseudo vertex. Inserting or deleting an isolated vertex equals to inserting or deleting edge  $\vec{v}, \infty$ . Thus, the edge-oriented operation model can support all updates over graphs. For simplicity, we only study edge insertion and deletion in this paper. Given an original graph  $\mathcal{G}$  with 10 edges in Fig. 1a, after a sequence of insertions  $\{\sigma_1, \sigma_2, \dots\}$  (Fig. 1c), we can obtain the updated graph in Fig. 1d.

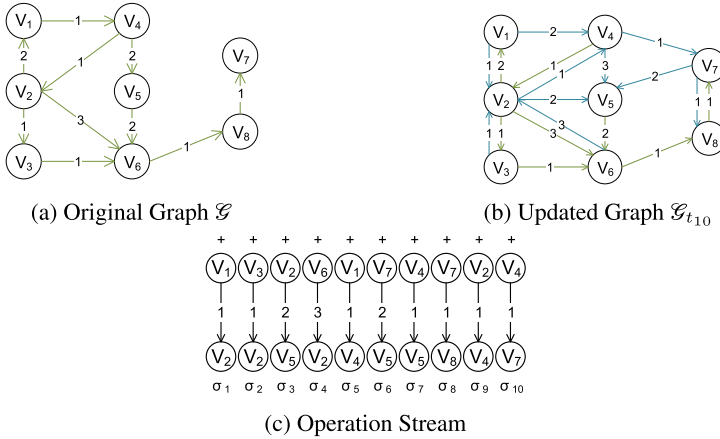


Fig. 1. Dynamic graph with operation stream

**Definition 2 (Query Primitives).** Each query primitive is denoted as  $q(C, t_i)$ , where  $C$  is a query condition and  $t_i$  defines the issue timestamp of query  $q$ . We define the following query primitives over graph snapshot  $\mathcal{G}_{t_i}$ :

1. **Edge Query:** If  $C = (u, v)$  where  $u$  and  $v$  are the node IDs, return  $\{(\vec{uv}, t, w) | (\vec{uv}, t, w) \in \mathcal{G}_{t_i}\}$ ; If the edge doesn't exist, return  $\phi$ .
2. **1-hop Successor Query:** If  $C = (u, \rightarrow)$  where  $u$  is a node ID, return  $\{(\vec{uv}, t, w) | (\vec{uv}, t, w) \in \mathcal{G}_{t_i}\}$ . If the edge doesn't exist, return  $\phi$ .
3. **1-hop Predecessor Query:** If  $C = (v, \leftarrow)$  where  $v$  is a node ID, return  $\{(\vec{uv}, t, w) | (\vec{uv}, t, w) \in \mathcal{G}_{t_i}\}$ . If the edge doesn't exist, return  $\phi$ .

Due to great parallel processing ability of GPU, in this paper, we aim to design an efficient *GPU-oriented data structure* to support high throughput updates and queries over the underlying graph. Different from CPU-oriented solutions, the dynamic graph on GPUs usually adopt the *batch-based* model [12]. The system buffer module batches edge updates on CPU side and periodically sends the updating batches to update module in GPU side. The update module process all edge updates (in one batch) simultaneously. There are two reasons of adopting batch based model in GPU-oriented dynamic graph processing. First, due to high parallel capability of GPU, the batch model enables GPU to process multiple updates and queries at the same time. Obviously, one operation-at-a-time (sequential model) limits parallel computing of GPU. Second, we need to minimize the cost of PCIe transferring on designing GPU-oriented systems. The one operation-at-a-time transferring model cannot reach the full capacity of PCIe bandwidth. For example, the 16-lane PCIe could reach  $64GB/s$  per second [11]. Formally, a batch defined as follows.

**Definition 3 (Operation Batch).** A batch  $B_t = \{\sigma_i\}$ , where  $\sigma_i$  specifies one edge insertion or deletion at time  $t_i$ . All operations sorted according to their associated timestamps. Furthermore, the batch size  $|B_t| \leq \theta$ , where  $\theta$  is a tuning parameter.

## 2.2 Existing CSR-Based Data Structure: GPMA+

The dynamic graph structures on GPUs could be divided into two categories: CSR based and neighbor list based. Since our proposed data structure (LPMA) belongs to the former one, to better understand the benefit of our method, we introduce GPMA+, a state-of-the-art CSR data structure for dynamic graphs. We review both categories in the related work section (Sect. 5). CSR [9] is the de-facto graph representation in existing GPU graph systems and applications (e.g., Gunrock [13] and Medusa [16]) due to compact representation, good memory locality and friendly supporting massive-parallel processing. CSR compresses an adjacent matrix (of a sparse graph) into three arrays: row offset array (corresponding to vertices), column array (corresponding to edges) and values array (corresponding to edge weights). However, the traditional CSR is costly for the dynamic updates to graphs, since all nodes' adjacent edges are stored consecutively in a sorted column array. Any edge insertion or deletion will lead to data movements in the sorted array. GPMA+ [12] adopts Packed Memory Array (PMA) [2,3] in CSR for dynamic graphs. PMA maintains a sorted array, but it leaves gaps to accommodate fast updates with a bounded gap ratio. GPMA+ extends PMA to GPU platform and use it to store the sorted arrays in CSR. They propose a segment-oriented operations to parallelize edge insertions/deletions in a lock-free model. Note that our proposed LPMA (leveled Packed Memory Array) is an optimized data structure compared with GPMA+.

Figure 2 shows an example of GPMA+ and segment-oriented update. GPMA+ divides the edge array into fixed size chunks known as *segments*. For simplicity, we assume that the lower and upper bound density threshold in each segment is  $[0, 80\%]$ . Given a batch of insert edges, GPMA+ sorts these edges according to source and destination nodes IDs. Then for each insert edge, GPMA+ assigns one thread to perform the binary search to locate the associated segments. In Fig. 2, the four associated segments are 0, 1, 2 and 3. GPMA+ initiates the first round update. A thread or a warp (depends

on the size of the segment) is called to handle one segment update. After the first round, segment 1 is updated successfully and the updates on segments 0, 2 and 3 fail due to the lack of space. For example, there are four insert edges that are located at segment 0, but there are only one empty position in segment 0. Edge  $\sigma_{14}$  updates the weight of the original edge  $\vec{v_1, v_4}$  but other three edges are new ones. Therefore, GPMA+ rolls the segment 0 up to the above level (i.e., Seglevel 1  $[0, 1]$ ), including segments 0 to 1 in the second round to probe empty space. Unfortunately, the second round also fails. Note that Seglevels 1, 2, 3 are index ranges that includes consecutive segments, which are not physical storage levels. Only Segleaf is a physical sorted array. In this example, even GPMA+ reaches the root Seglevel 2  $[0, 3]$  (including all segments in the ordinal array), there are no enough space to accommodate all insert edges.

Thus, GPMA+ triggers the space expansion that doubles the original sorted array (shown in the blue segments). Finally, GPMA+ merges all edges in the original array and insert edges; also re-balances these edges into eight segments in the expanded array (see Fig. 2). Since the re-balance involves all segments, we call it *global re-balancing*. GPMA+ handle the deletion as a dual process of insertion.

### 3 Leveled Packed Memory Array

In this section, we propose a novel data structure, called Leveled Packed Memory Array (LPMA). Actually, LPMA is designed to address costly expansion operation in GPMA+. We first discuss the bottleneck of expansion operation (Sect. 3.1), then propose our data structure (LPMA) (Sect. 3.2) and prove the superiority of LPMA (Sect. 3.3). Note that due to reduce the repetition, we only focus on insertion. Deletions will lead to *shrinking* of LPMA that is dual operation of expansion.

#### 3.1 Expansion in GPMA+

GPMA+ aims to maintain a continuous sorted array with public reserve space for future insertions. After re-balancing, new edges could be inserted into the reserve space. During the rebalancing, edges are sorted by nodes IDs. However, if new insert edges overflow the original array, GPMA+ will trigger the expansion. The expansion only happens when no enough reserve spaces in the whole original array can accommodate insert edges. Let us recall Fig. 2. After the third round insertion, each segment reaches the density threshold. GPMA+ doubles the sorted array from four segments to eight segments, and then inserts and re-balances edges to complete this update.

The expansion operation in GPMA+ is costly in two ways. Firstly, the whole array resize and re-balance requires massive memory access and reallocation. Since GPMA+ is stored in a continuous memory space, the system needs to allocate a double size memory space, re-balance the whole array, write into the new space and release the original one. The reallocation process could be costly when the edges number grows large. Furthermore, since GPMA+ always allocates new continuous memory space to hold all edges and release the original continuous memory space, this kind of allocation leads to a lot of potential memory fragmentation. Secondly, unnecessary data movements are



carried out during the re-balance. In each expansion & re-balance phase, GPMA+ collects all edges in the original array and merges them with insert edges. Finally, these edges are distributed to the whole new double-size array to balance the fragment density. Obviously, this is a global operation with  $O(|E|)$  complexity and many uninvolved segments also need to participate. Figure 2 shows an instance. In the first round, the insertion in the segment  $s_1$  is finished. In the fourth round, the segment  $s_1$  is re-balanced again after expansion.

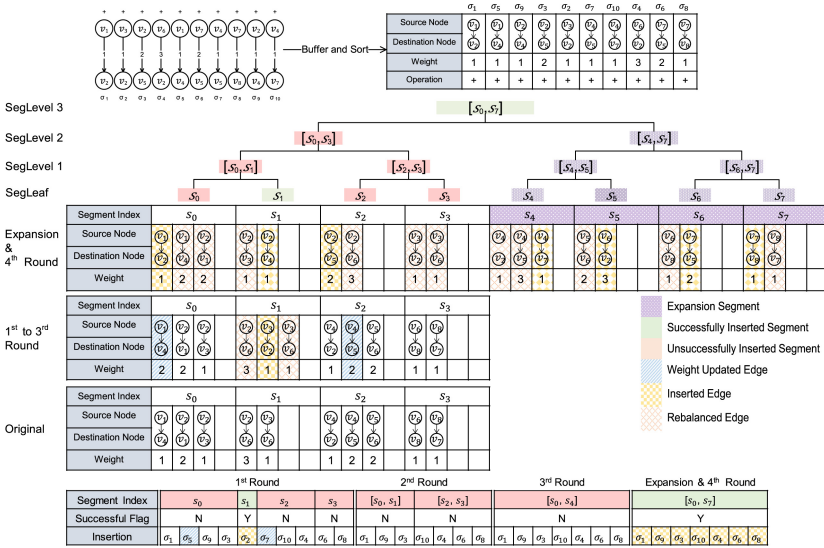


Fig. 2. GPMA+ update and expansion

### 3.2 LPMA Structure and Update

In order to reduce the expansion overhead in GPMA+, we propose a leveled structure LPMA to organize edge segments. Different from GPMA+, LPMA partitions edge segments into different levels. Except for the head level  $level_0$ , each  $x$ -th level ( $level_x$ ) ( $x = 1, \dots, n$ ) contains  $2^{x-1}$  segments, where segments in the same level are stored in a continuous memory space, but we do not store all levels consecutively. Logically, segments in each level except for  $level_0$  (in LPMA) form a *perfect binary tree*. The dash lines in Fig. 3 shows the tree's structure. Since LPMA is a perfect binary tree, we do not store the dash line (parent-child relation) physically in LPMA. The in-order traversal of the binary tree (LPMA) forms a sorted edge array, which corresponds to GPMA+. Note that head level  $level_0$  has a single segment that precedes all other segments. In a word, segments in LPMA form a sorted edge array through the in-order traversal, but it has different physical storage scheme from GPMA+.

Since LPMA is a perfect binary tree, it is easy to map the  $y$ -th segment in the  $x$ -th level (i.e.,  $level_x$ ) to the  $i$ -th segment in the sorted array and vice versa. We also call  $i$  as the sequential index of the segment. The following equations illustrates the process ( $m$  is the level number of LPMA and  $d$  is the greatest common divisor of  $n$  and  $2^m$ ):

$$i = \begin{cases} 0 & x = 1 \\ (2y + 1) \times 2^{m-x} & m \geq x \geq 2 \end{cases} \quad (1)$$

$$(x, y) = \begin{cases} (1, 0) & i = 0 \\ (m - d, \frac{i/2^d - 1}{2}) & i > 0 \end{cases} \quad (2)$$

Note that LPMA adopts the similar segment-oriented update procedure as GPMA+. Specifically, when the update batch arrives, LPMA first conducts binary search to locate the associated segments for insert edges. The system performs the segment oriented parallel insertion on the logical sorted array that is same with GPMA+. If the corresponding segment cannot accommodate insert edges, we will roll up the insertions. Figure 3 shows an update example of LPMA. Initially, LPMA has four segments. When edge insertion batch comes, the system sorts edges according to edge endpoint IDs and conducts binary search over LPMA to identify the associated segments. In the first round, new edges  $\sigma_5$  and  $\sigma_7$  updates corresponding original edges  $\overrightarrow{v_1v_4}$  and  $\overrightarrow{v_4v_5}$ , respectively. Edge  $\sigma_2 = \overrightarrow{v_3v_2}$  is successfully inserted into segment  $s_2$ . Other edge insertions fail due to insufficient space in the associated segments. All the updates and insertions are executed parallel. Then, we probe larger segment ranges ( $2^1$  and  $2^2$  segments) in the second and third round. Finally, there are still seven edges that cannot be inserted into the whole sorted array (LPMA) and they trigger expansion in LPMA.

LPMA is designed to address the expansion performance issue in GPMA+, thus, it has different expansion process. Recall Fig. 2. GPMA+ allocates a double-size sorted array, merges all original edges and insert ones, and then re-balances them in the extended array. After expansion, re-balancing in GPMA+ involves all edges and segments, thus, we call it *global re-balancing*. Furthermore, GPMA+ always needs a continuous memory space, the expansion often allocates a new double-size array and releases the original one. It not only wastes time in memory allocation and edge re-balancing but also leads to more memory fragmentation.

At expansion, LPMA only needs to allocate a new level and append it to the current tree. Figure 3 illustrates an expansion example, i.e., appending new  $level_3$ . After expansion, LPMA performs *localized re-balance* that does not always involve all segments. Since LPMA is a perfect binary tree, appending one new level, each segment (in the original LPMA tree) has an empty successor segment (shown in blue colour in Fig. 3) in the in-order traversal. Logically, after expansion, we append one empty segment after each original segment to obtain an extended LPMA. Figure 3 visualizes the sorted segments in the expanded LPMA. However, GPMA+ appends all new empty segments to the original sorted array (see Fig. 2).

In Fig. 3, there is one new empty segment  $s_4$  between  $s_0$  and  $s_2$ . Before expansion, seven edges still fail to be inserted. Edges  $\sigma_1, \sigma_9$  and  $\sigma_3$  should be inserted into segment

$s_0$ , but, there are insufficient spaces to hold three edges in segment  $s_0$ . After expansion,  $s_0$  has a new empty successor segment  $s_4$ . Therefore, we can re-balance six edges between  $s_0$  and  $s_4$ , including three edges in the original  $s_0$  and three insert edges. The re-balancing only happens between two consecutive segments, thus, we call it *localized re-balance*. The same happens over segment pairs  $(s_1, s_6)$  and  $(s_3, s_7)$  to handle edge insertions. In this example, segments  $s_2$  or  $s_5$  does not participate in re-balancing, which is different from global re-balance in GPMA+. Of course, if two consecutive segments cannot accommodate insert edges, we also need to roll up the insertion and check four consecutive segments. Although in the worst case all segments of LPMA participate in the re-balance process, which degrades to global re-balance in GPMA+, we theoretically prove that the worst case rarely happens and our method (LPMA) reduces the expected number of re-balancing segments significantly (see Lemma 1). Experiment results also confirm our analysis (see Subsect. 4.3).

**Query Processing:** Since LPMA maintains a sorted edge array, it is easy to answer query primitives (edge query and 1-hop successor/predecessor queries) using the binary search. Other graph queries can be decomposed into a series of query primitives that are evaluated over LPMA. Furthermore, to employ existing graph analysis library, we can also convert the sorted edge array in LPMA (corresponds to a graph snapshot) into CSR representation, since most GPU graph libraries are based on CSR format.

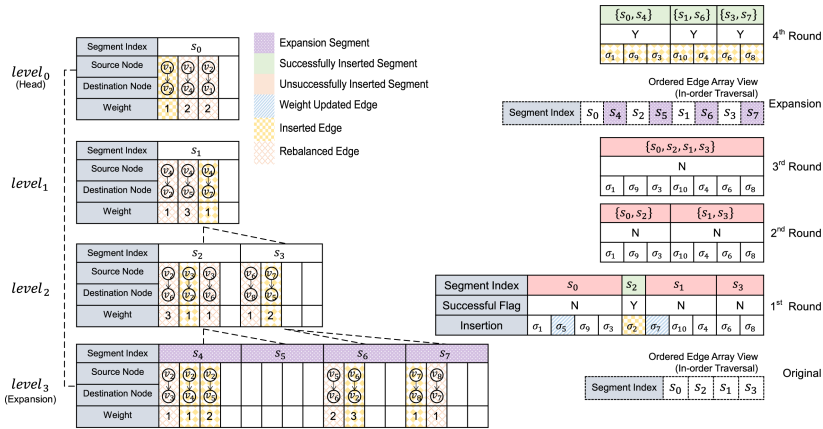


Fig. 3. LPMA update and expansion

### 3.3 Analysis

**Memory Efficiency:** Leveled memory allocation in LPMA does not affect memory access. Since GPU has smaller cache and supports parallel memory access, we can assign different warps to access different levels of LPMA in parallel. The non-continuous memory allocation is not the bottleneck as long as each segment has 128-byte continuous space for one warp access. That is different from CPU architecture due

to larger cache size and serial memory access. Furthermore, as discussed before, LPMA alleviates memory allocation cost and memory fragmentation issue.

**Localized VS. Global Re-balancing:** Both LPMA and GPMA+ employ the same update strategy on the sorted edge arrays, although LPMA maintains a sorted array using a perfect binary tree while GPMA+ stores a sorted array physically. They have the exactly same condition to trigger expansion. The only difference lies in the re-balancing process. To quantify the cost of data movement during re-balancing, we introduce the following the definition.

**Definition 4 (Re-balancing Size).** *Re-balancing size is the number of segments involved in the re-balancing process after expansion.*

Let us recall the running example in Figs. 2 and 3. Given the same insertions, after expansion, all eight fragments participate in re-balancing in GPMA+. However, in LPMA, segment  $s_2$  or  $s_5$  is not involved in re-balancing. Actually, according to localized and global re-balancing strategies in LPMA and GPMA+, it is easy to conclude the following claim.

*Claim.* Given the same edge insertion batch, the re-balancing size in LPMA is always smaller than that in GPMA+.

To better quantify the benefit of LPMA, we will make the following analysis. Let  $s$  be the number of segments in the original array (before expansion).  $x$  is the number of insert edges in a batch and  $n$  is the maximum number of edges in each segment. According to LPMA insertion strategy, given an insert edge  $\sigma$ , we first locate the associated segment in the original array. For the simplicity of analysis, we assume that edges are randomly inserted. The following lemma states the number of associated segment and non-associated segments.

**Lemma 1.** *The expected numbers of non-associated segments and associated segments are  $s \cdot (1 - \frac{1}{s})^x$  and  $s - s \cdot (1 - \frac{1}{s})^x$ , respectively.*

*Proof.* The edge insertion can be considered as a hashing process, where segments could be considered as buckets of a hash table. In other words, the above process can be considered hashing  $x$  items into a hash table with  $s$  slots. Thus, the number of non-associated segments equals to the number of empty slots in the hash table. According to [4], the expected number of non-associated segments is  $s \cdot (1 - \frac{1}{s})^x$  and the expected number of associated segments is  $s - s \cdot (1 - \frac{1}{s})^x$ .

In LPMA expansion and re-balancing process, only associated segments and their successor empty segments (in the new expanded layer of LPMA) participate in the re-balancing process. However, if an associated segment and its successor empty segment cannot accommodate the corresponding edge insertions, we may need to roll up the insertion and consider the consecutive four segments. Fortunately, we prove the rolling up rarely happens in practice. Considering typical parameters in our experiments, there are  $x = 10^5$  insert edges in a batch and  $10^6$  segments in the original LPMA. Each fragment can accommodate at most  $n = 16$  edges, which corresponds to 128-byte continuous space for one warp access. Consider an associated segment  $a$  and its successor

empty segment  $a'$ . We need to trigger rolling up process only when there are at least  $(n + 1) = 17$  edges to be inserted into  $a$ ; otherwise,  $a$  and  $a'$  can hold all insert edges. The number of edges to be inserted into segment  $a$  (denoted as  $y$ ) can be modeled as a binomial distribution  $y \sim B(x, \frac{1}{s})$ . According to typical parameters in experiments,  $y \sim B(10^5, 10^{-6})$ . Thus, the probability of non-rolling up is calculated as a cumulative distribution  $P(y \leq 17) = \sum_{y=0}^{17} C_{10^5}^y (\frac{1}{10^6})^y (1 - \frac{1}{10^6})^{10^5-y} > 99\%$ <sup>1</sup>. In other words, the rolling up rarely happens and most re-balancing is conducted between two consecutive segments (an associated segment  $a$  and its successor empty segment  $a'$ ).

According to Lemma 1 and typical parameters in experiments ( $s = 10^6$  and  $x = 10^5$ ), the proportion of non-associated segments is  $(1 - \frac{1}{s})^x \approx e^{-\frac{x}{s}} \geq 90\%$ . Since the rolling up rarely happens, the expected re-balancing size in LPMA is  $2s \cdot (1 - \frac{1}{s})^x$ . It means that LPMA saves more than 90% cost, since GPMA+'s re-balancing size is always  $2s$ .

## 4 Experimental Evaluation

### 4.1 Experimental Setup

**Datasets:** In this paper, we use 7 datasets obtained from SNAP [10] in experiments. The datasets statistics are given in Table 1.

**Table 1.** Datasets

Num	Dataset	Nodes	Edges	Average degree
D1	Orkut	3, 072, 441	117, 185, 083	38.14
D2	Graph500	1, 048, 576	125, 829, 120	120
D3	LiveJournal	4, 847, 571	68, 993, 773	14.23
D4	Pokec	1, 632, 803	30, 622, 564	18.75
D5	cit-Patents	3, 774, 768	16, 518, 948	4.38
D6	Road	1, 379, 917	1, 921, 660	1.39
D7	com-DBLP	317, 080	1, 049, 866	3.31

**Setup:** All experiments are implemented with CUDA 7.5 and GCC 4.8.5 and run on Red Hat 4.8.5 server that has Intel(R) Xeon(R) E5-2640 (6-cores, 2.60 GHz) with 128 GB main memory and NVIDIA Tesla P100 GPU with 16 GB main memory.

### 4.2 Methodology

We choose the GPMA+, faimGraph (sort version) and Hornet as the comparable structures. Although [1] is the state-of-art neighbor list based structure, the hash table of [1]

<sup>1</sup> For the ease of calculation, we can use a normal distribution to simulate the binomial distribution calculation when  $x$  is large enough.

requires the preset of neighbor list length. In many highly dynamic graphs of real world situations, the degree of each node may be variable and could not be predicted. If the original graph is null and the whole system is build from scratch, which is our experiment method, [1] is not suitable. Thus we rule out [1] from the comparable structures. We conduct the experiments on expansion, insertion, query primitives and CSR converting to evaluate the performance of LPMA. For the expansion and insertion experiments, we first shuffle the 7 datasets into the random order. We build these dynamic graph data structures from scratch and load these edges incrementally in batches. We tune the batch size from  $10^4$  to  $10^5$  to evaluate the performance vary with batch sizes. For the query primitives experiments, we generate  $10^4$  and  $10^5$  edge batches randomly and run the edge queries on the 7 datasets. We also generate  $10^3$  and  $10^4$  randomly nodes and run the sorted neighbor queries. For the CSR converting experiments, we fully load the 7 datasets into the comparable structures and convert the original structures into the CSR fashion.

### 4.3 Expansion and Insertion Performance

**Expansion Performance:** As mentioned before, LPMA is designed to address the expansion cost in GPMA+. So, we first evaluate the expansion cost in both LPMA and GPMA+. During the insertion, some batches would trigger the expansion and we denote them as *expansion batches*. We set LPMA in the same density thresholds and same segment size with GPMA+, thus the expansion batches would appear at the same time on both structures. We evaluate the average one-batch insertion time of expansion batches to compare the expansion performance between LPMA and GPMA+. Figure 4 shows the average one-batch insertion time of expansion batches on the four datasets with different batch sizes. When the edges arrive in  $10^4$  batches, LPMA has more than  $60\times$  speed up than GPMA+ on the 5 larger datasets and more than  $20\times$  speed up on the 2 smaller datasets. As we analysed in Sect. 3.3, GPMA+ always needs to re-balance the whole array during the expansion. The expansion cost depends on the size of the array. The array size grows quite large in the last few rounds on the 5 larger datasets, thus the expansion costs increase rapidly. For LPMA, the expansion cost depends on the distribution of the insertions that would not grow with the array size. If the batch size

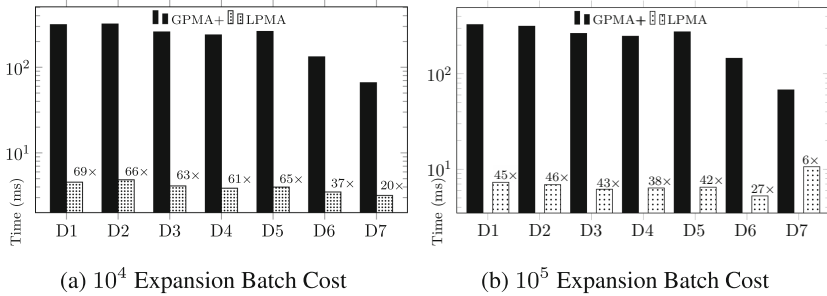


Fig. 4. Expansion performance

is small, the expansion cost remains insignificant in each expansion round. When the edges arrive in  $10^5$  batches, the LPMA has around  $40\times$  to speed up than GPMA+ on the 5 larger datasets,  $27\times$  speed up on Road and  $6\times$  speed up on com-DBLP. The advantage of LPMA weakens with the increment of batch size, since the expected numbers of non-associated segments (Lemma 1) is reduced.

**Overall Insertion Performance:** Figure 5 shows the average time of all the insertion batches (including expansion batches). When the edges arrive in  $10^4$  batches, LPMA has  $4\times$  to  $8\times$  speed up than GPMA+. In  $10^5$  batches, LPMA has  $5\times$  to  $10\times$  speed up than GPMA+. In the  $10^4$  batch sizes, LPMA has  $1.5\times$  to  $2.5\times$  speed up than faimGraph on the 5 larger datasets. On the 2 smaller datasets Road and com-DBLP, faimGraph has  $1.5\times$  to  $2.5\times$  speed up than LPMA. In the  $10^5$  batch sizes, LPMA has  $1.2\times$  speed up on the largest dataset Orkut and faimGraph has  $1.02\times$  to  $2\times$  speed up on the other datasets. The performance diversities on different datasets are caused by the different insertion methods. In faimGraph, the insertion batches are partitioned by the source nodes and merged into the associated neighbor lists. Each neighbor list is assigned a thread to process the insertion. On the larger datasets with higher average degrees, the lengths of the neighbor lists are more uneven than the smaller datasets. The uneven neighbor lists bring the unbalanced workloads to faimGraph. LPMA inherits the segment oriented parallel strategy from GPMA+ and the workloads among the threads are always balanced. On the other hand, when the batch size grows large in LPMA, the re-balance size grows large and drop the performance significantly. Thus LPMA has better insertion performance on the larger graph in smaller batches and faimGraph has better insertion performance on the smaller graph in larger batches. In the  $10^4$  batch sizes, Hornet has  $1.5\times$  to  $3\times$  speed up than LPMA. In the  $10^5$  batch sizes, LPMA

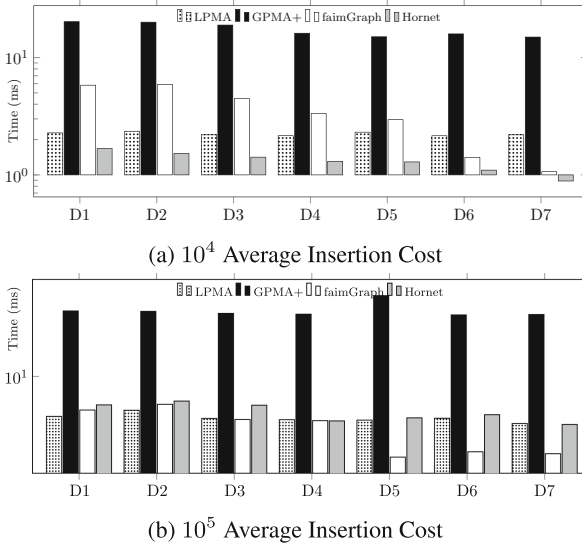


Fig. 5. Overall insertion performance

has  $1.3\times$  to  $0.9\times$  speed up than Hornet. Although Hornet is faster than LPMA in  $10^4$  batch, Hornet dose not maintain the sorted graph and has worse performance on the graph queries.

### 4.4 Query Performance

**Query Primitives:** The query primitives performance is the fundamental of the most graph algorithms like BFS,  $n$ -hop neighbors and SSSP, which are executed by performing query primitives iteratively. To evaluate query primitives performance of LPMA, we compare edge query and sorted 1-hop neighbor query with GPMA+, faimGraph and Hornet. We randomly choose query sets with  $10^4$  and  $10^5$ -size query batches and run on the 7 datasets. Then we compare the average processing time for the query batches. For the neighbor query, we randomly choose  $10^3$  and  $10^4$  nodes sets and run the sorted neighbor lists query on 7 datasets and compare the average time cost.

Figure 6 shows the results of the query primitives performance. Since the GPMA+ and LPMA has the same logical structure and same procedure for the query primitives, they have similar query performance. For the edge query, LPMA has  $12\times$  and  $6\times$  speed up than faimGraph in  $10^4$  and  $10^5$  batches. To run the edge query parallel, faimGraph assigns one thread for each query edge and runs binary search on the associated neighbor list. LPMA assigns one thread for each query edge and run the binary search on the whole edge array. Although LPMA has larger searching space for each query edge, the shared memory prefetch could reduce the main memory access and speed up the parallel query evidently. For the sorted neighbor list query, LPMA has  $1.5\times$  and  $1.2\times$  speed up than faimGraph in  $10^3$  and  $10^4$  batches. faimGraph assigns one thread for each query node to read the associated neighbor list and the uneven lengths of neighbor lists could cause unbalanced workload. LPMA first uses node oriented strategy to read the offset and locate the associated segments. Then LPMA uses segment oriented strategy to read the neighbor lists parallel that avoid the unbalanced workload. Since Hornet does not maintain the sorted edges, thus LPMA has  $5\times$  speed up on the edge query and  $3\times$  speed up on the sorted neighbor query.

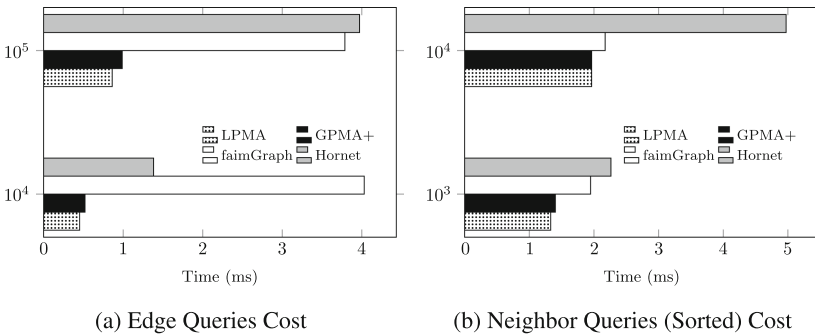


Fig. 6. Primitive Query Performance



**CSR Converting Performance:** As we discussed in Sect. 2.2, CSR is a de facto structure for many existing graph algorithms. To enable use existing graph analysis libraries, we need to convert a dynamic graph structure to CSR format. Thus, we compare the CSR converting time cost for these 4 structures. In experiments, we load the 7 datasets and convert the dynamic graph structure into a CSR. GPMA+ and LPMA maintain a sorted edge array. By compacting the empty cells in the array, both GPMA+ and LPMA could be converted into CSR in a small cost. Similar as sorted neighbor list query, faimGraph uses the node oriented strategy to extract the CSR and the performance is limited by the unbalanced workload. Hornet does not maintain the ordered graph and needs to be sorted first before the converting. Figure 7 shows that GPMA+ and LPMA has the same performance,  $30\times$  to  $7\times$  speed up than faimGraph and  $0.8\times$  to  $1.8\times$  speed up than Hornet. To be noticed, faimGraph has better CSR converting performance than LPMA on the Road dataset since the road graph has evenner distribution of node degrees than other graphs.

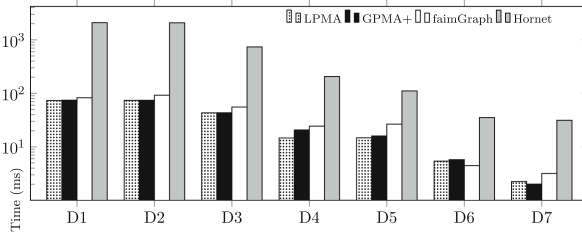


Fig. 7. CSR converting performance

## 5 Related Work

**CSR-Based Structure:** As an early effort, the dynamic compressed sparse row (DCSR) [9] is devised to handle the dynamic changes. By reserving empty cells in the column array, the insertions could be added into the data structure in a small cost. However if the insertions for row  $i$  over-range the reserve space, the offset of row  $i$  would be divided into discontinuous space in the column array. The defragmentation operation has to be conducted after the insertion to maintain the good locality of the data structure, thus, DCSR is not efficient for dynamic graphs due to high throughput insertions/deletions.

**Neighbor List Based Structure:** The neighbor list based structures maintain a nodes array and a neighbor list for each node. The main contribution of the neighbor list based structures is to design a variable array or list to store the neighbor lists. The STINGER [6] data structure was first introduced as a dynamic graph structure for both temporal and spatial graphs with meta-data for multi-core architectures. cuSTINGER [7] extends the STINGER structure to the GPUs. STINGER relies on blocked linked lists to store the neighbor lists, whereas cuSTINGER uses arrays for the neighbor lists. Hornet [5] designs a dynamic array management system to store the neighbor list of each node.

First, Hornet uses Block-arrays as store unit for the edges. The system assigns each node a default number of the cells for the neighbor list. Once the cells are full, the system reallocates the double size of the previous space. Hornet has a Vectorized Bit Tree for each Block-array to locate the empty cells and  $B^+$  Trees of block-arrays to manage the Block-arrays. Hornet places the neighbor lists in block sizes that are powers of two which sets the upper bound for the memory allocated for the entire graph evolution:  $2|E|$ . faimGraph [14] stores the adjacency lists in configurable size memory pages. The pages of one node's neighbor list are connected by the pointers. faimGraph uses a single memory pool on GPU to manage the memory pages. A hash table approach is made in Dynamic Graphs on the GPU [1]. Instead of array, this work uses hash table for the adjacency lists storage. The GPU memory is divided into fixed size chunks and each chunk presents a bucket of the hash table. This work deals the hash collisions with linked list achieves  $O(1)$  time complexity for single edge update. However this work requires the preset neighbor lists length for the hash table size that means it does not support the dynamic graph build from scratch.

## References

1. Awad, M.A., Ashkiani, S., Porumbescu, S.D., Owens, J.D.: Dynamic graphs on the GPU. In: 2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS), pp. 739–748. IEEE (2020)
2. Bender, M.A., Demaine, E.D., Farach-Colton, M.: Cache-oblivious b-trees. In: Proceedings 41st Annual Symposium on Foundations of Computer Science, pp. 399–409. IEEE (2000)
3. Bender, M.A., Hu, H.: An adaptive packed-memory array. *ACM Trans. Database Syst. (TODS)* **32**(4), 26-es (2007)
4. Bogart, K., Stein, C.: Discrete math in computer science. Department of Computer Mathematics and Department of Computer Science. Dartmouth College, Hanover, NH (2002)
5. Busato, F., Green, O., Bombieri, N., Bader, D.A.: Hornet: an efficient data structure for dynamic sparse graphs and matrices on GPUs. In: 2018 IEEE High Performance extreme Computing Conference (HPEC), pp. 1–7. IEEE (2018)
6. Ediger, D., McColl, R., Riedy, J., Bader, D.A.: Stinger: high performance data structure for streaming graphs. In 2012 IEEE Conference on High Performance Extreme Computing, pp. 1–5. IEEE (2012)
7. Green, O., Bader, D.A.: cuSTINGER: supporting dynamic graph algorithms for GPUs. In: 2016 IEEE High Performance Extreme Computing Conference (HPEC), pp. 1–6. IEEE (2016)
8. Hu, Y., Liu, H., Huang, H.H.: Tricore: parallel triangle counting on GPUs. In: SC 2018: International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 171–182. IEEE (2018)
9. King, J., Gilray, T., Kirby, R.M., Might, M.: Dynamic sparse-matrix allocation on GPUs. In: Kunkel, J.M., Balaji, P., Dongarra, J. (eds.) ISC High Performance 2016. LNCS, vol. 9697, pp. 61–80. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-41321-1\\_4](https://doi.org/10.1007/978-3-319-41321-1_4)
10. Leskovec, J., Krevl, A.: SNAP Datasets: Stanford large network dataset collection, June 2014. <http://snap.stanford.edu/data>
11. McGinnis, C.: PCI-SIG® fast tracks evolution to 32gt/s with PCI express 5.0 architecture. News Release, 7 June 2017
12. Sha, M., Li, Y., He, B., Tan, K.-L.: Accelerating dynamic graph analytics on GPUs. *Proc. VLDB Endow.* **11**(1), 107–120 (2017)

13. Wang, Y., Davidson, A., Pan, Y., Wu, Y., Riffel, A., Owens, J.D.: Gunrock: a high-performance graph processing library on the GPU. In: Proceedings of the 21st ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, pp. 1–12 (2016)
14. Winter, M., Mlakar, D., Zayer, R., Seidel, H.-P., Steinberger, M.: faimGraph: high performance management of fully-dynamic graphs under tight memory constraints on the GPU. In: SC 2018: International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 754–766. IEEE (2018)
15. Zeng, L., Zou, L., Tamer Özsü, M., Hu, L., Zhang, F.: GSI: GPU-friendly subgraph isomorphism. In: 2020 IEEE 36th International Conference on Data Engineering (ICDE), pp. 1249–1260. IEEE (2020)
16. Zhong, J., He, B.: Medusa: simplified graph processing on GPUs. *IEEE Trans. Parallel Distrib. Syst.* **25**(6), 1543–1552 (2013)



# Updating Maximal $(\Delta, \gamma)$ -Cliques of a Temporal Network Efficiently

Suman Banerjee<sup>1</sup> and Bithika Pal<sup>2</sup>

<sup>1</sup> Indian Institute of Technology, Jammu, India  
suman.banerjee@iitjammu.ac.in

<sup>2</sup> Indian Institute of Technology, Kharagpur, India  
bithikapal@iitkgp.ac.in

**Abstract.** Given a temporal network  $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{T})$ ,  $(\mathcal{X}, [t_a, t_b])$  (where  $\mathcal{X} \subseteq \mathcal{V}(\mathcal{G})$  and  $[t_a, t_b] \subseteq \mathcal{T}$ ) is said to be a  $(\Delta, \gamma)$ -clique of  $\mathcal{G}$ , if for every pair of vertices in  $\mathcal{X}$ , there must exist at least  $\gamma$  links in each  $\Delta$  duration in  $[t_a, t_b]$ . In this paper, we study the maximal  $(\Delta, \gamma)$ -clique enumeration problem in online setting; i.e.; the entire link set of the network is not known in advance. Suppose, the link set till time stamp  $T_1$  (i.e.,  $\mathcal{E}^{T_1}$ ), and its corresponding  $(\Delta, \gamma)$ -clique set are known. In the next batch (till time  $T_2$ ), a new set of links (denoted as  $\mathcal{E}^{(T_1, T_2]}$ ) is arrived. Now, the goal is to update the existing  $(\Delta, \gamma)$ -cliques to obtain the maximal  $(\Delta, \gamma)$ -cliques till time stamp  $T_2$ . We formally call this problem as the MAXIMAL  $(\Delta, \gamma)$ -CLIQUE UPDATION PROBLEM. We propose an efficient updation approach and show that the proposed methodology is correct. An extensive set of experiments have been conducted with four datasets. The obtained results show that the proposed methodology is efficient both in terms of time and space to enumerate maximal  $(\Delta, \gamma)$ -cliques in online setting. Compared to its off-line counterpart, the improvement caused by the proposed approach is in the order of hours and GB for computational time and space, respectively.

**Keywords:** Temporal network ·  $(\Delta, \gamma)$ -clique · Enumeration algorithm

## 1 Introduction

A pairwise relation among a group of agents is represented by a ‘network’ (also known as ‘graph’), where the set of agents forms the *vertex set*, and the links among them form the *edge set* of the network [3]. Examples include ‘social network’, ‘information network’ and many more. Analysis of such networks for different topological structures brings out many important characteristics. For example, a cohesive group of users in a social network can be interpreted as close friends. Most of the real-world networks are time varying in nature, which means the structure of the network are changing over time [9]. This kind of networks

---

Both the authors have contributed equally in this work and they are joint first authors.

are effectively modeled as *temporal network* (also known as the *time varying graph* or *link stream*) [9].

To analyze a static network, among several topological structures in the literature, widely studied structural pattern is *clique* [6]. Plenty of solution methodologies have been proposed to enumerate such structures present in a network [15]. Initially, E. A. Akkoyunlu [1] proposed an enumeration technique for maximal cliques. After that, a recursive technique has been proposed by Bron and Kerbosch [4]. Since then, a significant effort has been put to develop practical algorithms for enumerating maximal cliques in different scenarios such as for sparse graph [6], in large networks [5], in uncertain graphs [11], using map reduce framework [14] and so on. As the real world networks are time varying, none of the mentioned techniques can be applied for analyzing such networks.

To analyze a temporal network for cohesive structures, Viard et al. [13] introduced the notion of  $\Delta$ -clique. For a given temporal network, a  $\Delta$ -clique is a vertex subset and time interval pair, where for every  $\Delta$  duration of the interval, there exist at least one link between every pair of vertices in the subset. However, for practical applications such as community structure in temporal networks etc.  $\Delta$ -clique appears to be a too sparse structure. Hence, recently Banerjee and Pal [2] extended the notion of  $\Delta$ -clique to  $(\Delta, \gamma)$ -clique by incorporating frequency along with the duration, and proposed an enumeration algorithm for maximal  $(\Delta, \gamma)$ -cliques present in a temporal network. In all these studies, it is implicitly assumed that the whole temporal links are available before the enumeration algorithm starts execution. However in reality, the scenario may be little different which has been explained with a real-world example.

Consider a call graph which is a temporal network with a set of persons as vertex set and there is a link between the persons  $u$  and  $v$  at time  $t$  if there is phone call between them at that time. Now, to analyze their contact patterns (assume for hundred days), we need to wait till the entire data is collected. In many data collection scenarios, the required time is much more. As an example, for the ‘college message’ dataset [12], the duration for collecting the data was 193 days. However, without waiting till the end, we can start analyzing when substantial number of links are available and enumerate the cohesive groups. Later, we can update them as and when the new set of links arrive. So, the question here is that given a temporal network, its link set till time  $T_1$  and its corresponding maximal  $(\Delta, \gamma)$ -cliques, how to update these cliques (which we study in this work) when the next batch of links (till timestamp  $T_2$ ) are available such that the  $(\Delta, \gamma)$ -Cliques till timestamp  $T_2$  are obtained.

The contributions of the paper are (i) introduction of MAXIMAL  $(\Delta, \gamma)$ -CLIQUE UPDATION PROBLEM, where the links are coming as a batch in an iterative manner, (ii) efficient methodology to update the existing  $(\Delta, \gamma)$ -clique by considering new links in an online fashion, (iii) correctness of the proposition, (iv) extensive set of experiments with four real-world datasets with different data partitioning schemes, which show the efficiency of the updation approach to enumerate all the maximal  $(\Delta, \gamma)$ -cliques. Now, Sect. 2 contains basic concepts and problem definition. Section 3 discusses the proposed methodology. Section 4 contains the experimental results.

## 2 Preliminaries and Problem Definitions

**Definition 1 (Temporal Network) [7].** A temporal network is defined as a triplet  $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{T})$ , where  $V(\mathcal{G})$  and  $\mathcal{E}(\mathcal{G})$  ( $\mathcal{E}(\mathcal{G}) \subseteq \binom{\mathcal{V}(\mathcal{G})}{2} \times \mathcal{T}$ ) are the vertex and edge set of the network.  $\mathcal{T} (= [T_1, T_2])$  is the time interval during which the network is observed. Throughout the paper, we use  $|\mathcal{V}(\mathcal{G})| = n$  and  $|\mathcal{E}(\mathcal{G})| = m$ .

**Definition 2 ( $(\Delta, \gamma)$ -Clique) [2].** For a given time period  $\Delta$  and  $\gamma \in \mathbb{Z}^+$ , a  $(\Delta, \gamma)$ -clique of the temporal network  $\mathcal{G}$  is a vertex set, time interval pair, i.e.,  $(\mathcal{X}, [t_a, t_b])$  where  $\mathcal{X} \subseteq \mathcal{V}(\mathcal{G})$ ,  $|\mathcal{X}| \geq 2$ , and  $[t_a, t_b] \subseteq \mathcal{T}$ . Here  $\forall v_i, v_j \in \mathcal{X}$  and  $t \in [t_a, \max(t_b - \Delta, t_a)]$ , there must exist  $\gamma$  or more number of edges, i.e.,  $(v_i, v_j, t_{ij}) \in \mathcal{E}(\mathcal{G})$  and  $f_{(v_i, v_j)} \geq \gamma$  with  $t_{ij} \in [t, \min(t + \Delta, t_b)]$ . It is easy to observe, that a  $(\Delta, \gamma)$ -clique will be a  $\Delta$ -clique when  $\gamma = 1$ .

**Definition 3 (Maximal  $(\Delta, \gamma)$ -Clique).** A  $(\Delta, \gamma)$ -clique  $(\mathcal{X}, [t_a, t_b])$  of the temporal network  $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{T})$  will be maximal if neither of the following three is true; (i)  $\exists v \in \mathcal{V}(\mathcal{G}) \setminus \mathcal{X}$  such that  $(\mathcal{X} \cup \{v\}, [t_a, t_b])$  is a  $(\Delta, \gamma)$ -clique, (ii)  $(\mathcal{X}, [t_a - dt, t_b])$  is a  $(\Delta, \gamma)$ -clique. This applies only if  $t_a - 1 \geq T_1$ , (iii)  $(\mathcal{X}, [t_a, t_b + dt])$  is a  $(\Delta, \gamma)$ -clique. This applies only if  $t_b + 1 \leq T_2$ .

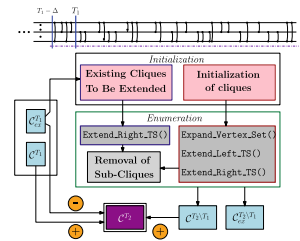
As mentioned, all the links of the temporal network may not be available at the beginning of the execution of any  $(\Delta, \gamma)$ -clique enumeration algorithm. Hence, it is required to study the Maximal  $(\Delta, \gamma)$ -clique Updation Problem, which is stated in Definition 4.

**Definition 4 (Maximal  $(\Delta, \gamma)$ -Clique Updation Problem).** Given a temporal network  $\mathcal{G}(V, E, \mathcal{T})$  with its link set and  $(\Delta, \gamma)$ -clique till time stamp  $T_1$ , and the links from time stamp  $T_1$  to  $T_2$ , i.e.  $\mathcal{E}^{(T_1, T_2]}$ , the problem of Maximal  $(\Delta, \gamma)$ -clique Updation is to update the existing maximal  $(\Delta, \gamma)$ -cliques till  $T_1$  to obtain the maximal  $(\Delta, \gamma)$ -cliques till  $T_2$ .

## 3 Proposed Solution Approach

Let,  $\mathcal{C}^{T_1}$ ,  $\mathcal{C}^{T_2}$ , and  $\mathcal{C}^{T_2 \setminus T_1}$  denote the set of maximal cliques till time  $T_1$ ,  $T_2$ , and from  $T_1$  to  $T_2$ , respectively. Also assume that for a clique  $(\mathcal{X}, [t_a, t_b])$ , the first and last  $\gamma$ -th occurrence timestamps of a static edge  $(u, v)$  within  $[t_a, t_b]$  are denoted by  $f_{uv}^\gamma$  and  $l_{uv}^\gamma$ , respectively, where  $u, v \in \mathcal{X}$ .

The proposed approach is described in Fig. 1. As the method updates the existing  $(\Delta, \gamma)$ -cliques with the new links, we call it as ‘Edge on Clique’ approach. It takes the maximal  $(\Delta, \gamma)$ -clique set till time stamp  $T_1$  (i.e.,  $\mathcal{C}^{T_1}$ ), the possible clique set to be extended ( $\mathcal{C}_{ex}^{T_1}$ ), the links arrived during  $T_1 - \Delta$  to  $T_2$  ( $\mathcal{E}^{[T_1 - \Delta, T_2]}$ ),  $T_1$ ,  $T_2$ ,  $\Delta$ , and  $\gamma$  as inputs. It produces the maximal  $(\Delta, \gamma)$ -cliques till  $T_2$  ( $\mathcal{C}^{T_2}$ ), and the cliques that will be extended in the next update (i.e., with the links  $\mathcal{E}^{(T_2, T_3]}$  for  $T_3 > T_2$ ).



**Fig. 1.** Diagram of the proposed methodology

*Description of the Proposed Approach:* The proposed approach is shown in Algorithm 1. It works in three parts; (i) extending the right timestamp of the cliques in  $\mathcal{C}_{ex}^{T_1}$  (Line 3–9); (ii) processing the links  $\mathcal{E}^{[T_1-\Delta, T_2]}$  through initialization, expanding vertex set, right and left timestamp (Line 10–23); (iii) removal of sub-cliques formed due to existence of the links before time  $T_1 - \Delta$  (Line 24). We first process the cliques in  $\mathcal{C}_{ex}^{T_1}$  to extend the right timestamp with the links coming till timestamp  $T_2$ . For the enumeration process, four clique sets:  $\mathcal{C}^I$  (for holding the cliques yet to be processed),  $\mathcal{C}_{im}$  (for keeping the cliques already or yet to be processed),  $\mathcal{C}^{T_2 \setminus T_1}$  (for storing the maximal cliques, whose entire or partial links are present in  $\mathcal{E}^{[T_1-\Delta, T_2]}$ ), and  $\mathcal{C}_{ex}^{T_2 \setminus T_1}$  (for storing the cliques to be extended in next update) are maintained. At first, for each clique  $(\mathcal{Z}, [t_x, t_y])$  in  $\mathcal{C}^I$ , if the right time stamp extension is possible using `Extend_Right_TS()`, the new clique is added in  $\mathcal{C}_{im}$  and  $\mathcal{C}^I$ , and set the `r_flag` as `FALSE` to indicate  $(\mathcal{Z}, [t_x, t_y])$  is non-maximal. Otherwise,  $(\mathcal{Z}, [t_x, t_y])$  is maximal and added to  $\mathcal{C}^{T_2 \setminus T_1}$  in Line 7. Now, if  $t_y \geq T_2$  for the currently popped clique, it is added into  $\mathcal{C}_{ex}^{T_2 \setminus T_1}$  as a possible candidate for the extension. In Line 10, it computes all the cliques having exact  $\gamma$  links of duration  $\Delta$ , for every pair of vertices in  $\mathcal{E}^{[T_1-\Delta, T_2]}$ .

---

**Algorithm 1:** Maximal  $(\Delta, \gamma)$ -clique enumeration using ‘Edge on Clique’ (EOC) Approach

---

**Data:** The Clique Set till Time  $T_1$ ; i.e.:  $\mathcal{C}^{T_1}, \mathcal{C}_{ex}^{T_1}$ , The link set  $\mathcal{E}^{[T_1-\Delta, T_2]}$  of  $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{T}, \Delta, \gamma, T_1, \text{ and } T_2$ .

**Result:** Initialized Clique Set for the links  $\mathcal{E}^{T_2 \setminus T_1}$ .

```

1  $\mathcal{C}^I \leftarrow \mathcal{C}_{ex}^{T_1}$  ;
2  $\mathcal{C}^{T_2 \setminus T_1} \leftarrow \emptyset, \mathcal{C}_{ex}^{T_2 \setminus T_1} \leftarrow \emptyset, \mathcal{C}_{im} \leftarrow \mathcal{C}^I$ ;
3 while  $\mathcal{C}^I \neq \emptyset$  do
4   take and remove  $(\mathcal{Z}, [t_x, t_y])$  from  $\mathcal{C}^I$ ;
5   r_flag = Extend_Right_TS $(\mathcal{Z}, [t_x, t_y], \mathcal{E}^{[T_1-\Delta, T_2]})$ ;
6   if r_flag == TRUE then
7     add  $(\mathcal{Z}, [t_x, t_y])$  to  $\mathcal{C}^{T_2 \setminus T_1}$ ;
8   if  $t_y \geq T_2$  then
9     add  $(\mathcal{Z}, [t_x, t_y])$  to  $\mathcal{C}_{ex}^{T_2 \setminus T_1}$ ;
10  $\mathcal{C}^I \leftarrow$  Execute Algorithm 1 of [2] on the links  $\mathcal{E}^{[T_1-\Delta, T_2]}$ ;
11  $\mathcal{C}_{im} \leftarrow \mathcal{C}_{im} \cup \mathcal{C}^I$ ;
12 while  $\mathcal{C}^I \neq \emptyset$  do
13   take and remove  $(\mathcal{Z}, [t_x, t_y])$  from  $\mathcal{C}^I$ ;
14   if  $t_y - t_x == \Delta$  then
15     Prepare the static graph  $G$  for the duration  $[t_x, t_y]$ ;
16     Associate  $N_G(\mathcal{Z})$  to  $(\mathcal{Z}, [t_x, t_y])$ ;
17     v_flag = Expand_Vertex_Set $(\mathcal{Z}, [t_x, t_y], N_G(\mathcal{Z}))$ ;
18     l_flag = Extend_Left_TS $(\mathcal{Z}, [t_x, t_y], \mathcal{E}^{[T_1-\Delta, T_2]})$ ;
19     r_flag = Extend_Right_TS $(\mathcal{Z}, [t_x, t_y], \mathcal{E}^{[T_1-\Delta, T_2]})$ ;
20     if v_flag  $\wedge$  l_flag  $\wedge$  r_flag == TRUE then
21       add  $(\mathcal{Z}, [t_x, t_y])$  to  $\mathcal{C}^{T_2 \setminus T_1}$ ;
22     if  $t_y \geq T_2$  then
23       add  $(\mathcal{Z}, [t_x, t_y])$  to  $\mathcal{C}_{ex}^{T_2 \setminus T_1}$ ;
24 EOC_Remove_Sub_Cliques $(T_1)$ ;
25  $\mathcal{C}^{T_2} \leftarrow \mathcal{C}^{T_2 \setminus T_1} \cup (\mathcal{C}^{T_1} \setminus \mathcal{C}_{ex}^{T_1})$  ;
26 return  $\mathcal{C}^{T_2}, \mathcal{C}_{ex}^{T_2 \setminus T_1}$ ;

```

---

In the next step, an arbitrary clique  $(\mathcal{Z}, [t_x, t_y])$  is taken from  $\mathcal{C}^I$  in Line 13. If  $t_y - t_x = \Delta$ , the static graph  $G$  is built with the links of  $[t_x, t_y]$  in Line 15. The neighboring vertices in  $G$ , which have communicated at least  $\gamma$  times with any vertex in  $\mathcal{Z}$ , forms the candidate vertex set  $N_G(\mathcal{Z})$  for the clique  $(\mathcal{Z}, [t_x, t_y])$  in Line 16. Next, the algorithm tries to expand  $(\mathcal{Z}, [t_x, t_y])$  in the following three ways: (i) by adding vertices (function `Expand_Vertex_Set()`), (ii) by stretching  $t_x$  towards its left (function `Extend_Left_TS()`), and (iii) by stretching  $t_y$  towards its right (function `Extend_Right_TS()`).

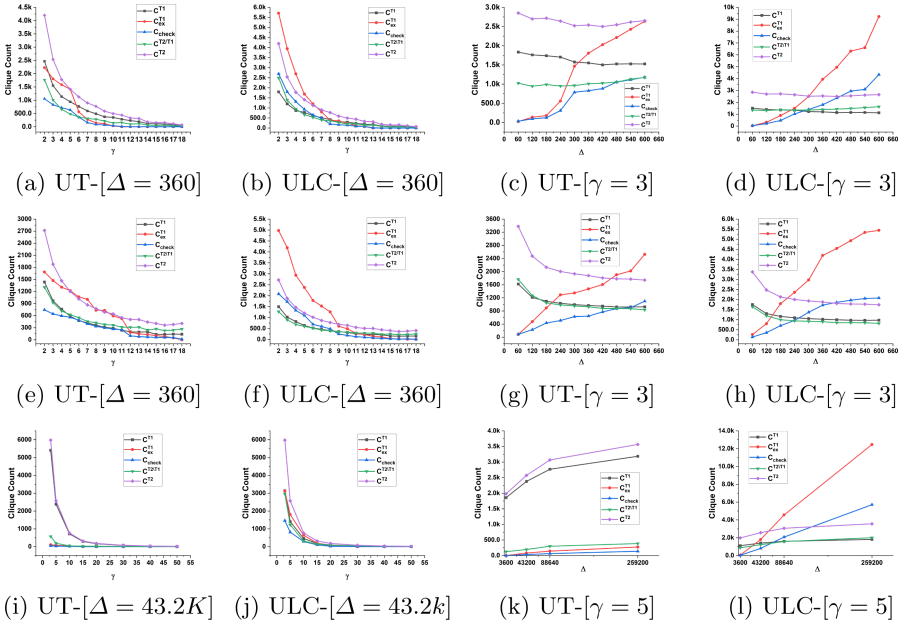
`Expand_Vertex_Set()` selects the neighbors of  $\mathcal{Z}$  in  $G$  within the interval  $[t_x, t_y]$  and checks whether the new tuple(vertex pair, time interval) holds the  $(\Delta, \gamma)$ -clique property. If the vertex addition is possible, the new clique is added in  $\mathcal{C}_{im}$  and  $\mathcal{C}^I$ , and it declares  $(\mathcal{Z}, [t_x, t_y])$  as non-maximal by setting  $v\_flag$  to *FALSE*. `Extend_Left_TS()` tries to extend  $t_x$  by  $t_{xl} - \Delta$  ( $t_{xl}$  is the latest time stamp from all the  $\gamma$ -th occurrence time stamps within  $[t_x - 1, t_y]$  of each possible vertex pair in  $\mathcal{Z}$ ) with the links  $\mathcal{E}^{[T_1 - \Delta, T_2]}$ . If this holds the  $(\Delta, \gamma)$ -clique property, the new clique is added in  $\mathcal{C}_{im}$  and  $\mathcal{C}^I$ , and it declares  $(\mathcal{Z}, [t_x, t_y])$  as non-maximal by setting  $l\_flag$  to *FALSE*. Similarly, in `Extend_Right_TS()`,  $t_y$  is extended as  $t_{yr} + \Delta$  ( $t_{yr}$  is the earliest time stamp from all the last  $\gamma$ -th occurrence time stamps within  $[t_x, t_y + 1]$  of each possible vertex pair in  $\mathcal{Z}$ ) and sets  $r\_flag$  to *FALSE* if the extension is possible. If any of the functions returns *FALSE* that means the clique  $(\mathcal{Z}, [t_x, t_y])$  is not maximal. Hence, we perform logical ‘AND’ operation among the flags in Line 20, and if the outcome is *TRUE*, that means the clique  $(\mathcal{Z}, [t_x, t_y])$  is a maximal clique, and it is added to  $\mathcal{C}^{T_2 \setminus T_1}$ . Next, if  $t_y$  is greater than  $T_2$ , the clique is added in  $\mathcal{C}_{ex}^{T_2 \setminus T_1}$ .

Now, it is important to observe that the maximal clique set  $\mathcal{C}^{T_2}$  can be obtained by the union of  $(\mathcal{C}^{T_1} \setminus \mathcal{C}_{ex}^{T_1})$ , and  $\mathcal{C}^{T_2 \setminus T_1}$ . However, the cliques which are formed with the links  $\mathcal{E}^{[T_1 - \Delta, T_2]}$ , unable to verify it’s extendibility towards left. This results in having  $t_x$  greater than its actual value from it’s maximal counterpart. Hence, the `EOC_Remove_Sub_Cliques()` function is invoked to remove those cliques which are identified falsely as maximal. For a fixed  $\mathcal{Z}$ , it tries to get the maximum duration, and for a fixed  $[t_x, t_y]$ , it tries to keep the maximum number of vertices in  $\mathcal{Z}$ . The procedure first finds out the cliques having  $t_x \leq T_1$  as possible candidates for removal check ( $\mathcal{C}_{check}$ ). For an identified clique  $(\mathcal{Z}, [t_x, t_y])$ , there are two possibilities; (i) with the same  $\mathcal{Z}$  their exist a  $[t_{x'}, t_{y'}]$  which contains  $[t_x, t_y]$ , or (ii) there exist a clique  $(\mathcal{Z}', [t_{x'}, t_{y'}])$  such that  $\mathcal{Z} \subset \mathcal{Z}'$  and  $[t_x, t_y] \subseteq [t_{x'}, t_{y'}]$ , resulting  $(\mathcal{Z}, [t_x, t_y])$  as non-maximal. Hence,  $(\mathcal{Z}, [t_x, t_y])$  is removed from  $\mathcal{C}^{T_2 \setminus T_1}$ . In this way, it ends up with getting the maximal cliques only. Finally, the algorithm returns the set of maximal cliques till  $T_2$  ( $\mathcal{C}^{T_2}$ ), and the possible cliques to be extended in next update cycle ( $\mathcal{C}_{ex}^{T_2 \setminus T_1}$ ), as output.

*Correctness Brief:* Assume that, the entire set of links till  $T_2$  ( $\mathcal{E}^{T_2}$ ) is being processed to get  $\mathcal{C}^{T_2}$ . Now, it is easy to observe that for a maximal clique  $(\mathcal{Z}, [t_x, t_y])$  in  $\mathcal{C}^{T_2}$ , if  $[t_x, t_y]$  lies entirely within  $[T_1 - \Delta, T_2]$ , the procedure `Expand_Left_TS()` can reach to the maximal clique by extending the start time stamp towards left. However, if  $T_1 - \Delta$  lies within the  $[t_x, t_y]$  and the first gamma edges of every pair of vertices in  $\mathcal{Z}$  does not belong to  $\mathcal{E}^{[T_1 - \Delta, T_2]}$ , the procedure `Expand_Left_TS()` will fail to get the maximal cliques from  $\mathcal{C}^I$ , initialized using  $\mathcal{E}^{[T_1 - \Delta, T_2]}$ . This scenario returns a non-maximal cliques by identifying them as falsely maximal. However, we do not miss the maximal ones as the algorithm uses the cliques from  $\mathcal{C}_{ex}^{T_1}$ , which has it’s  $t_x$  fixed and the  $t_y$  is extended correctly by Procedure `Expand_Right_TS()`.

The detailed algorithmic description of each procedure with its correctness proofs, analysis of time and space requirement will be available in its full version.





**Fig. 2.** Result for the change of clique count w.r.t  $\Delta$  and  $\gamma$  with different partition schemes; (a-d) Infectious, (e-h) Hypertext, (i-l) College Message dataset

### 4 Experimental Evaluation

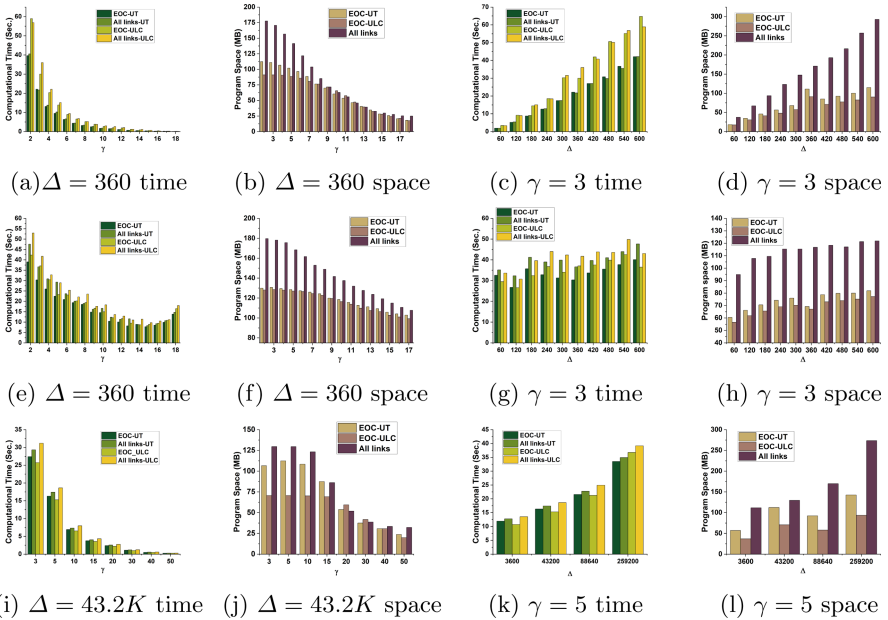
**Datasets and Experimental Setup.** In this study, we use the following four datasets, namely Infectious [8], Hypertext [8], College Message [12], and Autonomous Systems (AS180) [10].

Table 1 above gives the basic statistics of the datasets. The only setup required in our experiments is to partition the dataset. We use the following two techniques: (i) **Uniform Time Interval-Based Partitioning (EOC-UT)**: In this technique, the dataset is splitted into parts, where time duration in each part is equal. (ii) **Uniform Link Count-Based Partitioning (EOC-ULC)**: In this technique, the dataset is splitted into parts, where the number links in each part are equal. In our experiments, the number of partitions is two. We implement the proposed methodology in Python 3.6 along with NetworkX 2.2 environment. All the experiments have been carried out in a 32-core server with 256 GB RAM and 2.2 GHz processing speed. The implementation is available at <https://github.com/BITHIKA1992/Delta-gamma-Clique-Update>.

**Table 1.** Basic statistics of datasets

Datasets	#Nodes	#Links	#Static edges	Lifetime
Infectious	410	17298	2765	8 h
Hypertext	113	20818	2196	2.5 Days
College message	1899	59835	20296	193 Days
AS180	4002	2127983	8957	180 Days

**Experimental Results with Discussion.** To understand the improvement of computational time and space in case of partition-based scheme, it is necessary to analyze the size of different clique sets  $\mathcal{C}^{T_1}$ ,  $\mathcal{C}_{ex}^{T_1}$ ,  $\mathcal{C}^{T_2 \setminus T_1}$ ,  $\mathcal{C}^{T_2}$ ,  $\mathcal{C}_{check}$  with the change in  $\gamma$  and  $\Delta$  (Fig. 2). Based on the lifetime and link count at each timestamp, one suitable value of  $\Delta$  and  $\gamma$  is chosen for each of the datasets. For both the partition schemes, for a fixed value of  $\Delta$ , the size of the clique sets decreases exponentially with the increasing value of  $\gamma$  (Fig. 2 a, b, e, f, i, and j). However, for fixed  $\gamma$ , other than  $\mathcal{C}_{ex}^{T_1}$  and  $\mathcal{C}_{check}$ , all other clique set sizes decrease with the increasing value of  $\Delta$  (Fig. 2 c, d, g, and h), due to dense contact trace at each time stamp. An exception is observed in case of the college message dataset, due to the large change in delta, and their sparse contact patterns (Fig. 2 k and l).



**Fig. 3.** Results for Computational Time and Space for different datasets; (a–d) Infectious, (e–h) Hypertext, (i–l) College Message with fixed  $\Delta$  and  $\gamma$  settings

Now, Fig. 3 shows the result for computational time and space, in the similar setting of Fig. 2. In case of partition, we report the total time to compute  $\mathcal{C}^{T_2}$  by adding the time required to execute each partition. To compute the space, we report the maximum required space among both the partitions. From the Figs. 3 a, e, and i, it can be observed that the computational time decreases exponentially with the increase of  $\gamma$ . It is also observed that compared to the *all links*, the partition-based schemes lead to an improvement in computational time and the improvement is more when the  $\gamma$  value is less, due to the size of  $\mathcal{C}_{ex}^{T_1}$ .

For a fixed  $\Delta$ , when the value of  $\gamma$  increases, the space requirement decreases (Fig. 3 b, f, and j). Also, the partition schemes lead to an improvement in terms of space compared to All-Links. Similar result is reported for fixed  $\gamma$  case in Fig. 3 c, d, g, h, k, and l.

**Table 2.** Results for AS180

	$\Delta$	$\gamma$	With partition	Without partition
Time (h)	5	3	7.52344	20.42738
	10	3	8.24277	19.5564
	15	3	9.41666	17.89813
	20	3	9.52877	17.79293
	20	5	8.86788	>96
Space (GB)	5	3	77.42066	133.41058
	10	3	55.2932	92.70591
	15	3	47.7809	79.45884
	20	3	44.84441	74.34061
	20	5	50.54501	>256 GB

We experiment on a large dataset AS180 to observe improvement in both time and space on a broad scale. As the number of links in each time(day) is huge, we partition the entire dataset into three partitions with uniform link count based partition scheme. The result is reported in Table 2. For all the cases, the improvement compared to with and without partition is significant in both time and space.

## References

1. Akkoyunlu, E.A.: The enumeration of maximal cliques of large graphs. *SIAM J. Comput.* **2**(1), 1–6 (1973)
2. Banerjee, S., Pal, B.: On the enumeration of maximal  $(\delta, \gamma)$ -cliques of a temporal network. In: *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*, pp. 112–120. ACM (2019)
3. Barabási, A.L., et al.: *Network Science*. Cambridge University Press, Cambridge (2016)
4. Bron, C., Kerbosch, J.: Algorithm 457: finding all cliques of an undirected graph. *Commun. ACM* **16**(9), 575–577 (1973)
5. Cheng, J., Ke, Y., Fu, A.W.C., Yu, J.X., Zhu, L.: Finding maximal cliques in massive networks. *ACM Trans. Database Syst. (TODS)* **36**(4), 21 (2011)
6. Eppstein, D., Löffler, M., Strash, D.: Listing all maximal cliques in large sparse real-world graphs. *J. Exp. Algorithmics (JEA)* **18**, 3.1–3.21 (2013)
7. Holme, P., Saramäki, J.: Temporal networks. *Phys. Rep.* **519**(3), 97–125 (2012)
8. Isella, L., Stehlé, J., Barrat, A., Cattuto, C., Pinton, J.F., Van den Broeck, W.: What’s in a crowd? analysis of face-to-face behavioral networks. *J. Theor. Biol.* **271**(1), 166–180 (2011)
9. Kostakos, V.: Temporal graphs. *Phys. A: Stat. Mech. Appl.* **388**(6), 1007–1023 (2009)
10. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graphs over time: densification laws, shrinking diameters and possible explanations. In: *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pp. 177–187 (2005)
11. Mukherjee, A.P., Xu, P., Tirthapura, S.: Enumeration of maximal cliques from an uncertain graph. *IEEE Trans. Knowl. Data Eng.* **29**(3), 543–555 (2017)
12. Panzarasa, P., Opsahl, T., Carley, K.M.: Patterns and dynamics of users’ behavior and interaction: network analysis of an online community. *J. Am. Soc. Inf. Sci. Technol.* **60**(5), 911–932 (2009)

13. Viard, T., Latapy, M., Magnien, C.: Computing maximal cliques in link streams. *Theor. Comput. Sci.* **609**, 245–252 (2016)
14. Xiang, J., Guo, C., Aboulnaga, A.: Scalable maximum clique computation using MapReduce. In: 2013 IEEE 29th International Conference on Data Engineering (ICDE), pp. 74–85. IEEE (2013)
15. Xu, J.: *Topological Structure and Analysis of Interconnection Networks*, vol. 7. Springer, Boston (2013). <https://doi.org/10.1007/978-1-4757-3387-7>

# **Social Network**



# Web of Students: Class-Level Friendship Network Discovery from Educational Big Data

Teng Guo<sup>1</sup>, Tao Tang<sup>2</sup>, Dongyu Zhang<sup>1</sup>, Jianxin Li<sup>3</sup>, and Feng Xia<sup>2(✉)</sup>

<sup>1</sup> School of Software, Dalian University of Technology, Dalian 116620, China

<sup>2</sup> School of Engineering, IT and Physical Sciences, Federation University Australia, Ballarat, VIC 3353, Australia  
f.xia@ieee.org

<sup>3</sup> School of Information Technology, Deakin University, Melbourne, VIC 3125, Australia

**Abstract.** Classmate friendships are a major aspect of university social experience. Taking classes together is one of the main ways for students to build friendships. Consequently, class-level friendship networks have attracted tremendous attention from researchers. They are also very useful in student support and early intervention. However, these networks are normally invisible for educators. Discovering such an important web of students effectively is a pressing problem. Against this background, we propose a data-driven framework called CANDY which automatically discovers the class-level friendship networks based on educational big data. We first represent features through representation learning methods. Secondly, the data is augmented with the randomly shuffling method. Thirdly, a conditional generative adversarial network model is used to mine the class-level friendship networks. A deep adversarial optimization strategy is proposed here for problems caused by network sparsity. To evaluate the performance of the proposed approach, we build a real-world dataset that contains rich student information. Extensive experiments have been conducted and the results demonstrate the effectiveness of our framework.

**Keywords:** Social network analysis · Educational big data · Generative adversarial networks · Friendship networks

## 1 Introduction

Friendship plays an important role in everyone's life. Whether in personal development or social well-being, friendship matters a lot for everyone, especially for university students who are not physically and mentally matured [19, 25]. Taking classes together is one of the main ways for students to build friendships. Consequently, class-level friendship networks have attracted researchers tremendous attention [18, 23]. Studies indicate that abnormal behaviors including drinking, stress, depression, and suicide in various age of student groups are related to their repugnant friendships [20, 21]. However, such an important web of students is invisible and difficult to discover, which poses a challenge for the education management department. In this case, a significant topic of educational research is the discovery of class-level friendship networks based on known student information data stored in the education management system.

Discovering friendship networks among classmates faces tremendous challenges because their social choices might be impacted by various factors. Previous research shows that the social choice of students could be impacted by appearance features, psychology features, intellectual features, behavioral features, and various kinds of similar features as well [15, 32, 35]. From a methodological point of view, current research in this field can mainly be divided into two categories: questionnaire-based research [35] and link prediction-based research [22]. The questionnaire method is frequently used by statisticians to analyze the relationship between friendship and demographic characteristics through small batches of samples. In terms of efficiency, this method is costly and time-consuming, and difficult to use as a means of daily management. The link prediction aims to predict latent relationships based on known friendships. In a word, neither of these approaches can meet the requirements of practical applications that can infer the web of students automatically based on the data stored in the educational management system to assist the daily management of the university.

With the rise of artificial intelligence in this technological era [8, 11, 12], the research paradigm turns into the fourth stage. Big data technology has greatly advanced network science, which enriches the means for social network analysis [10, 34, 36]. These advancements provide us with an unprecedented opportunity to reveal the laws behind the web of students. Nevertheless, new challenges and limitations are introduced as well. Friendship relationships between people are interrelated rather than independent. Such a high-order network feature results in that we need to discover the overall friendship networks of students instead of predicting the existence of a single link independently. Therefore, for the aforementioned problem, a special solution framework is required.

Based on these observations, our research aims to discover an important web of students, i.e., students' class-level friendship networks. We are devoted to designing a framework for discovering friendship networks in each class through mining educational data stored in the university's management system, including ID photos (appearance feature), campus smart card records (behavior feature), course grades (intelligence feature), and psychological test scores (psychology feature). Therewith, we proposed a data-driven friendship network discovery framework, namely CANDY (C**l**A**s**s-level frie**N**dship network **D**iscover**Y**) by taking advantage of graph learning [29]. Firstly, we represent the features of each dimension as a dense vector through representation-learning related theory. Secondly, we design a  $G$  matrix to remove the interference of redundant information in the traditional adjacency matrix on the network generation experiment. Third, we use a random shuffle strategy on data augmentation to tackle the overfitting issue caused by the small dataset. Fourth, we use a Wasserstein distance-based conditional generative adversarial network (W-CGAN) [1] as the main generative model. In other words, we aim to train a generator which can generate class-level friendship networks based on the features mentioned above. In this step, we propose a deep adversarial optimization strategy for the training of the generative adversarial networks (GAN) based model to solve the problem caused by the sparsity of the matrix. Finally, we design  $P_G$ ,  $R_G$  and  $F1_G$ , as evaluation matrices which are the variants of precision, recall, and F1 score, to evaluate network generation comprehensively.

Our contributions could be summarized as follows:

- We design a framework for students' class-level friendship network discovery based on student information data in the education management system.

- We propose a deep adversarial optimization strategy for the GAN based model to solve the problem caused by the sparsity of adjacency matrices in the network discovery experiment.
- We conduct comprehensive experiments on the real-world dataset and the extensive results demonstrate the effectiveness of the CANDY framework.

This paper is organized as follows: In the next section, related work is reviewed briefly. The problem formulation is presented in Sect. 3. In Sect. 4, the proposed CANDY mining framework is introduced in detail. In Sect. 5, we introduce the details of our dataset. In Sect. 6, we explain the details of experiments and analyze their results. In the final section, we summarize the conclusion and future direction of our research.

## 2 Related Work

Research about social tie inferring attracts tremendous attention in the community of network analysis and network mining. Crandall et al. [4] propose a probabilistic framework to explore the connection between social relations and the number of co-occurrences, and demonstrated it based on the data of Flickr’s users, as early as of 2010. Since then, researchers pay great attention and effort to social tie inference based on various geo-located datasets. Olteanu et al. [17] carry out an experiment to explore the effect of co-location information that stems from social relationships between users on location privacy. The above-mentioned researchers cumulatively prefer to treat social networks extracted by co-occurrence as a feature for the various applications rather than to explore the detailed relationship between co-occurrence and diverse social relationships. Deng et al. [5] simulate the whole process of online friend-making (online friendship) through the construction of game models and performed some field experiments. Liu et al. [13] and Xia et al. [28] examine citation and collaboration networks of scholars through mining publication meta-data. Liu et al. [14] propose a model based on network representation learning, namely Shifu2, to discover advisor-advisee relationships hidden behind scientific collaboration networks.

Friendships of students always gain high popularity and are considered as a special kind of social behavior. Yao et al. [33] propose a semi-supervised method to detect the friend list of students. Based on the random model, they predict the friend list according to their co-occurrence record. They verify the correctness of the friend list by using it to predict academic performance. Xu et al. [30] consider that university students’ social behaviors show significant homophily in the aspect of major subject and course grade. Their research aims to find the social network of university students by eliminating the homophily effect. Khalil et al. [9] focus on the impact of course selection type on academic performance. The results demonstrate that students who took the same class with friends are more likely to achieve better grades than students who took the class alone or took the class with different students. Some studies attempt to reveal students’ social relationships, like [33]. However, they barely validate the methods effectively, except for simply using students’ co-presence as a characteristic to predict their achievement. Discovering student social relationships based on datasets from the education system remains an open topic.



### 3 Problem Formulation

In this section, we introduce notations in this paper and then formally define the research problem. Firstly, in our problem setting, we implement a directed graph  $\mathcal{G} = (V, E)$  to represent the class-level friendship network, where the node in  $V$  represents students and the edge in  $E$  represents friend relationships. The edge from node A to node B represents that student A considers student B to be his or her friend. The corresponding adjacency matrix of the friendship network  $\mathcal{G}$  is  $\mathbf{A}$ . For student  $i$ , we use  $\mathbf{x}_i = [x_{i1}, x_{i2}, \dots, x_{in}]$  to represent his or her personal features.  $\mathbf{X}_j$  represents the feature matrix for the class  $j$ . (The features used in this paper include appearance feature (ID photos), psychology feature (psychological test results), intelligence feature (course grades), and behavior feature (campus smart cards)). In this research, we assume that there is a mapping relationship between individual features and friendship networks, i.e.,  $\mathbf{A} = \mathcal{F}(\mathbf{X})$ . The goal of our study is to find this function  $\mathcal{F}$ .

**Friendship Network Discovery Problem:** Assuming there is a given feature matrix  $\mathbf{X}_j$  of a class, our aim is to discover the adjacency matrix of the corresponding friendship network  $\mathbf{A}$ .

### 4 Design of CANDY

In this section, we provide a specific description of the proposed framework, CANDY. This framework consists of five components: feature representation, network representation, data augmentation, generative model, and performance evaluation. The details of each component are as follows:

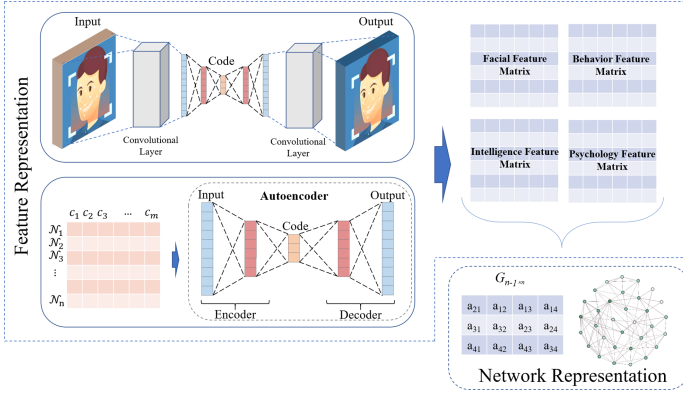
#### 4.1 Feature Representation

Previous research demonstrates that the social choice of an individual student could be impacted by other students' appearance features, psychological features, intellectual features, and behavioral features [15, 32, 35]. In this case, all these features are used in this research for friendship network discovery. To achieve more effective information mining, we process these features through the method of representation learning (shown in Fig. 1).

**Appearance Feature Representation.** In this subsection, we represent students' facial features by using their ID photos. To achieve a better and effective representation, the ID photo is processed by an auto-encoder, which is a neural network model used in a wide variety of fields [16, 31]. The auto-encoder is defined as follows:

$$\begin{aligned}
 \mathbf{h}_{(2)} &= f(\mathbf{W}_{(2)}\mathbf{h}_{(1)} + \mathbf{b}_{(2)}) \\
 \mathbf{h}_{(3)} &= f(\mathbf{W}_{(3)}\mathbf{h}_{(2)} + \mathbf{b}_{(3)}) \\
 &\dots \\
 \mathbf{h}_{(i)} &= f(\mathbf{W}_{(i)}\mathbf{h}_{(i-1)} + \mathbf{b}_{(i)}), i = 1, 2, \dots, k
 \end{aligned} \tag{1}$$

where  $f$  is the activation function, and  $\mathbf{W}_{(i)}$ ,  $\mathbf{b}_{(i)}$  are the transformation matrix and the bias vector, respectively.



**Fig. 1.** Feature representation in the CANDY framework.

**Behavior Feature Representation.** Student behavior similarity is measured by their co-occurrence of canteen in this research [32]. Each co-occurrence is defined as two students generating records in the canteen within a short time interval, set as 1 min [32]. The behavior feature representation is defined as follows: for student  $i$ , the similarity vector of behavior is  $s_i = [s_{i1}, s_{i2}, \dots, s_{ij}]$ , where  $s_{ij}$  represents the co-occurrence number of student  $i$  and student  $j$  in a particular place within a specified period of time. In this case, the co-occurrence matrix  $S_k$  for class  $k$  can be defined as follows:

$$\begin{pmatrix} 1 & s_{12} & \dots & s_{1n} \\ s_{21} & 1 & \dots & s_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ s_{n1} & s_{n2} & \dots & 1 \end{pmatrix}$$

**Intelligence Feature Representation.** In this sub-section, we use academic performance records to represent students' intelligence levels. The heterogeneity caused by the difference in the number of courses makes it difficult to be used as a feature for a machine learning model. For example, one student chooses courses A, B, and C and another student chooses courses C and D. In this case, the dimensions and contents of their academic performance features are different. To overcome the heterogeneity of students' academic performance, we employ the method mentioned in [7] for homogenization. Firstly, we embed their course through one-hot encoding and replace the 1 with the corresponding exam grade. In this way, we create the matrix  $C \in \mathbb{R}^{n \times m}$ , where  $n$  and  $m$  represent the number of students and the number of courses, respectively.

However, the number of courses taken by each student is much less than the total number of courses offered by the university. For example, in our dataset, the university offers 200 courses for students, and the number of courses students take per semester is around 18. Students are taking different classes, leading to the sparsity of the matrix  $C$ . To overcome this problem, we use an auto-encoder (Eq. 1) to reduce the dimension of high dimensions caused by the previous step for obtaining an effective representation of students' intelligence features.

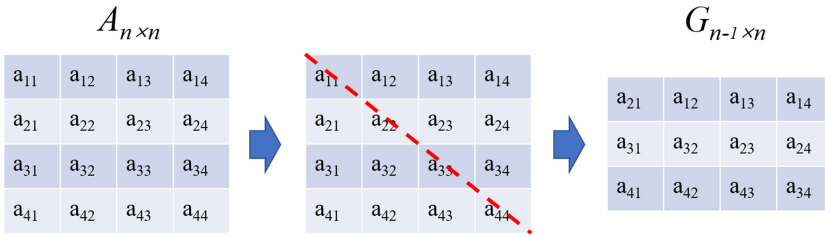


Fig. 2.  $G$  Matrix.

**Psychology Feature Representation.** Students’ psychological characteristics are collected through the Big Five personality traits which are widely used in student-related analysis research [26, 32]. The Big Five personality traits are considered as a traditional psychology model that includes five aspects: Openness, Conscientiousness, Extraversion (also often spelled as Extroversion), Agreeableness, and Neuroticism [6].

### 4.2 Network Representation

Generally, the adjacency matrix is used for representing the network structure. Here, we replace the adjacency matrix with its variant named  $G$  matrix, denoted by  $G$ . The initial idea is that the elements on the diagonal of the adjacent matrix do not need to be learned from data as they are constant to 0. In order to eliminate the redundancy information for efficient learning, we remove the diagonal of the adjacent matrix  $A_{n \times n}$  for obtaining  $G$  matrix  $G_{n-1 \times n}$ . An example is shown in Fig. 2 to illustrate this process.

### 4.3 Data Augmentation

Unlike routine prediction experiments, each sample label in this experiment corresponds to a small-scale friendship network. In general, this kind of data is difficult to collect on a large scale. Inspired by auxiliary task design in self-supervised learning [27], we design a data augmentation method to prevent overfitting from the small dataset. The basic idea of our method is to transform the organizational form of the data without changing the rules of node connection. Suppose there are  $n$  students in a class  $j$ , we number the students and use the number as the corresponding row or column number to form the adjacency matrix  $A_{j1}$ . Then we randomly shuffle the corresponding numbers of the students to regenerate the adjacency matrix  $A_{j2}$ . In this case, this process is repeated  $n$  times to get  $n$  adjacency matrices  $A_{j1}, A_{j2}, \dots, A_{jn}$ . (In the final experiment, these adjacency matrixes are processed by the methods in Sect. 4.2 to obtain the corresponding  $G$  matrix) For the feature matrix  $X_j$ , the corresponding row would be adjusted according to the student number change for obtaining  $n$  feature matrix:  $X_{j1}, X_{j2}, \dots, X_{jn}$ .

### 4.4 Generative Model

**W-CGAN.** In this research, we use a generative model to discover the topology of the friendship network i.e., this model could generate its corresponding adjacency matrix ( $G$  matrix). GAN (Generative Adversarial Networks) is a generative model based on neural network structure, which is widely used in various fields [2]. We aim to train a generator, which can generate the adjacency matrix of class-level friendship networks based on appearance features, psychology features, intellectual features, and behavioral features.

The main generative model used in this research is W-CGAN [1]. The generator in W-CGAN is the function  $\mathcal{F}$  in  $\mathcal{G} = \mathcal{F}(\mathbf{X})$  which is described in Sect. 3. Compared with classical GAN, WGAN solves the issue of vanishing gradients caused by Jensen-Shannon divergence through using smoothed Wasserstein Distance shown as follows:

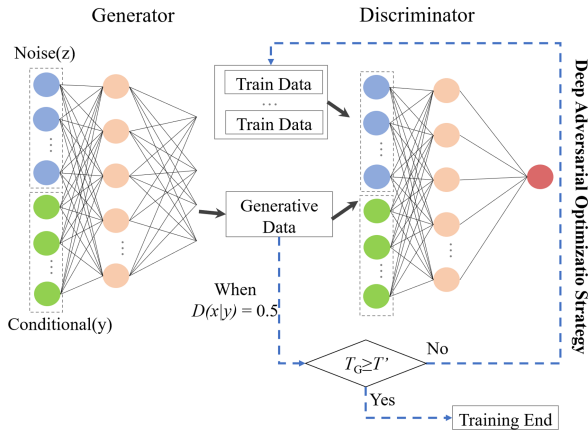
$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y) \sim \gamma} [\|x - y\|] \tag{2}$$

where  $\mathbb{P}_r$  is the real data distribution and  $\mathbb{P}_g$  is the distribution generated by the model.

The loss function of W-CGAN is based on the Kantorovich-Rubinstein duality which is clearly described as follows:

$$\min_G \max_{D \in \mathcal{D}} \mathbb{E}_{x \sim \mathbb{P}_r} [D(x|\mathbf{y})] - \mathbb{E}_{\tilde{x} \sim \mathbb{P}_g} [D(\tilde{x}|\mathbf{y})] \tag{3}$$

where  $G$  and  $D$  is generator and discriminator respectively.  $\mathcal{D}$  is the set of 1-Lipschitz functions. Actually,  $P_g$  is a distribution implicitly defined as:  $\tilde{x} = G(z)$ , where  $z$  is a combination of noises and features.



**Fig. 3.** The illustration of deep adversarial optimization strategy.

**Deep Adversarial Optimization Strategy.** Most friendship networks in the real world are sparse. Therefore, we can say there is no connection between most nodes. Likewise, it leads to most of the elements in the adjacency matrix being 0, i.e., the sparsity of the matrix. Without special treatment of sparsity, the model continuously generates all-0 matrices. In this situation, we propose an optimization strategy named as **Deep Adversarial Optimization Strategy** to overcome the target matrix's sparsity. The basic idea of this strategy is that, **lets the model learn from its own generated results to avoid the same mistakes** (shown in Fig. 3). The details are shown as follows:

1. **Step 1:** Train W-CGAN in the normal process based on real dataset  $R$  until that the discriminator is unable to differentiate between the two distributions, i.e.  $D(x) = 0.5$ . At this point, the generative result from the generator is  $R_{G(R)}$ .
2. **Step 2:** Choose an indicator  $T$  named Adversarial Indicator, which is an indicator used to evaluate the generated results from a certain aspect, and define the loop threshold value  $T'$ .
3. **Step 3:** Calculate the indicator  $T$  of  $R_{G(R)}$  and define it as  $T_G$ , and then define the adversarial condition like  $T_G \geq$  (or  $<$ )  $T'$ . If the conditions are met, the training ends. If not, put  $R_{G(R)}$  into the real dataset  $R$  as a negative sample, i.e., adversarial sample, and skip back to step 1.

#### 4.5 Performance Evaluation Methodology

Our goal is to generate the adjacency matrix corresponding to each class-level friendship network, which can be different from traditional classification and regression algorithms. Therefore, the traditional evaluation indicators like recall and precision are invalid here. To tackle this issue, we design a set of evaluation methods for network discovery: 1. We evaluate each of the generated matrices. 2. We take the mean value of the evaluation results for all generated matrices to get the final result.

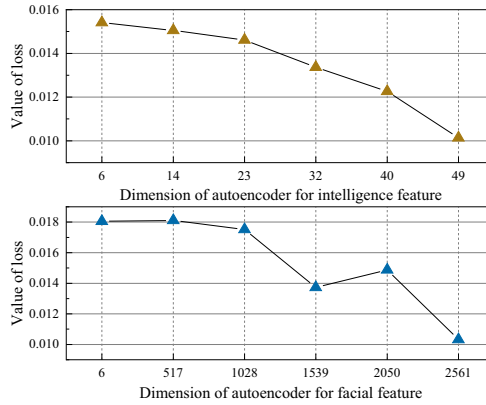
First, we evaluate the individual generated matrices. Intuitively, the result of the subtraction of the two matrices should be used as the evaluation criterion for the generative model. However, the sparseness of the data makes such evaluation methods ineffective. In this case, we introduce a confusion matrix to evaluate a single matrix sample. For the adjacent matrix, matrix elements only include zero and one. According to the labels of real data and their corresponding predicted results, we divide matrix elements into four categories: TO (True One), FO (False One), TZ (True Zero), and FZ (False Zero) and define the corresponding confusion matrix. Based on this confusion matrix, we define evaluation metrics for each generated matrix:  $P_G$  ( $G$  precision),  $R_G$  ( $G$  recall), and  $F1_G$  ( $G$  F1-score) as follows:

$$P_G = \frac{TO}{TO + FO} \quad (4)$$

$$R_G = \frac{TO}{TO + FZ} \quad (5)$$

$$F1_G = (1 + \beta^2) \frac{P_G \cdot R_G}{(\beta^2 \cdot P_G) + R_G} \quad (6)$$

In this work,  $F1_G$  is the harmonic mean of  $R_G$  and  $P_G$  ( $\beta = 1$ ). Then, we take the mean value of the evaluation results for all generated matrices to get the final results.



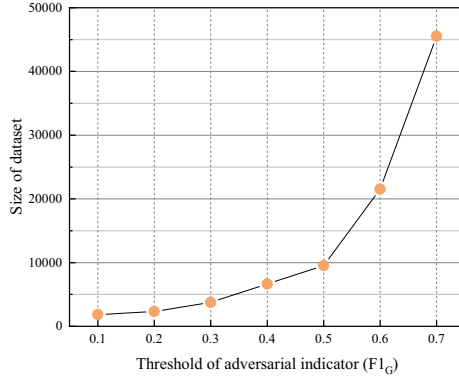
**Fig. 4.** The results of feature representation for facial feature and intelligence feature.

## 5 Dataset

The dataset used in this research includes 512 university students from the same Chinese university and all of them are freshmen who just finished their first semester exams. All participants are required to be more than 18-year-old freshmen (aged 18–20, mean = 19.03, SD = 0.21), who live in several specific freshman residential buildings (next to each other) in the same area. They come from 16 different classes. We use the data stored from the education management system, including their appearance feature (ID photos), psychology feature (psychological test results), intelligence feature (course grades), and behavior feature (campus smart card records) for discovering class-level friendship networks. This research was approved by the university's ethics committee.

The details of our dataset are introduced as follows:

1. **Friendship Network Data.** We use a questionnaire-based method for our data collection method in which each student is asked to write about 5–8 good friends. We collect friend relationships for each class in turn. Students from the same class are called to the lab and complete a friend relationship questionnaire. This activity takes place at the beginning of the student's second semester. Monetary compensation is given for participation in the study.
2. **ID Photo Data.** Our experiment utilizes personal one-inch photographs submitted by all participants at the time of enrollment in the university.
3. **Academic Performance Data.** Students' academic performance is generally recorded as the exam grade of each course. The academic performance data used in this research includes 13,234 records.
4. **Campus Smart Card Data.** In most universities, the smart card is used as a recognition tool for identifying a student. Generally, smart cards can be used for any scene of the behavior of university students and thus record tons of data for student behavior, such as bathing and eating. Financial data includes 259,513 records.



**Fig. 5.** The relation between the threshold of adversarial indicator and the size of the dataset.

5. Psychology Test Data. All university students are required to take psychological tests during their enrollment. We use that data to profile the psychological characteristics of the students involved in this experiment, which includes 30,720 records.

## 6 Experiments

### 6.1 Experimental Settings

For our research experiments, we use the friendship networks of twelve classes to train our model and the other four classes for testing. For the training set, we extend one network to thirty networks by using the data augmentation method mentioned in Sect. 4.3. In this case, we have 360 network samples in the training set in total. As discussed above, auto-encoders are used to obtain effective feature representation of facial features and intelligence features. We then test the performance on different dimensions (shown in Fig. 4) and the value of the loss function of the auto-encoder fluctuates slightly. Thus, we choose 6 as the final embedding dimension for computational efficiency.

In addition, as far as we know, there is no experiment exactly the same as our experiment. In this case, to verify the effectiveness of our framework, we design a comparison experiment based on binary classification: We consider the generation process of a friendship network with  $n$  nodes as  $n^2$  independent binary classification experiments. We make predictions about whether every two students are friends or not. For example, if a class has 32 students, we make 1024 ( $32 \times 32$ ) binary classification experiments to build an adjacency matrix of their friendship network (992 binary classifications for  $G$  Matrix). The classifiers used in this part are shown as follows:

- SVM (Support Vector Machine) [24]: SVM is a classic classification algorithm and is widely used in the field of data mining.
- XGBoost [3]: XGBoost is a boosting-tree-based method and is widely used in various data mining scenarios with good performance.

- DNN (Deep Neural Networks): DNN is a trendy model based on a multi-layer neural network and is widely used in various scenarios. (Our research implements a common three-layer neural network model)

The training set and testing set used for these binary experiments are synchronous with the main model proposed in this research.

## 6.2 Analysis of Results

The results of our generation experiments are shown in this part. The adversarial indicator  $T$  of the CANDY framework used in this paper is  $F1_G$  with adversarial condition  $F1_G \geq 0.7$  (The reasons are given below). The results of the CANDY framework and all comparison experiments are shown in Table 1. First, the CANDY framework proposed in this paper outperforms all comparison algorithms. The possible reason is that friendship between people is not independent of others. For example, A is a friend of B, and A is also a friend of C. Therefore, B and C may also be friends because they have mutual friends. In this case, independent binary classification experiments fail to catch these high-order network characteristics.

**Table 1.** Performance of all comparison experiments. CANDY ( $T_{(F1_G)} = 0.4$ ) represents the experiment results based on the CANDY framework when the adversarial indicator  $T$  is  $F1_G$  and the threshold is set to 0.4.

	$P_G$	$R_G$	$F1_G$
SVM	0.26976	0.26412	0.26691
XGBOOST	0.30108	0.29758	0.29932
DNN	0.42192	0.43217	0.42693
CANDY ( $T_{(F1_G)} = 0.4$ )	0.33499	0.33721	0.33609
CANDY ( $T_{(F1_G)} = 0.7$ )	0.52717	0.50165	0.51409

Although experiments verify the effectiveness of the CANDY framework, the overall performance is still not good enough as expected but the best F1-score is only 0.51409. The main reasons behind this are as follows.

First, each sample in this experiment corresponds to a matrix, so the complexity of the experiment is greatly increased. Second, this experiment is equivalent to conducting 992 interconnected binary classification tasks at the same time. For such a challenging task, the dataset used in this research is too small to support good performance. Although we propose a data augmentation method to mitigate the impact of the small-scale dataset, such a linear method can not completely solve this problem. More data involved in model training will help to achieve better results. Thirdly, as the purpose of this paper is to emphasize the effectiveness of the CANDY framework instead of pursuing high accuracy, we did not collect enough features for the experiment. For example, we only use the co-occurrence of the cafeteria as a feature to infer the friendship network. Co-occurrence in more places can also effectively help in improving the



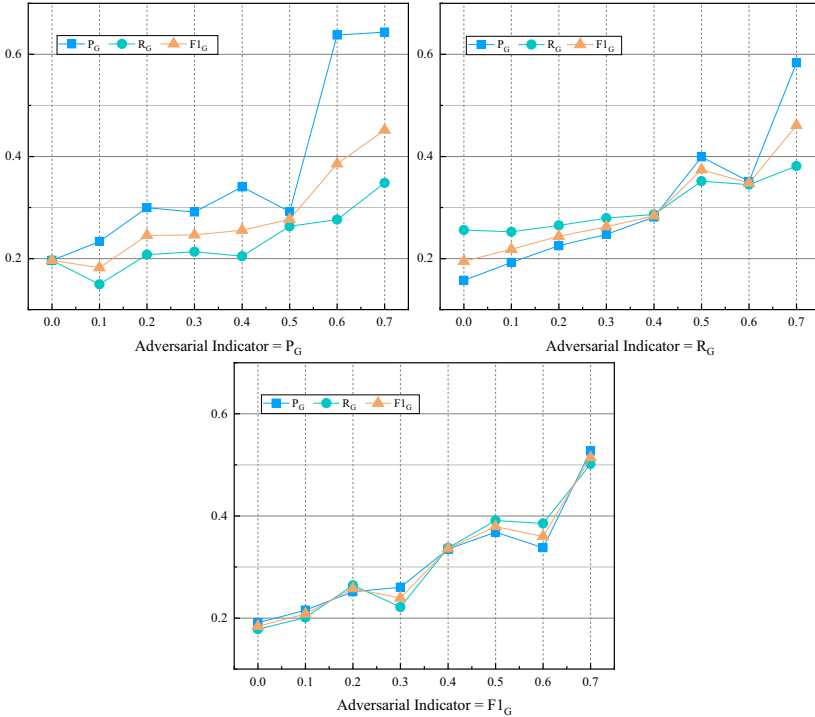


Fig. 6. The performances of CANDY with different adversarial conditions.

experimental results, e.g. that research by [30] uses the co-occurrence of 17 places to infer the friendship network. A more comprehensive feature is bound to result in higher performance.

In addition, the threshold of indicator  $T$  is set at 0.7 because the size of the training set increases with the time of training. According to the deep adversarial optimization strategy proposed in this research paper, ‘half-finished products’ from the generator will consistently be added to the training set as long as the requirement is not met, leading to that concerning the training set will increase with the time of training. For example, if we choose  $F1_G$  as the adversarial indicator, the changing trend of the size of the training set is shown in Fig. 5. An extensive training set leads to longer model training times. In this case, the adversarial indicator  $T$  is set to 0.7 to balance the training process and the algorithm’s performance.

Moreover, the selection of adversarial indicator  $T$  is crucial for our algorithm and its efficiency because the indicator can significantly impact the algorithm’s performance. We perform experiments to analyze this issue. We set the adversarial indicators as  $P_G$ ,  $R_G$ , and  $F1_G$ , separately, and for each adversarial indicator, we take values from 0.1 to 0.7 to test its impact on performance. We use  $P_G$ ,  $R_G$ , and  $F1_G$  to quantify the results, and the results are shown in Fig. 6. It can be observed that different indicators have different effects. The higher the threshold set on the training set, the better the performance

on the test set. The experimental results demonstrated that  $F1_G$  is an optimal candidate for the adversarial indicator, which is consistent with the intuition that  $F1_G$  is affected by both  $P_G$  and  $R_G$ .

**Table 2.** Performance with different variants. For concise presentation in the table, we use shorthand to represent each part of CANDY framework: R (Raw data), A (Data augmentation), W-CG (W-CGAN),  $G$  ( $G$  matrix).

	$P_G$	$R_G$	$F1_G$
R+W-CG	0.08912	0.12418	0.10377
R+A+W-CG	0.18057	0.17394	0.17719
R+A+W-CG+ $G$	0.19056	0.17794	0.18403
CANDY ( $T_{(F1_G)} = 0.4$ )	0.33499	0.33721	0.33609
CANDY ( $T_{(F1_G)} = 0.7$ )	0.52717	0.50165	0.51409

Besides, we compare the performance of the CANDY framework with its variants as well, and the results are shown in Table 2. Note that the W-CGAN means ordinary W-CGAN without a deep adversarial optimization strategy. First, feeding the raw data into W-CGAN achieves an inferior performance, because the raw dataset only contains 16 networks, causing serious overfitting issue. By contrast, the improvement of ‘R+A+W-CG’ demonstrates the effectiveness of our data augmentation. The  $G$  matrix improves the performance but not sharply. Nevertheless, it contributes to the training efficiency of the model by reducing redundant information. Finally, we added the deep adversarial optimization strategy (i.e., CANDY framework) with  $T = F1_G$  and the performance is greatly improved. In other words, the experimental results in this part demonstrate the effectiveness of each part of the CANDY framework.

## 7 Conclusion

In this work, we propose a data-driven framework to discover students’ class-level friendship networks based on student data from the education management system, including students’ ID photos, psychological test results, course grades, and the record of campus smart cards. First, we represent features as low-dimensional dense vectors through representation learning. Secondly, we use conditional GAN with Wasserstein distance as the main generative model and propose a deep adversarial optimization strategy to tackle the problem caused by the sparsity of adjacency matrices. Finally, we transplant the evaluation system of classification experiments into the network generation for achieving a comprehensive evaluation. The performance on a real-world dataset validates the effectiveness of the proposed framework. In future work, in order to achieve a comprehensive understanding of the social patterns among students, a series of indicators will be explored to quantify students’ social patterns from a dynamic and static perspective, respectively.

## References

1. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: Proceedings of the International Conference on Machine Learning 2017, pp. 214–223. PMLR (2017)
2. Chen, J., Li, Y., Ma, K., Zheng, Y.: Generative adversarial networks for video-to-video domain adaptation. In: Proceedings of the 32th AAAI Conference on Artificial Intelligence, pp. 3462–3469. AAAI Press (2020)
3. Chen, T., Guestrin, C.: XGBoost: a scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794. ACM (2016)
4. Crandall, D.J., Backstrom, L., Cosley, D., Suri, S., Huttenlocher, D., Kleinberg, J.: Inferring social ties from geographic coincidences. *Proc. Natl. Acad. Sci.* **107**(52), 22436–22441 (2010)
5. Deng, X., Song, D., Wei, L.: A dynamic game model analysis for friendship selection. *J. Intell. Fuzzy Syst.* **1**, 1–9 (2019)
6. Goldberg, L.R.: An alternative ‘description of personality’: the big-five factor structure. *J. Pers. Soc. Psychol.* **59**(6), 1216 (1990)
7. Guo, T., et al.: Graduate employment prediction with bias. In: Proceedings of the 32th AAAI Conference on Artificial Intelligence, pp. 670–677. AAAI Press (2020)
8. Hernández, I., Rivero, C.R., Ruiz, D.: Deep web crawling: a survey. *World Wide Web* **22**(4), 1577–1610 (2019)
9. Khalil, L.J., Khair, M.G.: Social network analysis: friendship inferred by chosen courses, commuting time and student performance at university. *Int. J. Reason.-based Intell. Syst.* **10**(1), 59–67 (2018)
10. Lande, D., Fu, M., Guo, W., Balagura, I., Gorbov, I., Yang, H.: Link prediction of scientific collaboration networks based on information retrieval. *World Wide Web* **23**(4), 2239–2257 (2020)
11. Liu, J., et al.: Artificial intelligence in the 21st century. *IEEE Access* **6**, 34403–34421 (2018)
12. Liu, J., et al.: Data mining and information retrieval in the 21st century: a bibliographic review. *Comput. Sci. Rev.* **34**, 100193 (2019)
13. Liu, J., Ren, J., Zheng, W., Chi, L., Lee, I., Xia, F.: Web of scholars: a scholar knowledge graph. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2153–2156 (2020)
14. Liu, J., et al.: Shifu2: a network representation learning based model for advisor-advisee relationship mining. *IEEE Trans. Knowl. Data Eng.* **33**(4), 1763–1777 (2021)
15. Morelli, S.A., Ong, D.C., Makati, R., Jackson, M.O., Zaki, J.: Empathy and well-being correlate with centrality in different social networks. *Proc. Natl. Acad. Sci.* **114**(37), 201702155 (2017)
16. Muhammed, Fatih, B., Abubakar, A., James Y, Z.: Concrete autoencoders for differentiable feature selection and reconstruction. In: Proceedings of the 36th International Conference on Machine Learning, pp. 444–453. PMLR (2019)
17. Olteanu, A.M., Huguenin, K., Shokri, R., Humbert, M., Hubaux, J.P.: Quantifying inter-dependent privacy risks with location data. *IEEE Trans. Mobile Comput.* **16**(3), 829–842 (2016)
18. Overgoor, J., Adamic, L.A., et al.: The structure of us college networks on Facebook. In: Proceedings of the International AAAI Conference on Web and Social Media, vol. 14, pp. 499–510 (2020)
19. Parkinson, C., Kleinbaum, A.M., Wheatley, T.: Similar neural responses predict friendship. *Nat. Commun.* **9**(1), 332 (2018)

20. Pickering, T.A., et al.: Diffusion of a peer-led suicide preventive intervention through school-based student peer and adult networks. *Front. Psychiatry* **9**, 598 (2018)
21. Ream, G.L.: The interpersonal-psychological theory of suicide in college student suicide screening. *Suicide Life-Threat. Behav.* **46**(2), 239–247 (2016)
22. Ren, J., et al.: Matching algorithms: fundamentals, applications and challenges. *IEEE Trans. Emerging Top. Comput. Intell.* **5**(3), 332–350 (2021)
23. Rodríguez-Triana, M.J., Prieto, L.P., Holzer, A., Gillet, D.: Instruction, student engagement, and learning outcomes: a case study using anonymous social media in a face-to-face classroom. *IEEE Trans. Learn. Technol.* **13**(4), 718–733 (2020)
24. Scholkopf, B., Smola, A.J.: *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge (2001)
25. Van Duijn, M.A., Zeggelink, E.P., Huisman, M., Stokman, F.N., Wasseur, F.W.: Freshmen into a friendship network. *J. Math. Sociol.* **27**(2–3), 153–191 (2003)
26. Vedel, A.: The big five and tertiary academic performance: a systematic review and meta-analysis. *Pers. Individ. Differ.* **71**, 66–76 (2014)
27. Wang, X., He, K., Gupta, A.: Transitive invariance for self-supervised visual representation learning. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1329–1338 (2017)
28. Xia, F., Liu, J., Ren, J., Wang, W., Kong, X.: Turing number: how far are you to AM Turing award? *ACM SIGWEB Newsl. (Autumn)*, 1–8 (2020)
29. Xia, F., et al.: Graph learning: a survey. *IEEE Trans. Artif. Intell.* **2**, 109–127 (2021)
30. Xu, J.Y., Liu, T., Yang, L.T., Davison, M.L., Liu, S.Y.: Finding college student social networks by mining the records of student id transactions. *Symmetry* **11**(3), 307 (2019)
31. Xu, W., Tan, Y.: Semisupervised text classification by variational autoencoder. *IEEE Trans. Neural Netw. Learn. Syst.* **31**(1), 1–14 (2019)
32. Yao, H., Lian, D., Cao, Y., Wu, Y., Zhou, T.: Predicting academic performance for college students: a campus behavior perspective. *ACM Trans. Intell. Syst. Technol. (TIST)* **10**(3), 24 (2019)
33. Yao, H., Nie, M., Su, H., Xia, H., Lian, D.: Predicting academic performance via semi-supervised learning with constructed campus social network. In: Candan, S., Chen, L., Pedersen, T.B., Chang, L., Hua, W. (eds.) *DASFAA 2017*. LNCS, vol. 10178, pp. 597–609. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-55699-4\\_37](https://doi.org/10.1007/978-3-319-55699-4_37)
34. Yin, H., Yang, S., Song, X., Liu, W., Li, J.: Deep fusion of multimodal features for social media retweet time prediction. *World Wide Web* **24**, 1027–1044 (2020)
35. Zhang, D., et al.: Judging a book by its cover: the effect of facial perception on centrality in social networks. In: *Proceedings of the Web Conference 2019*, pp. 2290–2300. ACM (2019)
36. Zhou, Y., et al.: Extracting representative user subset of social networks towards user characteristics and topological features. *World Wide Web* **23**(5), 2903–2931 (2020)



# Event Cube for Suicidal Event Analysis: A Case Study

Qing Li<sup>1</sup>, Zhihan Yan<sup>2</sup>, Jun Li<sup>1</sup>, Zhenguo Yang<sup>3(✉)</sup>, Zehang Lin<sup>1(✉)</sup>,  
Hong Va Leong<sup>1</sup>, Lei Chen<sup>4</sup>, and Nancy Xiaonan Yu<sup>5,6</sup>

<sup>1</sup> Department of Computing, The Hong Kong Polytechnic University, Hong Kong, China

<sup>2</sup> Department of Electronic Electrical Engineering, University College London, London, UK

<sup>3</sup> School of Computer Science, Guangdong University of Technology, Guangzhou, China

<sup>4</sup> Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong, China

<sup>5</sup> Department of Social and Behavioural Sciences, City University of Hong Kong, Hong Kong, China

<sup>6</sup> Shenzhen Research Institute of City University of Hong Kong, Shenzhen, China

**Abstract.** The publicly available data, such as the massive and dynamically updated news and social media data streams (a.k.a. big data), cover a wide range of social activities, personal views, and expressions. Effective research and application rely heavily on the ability of comprehending and discovering the knowledge patterns underlying this big data, from which the notion of an event serves as a cornerstone in building up more complex knowledge structures. Establishing methodologies and techniques for discovering real-world events from such large amounts of data, as well as for managing and analyzing such events in an efficient and aesthetic manner, is crucial and challenging. In this paper, we present an event cube framework devised to support various collection, consolidation, fusion, and analysis tasks for suicidal events. More specifically, we present a mechanism for data collection over multiple data sources in both passive and active manners, and promote the mappings constructed from various representation spaces for data consolidation. Furthermore, multimodal fusion is devised to integrate multiple data intrinsic structures and learn discriminative data representations so as to process heterogeneous multimodal data efficiently. Finally, the event cube model is developed to support event organization and contextualization with hierarchical and analytical operations. A case study is provided to demonstrate the capabilities and benefits of our event cube facilities supporting on-line analytical processing of suicidal events and their relationships.

**Keywords:** Event cube · Suicidal event · Social media

## 1 Introduction

With the advent of the 5G technology, more and more people are sharing what is happening around them in different forms (e.g., text, image and video) through social media platforms (e.g., Flickr, Twitter and Weibo). This provides us the possibility to mine

from the huge amount of data (a.k.a. big data) on these platforms for something useful, like real-world event detection. Real-world event detection means to detect real-world events that have occurred, are occurring, or are about to occur, which can help us better understand and predict the patterns of these events. In fact, real-world events can be large public events (e.g., earthquake, fire, etc.) or small personal events (e.g., suicide, crime, etc.). In this paper, we focus on suicidal event detection, hoping to reduce the risk of suicide in advance by learning the patterns of events in this category.

Currently, research works on event detection can be divided into two categories according to the data source: news media and social media. News media refers to a number of online news stories which are texts written by a few professional news staff. Most of the traditional event detection work is done through news media to discover patterns of public events that have occurred and attracted widespread attention. For example, Kumaran et al. [1] use text classification and named entities for event detection on news media. However, it is difficult to find out what is going to happen through news media because it only reports the content of events that have already happened. Social media refers to some social platforms (e.g., Flickr, Twitter and Weibo) that allow most users to publish their own content, which is more diverse in forms (e.g., image, text and video). In recent years, with the growth and popularity of social media, a great deal of research work has been devoted to detecting events from social media, as it may contain elements of upcoming events that can help detect events that are going to happen. For instance, Takeshi et al. [2] use Twitter data for the detection of public events (i.e., earthquakes). Particularly for suicidal events, social media data can identify people with suicidal ideation in advance. For example, Sawhney et al. [3] utilize Twitter data for suicide ideation detection. However, social media data suffers from inconsistent data formats, with severe noise levels. Therefore, we have collected suicidal events data by combining news media and social media data hoping to balance out their undesired impact.

In this paper, we present an event cube framework for suicidal events, which consists of four components: collection, consolidation, fusion and analysis. In particular, we first develop a mechanism that automatically (actively and passively) collects multimodal data to integrate the data. We then represent the consolidated data by using multimodal fusion learning, so as to effectively address the problematic gap between heterogeneous data. Finally, we utilize the event cube (EC) model for multiple on-line operations to analyze event relationships. Through a case study, we demonstrate the capabilities of the event cube, as well as how the on-line operations and event analysis facilitate the analysis of suicide events.

The rest of this paper is organized as follows. Section 2 describes the related work. Section 3 describes the methods, including the model framework, data collection, data consolidation, data fusion and analyzing facilities of EC. Section 4 provides a case study of EC used for suicidal events. Finally, Sect. 5 concludes this paper.

## 2 Related Work

### 2.1 Event Detection

Event detection aims to discover real-world events from the Internet platforms, e.g., social media, which can be divided into *targeted event detection* (TED) and *untargeted event detection* (UED) according to whether labels are used or not. Specifically, TED usually utilizes supervised classification models. For instance, Takeshi et al. [2] detect natural disasters by combining spatial, temporal, and keyword features with a Kalman Filter or Particle Filter and a support vector machine classifier. Thien et al. [4] propose an event detection neural network model based on graph convolutional networks over dependency trees and entity mention-guided pooling. However, the aforementioned methods do not take into account the multi-source nature of data. UED exploits unsupervised clustering models to discover possible events without labeling information. Weng et al. [5] propose a wavelet-based method for detecting Twitter events. Feng et al. [6] present an online multi-scale event detection approach based on hashtags clustering in geo-tagged documents. However, these approaches either exploit only limited features or have a high computational cost when fusing different modalities.

### 2.2 Event Relationship Analysis

Textual data has been used in event relationship analysis. Yang et al. [7] propose to utilize the temporal relationship, event similarity, temporal proximity, and document distributional proximity to identify the event evolution relationships between events in a terror attack incident. Deng et al. [8] introduce the concept of public opinion field to distinguish event information and public opinion in the text corpus and analyze event relationships according to the public opinion field. For modeling how an event is dependent on another event within a topic, Huang et al. [9] explore three kinds of event relationships: co-occurrence dependence relationship, event reference relationship, and temporal proximity relationship. Zhou et al. [10] model event relationships using TFIEF and temporal distance cost factor and construct event relationship graphs with event sequence, event content similarity, and event temporal cost function.

### 2.3 Online Analytical Processing

Online analytical processing (OLAP) is based on the data cube to operate and analyze multi-dimensional data. Gray et al. [11] define the data cube concept in the data warehousing and mining field, which is used to store a measure of interest along multiple dimensions. Li et al. [12] introduce a data cube for dominant relationship analysis, which captures the dominant relationships between products and customers. In summary, the data cube is suitable for storing and modeling multi-dimensional data, and it can perform data queries and analyses from multiple angles and multiple levels. In contrast to data cube, event cube [13] stores an event in each cell, the dimensions of which include both structured and unstructured data.

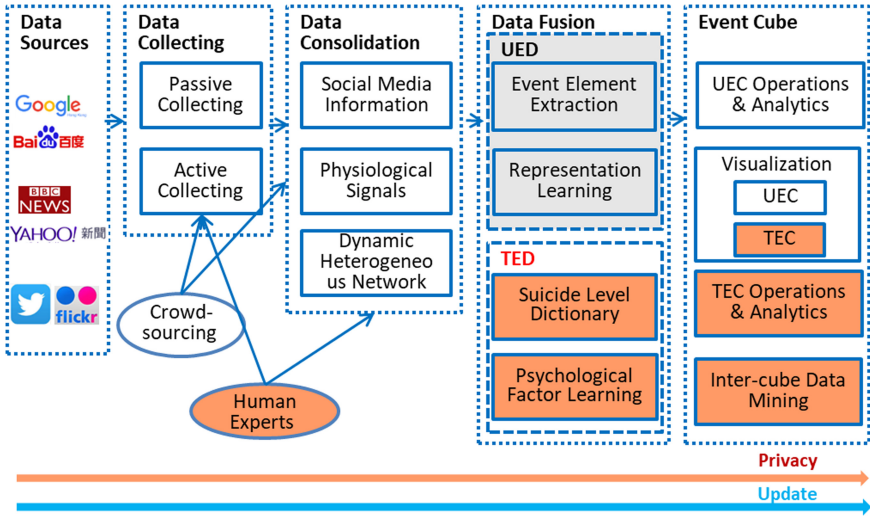


Fig. 1. Overview of the EC framework.

### 3 The Event Cube Paradigm

#### 3.1 Overview of the Framework

The architecture of the proposed EC framework is shown in Fig. 1. Owing to multiple data sources, passive and active data collection mechanisms are developed for data collection. In particular, crowd-sourcing techniques and simulated distribution models have been developed to ensure data quality while preserving privacy. In terms of data consolidation, the mappings are built from different representation spaces for data integration, along with knowledge base and crowd sourcing being combined for data denoising. Furthermore, multimodal fusion is conducted to integrate multiple data intrinsic structures and learning discriminative data representations, so as to process heterogeneous multimodal data efficiently. Finally, an event cube (EC) model has been proposed with hierarchical and analytical operations to support event organization and contextualization [13].

#### 3.2 Data Collection

Generally, the data can be categorized into passive and active ones. The passive data is collected from social media on the Internet regarding general person-specific information and contextual occurrences. Meanwhile, active data can be collected by using wristband devices that could capture and return human physiological signals, in addition to conventional accelerometer signals; the former incorporates textual and visual data, while the latter comprises body temperature, Blood Volume Pulse (BVP signal to determine heart beat rate), and Electrodermal Activity (EDA signal to evaluate skin conductivity).



**Social Media Data Crawling and Collection.** For data collection from social media, e.g., Weibo, Twitter, the type of information to be collected varies significantly. Individual user accounts are linked to the collected data, on which sentiment analysis is performed. Contextual data is linked to the context of posts, such as key events like the COVID-19 epidemic. Extracted posts are fed into a deep-learning network, which learns the latent space of words and maps them to emotional status in six dimensions. Emotional signals reflected in the posts are consolidated into multi-dimensional time series that are saved for efficient calculation and retrieval. This time series data is referred to as a *user's emotional journal*. Suicidal situations as a rather particular type of uncommon domains correspond to black swan-type circumstances. To collect enough positive examples, we start with confirmed cases and go backward to collect enough relevant posts from a victim and his or her “friends” or associated components in the social network, as well as a type of “follower” idea. For more effective machine learning, data augmentation is used to generate more cases at various stages of suicidal attempts. For confirmed cases, a general trend of going towards more serious warning levels can be detected, and early alert for following up actions will be devised.

**Physiological Data Collection.** For useful movement and physiological signals, the data can be collected from individuals using a data-collecting wristband. The availability of physiological data is limited due to the requirement of wearing the wristband and some sort of intermittent self-reporting of emotional status, whereas subject movement data is more readily available. To transform the raw signal to more relevant signals, signal processing is required, such as extracting the base signal (tonic component) from an EDA signal in order to elicit the true fluctuation (rapid fluctuating phasic part) indicative of human emotion. These physiological signals have been discovered to be beneficial in detecting human activities and stress. As is known to all, human stress is strongly linked to suicidal attempts, therefore those who are constantly stressed can be identified and referred to appropriate counseling.

### 3.3 Data Consolidation

Since the data to be integrated comes from various sources and can be of multiple modalities, a knowledge base needs to be built in addition to a database, so as to supply data instances for the generation of event cubes, as well as for machine learning classification and event identification.

**Consolidation of Social Media Information.** A graph database is used to store the intra-subject and inter-subject relationships between extracted subjects and their posts. There exist relationships between subjects (friends, family, etc.) and relationships between posts (quoting or similarity in content) either explicitly or implicitly. Such kinds of complicated relationships are best modeled as a large graph with parallel edges and hyper-edges, where edges indicate different degrees of impact. The related data contained in the graph database can be queried efficiently. The integrated data along with the discovered changes in a user's emotional journal can facilitate detecting potential cases from the connected components even earlier.

**Consolidation of Physiological Signals.** We have adopted a machine learning approach to build models which can classify human emotion status based on the collected data. SVM and other models have been shown to be able to learn from data instances generated from data for human stress, as well as human fall (mostly due to accidents). Wong et al. [14] revealed strong accuracy on stress detection and excellent accuracy on activity recognition using SVM, logistic regression, decision trees, and kNN. Fall detection in various directions, an essential activity type, is also explored with good accuracy [15]. Likewise, the identified emotions can be integrated with the emotion signal time series obtained from social media for the same individual.

**Dynamic Heterogeneous Network.** Heterogeneous networks and graphs have been built by using multi-sourced data in multiple modalities to support various application scenarios, such as knowledge extraction [16], anomaly detection [17], etc. Relational constraints [18] allow users to specify fine-grained connection requirements between vertices in heterogeneous networks, as well as a community detection algorithm with near-linear time complexity. Furthermore, scalable network representation learning [19] and random walk model abstraction allow users to easily explore different transition probabilities by specifying dynamic edge weights and random walk status. For different knowledge graph tasks, Path-interstellar (Interstellar) Search Recurrent Architecture [20] is proposed for the representation of short-term and long-term information along the paths of knowledge graph. In real-world applications, efficient approximate solutions for large-scale data [21] are often preferred over inefficient exact solutions, especially for some fundamental problems like trip planning, task assignment, facility location planning, scoring function search [22]. Metric embedding aims to map the geometric data from the original space into a simpler or more special space (e.g., from graphs to trees, from high-dimensional spaces to low-dimensional spaces), so that one only needs to solve the same problem on this simple embedding space in the context of metric embedding. Due to the high time and space complexity of existing Hierarchically Separated Tree (HST)-based solutions, a dynamic programming-based method [23] can be devised to improve efficiency. In real-world applications, member (not limited to data) updates are not uncommon, thus approaches like the Stable Learned Bloom Filters (SLBF) [24] are proposed to address the performance decay issue on intensive insertion workloads by combining classifier with updatable backup filters. In terms of pre-trained multi-source models that are available, there is no principled way to select appropriate models for reusing. Consequently, an ensemble method [25] can help select a subset of source models that achieves the best performance for a target task.

### 3.4 Data Fusion for Event Detection

For heterogeneous data fusion, two alternative frameworks can be adopted, one based on matrix factorization and the other based on deep learning. On one hand, event detection can focus on utilizing data from multimedia domains, such as news and social media. In-domain and cross-domain Laplacian regularization (ICLR) model [26] has been used to learn effective data representation for both textual news reports contributed by journalists in the news media domain, and image posts shared by amateur users in the

social media domain. Classification and clustering strategies help leverage the obtained data representation to discover existing and novel events, respectively. More specifically, ICLR is devised to construct respective Laplacian regularization terms based on the inter-domain and intra-domain label consistency properties, which can be optimized using an alternating optimization technique with theoretical convergence guarantees.

On the other hand, shared semantic space with correlation alignment [27, 28] can align nonlinear correlations of multimodal data distributions in deep neural networks designed for heterogeneous data. A neural network has been designed by using convolutional layers and fully-connected layers to extract features from images, including images on Flickr-like social media, in the context of cross-modal (event) retrieval. Simultaneously, a fully-connected neural network is exploited to extract semantic features from text documents, such as news articles from the news media. In particular, during the joint training of the two neural networks, nonlinear correlations of layer activations in the two neural networks are aligned by correlation alignment.

### 3.5 Event Cube for Analysis

To facilitate better understanding and discovery of the knowledge patterns underlying big data, and to support the development of methodologies and techniques for discovering real-world events, an event cube (EC) model [13] has been designed to accommodate a variety of event queries and analytical tasks; such events include those discovered by techniques of untargeted event detection (UED) and targeted event detection (TED) from multi-sourced data. Specifically, based on the essential event elements of “5W1H” (i.e., When, Where, Who, What, Why, and How), the EC model is developed to organize the discovered events from multiple dimensions, to operate on the events at various levels of granularity, so as to facilitate analyzing and mining hidden/inherent relationships among the events effectively. The intra-cube operations include Roll-up, Drill-down, Slice, Dice, Pivot, and X-validate, and the inter-cube operations include Union, Intersect, Subtract, and Scoping [13].

Since each event is described by a set of dimensions, such as When, Where, Who, How, Why, etc., the atomic operations were first defined on single dimensions, based on which we derive more complex relationships, such as content similarity relationships, content dependence relationships, etc., in terms of inter-cube event relation analysis. For cubes sharing the required dimensions, the aforementioned intra-cube relationship analyzing operations can still be applied for inter-cube event relation analysis. The term and technique were borrowed from data mining to coin these relations as association relationships, and then we extract them using association rule learning on multiple dimensions. In terms of the Why dimension, the causal relationships were explored to find the reasons for different emotions from social media data [29, 30].

## 4 Case Study

### 4.1 Datasets

As part of our research, we have collected reasonably large suicidal datasets, including 367 suicidal ideation ones from Reddit, and 13 student suicidal events (containing 3

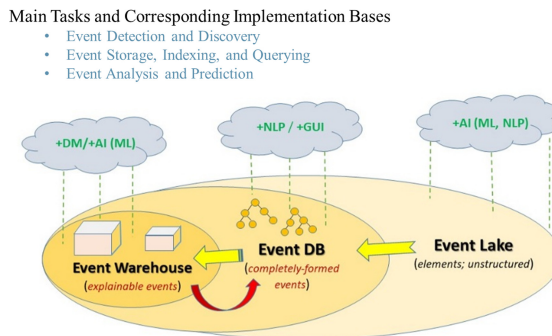
**Table 1.** Samples of the suicide event dataset.

<b>When</b>	2019-12-15	2020-05-11	2020-06-30
<b>Where</b>	A hotel near Peking University, Haidian District, Beijing, China	Communication University of China, Chaoyang District, Beijing, China	Sunshine Shangdong Community, Baoji High-tech Zone, Baoji City, Shaanxi, China
<b>Who</b>	Gao XX	Huang XX	Chen XX
	College Student	Postgraduate student	Primary school student
<b>What</b>	A female college student was tortured by her boyfriend and committed suicide	A female postgraduate student killed herself	A sixth-grade girl committed suicide
<b>Why</b>	Boyfriend abuse	Her supervisor disagreed with her graduation defense	Dissatisfaction with the exam results
<b>How</b>	Take drug	Jump off	Jump off

subevents) in China between 2019 and 2020 from online news (e.g., Sina, Sohu, etc.) and social media (e.g., Weibo). For our case study here, we focus on the latter ones since each of these events includes the “5W1H” elements, i.e., time (When), location (Where), person and group (Who), title and content (What), reason (Why), and method (How). As shown in Table 1, we depict three of these events for illustration purpose. In particular, for what, we only show the brief content as the content is too long.

### 4.2 Main Architecture and Components

As shown in Fig. 2, we have adopted a 3-tiered architecture in the EC framework based on the technologies of data lake, data warehouse, and database, to support the functions of event detection and discovery, event storage and querying, as well as event analysis and prediction, respectively.



**Fig. 2.** The main architecture and implementation bases of the EC prototype.

For the backend, PostgreSQL is used for the management of the consolidated data, and is intended for subsequent expansion of the dataset. Meanwhile, we have also developed a web front-end by using Flask for the user interface, as shown in Fig. 3.

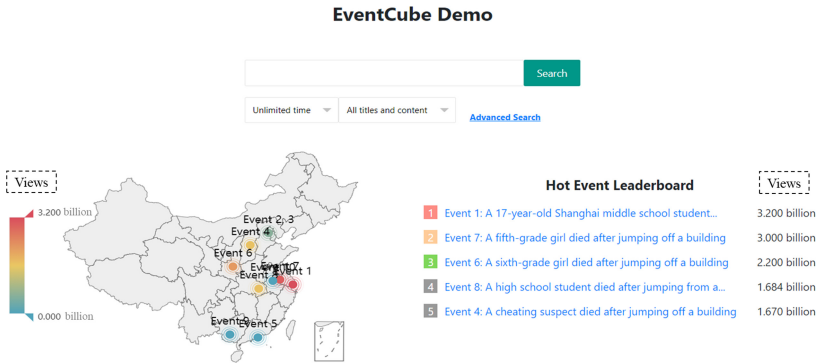


Fig. 3. The main interface of the EC prototype.

**Main Interface Facilities.** In order to make it intuitive and easy to use for users, a graphical interface is designed to guide and help users to quickly understand and analyze suicide events. As shown in Fig. 3, the main interface consists of three parts, i.e., the search bar, the hot event leaderboard and the event visualization map. Each component is described as follows.

- 1) **Search Bar:** We adopt keywords for the search and provide the user with constraints on time (e.g., unlimited time, nearly a week, nearly a month and nearly a year) and scope (i.e., all titles and content, only title, and only content), allowing the user to find specific events quickly. As shown in Fig. 4, when we search events with the keyword “jump”, the system returns the desired search results and allows the user to learn more about the events by clicking on a specific event. In addition, we also

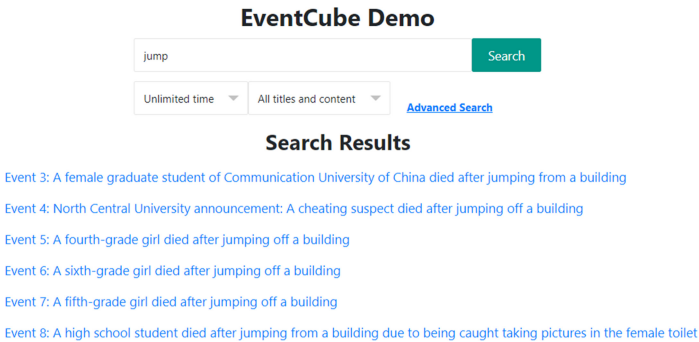


Fig. 4. The search results with the search bar.

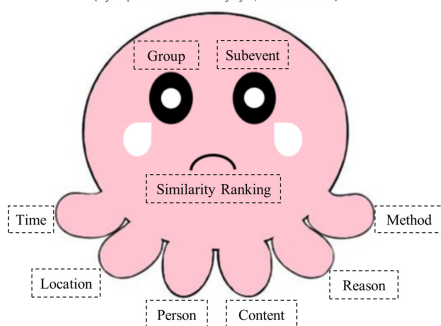
develop an advanced search function based on the characteristics of suicide events, which will be illustrated later in conjunction with the EC operations.

- 2) **Hot Event Leaderboard:** In order to give users a quick overview of influential events, we create the hot event leaderboard, which is ranked according to the hotness of the event. In particular, we utilize real-time crawling technology on Weibo Hotspots to capture user clicks on an event in real time to reflect its hotness.
- 3) **Event Visualization Map:** The event visualization map reflects popular events from a different perspective (i.e., location). We present all events in a map combined with a heat map, showing the two dimensions of where the event is taking place and how hot it is, which gives users a clearer and more intuitive view of all the events.

**Event Interface.** Just like any type of events, a suicidal event also has the “5W1H” elements, so we adopt the octopus’s icon as the graphical representation of the event. As shown in Fig. 5 (here we take Event 6 as an example), the unhappy octopus represents Event 6 itself, with its eyes, mouth and feet all having their own functions. Specifically, the two eyes represent group and subevent, respectively; a mouth represents similarity ranking, and six feet represent time, location, person, content, reason and method, respectively. We can observe that, with the exception of subevent and similarity ranking, the other parts correspond to the “5W1H” elements. Note that the eye on subevent refers to subevents of the shown event, which can be subsequent events or closely related ones of the event. Similarity ranking means the combined similarity ranking of the event with respect to other events in terms of the dimensions of “5W1H”. Their corresponding functions are to be described further in the following sections in conjunction with EC operations.

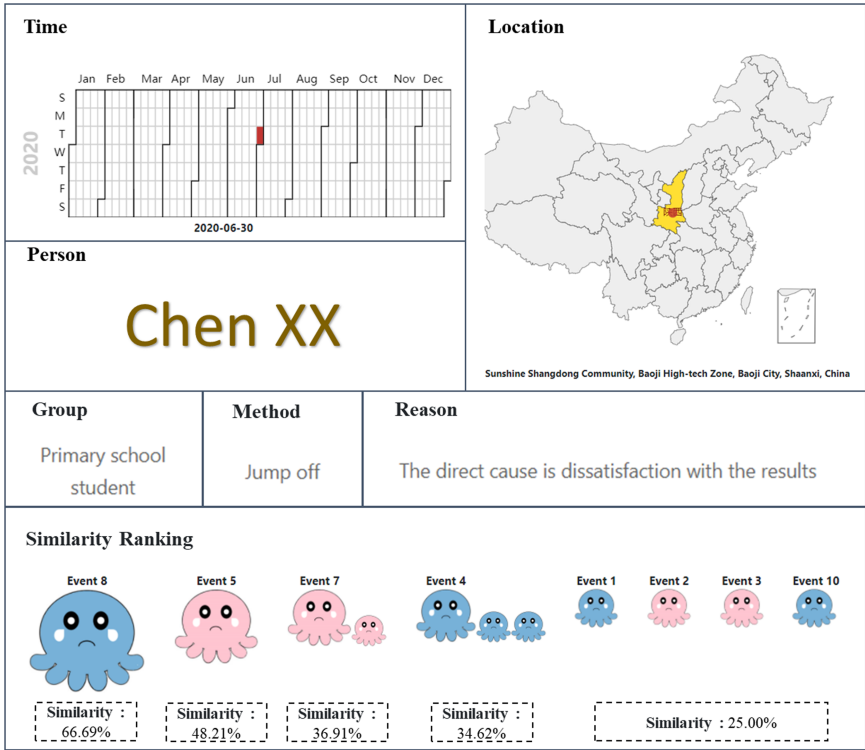
**Event 6: A sixth-grade girl died after jumping off a building**

(Try to put the mouse on my eyes, mouth and feet)



**Fig. 5.** The event interface of the EC prototype (Event 6 as an example).

When a user clicks on the various parts of an octopus, he/she can retrieve specific information visually, as shown in Fig. 6. For the similarity ranking part, the color of the octopus represents the sex of the suicidal case (i.e., pink for female and blue for male), the size represents the similarity level, and the small octopus in the bottom right corner of a larger octopus represents its subevents. With these interactive functions, users can quickly find out similar events and identify the specific event they are interested in.



**Fig. 6.** The detail of each part in the octopus (Event 6 as an example). Note that the content is ignored here due to space limitations.

### 4.3 Case Study with EC Operations

To better understand the EC model, we illustrate some cases of our system with the operations defined by the EC model. Considering that we are only studying suicidal events, the operations for the cases here only involve the intra-cube operations, i.e., Drill-down operations, Slice operations and event relationship analysis.

**Drill-Down.** The Drill-down operation refers to the conversion of a high-level concept to a low-level concept via a concept hierarchy, which makes a concept more refined in a certain dimension. As illustrated in Fig. 7, when we perform the Drill-down operation on Event 4 (here we use the Event hierarchy as an example), we can obtain its subevents, i.e., Event 4-1 (the related event) and Event 4-2 (the subsequent event). The subevents also have the same functions and actions as their parent event, which allows users to get a clearer and more detailed picture of all aspects related to this suicidal event.

**Slice.** The Slice operation refers to the filtering of events that satisfy the conditions with respect to a dimension of the events. As introduced in Sect. 4.2.1, the keyword search through the Search Bar can be seen as the Slice operation on the what dimension. Furthermore, we also propose an advanced search function based on the idea of Query

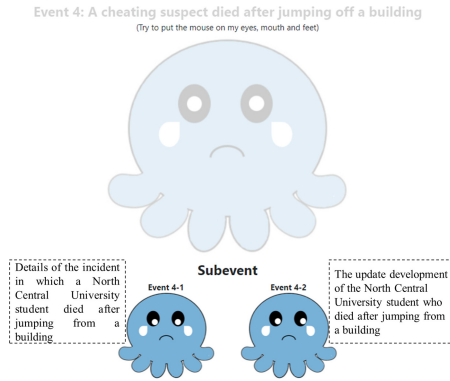


Fig. 7. The subevent of an octopus (Event 4 as an example).

By Example (QBE). As shown in Fig. 8, we apply the Slice operation to the Time (When) dimension, limiting the time of the event to 2020 to 2021. After the Slice operation, users can filter some events according to their desired criteria and then analyze/dig out more useful information.

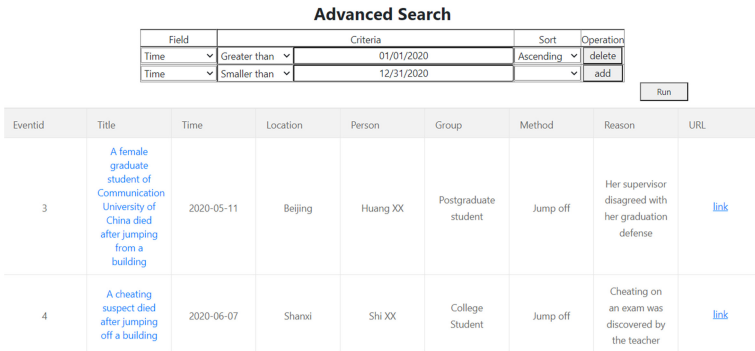
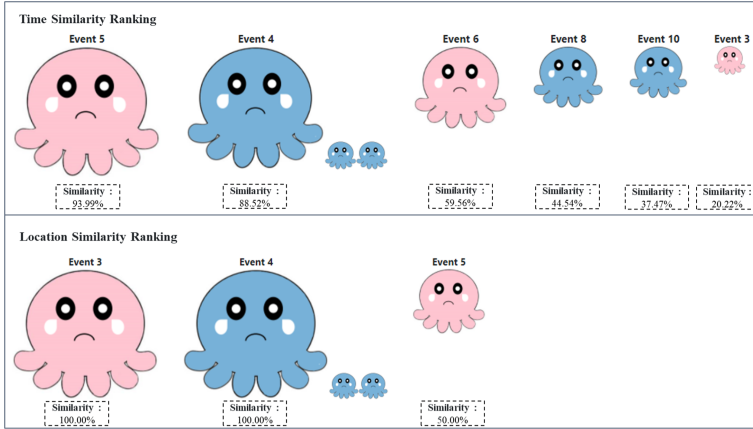


Fig. 8. The Slice operation of EC (here only the first two events are listed due to space constraint).

**Event Relationship Analysis.** Event relationship analysis refers to the measuring the similarity between two events with a given similarity metric, which can be conducted for single-dimensional and multi-dimensional event relationship types. For single-dimensional event relationship analysis (using the time and location dimensions of Event 7 as an example), we use the corresponding similarity formulations defined by the EC model [13] for event relationship analysis. As shown in Fig. 9, we obtain the similarity ranking with Event 6 in the corresponding dimensions (i.e., time and location), which allows users to analyze other related events in terms of these dimensions. For multi-dimensional event relationship analysis (using the “5W1H” dimensions of Event 6 as an example), the similarity ranking of the event is obtained by weighing the similarity relationships calculated for each dimension, and obtain the final similarity score through



weighted sum. Such an aggregated similarity ranking can be more informative for users as it combines the similarity relationships of multiple event dimensions, which provides with a holistic view of inter-event relationship analysis.



**Fig. 9.** Two example (When and Where) dimensions for the similarity ranking function (Event 6 as an example).

## 5 Conclusion

In this paper, we present an event cube (EC) framework which is utilized for the collection, consolidation, fusion and analysis tasks of suicidal events. Specifically, a data collection mechanism capable of actively and passively collecting multiple data sources of social media is devised for data integration. At the same time, multimodal fusion learns the intrinsic representation of data from multiple sources, enabling efficient handling of heterogeneous data. In addition, the proposed EC model supports multiple functions and provides intensive event relationship analysis. A case study is provided to demonstrate the capabilities and benefits of our EC facilities supporting on-line analytical processing of suicidal events and their relationships. Our on-going research focus on event causality relationships discovery and prediction, with an aim towards *explainable events* through mining relationships along the Why dimension.

**Acknowledgements.** We are greatly indebted to Dr. Xingyun Liu for her valuable comments and insightful suggestions to our event cube prototype and suicidal analysis. The research described in this paper has been supported by the Hong Kong Research Grants Council through a Collaborative Research Fund (project no. C1031-18G) and Shenzhen Philosophy and Social Sciences Fund in the 13th Five-year Plan (project no. SZ2018B020), P. R. China.

## References

1. Kumaran, G., Allan, J.: Text classification and named entities for new event detection. In: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 2004, pp. 297–304 (2004)
2. Sakaki, T., Okazaki, M., Matsuo, Y.: Earthquake shakes Twitter users: real-time event detection by social sensors. In: Proceedings of the 19th International Conference on World Wide Web, April 2010, pp. 851–860. ACM (2010)
3. Sawhney, R., Joshi, H., Gandhi, S., Shah, R.: A time-aware transformer based model for suicide ideation detection on Social Media. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), November 2020, pp. 7685–7697 (2020)
4. Nguyen, T., Grishman, R.: Graph convolutional networks with argument-aware pooling for event detection. In: Proceedings of the AAAI Conference on Artificial Intelligence, April 2018, vol. 32, no. 1 (2018)
5. Weng, J., Lee, B.S.: Event detection in twitter. In: Proceedings of the International AAAI Conference on Web and Social Media, July 2011, vol. 5, no. 1 (2011)
6. Feng, W., et al.: STREAMCUBE: hierarchical spatio-temporal hashtag clustering for event exploration over the Twitter stream. In: 2015 IEEE 31st International Conference on Data Engineering, April 2015, pp. 1561–1572. IEEE (2015)
7. Yang, C.C., Shi, X., Wei, C.-P.: Tracing the event evolution of terror attacks from on-line news. In: Mehrotra, S., Zeng, D.D., Chen, H., Thuraisingham, B., Wang, F.-Y. (eds.) *Intelligence and Security Informatics*, pp. 343–354. Springer, eidelberg (2006). [https://doi.org/10.1007/11760146\\_30](https://doi.org/10.1007/11760146_30)
8. Deng, L., Xu, B., Zhang, L., Han, Y., Zou, P.: Event evolution analysis in microblogging based on a view of public opinion field. In: 2013 Sixth International Symposium on Computational Intelligence and Design, October 2013, vol. 2, pp. 193–197. IEEE (2013)
9. Huang, D., Hu, S., Cai, Y., Min, H.: Discovering event evolution graphs based on news articles relationships. In: 2014 IEEE 11th International Conference on e-Business Engineering, November 2014, pp. 246–251. IEEE (2014)
10. Zhou, P., Wu, B., Cao, Z.: Emmbtt: a novel event evolution model based on TFxIEF and TDC in tracking news streams. In: 2017 IEEE Second International Conference on Data Science in Cyberspace (DSC), June 2017, pp. 102–107. IEEE (2017)
11. Gray, J., et al.: Data cube: a relational aggregation operator generalizing group-by, cross-tab, and sub-totals. *Data Min. Knowl. Disc.* **1**(1), 29–53 (1997)
12. Li, C., Ooi, B.C., Tung, A.K., Wang, S.: Dada: a data cube for dominant relationship analysis. In: Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data, June 2006, pp. 659–670 (2006)
13. Li, Q., Ma, Y., Yang, Z.: Event cube – a conceptual framework for event modeling and analysis. In: Bouguettaya, A., et al. (eds.) *Web Information Systems Engineering – WISE 2017: 18th International Conference, Puschino, Russia, October 7-11, 2017, Proceedings, Part I*, pp. 499–515. Springer International Publishing, Cham (2017). [https://doi.org/10.1007/978-3-319-68783-4\\_34](https://doi.org/10.1007/978-3-319-68783-4_34)
14. Wong, J.C.Y., Wang, J., Fu, E.Y., Leong, H.V., Ngai, G.: Activity recognition and stress detection via Wristband. In: Proceedings of the 17th International Conference on Advances in Mobile Computing & Multimedia, December 2019, pp. 102–106 (2019)
15. Fu, E.Y., Wong, C.Y., Lau, K.T., Leong, H.V., Ngai, G.: Your body signals expose your fall. In: Proceedings of the 21st International Conference on Information Integration and Web-Based Applications & Services, December 2019, pp. 689–693 (2019)
16. Xin, H., Lin, X., Chen, L.: CaSIE: canonicalize and informative selection of the OpenIE. In: Proceedings of ICDE 2021 (To appear) (2021)

17. Li, J., Di, S., She, Y., Chen, L.: FluxEV: a fast and effective unsupervised framework for time-series anomaly detection, WSDM 2021 (To appear) (2021)
18. Jian, X., Wang, Y., Chen, L.: Effective and efficient relational community detection and search in large dynamic heterogeneous information networks. *Proc. VLDB Endowm.* **13**(10), 1723–1736 (2020)
19. Yao, X., Shao, Y., Cui, B., Chen, L.: UniNet: scalable network representation learning with metropolis-hastings sampling. In: *Proceedings of ICDE* (2020)
20. Zhang, Y., Yao, Q., Chen, L.: Interstellar: searching recurrent architecture for knowledge graph embedding. In: *Proceedings of NeurIPS 2020* (2020)
21. Wang, Y., et al.: Disk: a distributed framework for single-source simrank with accuracy guarantee. *Proc. VLDB Endowm.* **14**(3), 351–363 (2020)
22. Di, S., Yao, Q., Zhang, Y., Chen, L.: Efficient relation-aware scoring function search for knowledge graph embedding. *ICDE 2021* (To appear) (2021)
23. Zeng, Y., Tong, Y., Chen, L.: HST+: an efficient index for embedding arbitrary metric spaces. In: *Proceedings of ICDE 2021* (2021)
24. Liu, Q., Zheng, L., Shen, Y., Chen, L.: Stable learned bloom filters for data streams. *Proc. VLDB Endowm.* **13**(12), 2355–2367 (2020)
25. Li, Y., Shen, Y., Chen, L.: Palette: towards multi-source model selection and ensemble for reuse. In: *Proceedings of ICDE 2021* (To appear) (2021)
26. Yang, Z., Li, Q., Xie, H., Wang, Q., Liu, W.: Learning representation from multiple media domains for enhanced event discovery. *Pattern Recogn.* **110**, 107640 (2021)
27. Yang, Z., Lin, Z., Kang, P., Lv, J., Li, Q., Liu, W.: Learning shared semantic space with correlation alignment for cross-modal event retrieval. *ACM Trans. Multim. Comput. Commun. Appl.* **16**(1), 1–22 (2020)
28. Yang, Z., Lin, Z., Guo, L., Li, Q., Liu, W.: MMED: a multi-domain and multi-modality event dataset. *Inf. Process. Manag.* **57**(6), 102315 (2020)
29. Chen, X., Li, Q., Wang, J.: A unified sequence labeling model for emotion cause pair extraction. In: *Proceedings of the 28th International Conference on Computational Linguistics, December 2020*, pp. 208–218 (2020)
30. Chen, X., Li, Q., Wang, J.: Conditional causal relationships between emotions and causes in texts. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), November 2020*, pp. 3111–3121 (2020)



# Cross-modal Attention Network with Orthogonal Latent Memory for Rumor Detection

Zekai Wu<sup>1</sup>, Jiaxin Chen<sup>1</sup>, Zhenguo Yang<sup>1</sup>(✉), Haoran Xie<sup>2</sup>, Fu Lee Wang<sup>3</sup>,  
and Wenyin Liu<sup>1,4</sup>(✉)

<sup>1</sup> Guangdong University of Technology, Guangzhou, China  
{yzg, liuwy}@gdut.edu.cn

<sup>2</sup> Lingnan University, Hong Kong, China  
hrxie@ln.edu.hk

<sup>3</sup> Hong Kong Metropolitan University, Hong Kong, China  
pwang@hkmu.edu.hk

<sup>4</sup> Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen, China

**Abstract.** In this paper, we design a cross-modal attention fusion network with orthogonal latent memory (CALM) to fuse multi-modal social media data for rumor detection. Given multimodal content features extracted from text and images, we devise a cross-modal attention fusion (CAF) mechanism to extract critical information underlying the modalities by intra-modality attention, and model the underlying relations among the modalities by inter-modality attention. In terms of the text, the natural sequential characteristics are critical to semantic understanding, while existing sequence models suffer from losing the information conveyed by the former words. To this end, we propose a Bi-GRU with orthogonal latent memory to extract the sequential features from the text, where the memory captures independent patterns. The fused content features and the sequential features can be used for rumor detection seamlessly. Extensive experiments conducted on two real-world datasets show the outperformance of the proposed CALM. (e.g.,  $F_1$ -score is improved from 0.823 to 0.846 on Weibo dataset).

**Keywords:** Rumor detection · Multi-modal · Social media

## 1 Introduction

Social media has revolutionized the way for people to acquire information, while it may foster the propagation of fake news like rumors in turn. Some offenders even use rumors to guide public opinion, damage the credibility of the government and even interfere with the general election [1]. Rumor detection aims to identify the rumors distributed on social media like platforms, where the data usually are in multiple modalities such as text, image, and videos, etc., being verisimilitude to the interest of most people.

In terms of the methodologies, the early works usually focus on textual news. For instance, Castillo et al. [4] extract message-based and topic-based features from the textual content and exploit a decision tree method to classify posts. Yu et al. [16] use a convolutional approach to extract key features and shape high-level interactions from textual content of the relevant posts. Recent studies have shown that detecting rumors in a multi-modal manner can achieve better performance, especially with deep learning methods. For instance, Khattar et al. [7] propose a novel VAE model to learn a shared representation of the modalities for detecting rumors. Yang et al. [15] apply a Ti-CNN method to detect rumors by extracting both explicit and latent multi-modal features within news content. In terms of fusing the heterogeneous modalities in the context of rumor detection, quite a few fusion strategies show impressive performance. For instance, Jin et al. [6] propose an attention mechanism to fuse visual, textual and social context features. Chen et al. [5] propose a self-attentive fusion mechanism to integrate the textual features with visual features. The aforementioned approaches suffer from a few deficiencies. Firstly, these methods either pay more attention to the semantic information or sequential information in social media textual data merely. Secondly, the existing approaches usually concatenate the multimodal features or introduce attention mechanism to weight the importance of modalities, neglecting correlations and interactions underlying the modalities.

In this paper, we design a cross-modal attention fusion network with orthogonal latent memory (denoted as CALM) to detect rumors from multimodal social media data. On one hand, we propose a cross-modal attention fusion mechanism with intra-modality and inter-modality attentions, where intra-modality attention extracts critical information underlying the single modalities, and inter-modality attention establishes the relations among multiple modalities. On the other hand, we extend Bi-GRU with orthogonal latent memory to capture long-distance temporal dependencies in the sequential models, avoiding gradient vanishing and exploding. In particular, orthogonal constraint on the latent memory ensures the diversity of the underlying patterns from global viewpoint.

The main contributions are summarized as follows.

- We propose a cross-modal attention fusion framework with intra-modality and inter-modality attentions to capture the modality-specific information and model the underlying relations among the multiple modalities.
- We devise an orthogonal latent memory to keep diverse latent patterns from the global viewpoint, which can be plugged in GRU-like sequential models to capture the long-distance temporal dependencies.
- We conduct extensive experiments on two real-word datasets, which show the outperformance of the proposed approach compared with the state-of-the-art baselines.

The rest of this paper is organized as follows. Section 2 summarizes the related works. Section 3 presents the proposed CALM. Section 4 shows the experiments and analyzes the experimental results. Section 5 concludes the work.

## 2 Related Work

In this section, we briefly review the works on multi-modal rumor detection and multi-modal data fusion.

### 2.1 Multi-modal Rumor Detection

Social media has become the main platform for people to obtain and share information, which may lead to the spread of rumors extremely fast in turn. The research attention on rumor detection has shifted from text-based approaches to multi-modal ones recently. For instance, Zhang et al. [18] employed a pre-trained BERT model to identify rumors and used a domain classifier to remove event-specific dependency. Zhang et al. [17] designed a knowledge-aware network and an event memory network for social media rumors. Zhou et al. [19] exploited multi-modal and relational information to learn the representation of articles and predict rumors. However, the textual extractor employed by prior studies either mainly focused on the semantic information or sequential information.

### 2.2 Multi-modal Data Fusion

Multi-modal data fusion aims to combine multi-aspect information from multiple data modalities, which are critical for various machine learning tasks [8, 10]. In the context of rumor detection, quite a few multi-modal data fusion approaches have been devised to deal with the multimodal data. For instance, Wang et al. [14] concatenated the visual features and textual features of social media data to get a multi-modal feature. Jin et al. [6] proposed a recurrent neural network with an attention mechanism to fuse image and text features. Chen et al. [5] proposed a self-attentive fusion mechanism to integrate the textual features with visual features for detecting rumors. The aforementioned methods can hardly discover latent correlations among the multiple modalities as the complementarity among multimodal features has not been fully explored.

## 3 Methodology

### 3.1 Overview of the Framework

The overall framework of the proposed CALM is shown in Fig. 1, which consists of four components, i.e., the visual extractor, the textual extractor, the cross-modal attention fusion (CAF) network and the rumor detector. The visual extractor and textual extractor extract visual and textual features from social media data. Specifically, the textual extractor can extract both semantic features and sequential features. Furthermore, the CAF component fuses multimodal content features extracted from text and images by inter-modality attention and intra-modality attention. Finally, the rumor detector concatenates the learned features as input to predict whether the social media data is rumor or non-rumor.

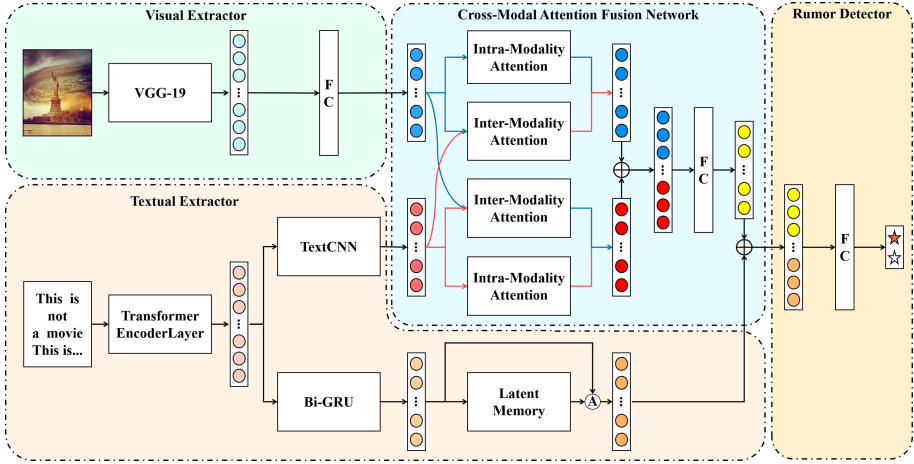


Fig. 1. Overview of CALM.

### 3.2 Visual Extractor

The attached image  $v$  of the social media data is fed into the visual extractor. We employ the VGG-19 network [11] to extract visual features, which has achieved impressive performance on multiple computer visual tasks. We extract the CNN feature for image from the fc-7 layer in VGG-19 and feed it into a fully connected layer to reduce the dimension down to  $d_m$ . The visual features  $V \in R^{d_m}$  can be obtained as follows:

$$V = \sigma(W_v \cdot VGG(v)) \tag{1}$$

where  $VGG(\cdot)$  is the pre-trained VGG-19 model,  $W_v$  is the weight matrix of the fully connected layer and  $\sigma(\cdot)$  is the activation function used.

### 3.3 Textual Extractor

We divided textual extractor into two sub-modules, content feature extraction and sequential feature extraction.

**1) Content Feature Extraction.** The textual input  $t$  to the textual extractor is the sequential list of words in the posts,  $t = [t_1 t_2 \dots t_n]$ , where  $n$  is the number of words in the text. Each word  $t_i \in t$  is represented as a word embedding vector, which is extracted with a pre-trained word2vec model. In order to obtain better understanding of the language structure, we employ the Transformer Encoder [12] to calculate and assign weights for different words in  $t$ . With  $E$  denotes as the encoder output, the operation can be obtained as follows:

$$E = TransformerEncoder(t) \tag{2}$$

noted that  $E = [E_1 E_2 \dots E_n]$ , where  $E_i$  is the encoder result of  $t_i$ .

More specifically, to capture the semantic features from text of the social media data, the content feature extraction exploits Text-CNN [9] to automatically capture semantic features in different granularities. Furthermore, the feature map produced by Text-CNN is fed into a fully connected layer to ensure the semantic features have the same dimension as the visual features  $V$ . Given the encoder output  $E$ , the semantic features  $T \in R^{d_m}$  can be calculated as follows:

$$T_t = TextCNN(E) \tag{3}$$

$$T = \sigma(W_t \cdot T_t) \tag{4}$$

where  $TextCNN(\cdot)$  is the Text-CNN model and  $W_t$  is the weight matrix in the fully connected layer.

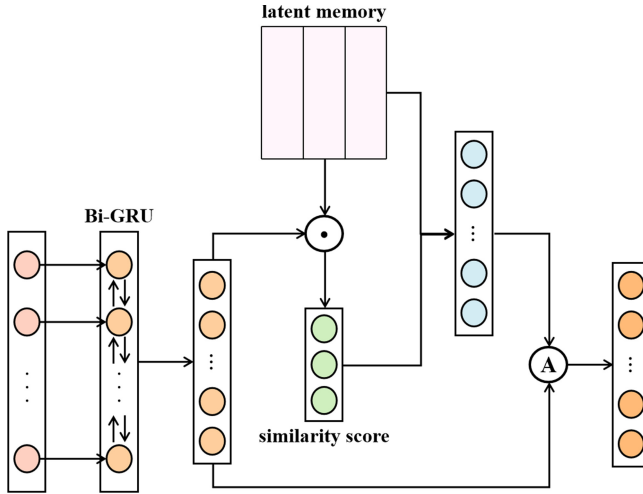


Fig. 2. The details of sequential feature extraction.

**2) Sequential Feature Extraction.** Existing sequence models suffer from a problem of vanishing and exploding gradients. This leads to the model learning inefficient dependencies between words that are a few steps apart. To overcome this problem, a latent memory network is introduced to improve Bi-GRU, which can not only make up for the defects of the sequence models, but also output the extra global latent patterns information shared by rumors. The details of the sequential feature extraction are provided in Fig. 2.

More specifically, given the input  $E$ , we use a Bi-GRU to compute the hidden state for each element and concatenate the last hidden state from both directions, denoted as  $R_{gru} \in R^{2*d_m}$ . Subsequently, we pass the  $R_{gru}$  through a fully connected layer to calculate the preliminary sequence features  $F_g$ . The operation can be represented as follows:

$$R_{gru} = GRU_{bi}(E) \tag{5}$$

$$F_g = \sigma(W_g \cdot R_{gru}) \tag{6}$$



where  $GRU_{bi}(\cdot)$  represents the Bi-GRU model and  $W_g$  is the weight matrix of the fully connected layer.

Furthermore, the patterns information in memory are chosen to strengthen the sequence features. In particular, the memory network is denoted as  $M \in R^{num \times d_m}$ , where  $num$  is depended on the number of latent patterns underlying the social media data. We calculate the similarity score  $M_{score}$  between the sequence features  $F_g$  and the latent patterns, which can be obtained by conducting softmax function on their dot product as follows:

$$M_{score} = softmax(M^T \cdot F_g) \tag{7}$$

Finally, we extract the closest patterns based on the similarity score and merge the resulting patterns information  $F_m$  with the sequence features  $F_g$  through conducting average operation. The final sequence features  $T_g \in R^{d_m}$  can be obtained as follows:

$$F_m = (M \cdot M_{score}) \tag{8}$$

$$T_g = avg(F_g, F_m) \tag{9}$$

where  $avg(\cdot)$  represents the average operation.

### 3.4 Cross-modal Attention Fusion Network (CAF)

In terms of multi-modal feature fusion, the visual features and semantic features are extracted by different methods, meaning it is not suitable to concatenate

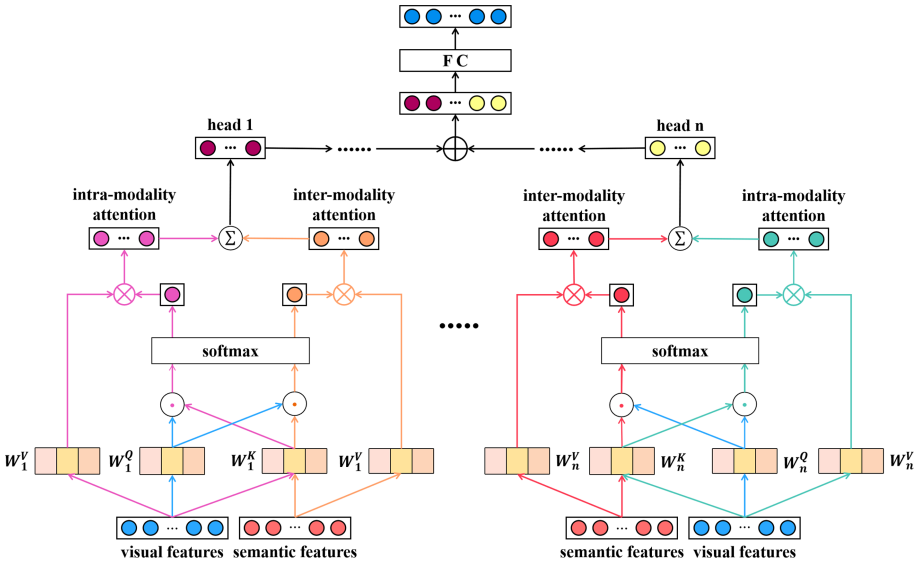


Fig. 3. The architecture of CAF.

them together directly. To this end, we devise a cross-modal attention fusion mechanism with intra-modality and inter-modality attentions to improve traditional fusion strategy. As shown in Fig. 3, each modality should not only pay attention to its own characteristics but also focus on other modal features. In particular, the multi-head mechanism allows the CAF to extract information from different feature spaces, which help the model explore different attention patterns in a variety of angles.

**1) Intra-Modality Attention.** Given the multimodal content features, we first produce a set of query, key and value pair by linear transformations for single modality. Taking the visual features  $V$  as an example, the operation can be obtained as follows:

$$V_Q = Linear(V, W^Q) \tag{10}$$

$$V_K = Linear(V, W^K) \tag{11}$$

$$V_V = Linear(V, W^V) \tag{12}$$

where ‘Linear’ denotes a fully connected layer,  $W^Q, W^K, W^V \in R^{d_m \times d_h}$  are the weight matrices and  $d_h$  represents the common dimension of the transformed features obtained from multiple modalities. Similarly, the corresponding linear transformations for the semantic features  $T$  can be represented as  $T_Q, T_K$  and  $T_V$ .

More specifically, we calculate the scaled dot product as the intra-modality attention weight. Given the  $V_Q$  and  $V_K$ , the operation can be obtained as follows:

$$V_{intra} = \frac{(V_Q \cdot V_K^T)}{\sqrt{d_h}} \tag{13}$$

where  $V_{intra}$  represents the intra-modality attention weight for  $V$ . Correspondingly, the semantic intra-modality attention weight  $T_{intra}$  can be calculated as follows:

$$T_{intra} = \frac{(T_Q \cdot T_K^T)}{\sqrt{d_h}} \tag{14}$$

**2) Inter-modality Attention.** As for inter-modality attention, to model the underlying relations among multiple modalities, we learn the inter-modality attention weight in a similar way.

$$V_{inter} = \frac{(V_Q \cdot T_K^T)}{\sqrt{d_h}} \tag{15}$$

where  $V_{inter}$  represents the inter-modality attention weight for  $V$ .

Furthermore, the softmax function is used to normalize the intra-modality and inter-modality attention weights. Then the visual resulting features  $V_C$  can be obtained by weighted summation over the different modalities.

$$V_C = softmax([V_{intra}, V_{inter}]) \begin{bmatrix} V_V \\ T_V \end{bmatrix} \tag{16}$$

In particular, the CAF calculates the intra-modality and the inter-modality attentions  $h$  times respectively and concatenates the multi-head features together. For clarity, we define that  $V_{C_i}$  is the attention outcome in the  $i^{th}$  head and  $W_i^Q, W_i^K, W_i^V$  are the weight matrices used in the corresponding linear transformations. In addition, we exploit a weight matrix to reduce the dimension for each modality. The operation can be obtained as follows:

$$F_v = W_o \cdot [V_{C_1} \oplus V_{C_2} \oplus \dots \oplus V_{C_h}] \quad (17)$$

where  $\oplus$  denotes the concatenation operation,  $W_o \in R^{h*d_n \times d_m}$  is the weight matrix and  $F_v$  represents the visual resulting features obtained from the cross-modal attention.

Relatively, the cross-modal attention outcome for the semantic features  $T$  can be achieved in a similar way, which is denoted as  $F_t$ .

$$F_t = W_o \cdot [T_{C_1} \oplus T_{C_2} \oplus \dots \oplus T_{C_h}] \quad (18)$$

Finally, we concatenate multimodal resulting features together and exploit a fully connected layer to calculate the final fused content features  $T_f \in R^{d_m}$  as follows:

$$T_f = \sigma(W_f \cdot (F_v \oplus F_t)) \quad (19)$$

where  $W_f$  is the weight matrix of the fully connected layer.

### 3.5 Rumor Detector

The goal of the rumor detector is to identify whether a social media data is a rumor or non-rumor. Given the fused content features  $T_f$  and the sequence features  $T_g$ , the rumor detector concatenates above features seamlessly and feeds the features into two fully connected layers to output the predicted result  $\tilde{y}$ . The operation of the detector can be represented as follows:

$$\tilde{y} = softmax(W_{r_2} \cdot \sigma(W_{r_1} \cdot (T_f \oplus T_g))) \quad (20)$$

where  $W_{r_1}, W_{r_2}$  are the weight matrices of the fully connected layers.

### 3.6 Loss Function

In terms of loss function used, we design an orthogonal constraint to make the latent memory keep its orthogonality and exploit a rumor detection loss function to identify rumors.

**1) Orthogonal Constraint.** The orthogonal constraint aims to minimize the pairwise cosine similarity between the patterns in the latent memory, which ensures the variety of the patterns to improve the discriminative power of the

memory. More specifically, given the latent memory network  $M$ , the proposed constraint can be represented below:

$$C_\beta(M) = \beta \|M^T M \odot (1 - I)\|_F^2 \quad (21)$$

where 1 denotes a matrix with all elements set to 1,  $\odot$  represents the element-wise product,  $I$  is the identity matrix and  $\beta$  is a hyperparameter.

**2) Rumor Detection Loss.** To identify rumors, we define a loss term  $\mathcal{L}$  by using cross entropy as follows:

$$\mathcal{L} = \sum_i^N -[y_i \times \log(\tilde{y}_i) + (1 - y_i) \times \log(1 - \tilde{y}_i)] \quad (22)$$

where  $\tilde{y}_i$  is the predicted result obtained from rumor detector for the  $i^{th}$  sample, and  $y_i$  is the corresponding ground-truth.  $N$  is the total number of social media samples.

Finally, the loss function of CALM can be written as follows:

$$\mathcal{L}_{CALM}(\theta, M) = \mathcal{L} + C_\beta(M) \quad (23)$$

where  $\theta$  is denoted as the parameter set of the proposed CALM.

The detailed steps of the proposed model CALM are summarized in Algorithm 1.

---

**Algorithm 1.** The CALM algorithm

---

**Input:** label  $y = \{y_i\}_{i=1}^N$ , textual input  $t = \{t_i\}_{i=1}^N$ , visual input  $v = \{v_i\}_{i=1}^N$ ,  $\beta$  for orthogonal constraint, the latent memory  $M$ , learning rate  $lr$ .

- 1: **Initialize the model parameters**
  - 2: Set the status of the model for training
  - 3: **for** number of training iterations **do**
  - 4:    $T, T_g = \text{TextualExtractor}(t, M)$
  - 5:    $V = \text{VisualExtractor}(v)$
  - 6:    $T_f = \text{CAF}(T, V)$
  - 7:    $\tilde{y} = \text{RumorDetector}(T_f \oplus T_g)$
  - 8:   Compute the loss using the loss function  $\mathcal{L}_{CALM}(\theta, M)$  with  $\tilde{y}$  and  $y$
  - 9:   Decay learning rate  $lr$  according to the number of the training iterations
  - 10: **end for**
- 

## 4 Experiments

### 4.1 Datasets

1) **Twitter Dataset** [3], is comprised of 514 images and 18,264 Tweets. We filter out the Tweets with noise and unclear labels, resulting in 379 images and 15,629 Tweets being related 9,405 rumors and 6,224 non-rumors.

**2) Weibo Dataset** [6], consists of 9,527 posts being related 4,748 rumors and 4,779 non-rumors. We split the dataset into training, validation, and testing sets in a ratio of 7:1:2.

## 4.2 Baselines

We compare the proposed methods with the following baselines:

- 1) **VQA** [2], aims to answer the questions about the given images. We improve the original VQA model to adapt to the rumor detection.
- 2) **NeuralTalk** [13], averages the outputs of RNN at each time step to obtain the latent representations and generates corresponding description for the given images.
- 3) **att-RNN** [6], uses the attention mechanism to fuse the visual, textual and social context features for rumor detection.
- 4) **EANN** [14], designs three components for multimodal rumor detection, including multimodal feature extractor, fake news detector and event discriminator.
- 5) **MVAE** [7], devises a multi-modal VAE structure to obtain shared representation and employs a binary classifier to detect rumors.
- 6) **MFN** [5], exploits a self-attentive mechanism to integrate multi-modal information and introduces a latent topic network to detect upcoming rumors.
- 7) **BDANN** [18], employs a BERT-based approach to extract multi-modal features and proposes a domain classifier to remove the event-specific dependency. As the domain classifier requires event labels, for a fair comparison, we remove the domain classifier in BDANN.

In terms of evaluations, accuracy, precision, recall, and  $F_1$ -score are adopted.

## 4.3 Performance of the Approaches

Table 1 summarizes the performance of the approaches on two datasets, from which we have some observations. 1) The multi-modal rumor detection models, e.g., att-RNN, EANN and CALM, outperform the multimodal fusion methods for rumor detection, such as VQA and NeuralTalk. The reason may be that the rumor detection models make full use of information about rumor and non-rumor events, e.g., global latent rumor patterns, event information, etc. 2) In terms of the rumor detection approaches, CALM significantly outperforms the baselines, benefiting from the cross-modal attention fusion mechanism to integrate multi-modal information and the orthogonal latent memory to capture robust representations.

**Table 1.** Performance of the approaches on two datasets

Dataset	Method	Accuracy	Rumors			Non-Rumors		
			Precision	Recall	$F_1$	Precision	Recall	$F_1$
Twitter	VQA	0.753	0.719	0.601	0.655	0.769	0.849	0.807
	NeuralTalk	0.667	0.570	0.593	0.582	0.733	0.714	0.723
	att-RNN	0.756	0.724	0.604	0.658	0.771	0.853	0.810
	EANN	0.757	0.728	0.601	0.658	0.770	0.856	0.811
	MVAE	0.805	0.869	0.588	0.702	0.782	<b>0.943</b>	0.855
	MFN	0.808	0.850	0.616	0.715	0.791	0.931	0.855
	BDANN	0.827	<b>0.872</b>	0.652	0.746	0.808	0.939	0.869
	CALM	<b>0.845</b>	0.785	<b>0.831</b>	<b>0.807</b>	<b>0.888</b>	0.855	<b>0.871</b>
Weibo	VQA	0.736	0.797	0.634	0.706	0.695	0.838	0.760
	NeuralTalk	0.726	0.794	0.613	0.692	0.684	0.840	0.754
	att-RNN	0.788	<b>0.862</b>	0.686	0.764	0.738	<b>0.890</b>	0.807
	EANN	0.816	0.820	0.820	0.820	0.810	0.810	0.810
	MVAE	0.824	0.854	0.769	0.809	0.802	0.875	0.837
	MFN	0.803	0.811	0.806	0.808	0.794	0.800	0.797
	BDANN	0.814	0.800	0.860	0.830	0.840	0.760	0.800
	CALM	<b>0.846</b>	0.843	<b>0.864</b>	<b>0.853</b>	<b>0.851</b>	0.828	<b>0.839</b>

#### 4.4 Ablation Study

**Table 2.** Performance of the variations of CALM

Dataset	Method	Accuracy	Precision	Recall	$F_1$
Twitter	CALM_CA	0.812	0.807	0.793	0.798
	CALM_LM	0.824	0.815	0.813	0.814
	CALM_OC	0.827	0.824	0.840	0.824
	CALM	<b>0.845</b>	<b>0.836</b>	<b>0.843</b>	<b>0.839</b>
Weibo	CALM_CA	0.821	0.821	0.822	0.821
	CALM_LM	0.824	0.832	0.826	0.823
	CALM_OC	0.831	0.832	0.832	0.831
	CALM	<b>0.846</b>	<b>0.847</b>	<b>0.846</b>	<b>0.846</b>

CALM consists of a cross-modal attention fusion (CAF) mechanism to combine multimodal content features and an orthogonal latent memory network to keep diverse latent patterns. For clarity, let CALM\_CA denote CALM without CAF module and CALM\_LM denote CALM without latent memory network. Furthermore, we remove orthogonal constraint to evaluate the effectiveness of preserving orthogonality among latent patterns, which is denoted as CALM\_OC. The performance of the variations of CALM are summarized in Table 2, from which we have the following observations. 1) CALM with all the components achieves the best performance on both datasets, demonstrating the significance of each

module. 2) We can observe that the performance of CALM drops dramatically without the CAF module. The reason is that the CAF module extracts critical information underlying the single modalities by intra-modality attention, and models the underlying relations among the modalities by inter-modality attention.

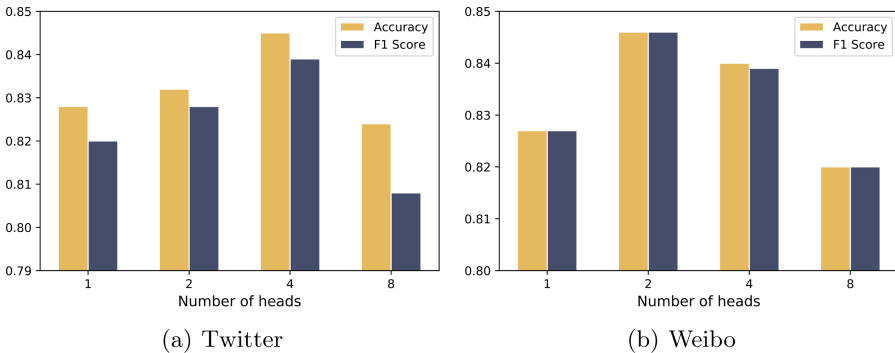
#### 4.5 Effectiveness of CALM on Multimodal Fusion

**Table 3.** Performance of CALM using single or multiple modalities

Dataset	Method	Accuracy	Precision	Recall	$F_1$
Twitter	CALM_T	0.786	0.804	0.815	0.786
	CALM_V	0.775	0.781	0.725	0.737
	CALM	<b>0.845</b>	<b>0.836</b>	<b>0.843</b>	<b>0.839</b>
Weibo	CALM_T	0.816	0.819	0.818	0.816
	CALM_V	0.580	0.586	0.573	0.560
	CALM	<b>0.846</b>	<b>0.847</b>	<b>0.846</b>	<b>0.846</b>

Table 3 summarizes the performance of CALM using single or multiple modalities on both datasets, from which we have two-fold observations. 1) CALM uses text and image jointly outperforms it uses either text (CALM\_T) or image (CALM\_V) merely, indicating the necessity of multimodal fusion. 2) In terms of the single data modality, text is more effective than images as text conveys certain semantic information that is easy to understand by humans or machines.

#### 4.6 Impact of the Number of Heads in CAF



**Fig. 4.** Impact of the number of heads in CAF.

Figure 4 summarizes the performance of CALM with different number of heads in the CAF module. We can find that maintaining a large number of heads may not necessarily improve the performance on both datasets. The reason could be that a large number of heads will increase the complexity of the model, while they cannot capture more attentional patterns than the certain number underlying the datasets.

#### 4.7 Impact of the Number of Patterns in Latent Memory

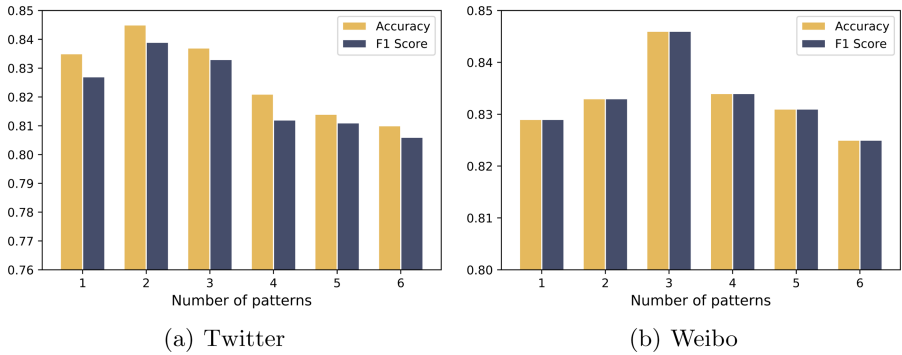


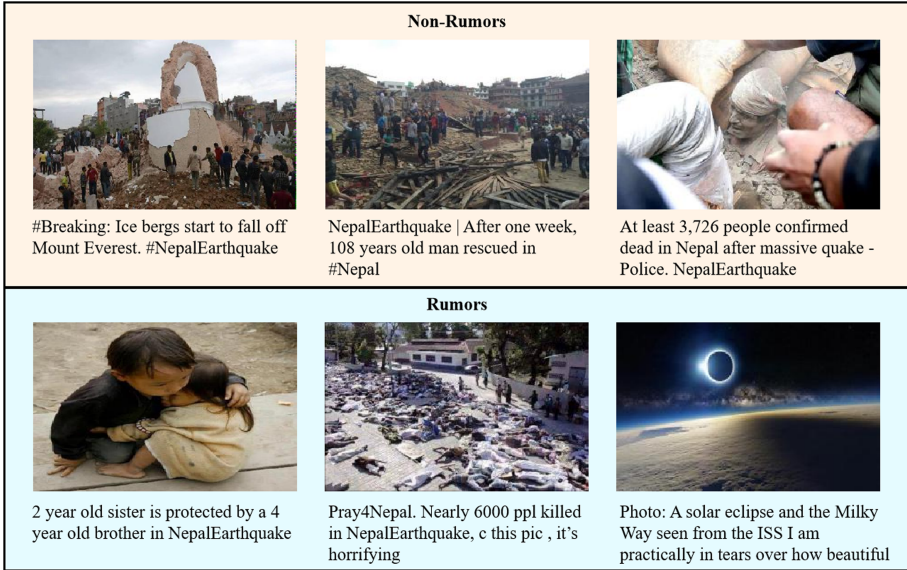
Fig. 5. Impact of the number of patterns in latent memory.

Figure 5 summarizes the performance of CALM with different number of patterns in the latent memory. We can observe that the performance of CALM can be improved with the increasing number of patterns at the beginning, while too many patterns will lead to poor result on both datasets. This is probably because that the number of latent patterns underlying the datasets is limited.

#### 4.8 Failure Cases Study

Figure 6 shows some examples that are predicted falsely by CALM, from which we have some observations. 1) In terms of non-rumors predicted as rumors by CALM, we can observe that the images have not shown discriminative information and the textual descriptions seem to exaggerate the facts more or less. 2) In terms of rumors predicted as non-rumors by CALM, we can observe that the textual and visual contents are quite consistent and relevant, which may confuse the model and it is even hard for humans to make identification.





**Fig. 6.** Failure cases of CALM. The Tweets in orange background shows non-rumors which are predicted as rumors, and the Tweets in blue background shows rumors that are recognized as non-rumors. (Color figure online)

## 5 Conclusion

In this paper, we propose a cross-modal attention fusion network with orthogonal latent memory for rumor detection. Specifically, we exploit a cross-modal attention mechanism with intra-modality and inter-modality attentions to integrate the modality-critical information and fully explore potential hidden correlations among the modalities. In particular, the proposed network introduces an orthogonal latent memory to store global latent patterns information shared by the rumor events, which can improve sequential models to capture the long-distance temporal dependencies. The experiments conducted on two popular datasets show the effectiveness of the proposed CALM for rumor detection.

**Acknowledgement.** This work is supported by the National Natural Science Foundation of China (No. 62076073), the Guangdong Basic and Applied Basic Research Foundation (No. 2020A1515010616), Science and Technology Program of Guangzhou (No. 202102020524), the Guangdong Innovative Research Team Program (No.2014ZT05G157), HKIBS Research Program Grant Application (HCRG-201-002) and the Faculty Research Grant (DB21B6) of Lingnan University, Hong Kong.

## References

1. Allcott, H., Gentzkow, M.: Social media and fake news in the 2016 election. *J. Econ. Perspect.* **31**(2), 211–36 (2017)

2. Antol, S., et al.: Vqa: Visual question answering. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2425–2433 (2015)
3. Boididou, C., et al.: Verifying multimedia use at mediaeval 2015. *Media Eval.* **3**(3), 7 (2015)
4. Castillo, C., Mendoza, M., Poblete, B.: Information credibility on twitter. In: Proceedings of the 20th International Conference on World Wide Web (2011)
5. Chen, J., Wu, Z., Yang, Z., Xie, H., Wang, F.L., Liu, W.: Multimodal fusion network with latent topic memory for rumor detection. In: 2021 IEEE International Conference on Multimedia and Expo (ICME), pp. 1–6. IEEE (2021)
6. Jin, Z., Cao, J., Guo, H., Zhang, Y., Luo, J.: Multimodal fusion with recurrent neural networks for rumor detection on microblogs. In: Proceedings of the 25th ACM International Conference on Multimedia (2017)
7. Khattar, D., Goud, J.S., Gupta, M., Varma, V.: Mvae: Multimodal variational autoencoder for fake news detection. In: The World Wide Web Conference (2019). <https://doi.org/10.1145/3308558.3313552>
8. Kim, Y., Lee, H., Provost, E.M.: Deep learning for robust feature generation in audiovisual emotion recognition. In: IEEE International Conference on Acoustics (2013)
9. Kim, Y.: Convolutional neural networks for sentence classification (2014)
10. Owens, A., Efron, A.A.: Audio-visual scene analysis with self-supervised multisensory features. In: Proceedings of the European Conference on Computer Vision (ECCV) (2018)
11. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2015)
12. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems (2017)
13. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell: a neural image caption generator. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3156–3164 (2015)
14. Wang, Y., et al.: Eann: event adversarial neural networks for multi-modal fake news detection. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery Data Mining, pp. 849–857. KDD '18, Association for Computing Machinery, New York, NY, USA (2018). <https://doi.org/10.1145/3219819.3219903>
15. Yang, Y., Zheng, L., Zhang, J., Cui, Q., Li, Z., Yu, P.S.: Ti-cnn: Convolutional neural networks for fake news detection (2018)
16. Yu, F., Liu, Q., Wu, S., Wang, L., Tan, T.: A convolutional approach for misinformation identification. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17, pp. 3901–3907 (2017). <https://doi.org/10.24963/ijcai.2017/545>
17. Zhang, H., Fang, Q., Qian, S., Xu, C.: Multi-modal knowledge-aware event memory network for social media rumor detection. In: Proceedings of the 27th ACM International Conference on Multimedia (2019). <https://doi.org/10.1145/3343031.3350850>
18. Zhang, T., et al.: Bdann: bert-based domain adaptation neural network for multi-modal fake news detection. In: 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1–8 (2020). <https://doi.org/10.1109/IJCNN48605.2020.9206973>
19. Zhou, X., Wu, J., Zafarani, R.: Safe: similarity-aware multi-modal fake news detection. *Adv. Knowl. Discovery Data Mining* **12085**, 354 (2020)



# OMT: An Operate-Based Approach for Modelling Multi-topic Influence Diffusion in Online Social Networks

Chenting Jiang<sup>1</sup>, Weihua Li<sup>2</sup>(✉), Shiqing Wu<sup>1</sup>, and Quan Bai<sup>1</sup>

<sup>1</sup> University of Tasmania, Hobart, TAS 7005, Australia  
{cjiang5, shiqing.wu, quan.bai}@utas.edu.au

<sup>2</sup> Auckland University of Technology, Auckland 1010, New Zealand  
weihua.li@aut.ac.nz

**Abstract.** Influence diffusion modelling in online social networks has been widely studied and applied in public opinion management, viral marketing, and rumour detection. Most existing studies focus on the network topology and the complex user characteristics while ignoring the diverse topic features of the information, especially the cross-impact of multiple topics on the information propagation. In this paper, we propose the *Operator-based Multi-Topic (OMT)* model by considering user topic interest, topic penetration, and topic correlation to explain the topic effects on influence diffusion fully. Meanwhile, the operator-based approach inherits the advantages of the heat diffusion-based model and the agent-based model. Accordingly, the OMT is recognized as a user context-aware and topic-aware prediction model, which can improve the practicability, quality, and simulation of influence diffusion modelling in multi-topic social networks. In the experiments, real-world datasets are adopted to evaluate the performance of the proposed OMT. The experimental results demonstrate that the OMT performs effectively in diffusion simulation and influence maximization.

**Keywords:** Influence diffusion · Multiple topics · Online social networks · Influence maximization

## 1 Introduction

Online information dissemination by posting, retweeting or commenting on the messages on social media is an important channel for marketing and e-commerce to influence people's opinion toward a particular topic, product, or brand [5]. Modelling the influence diffusion aims to predict the realistic spreading process of target information over time and its maximized influence result [9]. Most existing studies explore the network topology and the influence probability with diversified user characteristics to establish the models [10]. As a piece of information mainly highlights one topic [25], one of the critical challenges in influence diffusion modelling is to explore complex topic factors in the multi-topic network

environment. The topic factors are beneficial to describe the real and complex circumstances of influence propagation and improve their models' quality and practicability in the real world [24].

Specifically, the topic interest [13] or topic preference [6] of the users is vital to express the topic-to-user effect in influence propagation because an interesting topic is more likely to be transmitted and influence users [25]. Meanwhile, the current propagation status of a topic, referring to how much influence a topic penetrates in the entire network, is dynamically changing and constantly impacts its relevant topics' influence diffusion [20]. For example, users prefer to retweet a hot topic with a high propagation status on the social platform, influencing their followers potentially. Furthermore, the topic correlation [12] describes the cross-topic effects of influence diffusion [18] and reflects the complexity of multi-topic influence diffusion by the topic-to-topic effect. For instance, two topics with a strong correlation lead to the indirect influence diffusion between each other by message transmissions easily. Accordingly, a comprehensive analysis of the above topic factors is the critical objective to explain the multi-topic features of influence dissemination, which is also a deficiency of state-of-the-art models.

In this paper, we propose a novel influence diffusion model, i.e., the Operator-based Multi-Topic (OMT) model, based on the OBM [7]. As the OBM is beneficial for modelling the single-topic influence diffusion in a complex and reversible interactive network by combining the diffusion dynamics (i.e., the operator driven the heat differential equation) of the Heat Diffusion (HD) model and user contextual factors (i.e., user preference and capacity) of the Agent-Based Model (ABM), the OMT model can inherit the OBM advantages to exhibit the property of user context-awareness. User context, seen as a user's memory of previously exchanged messages, reflects the complexity of user characteristics. Meanwhile, the OMT model focuses on exploring three topic factors (i.e., user topic interest, topic penetration, and topic correlation) in multi-topic networks to extend the operator of the OBM further. The multi-topic analysis fully demonstrates the topic effects from three aspects of influence propagation, reveals the complexity of information features, widens the model's application range and confers its topic-aware property to improve its quality and performance. Accordingly, as the driver of the diffusion equation, the new operator of the OMT contains both user context and multi-topic features. The OMT is recognized as a user context-aware and topic-aware comprehensive model in multi-topic online interactive networks. Evaluated by the simulation experiment and applied to the influence maximization problem, the OMT model has been verified an excellent prediction performance on the spreading process and influence maximized result.

The major contributions of this paper are summarized as follows:

- We consider the user topic interest, current topic propagation status, and the topic inter-connectivity in an influence diffusion model, comprehensively revealing the significance of topic effects in a multi-topic network.
- We propose an Operator-based Multi-Topic diffusion (OMT) model with user-context-awareness and topic-awareness to enhance the model's practicability, quality, and simulation.

- We develop a novel seed selection algorithm based on the OMT, called the MGTS-greedy (Multi-topic Global Topical Support) algorithm, on the influence maximization problem to achieve optimal influence of the target topic by finding a multiple topic seed set.

## 2 Related Work

For topic-based influence diffusion modelling, the topic factor presents the unique perspective to interpret the topic effect on different aspects of influence diffusion. Barbieri et al. [1] primarily develop the topic-aware extensions of the IC and LT models, i.e., TIC and TLT. The propagation probability of these two extended models focuses on user authoritativeness and interests in a topic. Du et al. [4] propose a topic cascade model by capturing the topic distribution to modulating the transmission likelihood. This topic factor contains the topic preference of the transmission channels and the published topic. Yu et al. [26] extend the topic factor of the topic cascade model by including the topic influence matrix of the senders and the topic receptivity matrix of the receivers. Pinto and Chahed [21] explore the effect of multiple topics' interactions in information diffusion modelling by proposing a coupling of the Latent Dirichlet Allocation (LDA) topic models [2] and Hawkes processes [14]. Wu et al. [24] propose a Topic-Based Information Diffusion (TBID) model by considering social trends and topic semantics. Mahdizadehghadam et al. [16] propose a layered and interconnected network to explore the intrinsic interactions in one layer and the topic connectivity between the different layers. The inter-layer connectivity matrix is defined to describe the relationship between topics. Pan et al. [20] develop the Dynamic Influence Propagation (DIP) model by allowing the current propagation status of topics to change the propagation rates. Nolasco and Oliveira [18] develop a system to assess the topic relationship using the Kullback-Leibler (KL) divergence in the influence diffusion process. Wang et al. [23] develop the temporal topic influence (TTI) model by being aware of time, content and network structure evolution. The topic factors mainly reflect on the individual topic distributions and their similarities.

However, most existing works ignore the comprehensive analysis of the topic effects on users, message exchanges, and other topics during diffusion. Therefore, we grasp this opportunity to remedying the limitation by a synthetic analysis of topic effects on the influence diffusion process.

The topic-based models, cooperating with the seed selection algorithm, are applied to address the influence maximization problem. Chen et al. [3] design a faster topic materialization-based algorithm to address the topic-aware influence maximization problem by regarding topic preference. Singh et al. [22] propose a Community based Context-aware Influence Maximization (C2IM) algorithm by considering user interest in topics to improve the effectiveness of the selected seed. Finally, Min et al. [17] develop a topic based time-sensitive dynamic propagation model and heuristic algorithm by considering user topic preference and interaction delay.

However, the above seed selection methods cannot consider the interrelationships, cross-impacts of topics, and the current topic propagation status on influence maximization. So, we develop a seed selection algorithm by focusing on the topic correlation in the multi-topic environment, supporting the proposed diffusion model in influence maximization application.

### 3 The Operator-Based Multiple-Topic (OMT) Modelling

#### 3.1 Multi-Topic Influence Framework

In the multi-topic network environment, the influence diffusion directly occurs in the same topic communication between users and their neighbours and indirectly exists in their interactions about relevant topics. In terms of the direct influence transmission, when a pair of users have interactions, they will exchange the messages of a specific topic from the sender to the recipient, which leads to the transmission of the topic influence. On the other hand, the indirect influence diffusion stems from other relevant topic information exchanges. This means the message exchanges of a particular topic lead to propagating other topic influences between users. This indirect influence diffusion is based on the hybrid effect of the correlation of two topics related to the topic penetration and user topic interest in a multi-topic environment. Thus, our multi-topic influence framework demonstrates a strong capacity for interpreting both direct and indirect influence diffusion phenomenons in the multi-topic, complex, actual online social networks.

#### 3.2 Formal Definitions

**Definition 1.** A User  $v_i \in V$  is defined as a vertex in a directed and weighted social network as a graph  $G = (V, E)$ , where a set of users display as  $V = \{v_1, v_2, \dots, v_n\}$  and a set of edges is represented as  $E = \{e_{ij} | 1 \leq i, j \leq n\}$ . Edge  $e_{ij} = (v_i, v_j) \in E$  denotes the direct influence relationship between  $v_i$  and  $v_j$ , where  $e_{ij} \neq e_{ji}$ . The weight of the influence relationship  $e_{ij}$  expresses the trust degree of the user  $v_j$  to the user  $v_i$ , which will be explained in Definition 4.

User  $v_i$  has a set of sender-neighbors  $\mathcal{N}_i = \{v_j | v_j \in V, v_j \neq v_i, e_{ji} \in E\}$ . Similarly, the set of receiver-neighbours is defined as  $\bar{\mathcal{N}}_i = \{v_j | v_j \in V, v_j \neq v_i, e_{ij} \in E\}$ .

At time  $t$  of the influence diffusion, each user  $v_i$  has a set of messages  $I_i(t)$  and each message belongs to one topic  $x$ . Thus there are several subsets of messages  $I_i^x(t) \subseteq I_i(t)$ . The number of message about topic  $x$  is presented as  $m_i^x$ .

Moreover, each message transmission of a user occurs within a particular user context. User context is a comprehensive concept to express the interactions' situation surrounding, which consists of the previously exchanged messages of a topic, user preference and user capacity.

**Definition 2.** User Preference represents the user emotional attitude toward a topic on messages.  $m_i^{x+}$  denotes the number of messages in the context of the user  $v_i$  in favor of the topic  $x$ . On the contrary,  $m_i^{x-}$  represents the quantity of neutral or negative message of topic  $x$  in the context of the user  $v_i$ .

**Definition 3. User Capacity**  $c_i$  denotes the ability of holding messages in the context of user  $v_i$ . It is a constant, defined as  $c_i \in \mathbb{R}$ . We assume  $c_i = \sum_{x \in Q_i} c_i^x = \sum_{x \in Q_i} m_i^{x+} + \sum_{x \in Q_i} m_i^{x-}$ , where  $c_i^x$  represents the capacity of user  $v_i$  on topic  $x$ , and  $Q_i$  denotes a set of topics involved by user  $v_i$ .

**Definition 4. User Trust**  $w_{ij}$  denotes the trust relationship between  $v_i$  and  $v_j$ . For each pair of users  $(v_i, v_j)$ , a trust value  $w_{ji} \in [0, 1]$  quantifies the level of which user  $v_i$  trusts user  $v_j$ . If  $w_{ji} = 1$ , user  $v_i$  trusts all the messages posted by  $v_j$ . If  $w_{ji} = 0$ , the activity of user  $v_j$  has no direct impact on user  $v_i$ . Consequently,  $w_{ij} = 0$  if and only if  $e_{ij} \notin E$ .

We assume that the user trust of a specific topic is related to the topic interest, i.e.,  $w_{ij}^x = \alpha_j^x w_{ij}$ . Furthermore, user  $v_i$  is assumed to interact with its neighbours according to a Poisson process with rate  $R_i$ . Each neighbour is equally likely to participate in an interaction with the user  $v_i$ .

**Definition 5. A Topic**  $x \in Q$  is defined as a label of messages with the same major content, and  $Q$  denotes the set of topics in the multi-topic networks. The message transmissions within the same topic construct one layer subnetwork of influence diffusion  $G^x = (V^x, E^x)$ .

We suppose that each message’s topic has been identified in this model, which aims to simplify the topic extraction and topic labeling.

**Definition 6. User Topic Interest**  $\alpha_i^x \in [0, 1]$  refers to inherent personal likeability on a topic. If  $\alpha_i^x = 0$ , user  $v_i$  does not have any interest in topic  $x$ , which means  $v_i$  cannot be influenced by this topic. On the contrary, if  $\alpha_i^x = 1$ , user  $v_i$  is only interested in topic  $x$ , which means  $v_i$  can not be influenced by other topics besides topic  $x$ . A user usually shows interest in several topics and participates in these topics’ influence diffusion. Correspondingly, the sum of these topics’ interest equals 1, i.e.,  $\sum_{x \in Q_i} \alpha_i^x = 1$ .

**Definition 7. Topic Penetration**  $\beta^x \in [0, 1]$  denotes the topic dynamic weight in the networks to express the current diffusion state of a topic  $x$ . As a time-varying parameter,  $\beta^x$  is calculated by the real-time propagation percentage of the topic  $x$  in the entire networks.

Topic penetration reflect the topic effect on user interactions and message transmissions in the dynamic topic influence diffusion network. It has the equivalent impact on the interactions with the same topic at the same time.

**Definition 8. Topic Correlation**  $\theta^{xy} \in [0, 1)$ , a non-directional parameter, is defined as a probability of the topic-to-topic influence to reveal cross-topic effects of influence dissemination. If  $\theta^{xy} = 0$ , topic  $x$  and  $y$  do not have any relation, and they cannot impact each other. Whereas if  $\theta^{xy}$  infinitely closes to 1, two topics display a very strong correlation in indirect influence diffusion. Normally, the value of  $\theta^{xy}$  more, the topic correlation between  $x$  and  $y$  stronger. We assume that  $\theta^{xy}$  is determined by  $D(x||y)$  and  $D(y||x)$ :

$$\theta^{xy} = \frac{1}{D(x||y) + D(y||x)}, \tag{1}$$

where  $D(x||y)$  denotes how much affect of the relevant topic  $y$  on the target topic  $x$ , calculated by the Kullback-Leibler divergence (i.e., KL-divergence, also called relative entropy) [18]. In the indirect influence dissemination,  $D(x||y)$  expresses the message transmission of topic  $y$  results in the probability of the influence diffusion of topic  $x$  in the entire network state.

$$D(x||y) = \sum_{m \in I} x(m) \log \frac{x(m)}{y(m)}, \tag{2}$$

where  $x(m)$  denotes the probability distribution of the number of topic  $x$  messages. Due to  $D(x||y) \neq D(y||x)$ , a relative entropy  $D(x||y) \in [0, +\infty)$  reveals the one-way correlation of two topic distribution.  $D(x||y) = 0$  indicates the highest correlation from topic  $y$  to  $x$ . When the value of  $D(x||y)$  increases, the correlation of a pair of topics decreases.

Therefore, topic correlation is vital to express the complexity of multi-topic influence propagation, especially in the indirectly spreading a topic influence. It also quantifies the relationships between topics effectively in the entire network.

### 3.3 Derivation

In multi-topic social networks, message transmission with a particular opinion of a topic is the serious precondition for influence propagation. The probability of message transmission of a topic determines its influence spreading possibility and partly impacts the possibility on relevant topics' influence diffusion. Specifically, it depends on users trust relationship and context under a specific topic. Therefore, the positive message transmission probability of topic  $x$  from user  $v_j$  to user  $v_i$  is formulated as:

$$P_{ji}^{x+} = w_{ji}^x \frac{m_i^{x+} + m_j^{x+}}{c_i^x + c_j^x} = \alpha_i^x w_{ji} \frac{m_i^{x+} + m_j^{x+}}{c_i^x + c_j^x}. \tag{3}$$

Similarly, the negative message transmission probability is:

$$P_{ji}^{x-} = w_{ji}^x \frac{m_i^{x-} + m_j^{x-}}{c_i^x + c_j^x} = \alpha_i^x w_{ji} \frac{m_i^{x-} + m_j^{x-}}{c_i^x + c_j^x} = \alpha_i^x w_{ji} - P_{ji}^{x+}. \tag{4}$$

Equations 3 and 4 should satisfy the following criteria:

1. If  $m_i^{x+} = m_j^{x+} = 0$ , then  $P_{ji}^{x+} = 0$ .
2. If  $m_i^{x-} = m_j^{x-} = 0$ , then  $P_{ji}^{x-} = 0$ .
3. If  $\alpha_i^x = 0$ , then  $P_{ji}^{x+} = P_{ji}^{x-} = 0$ .
4. If  $m_i^{x+}$  increases, then  $P_{ji}^{x+}$  increases, and  $P_{ji}^{x-}$  decreases.
5. If  $m_j^{x+}$  increases, then  $P_{ji}^{x+}$  increases, and  $P_{ji}^{x-}$  decreases.



Similar to Eqs. 3 and 4, the message transmission probability of all relevant topics can be gained. After message transmissions of multiple relevant topics, user  $v_i$  has a probability of being influenced by user  $v_j$  on a targeted topic. The target-topic influence probability depends on the relevant message transmission probability of this topic from every neighbor, the receiving capacity of user  $v_i$  on relevant topics, and the topic correlation between the target topic and others. Therefore, the topic positive influence probability for a user denotes user  $v_j$  persuades user  $v_i$  to adopt one message in favor of topic  $x$  after an interaction, given by:

$$\mathbb{P} \{ \Delta_t m_i^{x+} = 1 \} = \sum_{j \in \mathcal{N}_i} \sum_{y \in \mathcal{Q}_i} \frac{1}{d_i} P_{ji}^{y+} \theta^{xy} \frac{m_i^{y-}}{c_i^y}. \tag{5}$$

Similarly, the topic negative influence probability for a user denotes user  $v_j$  persuades user  $v_i$  to repudiate one positive message after an interaction, given by:

$$\mathbb{P} \{ \Delta_t m_i^{x+} = -1 \} = \sum_{j \in \mathcal{N}_i} \sum_{y \in \mathcal{Q}_i} \frac{1}{d_i} P_{ji}^{y-} \theta^{xy} \frac{m_i^{y+}}{c_i^y}. \tag{6}$$

The probability that user  $v_j$  does not persuade user  $v_i$  to adopt or repudiate one positive message is given by:

$$\mathbb{P} \{ \Delta_t m_i^{x+} = 0 \} = \sum_{j \in \mathcal{N}_i} \sum_{y \in \mathcal{Q}_i} \frac{1}{d_i} (1 - P_{ji}^{y+}) \theta^{xy} \frac{m_i^{y-}}{c_i^y} + \frac{1}{d_i} (1 - P_{ji}^{y-}) \theta^{xy} \frac{m_i^{y+}}{c_i^y}, \tag{7}$$

where  $d_i$  represents the number of users in  $\mathcal{N}_i$ . Equations 5, 6, and 7 imply the relationship of these three influence probabilities as:  $\mathbb{P} \{ \Delta_t m_i^{x+} = 1 \} + \mathbb{P} \{ \Delta_t m_i^{x+} = -1 \} + \mathbb{P} \{ \Delta_t m_i^{x+} = 0 \} = 1$ . Accordingly, the change of target topic influence status of a user over time is implied:

$$\mathbb{E} [ \Delta_t m_i^{x+} ] = \frac{1}{d_i} \sum_{j \in \mathcal{N}_i} \sum_{y \in \mathcal{Q}_i} \frac{\theta^{xy}}{c_i^y} (P_{ji}^{y+} m_i^{y-} - P_{ji}^{y-} m_i^{y+}). \tag{8}$$

In the asymptotic diffusion process, the evolution of network influence state on topic  $x$  is described as a derived function, as follow:

$$\begin{aligned} \partial_t \mathbb{E} [ m_i^{x+} ] &= \lim_{h \rightarrow 0} \frac{1}{h} ( \mathbb{E} [ m_i^{x+}(t+h) ] - \mathbb{E} [ m_i^{x+}(t) ] ) \\ &= \lim_{h \rightarrow 0} \frac{1}{h} \mathbb{E} [ m_i^{x+}(t+h) - m_i^{x+}(t) ] \\ &= \lim_{h \rightarrow 0} \frac{1}{h} \mathbb{E} [ \Delta_t m_i^{x+} R_i^x h ] \\ &= R_i \beta^x \mathbb{E} [ \Delta_t m_i^{x+} ]. \end{aligned} \tag{9}$$

Equation 9 expresses user  $v_i$ 's expected states during the influence diffusion process. According to the superposition principle, the network state's expected target-topic influence is  $m^x$ . Moreover, we will replace  $m_i^{x+}$  with  $m^x$ , since

$m_i^{x-} = c_i^x - m_i^x$ . Combining Eq. 8 and Eq. 9, the propagation of the network influence state is:

$$\partial_t m_i^x = \frac{\beta^x R_i}{d_i} \sum_{j \in \mathcal{N}_i} \sum_{y \in \mathcal{Q}_i} \frac{\theta^{xy}}{c_i^y} [P_{ji}^{y+} c_i^y - (P_{ji}^{y+} + P_{ji}^{y-}) m_i^y]. \quad (10)$$

Then, we substitute the message transmission probability (i.e., Eqs. 3 and 4) into Eq. 10.

$$\begin{aligned} \partial_t m_i^x &= \frac{\beta^x R_i}{d_i} \sum_{j \in \mathcal{N}_i} \sum_{y \in \mathcal{Q}_i} \frac{\theta^{xy}}{c_i^y} \left( \alpha_i^y w_{ji} \frac{m_i^y + m_j^y}{c_i^y + c_j^y} c_i^y - \alpha_i^y w_{ji} m_i^y \right) \\ &= \frac{\beta^x R_i w_{ji}}{d_i} \sum_{j \in \mathcal{N}_i} \sum_{y \in \mathcal{Q}_i} \frac{\alpha_i^y \theta^{xy}}{c_i^y} \left( \frac{c_i^y}{c_i^y + c_j^y} m_j^y - \frac{c_j^y}{c_i^y + c_j^y} m_i^y \right). \end{aligned} \quad (11)$$

We can combine the interactions in Eq. 11 into a matrix, giving the differential equation:

$$\partial_t f = H f(t) [\partial_t m = H m], \quad (12)$$

In Eq. 12,  $H$  is defined by:

$$H_{ji}^x = \begin{cases} \frac{\beta^x R_i w_{ji}}{d_i} \sum_{y \in \mathcal{Q}_i} \frac{\alpha_i^y \theta^{xy}}{c_i^y + c_j^y} & i \neq j, \\ -\frac{\beta^x R_i}{d_i} \sum_{k \in \mathcal{N}_i} \sum_{y \in \mathcal{Q}_i} w_{ki} \frac{\alpha_i^y c_k^y \theta^{xy}}{c_i^y (c_i^y + c_k^y)} & i = j. \end{cases} \quad (13)$$

### 3.4 Network Diffusion Process

According to Eq. 12, the operator  $H_{OMT}^x$  determines the influence spreading process of the network over time. Recalling the solution of the differential equation, it approximately provides a closed form solution to describe an asymptotic network diffusion process, as follow:

$$f(t) = e^{\Delta t H_{OMT}} f(0), \quad (14)$$

where  $e^{\Delta t H_{OMT}}$  is the matrix exponential driven by the synthetic operator  $H_{OMT}$  (Eq. 13) over time. To reduce the computational load of this matrix function, we leverage the matrix-vector multiplication to approximate the network iterative process quickly:

$$e^{\Delta t H_{OMT}} \approx \sum_{m=0}^M \frac{\Delta t^m H_{OMT}^m}{m!} \approx \left( I + \frac{\Delta t H_{OMT}}{M} \right)^M. \quad (15)$$

Combining Eqs. 13, 14, and 15, the network diffusion process has been expressed by heat diffusion mechanism, user contextual characteristics, and multi-topic effects.

$$f(t) = \left( I + \frac{\Delta t H_{OMT}}{M} \right)^M f(0). \quad (16)$$

In the multi-topic environment, the operator  $H_{OMT}$  contains all relevant topics' correlations of the target topic  $x$ . Meanwhile, we can split each relevant topics influence operator of the target topic from the multi-topic operator to exploring different topics influences on the target topic or return to the single topic influence diffusion in the OBM.

## 4 The Influence Maximization in Multiple-Topic Social Networks

In multi-topic social networks, influence maximization focuses on achieving the maximum influence on the target topic by selecting a limited set of beginners [8, 11]. The application of the OMT model aims to support marketers realizing the maximized positive influence of a specific brand, product, or public topic in real-world complex social networks. With this objective, a novel seed selection approach, named the MGTS-greedy (Multi-topic Global Topical Support) algorithm, is proposed. It utilizes the GTS [7] as the indicator to measure the positive influence propagation of the target in a global view, i.e.,  $\sigma_{GTS}^x = \sum_{i \in V} m_i^x$ , and the algorithm logic. The innovative point of the new algorithm is comparing the margin value of every node's target topic influence by each pair of topics' operator and the diffusion equation.

Specifically, based on the OMT model, the initial setting of the MGTS-greedy algorithm assumes the same opportunity and value for each node in the network. Then, the margin in MGTS-greedy algorithm is based on how much influence each user on different topic layers to produce to neighbours on the target topic layer (i.e.,  $f^{yx}(t) = e^{t\alpha H} f^{yx}(0)$ ). The margin value reflects the influence ability of the user on the target topic based on the cross-impacts of all other topics (i.e.,  $\Delta inf s_i^{yx} = u_i^{yx}(t) - u_i^{yx}(0)$ ). After that, the margin of each node's influence on every topic is ranked in a list  $I$  from largest to smallest. Finally, according to the seed set budget, the top  $k$  nodes of a larger margin value from the list  $I$  are selected into the seed set to guarantee a relatively optimal solution of influence maximization. The detailed MGTS-greedy algorithm is described as Algorithm 1. Particularly, the selected seeds in the multi-topic networks could come out of the different topic layers, which represents the diversified relations and influence abilities of other topics on the target topic.

## 5 Experiments and Discussion

Two experiments are conducted for this study to evaluate the performance of the OMT model. The first experiment simulates several topics' influence diffusion processes by two models and compares the distances between the simulated curves and their corresponding actual propagation curves to demonstrate the simulation performance of the OMT model and the significance of topic factors in the multi-topic environment. The second experiment evaluates the ability of the OMT model cooperating with the MGTS-greedy algorithm on the influence maximization problem.

---

**Algorithm 1.** The Multi-topic GTS-Greedy algorithm

---

Input: Graph of a social network  $G = (V, E)$  with a set of topic layers  $\{x, y, \dots, n\} = Q$

Output: Seed set  $S^x(k)$

- 1: **for** each *Individual*  $i$  **do**
  - 2:  $f^{yx}(0) = 0; f_i^{yx}(0) = m_0; u_i^{yx}(0) = \text{sum}[f^{yx}(0)] = m_0;$
  - 3: Run  $f^{yx}(t) = e^{t\alpha H} f_i^{yx}(0)$  on a pair of topics between any topic  $y$  and target topic  $x$ ;
  - 4: Generate  $u_i^{yx}(t) = \text{sum}[f^{yx}(t)];$
  - 5: Calculate  $\Delta \text{infs}_i^{yx} = u_i^{yx}(t) - u_i^{yx}(0);$
  - 6: Ranking  $\Delta \text{infs}_i^{yx}$  into the list  $I$ ;
  - 7: **end for**
  - 8: **for** any seed budget  $l = 1$  to  $k$  **do**
  - 9: Select top- $k$   $\Delta \text{infs}_i^{yx}$  for individual  $i^{yx}$  into set  $S^x(k)$ ;
  - 10: Output Seed set  $S^x(k) = \{i^x, j^{yx}, \dots, k^{nx}\};$
  - 11: **end for**
- 

### 5.1 Experiment Setup

**Dataset.** Musical influence dataset is used to conduct our experiments, which is published by Kaggle [19]. This dataset includes 20 musical types, denoting as 20 topics. We suppose that all musical topics diffusion can be seen as the relevant and positive influence. The influencers (i.e., the senders) and the followers (i.e., the receivers) possess one specific musical type in each influence transmission. Each pair of users may own the same musical types, referring to the direct influence propagation, or different musical types, revealing the indirect influence diffusion.

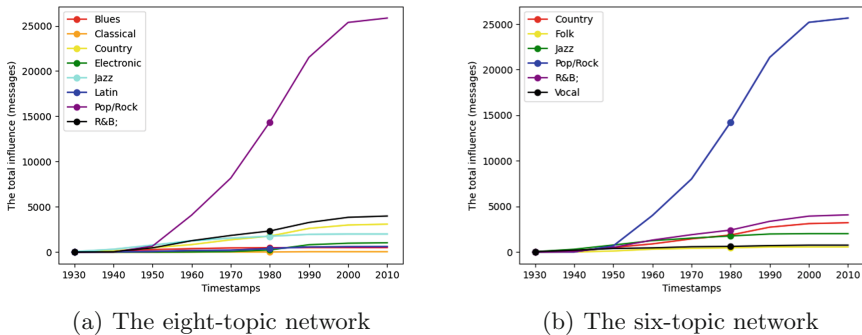
We extract two multi-topic network samples from 20 musical topics environment of the original dataset for our experiments. They include 8 topics and 6 topics, respectively. Every sample includes all the nodes, edges, and timestamps of the selected topics from the original dataset to keep the real topology and diffusion features. Specifically, Sample 1 has 4814 nodes and 37153 edges by including eight topics (i.e., “Pop/Rock”, “R & B”, “Jazz”, “Country”, “Blues”, “Classical”, “Latin” and “Electronic”). Sample 2 has 4496 nodes and 36343 edges by including six topics (i.e., “Pop/Rock”, “R & B”, “Jazz”, “Country”, “Folk”, “Vocal”).

**Model Setup.** In the OMT model, the parameters of user characteristics are derived from the real datasets. Specifically, user capacity  $c_i^x$  is established based on the users’ maximum interaction distribution, ranging from 1 to 10. User topic interest  $\alpha_i^x$  is a fixed value, relying on the topic percentage of all the received messages. User interaction  $R_i$  depends on the summation of in-degree and out-degree of actual users. Meanwhile, the user trust relationship represents the weight factor of the influence process, set as  $w_{ji} \approx \frac{\text{Number of instances where } v_j \text{ influences } v_i}{\text{Total number of influences affecting } v_i}$ . On the other hand, two topic parameters represent the actual topic effects of the real-world multi-topic datasets. Topic correlation, as a fixed value, reveals the

intrinsic relationship between topics, yielding by the actual network (refer to Definition 9). The details will be introduced in Environment Setup and Table 1. The topic penetration, as a time-varying value, reflects the dynamic propagation of each topic, generated by  $\beta^x \approx \frac{\text{Number of topic } x\text{'s diffused messages at time } t}{\text{Total number of overall topics' diffused messages at time } t}$ . Besides that, we leverage the diffused messages to measure the total influence state of the simulated and actual diffusion in the network because every diffused message serves as the carrier for propagating topics' influence.

**Target Topic.** In the multi-topic environment, the target topic denotes a particular topic that intends to be widely propagated to achieve a huge impact. We choose a target topic in these two network samples to explore its influence diffusion simulation in Experiment 1 and its influence maximization prediction in Experiment 2. According to the actual influence diffusion of each topic in two network samples, shown in Fig. 1, we select the Pop/Rock topic as the target. Obviously, the influence of “Pop/Rock” shows a more significant increase in two networks by measuring the number of diffused messages, which matches the practical marketing or commercial demands.

**Environment Setup.** Recalling Eqs. 1 and 2 to calculate the topic correlations, shown in Table 1, we reveal the topic connectivities for these multi-topic network environments and reflect the possibility of a cross-topic impact on the three common topics diffusion in two networks. Specifically, for the target topic, “Pop/Rock” has a stronger correlation with “classical”, presenting  $\theta^{xy} = 0.0338$ , while it shows a weaker relation with “R&B”, where  $\theta^{xy} = 0.0016$  in Sample 1. The target topic correlations in Sample 2 are less than 0.01, meaning other topics have minor effects on its influence diffusion. Furthermore, calculating topic correlations of “Jazz” & “R&B” is the preparation for simulating multiple topics' influence diffusion in Experiment 1.



**Fig. 1.** Real influence diffusion trends by diffused messages

**Table 1.** The topic correlations

Correlation ( $\theta^{xy}$ )	Sample 1 (eight-topic network)			Sample 2 (six-topic network)		
	Pop/Rock*	Jazz	R& B	Pop/Rock*	Jazz	R& B
Pop/Rock	1	0.0040	0.0016	1	0.0039	0.0016
Jazz	0.0040	1	0.0077	0.0039	1	0.0079
R& B	0.0016	0.0077	1	0.0016	0.0079	1
Country	0.0023	0.0494	0.0269	0.0023	0.0526	0.0281
Blues	0.0051	0.0383	0.0141	–	–	–
Classical	0.0338	0.5464	0.0303	–	–	–
Electronic	0.0041	0.2577	0.0384	–	–	–
Latin	0.0172	0.0417	0.0686	–	–	–
Folk	–	–	–	0.0032	0.1850	0.1640
Vocal	–	–	–	0.0057	0.0156	0.0104

\* represents the target topic in the multi-topic environment.

### 5.2 Experiment 1

The first experiment aims to verify the strong simulation capacity of the OMT model in the multi-topic environments by comparing the distances of the OMT and OBM simulations to the actual propagation. As mentioned, the difference between these two models is reflected as the consideration of the topic factors. We simulate the influence diffusion trends of “Pop/Rock”, “Jazz”, and “R&B”, which are common topics in two samples.

Figures 2(a) and 2(b) show three topics’ OMT and OBM simulated influence diffusion trends and their real propagation processes in two sample networks. Comparing the distances of the OMT and OBM simulated curve to Real Diffusion (RD), all the three topics’ OMT simulation is closer to the RD than the OBM simulation. Particularly, the “Jazz” and “R&B” simulated trends almost coincide to the RD. In terms of “Pop/Rock”, its OMT simulation displays a diffusing trend around the RD and presents a shorter distance to the RD than that of the OBM simulation, The OMT and RD curves intersect between 1980 and 1990. Therefore, the OMT model can simulate several topics’ influence diffusion processes and achieve excellent performances in multi-topic environments.

### 5.3 Experiment 2

In the second experiment, the combination of the OMT model and the MGTS-Greedy algorithm is employed to address the influence maximization problem to evaluate the application ability of the OMT model. The OMT model plays a role as the influence diffusion model for each algorithm due to the multi-topic network setup and its nearly realistic simulation. We select three classical seed selection approaches (i.e., random selection [11], degree ranking selection [11], K-step greedy selection [15]) and the MGTS-Greedy algorithm (Algorithm 1) to compare the different applied performances of the OMT.

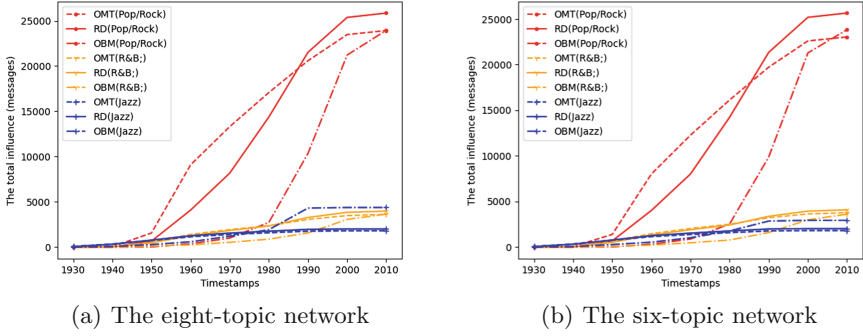


Fig. 2. Four topic influence diffusion simulations by diffused messages

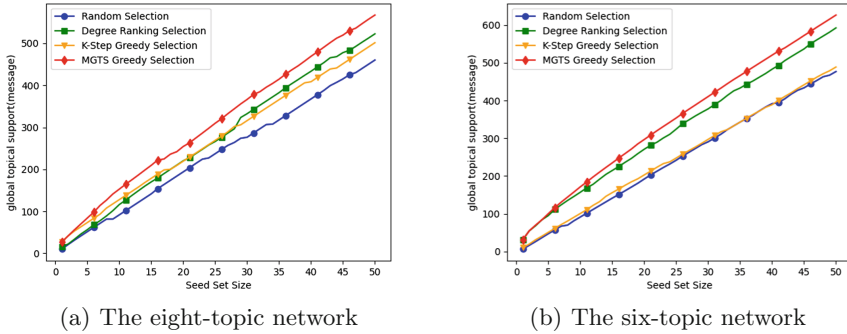


Fig. 3. The algorithms comparison by maximizing Pop/Rock influence

Figure 3(a) and 3(b) display the OMT model applications on four algorithms by maximizing Pop/Rock influence in different multi-topic networks. Obviously, the MGTS-greedy algorithm shows a higher global topical support in both multi-topic environments. That means the target topic can realize a higher positive influence in the entire social network through applying the OMT model with MGTS-greedy approach. Therefore, the outstanding performances of the combination of the OMT model and the MGTS-greedy algorithm in the influence maximization field can be evidence for the strong ability of the OMT model’s practical application.

### 5.4 Discussion

In the experiments, we employ the OMT model to simulate the influence diffusion process and apply the MGTS-greedy algorithm to realize the influence maximization in multi-topic environments with outstanding performances. There are several insights found by us: (1) In a multi-topic network, the topic influence received is not always the same as the sent influence, which results in a pair of users could hold different topics after one-time influence transmission; (2)

The topic penetration is significant to dynamically present the topic importance from the overall view; (3) The topic correlation is vital in revealing two topics' cross-impact of their intrinsic relationship; (4) A comprehensive analysis of topic factors is crucial to the OBM high-quality simulation; (5) Due to its extensive practicality, the OMT model provides an approach to predict the influence diffusion in the real networks.

## 6 Conclusion

In this paper, we propose an operator-based multi-topic model to explain the influence propagation process in the multi-topic complex social network. The OMT model leverages the benefits of the OBM and the topic factors to analyze the real influence propagation with user context-awareness and topic-awareness comprehensively, revealing the complexity of the user and information features. Particularly, we focus on the cross-impact of multiple topics and each topic current influence status. The experimental results reveal the importance of these topic factors in influence diffusion modelling. The OMT model is capable of simulating multiple topics' influence diffusion with remarkable similarity to the real-world influence propagation process. Furthermore, the MGTS-greedy algorithm is proposed to cooperate with the OMT model, achieving an outstanding performance on influence maximization. Finally, the future work of this research will continually improve the efficiency of the MGTS-greedy algorithm for seed selection in multi-topic networks.

## References

1. Barbieri, N., Bonchi, F., Manco, G.: Topic-aware social influence propagation models. *Knowl. Inform. Syst.* **37**(3), 555–584 (2013)
2. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. *J. Mach. Learn. Res.* **3**, 993–1022 (2003)
3. Chen, S., Fan, J., Li, G., Feng, J., Tan, K.l., Tang, J.: Online topic-aware influence maximization. *Proc. VLDB Endowment* **8**(6), 666–677 (2015)
4. Du, N., Song, L., Woo, H., Zha, H.: Uncover topic-sensitive information diffusion networks. In: *Artificial Intelligence and Statistics*, pp. 229–237. PMLR (2013)
5. Dwivedi, Y.K., et al.: Setting the future of digital and social media marketing research: perspectives and research propositions. *Int. J. Inform. Manage.* **59**, 102168 (2021)
6. Huiyu, M., Jiuxin, C., Tangfei, Y., Liu, B.: Topic based time-sensitive influence maximization in online social networks. *World Wide Web* **23**(3), 1831–1859 (2020)
7. Jiang, C., D'Arienzo, A., Li, W., Wu, S., Bai, Q.: An operator-based approach for modeling influence diffusion in complex social networks. *J. Soc. Comput.* **2**(2), 166–182 (2021). <https://doi.org/10.23919/JSC.2021.0007>
8. Li, C.T., Huang, M.Y., Yan, R.: Team formation with influence maximization for influential event organization on social networks. *World Wide Web* **21**(4), 939–959 (2018)
9. Li, M., Wang, X., Gao, K., Zhang, S.: A survey on information diffusion in online social networks: Models and methods. *Information* **8**(4), 118 (2017)



10. Li, Q., Wang, Z., Wu, B., Xiao, Y.: Competition and cooperation: dynamical interplay diffusion between social topic multiple messages in multiplex networks. *IEEE Trans. Comput. Soc. Syst.* **6**(3), 467–478 (2019)
11. Li, W., Bai, Q., Zhang, M.: A multi-agent system for modelling preference-based complex influence diffusion in social networks. *Comput. J.* **62**(3), 430–447 (2019)
12. Li, Y., Chen, Y., Wang, Q.: Evolution and diffusion of information literacy topics. *Scientometrics* **126**(5), 4195–4224 (2021)
13. Liang, Z., Jia, Y., Zhou, B., Zhang, B.: Topic diffusion behavior tracking in online social network. In: Zhao, M., Sha, J. (eds.) *ICCIP 2012. CCIS*, vol. 289, pp. 725–733. Springer, Heidelberg (2012). [https://doi.org/10.1007/978-3-642-31968-6\\_86](https://doi.org/10.1007/978-3-642-31968-6_86)
14. Liniger, T.J.: *Multivariate hawkes processes*. Ph.D. thesis, ETH Zurich (2009)
15. Ma, H., Yang, H., Lyu, M.R., King, I.: Mining social networks using heat diffusion processes for marketing candidates selection. In: *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, pp. 233–242 (2008)
16. Mahdizadehghadam, S., Wang, H., Krim, H., Dai, L.: Information diffusion of topic propagation in social media. *IEEE Trans. Signal Inform. Process. Over Netw.* **2**(4), 569–581 (2016)
17. Min, H., Cao, J., Yuan, T., Liu, B.: Topic based time-sensitive influence maximization in online social networks. *World Wide Web* **23**(3), 1–29 (2020)
18. Nolasco, D., Oliveira, J.: Mining social influence in science and vice-versa: a topic correlation approach. *Int. J. Inform. Manag.* **51**, 102017 (2020)
19. Ophelia: Music artists/influence data, February 2021. <https://www.kaggle.com/chuninghe/music-artistsinfluence-data>
20. Pan, T., Kuhnle, A., Li, X., Thai, M.T.: Popular topics spread faster: New dimension for influence propagation in online social networks. *arXiv preprint arXiv:1702.01844* (2017)
21. Pinto, J.C.L., Chahed, T.: Modeling multi-topic information diffusion in social networks using latent dirichlet allocation and hawkes processes. In: *2014 Tenth International Conference on Signal-Image Technology and Internet-Based Systems*, pp. 339–346. IEEE (2014)
22. Singh, S.S., Kumar, A., Singh, K., Biswas, B.: C2im: community based context-aware influence maximization in social networks. *Physica A Stat. Mech. Appl.* **514**, 796–818 (2019)
23. Wang, Q., Jin, Y., Yang, T., Cheng, S.: An emotion-based independent cascade model for sentiment spreading. *Knowl.-Based Syst.* **116**, 86–93 (2017)
24. Wu, D., Li, C., Lau, R.Y.: Topic based information diffusion prediction model with external trends. In: *2015 IEEE 12th International Conference on e-Business Engineering*, pp. 29–36. IEEE (2015)
25. Yu, M., Gupta, V., Kolar, M.: An influence-receptivity model for topic based information cascades. In: *2017 IEEE International Conference on Data Mining (ICDM)*, pp. 1141–1146. IEEE (2017)
26. Yu, M., Gupta, V., Kolar, M.: Estimation of a low-rank topic-based model for information cascades. *J. Mach. Learn. Res.* **21**(71), 1–47 (2020)



# Modeling User Profiles Through Multiple Types of User Interaction Behaviors

Yimin Lv<sup>1,2</sup>, Xinzhou Dong<sup>1,2</sup>, Beihong Jin<sup>1,2(✉)</sup>, and Wei Zhuo<sup>3</sup>

<sup>1</sup> State Key Laboratory of Computer Science, Institute of Software,  
Chinese Academy of Sciences, Beijing, China  
Beihong@iscas.ac.cn

<sup>2</sup> University of Chinese Academy of Sciences, Beijing, China

<sup>3</sup> MX Media Co., Ltd., Singapore, Singapore

**Abstract.** Constructing user profiles for online services with missing user data and sparse user behaviors has been a challenge. For users of online video services, we distinguish multiple interaction behaviors between users and videos, and propose the Multi-behavior Matrix Factorization (MMF) method for constructing user profiles. MMF improves the alternating least squares method for jointly decomposing multiple behavior-induced matrices and realizes the parallel solving of the user latent matrix and attribute latent matrix. In the paper, we give the workflow of MMF and show an example of achieving the video cold-start recommendation by the constructed user profiles. The resulting recalls are all above 0.7, which illustrates the availability of user profiles constructed by MMF.

**Keywords:** User profiles · Multi-behavior modeling · Matrix factorization · Recommendation

## 1 Introduction

Constructing profiles for users of online services is of great importance to locating the core population of online services, estimating the market size, helping make business decisions and achieving personalized marketing. In user profiles, various tags are used to identify multi-dimensional features of users. Traditionally, user profiles are constructed by statistical and rule-based methods. For example, in e-commerce websites, initial user profiles can be developed by counting user age, gender, consumption amount of online shopping, etc. These methods are easy to implement, moreover, the results match intuitions and have good explainability.

However, in many online service scenarios, missing user data and sparse user behaviors are very common. Take online video services as an example. Usually, users visit video services anonymously. Video services do not know the demographic characteristics of users, but they provide each user with an ID and record

---

This work was supported by the National Natural Science Foundation of China under Grant No. 62072450 and the 2021 joint project with MX Media.

their behaviors, including watching, liking, sharing, downloading videos, adding videos to the watchlist, etc. The last four behaviors can convey the user preference on the video and they are called explicit feedback. In contrast, the watching behavior is called implicit feedback. Furthermore, we find that the interactions between users and videos are sparse, and the amount of implicit feedback is much more than that of explicit feedback. In particular, users might not always give some kind of explicit feedback. For example, some user never gives any video a thumb up. These features of online services make it difficult to adopt traditional methods to construct user profiles. Thus, machine learning methods are needed.

Recently, some work focuses on constructing the profiles for users on social media platforms, where users interact with textual data such as tweets and microblogs. For example, Abel et al. [1] introduce a framework for modeling users on Twitter. Wang et al. [10] construct profiles for micro-blogging users by adopting user interest, social relationship and personal influence as tags. Shmueli et al. [8] also use user interest and social relationship to build user profiles, and then they recommend news stories to interested users by using content-based and collaborative filtering methods comprehensively. Zhao et al. [15] take google+ users as the target and improve the quality of user topic interest profiles through behavior factorization. On the other hand, some work serves users using search engines. For instance, Matthijs et al. [7] extract keywords from the user's browsing history and calculate the weights of keywords. Authors use these keywords and corresponding weights to construct user profiles and apply them to the re-ranking of web search results. Further, Bennett et al. [2] distinguish the short-term and long-term interactions between users and a search engine and then extract features from the user's browsing and query history to construct user profiles. With the popularity of topic models, Majumder et al. [6] propose a model named LTP (latent topic personalization), which mines the differences between output of personalized and vanilla services and identify user preferences on topics, obtaining personalized user topic interest vectors, that is, user profiles. Harvey et al. [4] extract the topic space from user's query log and build user profiles by analyzing the URLs that the user has clicked on in the topic space. The resulting user profiles are used for the training of the model which ranks search results.

We note that user behaviors and their relations can be modeled by employing the self-attention mechanism [16], multi-task learning [3], Transformer [11, 12] and graph neural network [5, 14, 17]. However, these methods cannot provide enough explainability required by user profiles. Thus, in the paper, for the anonymous users of an online video service, we propose a user profile construction method named MMF (Multi-behavior Matrix Factorization) which distinguishes multiple types of user behaviors. In particular, we collect interaction behaviors between users and videos, including explicit and implicit feedback, classify them into a dominant behavior and other supplementary behaviors, and construct behavior-induced matrices. Further, we propose an improved alternating least squares method for jointly decomposing multiple matrices and give a parallel solution for user latent matrix and attribute latent matrix. The user profile given by our method can accurately capture the user preference on video attributes, which not only has strong explainability, but also can be applied to downstream tasks such

as the video cold-start recommendation. Compared with the work of Yan et al. [13], our MMF method distinguishes the dominant behavior from supplementary behaviors and deals with behaviors of different categories by different strategies.

In the following sections, we give the overview of our MMF method, and show an example of achieving the video cold-start recommendation by the constructed user profiles.

## 2 Overview of MMF

For clarifying our goal and method, we give the following definitions first.

**Definition 1 (Categories of user-item interaction behaviors):** Generally, there are multiple types (for example,  $t$  types) of interaction behaviors between users and items in an online service. We take the one with the largest number of behaviors as user’s dominant behavior, and others as supplementary behaviors. We denote the set of user behaviors by  $\Gamma = \{B_0, B_1, \dots, B_t\}$ , where  $B_0$  is the dominant behavior and  $B_t (t \neq 0)$  represents the supplementary behavior.

For online video services, we take “watching a video longer than 3 min” as the dominant behavior, which is denoted by  $B_0$ , and the liking, sharing, downloading videos and adding videos to the watchlist as supplementary behaviors, which are denoted by  $B_1, B_2, B_3$  and  $B_4$ , respectively.

**Definition 2 (User-item attribute matrix):** We use matrix  $\mathbf{R}_{m \times n}$  to represent interactions between users and items in online services, where  $m$  is the number of users and  $n$  is the number of distinct attribute values of items. In matrix  $\mathbf{R}_{m \times n}$ , the element  $r_{ui}$  is set to the number of interactions between the user  $u$  and the item with the attribute value  $i$  at first, then the elements in the same row are grouped by attribute and normalized by group. Therefore,  $r_{ui}$  represents the relative preference of user  $u$  on attribute value  $i$ .

We decompose  $\mathbf{R}_{m \times n}$  into two low-rank matrices: user latent matrix  $\mathbf{X}_{m \times f}$  and attribute latent matrix  $\mathbf{Y}_{n \times f}$ , where  $f$  is the dimension of latent space. Further, the latent vector  $\mathbf{x}_u \in \mathbb{R}^f$  of user  $u$  is defined as  $\mathbf{x}_u = (\mathbf{X}_{m \times f}(u, :))^T$ , which indicates the degrees of interest of user  $u$  in different latent factors; the latent vector  $\mathbf{y}_i \in \mathbb{R}^f$  of video attribute value  $i$  is defined as  $\mathbf{y}_i = (\mathbf{Y}_{n \times f}(i, :))^T$ , which indicates the strength of the attribute value  $i$  of some attribute under different latent factors.

**Definition 3 (User-item attribute matrices under different behaviors):** We use  $\mathbf{R}_{m \times n}^B$  to denote the interaction behavior  $B$  between users and items. Next, we use  $\mathcal{M} = \{\mathbf{R}_{m \times n}^B\}, B \in \Gamma$  to denote the interactions between users and items. Correspondingly,  $\mathbf{X}_{m \times f}^B$  represents the user latent matrix under behavior  $B$ , obtained by decomposition of  $\mathbf{R}_{m \times n}^B$ , and  $\mathbf{x}_u^B$  is the latent vector of user  $u$  under behavior  $B$ .

**Definition 4 (User Profile):** In the paper, the user profile is defined as user preferences on attribute values of items, which can be represented as follows.

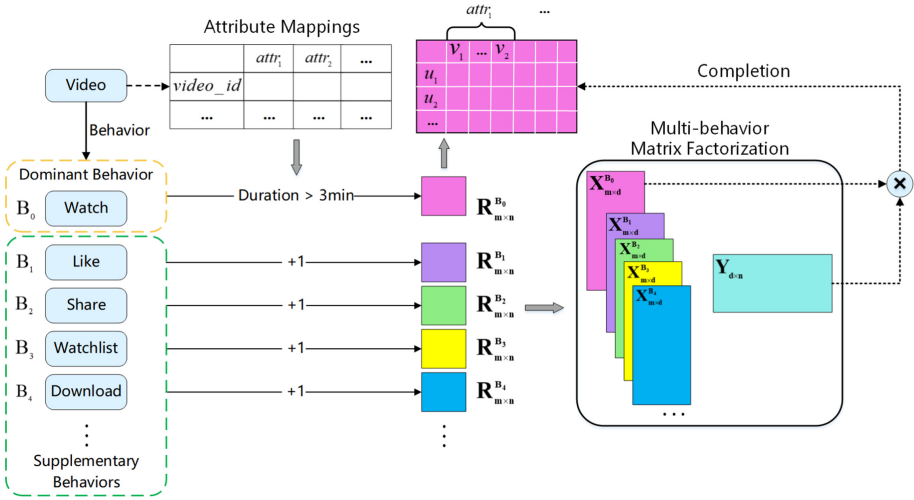


Fig. 1. Process of MMF

$$\mathcal{P} = \{P_u = \{r_{ui}\}\}, r_{ui} = (\mathbf{x}_u^{B_0 T} \mathbf{y}_i), u \in U, i \in V$$

where  $P_u$  represents the profile of user  $u$ ,  $r_{ui}$  represents the preference of user  $u$  on attribute value  $i$ ,  $\mathbf{x}_u^{B_0}$  is the latent vector of user  $u$  under dominant behavior  $B_0$ ,  $\mathbf{y}_i$  represents the latent vector of attribute value  $i$ ,  $U$  is the user set,  $V$  is the attribute value set.

Our goal is to construct profiles for users, that is, given  $\mathcal{M} = \{R_m^B \times n\}, B \in \Gamma$ , output  $\mathcal{P} = \{P_u = \{r_{ui}\}\}, u \in U, i \in V$ .

Our method MMF is to construct user profiles in three steps. The first step is to generate the matrix  $R_m^B \times n$  for each behavior; the second step is to obtain the user latent matrices under different behaviors and the attribute latent matrix through multi-behavior matrix factorization. The third step is to perform matrix completion based on the matrices obtained in the previous step to construct the user profile. Figure 1 shows the steps of the MMF method.

### Multi-behavior Matrix Factorization

To obtain the latent vectors of users and attributes for constructing the user profiles, we set the objective function as follows:

$$\min_{x_*, y_*} \sum_{B \in \Gamma} \sum_{u, i} c_{ui}^B (p_{ui}^B - \mathbf{x}_u^{B T} \mathbf{y}_i)^2 + \lambda \left( \sum_{B \in \Gamma} \sum_u \|\mathbf{x}_u^B\|^2 + \sum_i \|\mathbf{y}_i\|^2 \right), \quad (1)$$

where  $\Gamma = \{B_0, B_1, \dots, B_t\}$ ,  $\mathbf{x}_u^B$  represents the latent vector of user  $u$  under behavior  $B$ ,  $\mathbf{y}_i$  represents the latent vector of attribute value  $i$ , and  $p_{ui}^B$  is the preference coefficient of user  $u$  for attribute value  $i$  under behavior  $B$ . When  $r_{ui}^B > 0$ ,  $p_{ui}^B = 1$ , otherwise  $p_{ui}^B = 0$ .  $c_{ui}^B$  represents the confidence of user  $u$ 's preference coefficient for attribute value  $i$  under behavior  $B$ , which is defined as

$c_{ui}^B = 1 + \alpha r_{ui}^B$ . Moreover, we define the  $n \times n$  diagonal matrix  $\mathbf{C}^{uB}$  as  $c_{ii}^{uB} = c_{ui}^B$ , and the vector  $\mathbf{p}_u^B \in \mathbb{R}^n$  as  $\mathbf{p}_u^B = [p_{ui}^B], i \in V$ . Furthermore, we define the  $m \times m$  diagonal matrix  $\mathbf{C}^{iB}$  as  $c_{uu}^{iB} = c_{ui}^B$ , and the vector  $\mathbf{p}_i^B \in \mathbb{R}^m$  as  $\mathbf{p}_i^B = [p_{ui}^B], u \in U$ .

We improve the alternating least squares method to optimize the formula 1. We first fix attribute latent vectors to solve user latent vectors. When the attribute latent vectors are fixed, that is, the attribute latent matrix is constant, formula 1 can be transformed into the following formula:

$$\min_{x_*} \sum_{B \in \Gamma} \sum_u \sum_i c_{ui}^B (p_{ui}^B - \mathbf{x}_u^{B^T} \mathbf{y}_i)^2 + \lambda \sum_{B \in \Gamma} \sum_u \|\mathbf{x}_u^B\|^2. \quad (2)$$

When solving the latent vector of each user under the behavior  $B$ , we do not need to use other users' latent vectors and other behaviors, so we can transform the formula 2 into the formula 3:

$$\sum_{B \in \Gamma} \sum_u \min_u \sum_i c_{ui}^B (p_{ui}^B - \mathbf{x}_u^{B^T} \mathbf{y}_i)^2 + \lambda \|\mathbf{x}_u^B\|^2. \quad (3)$$

Using  $L_u^B$  to represent the expression after  $\min_u$  in formula 3, then taking the derivative of  $L_u^B$  with respect to  $\mathbf{x}_u^B$  and setting the derivative to 0. Through mathematical transformation, we can get

$$\mathbf{x}_u^B = (\mathbf{Y}^T \mathbf{C}^{uB} \mathbf{Y} + \lambda \mathbf{I})^{-1} \mathbf{Y}^T \mathbf{C}^{uB} \mathbf{p}_u^B. \quad (4)$$

Next, we fix the user latent vectors under all behaviors to solve the attribute latent vectors, that is, formula 1 is transformed into

$$\min_{y_*} \sum_{B \in \Gamma} \sum_u \sum_i c_{ui}^B (p_{ui}^B - \mathbf{x}_u^{B^T} \mathbf{y}_i)^2 + \lambda \sum_i \|\mathbf{y}_i\|^2. \quad (5)$$

When solving the latent vector of a certain attribute, we do not need to use the latent vectors of other attributes, so formula 5 can be transformed into

$$\sum_i \min_i \sum_{B \in \Gamma} \sum_u c_{ui}^B (p_{ui}^B - \mathbf{x}_u^{B^T} \mathbf{y}_i)^2 + \lambda \|\mathbf{y}_i\|^2. \quad (6)$$

Using  $L_i$  to represent the expression after  $\min_i$  in formula 6, then taking the derivative of  $L_i$  with respect to  $\mathbf{y}_i$  and setting the derivative to 0. Through mathematical transformation, we can get

$$\mathbf{y}_i = \left( \sum_{B \in \Gamma} \mathbf{X}^{B^T} \mathbf{C}^{iB} \mathbf{X}^B + \lambda \mathbf{I} \right)^{-1} \sum_{B \in \Gamma} \mathbf{X}^{B^T} \mathbf{C}^{iB} \mathbf{p}_i^B. \quad (7)$$

So far, we have obtained the  $\mathbf{x}_u^B$  and  $\mathbf{y}_i$ , as shown in formula 4 and formula 7, respectively. To further simplify the calculation, we convert formula 4 and formula 7 into formula 8 and formula 9, respectively, as shown below:

$$\mathbf{x}_u^B = \left( \mathbf{Y}^T \mathbf{Y} + \sum_{i \in r_{u^*}^B} (c_{ui}^B - 1) \mathbf{y}_i \mathbf{y}_i^T + \lambda \mathbf{I} \right)^{-1} \sum_{i \in r_{u^*}^B} c_{ui}^B \cdot 1 \cdot \mathbf{y}_i, \quad (8)$$

$$\mathbf{y}_i = \left( \sum_{B \in \Gamma} \mathbf{X}^{B^T} \mathbf{X}^B + \sum_{B \in \Gamma} \sum_{u \in r_{u^*}^B} (c_{ui}^B - 1) \mathbf{x}_u^B \mathbf{x}_u^{B^T} + \lambda \mathbf{I} \right)^{-1} \sum_{B \in \Gamma} \sum_{u \in r_{u^*}^B} c_{ui}^B \cdot 1 \cdot \mathbf{x}_u^B. \quad (9)$$

Converting formula 8 into the form  $\mathbf{A}\mathbf{x} = \mathbf{b}$ , we can get

$$\mathbf{A} = \mathbf{Y}^T \mathbf{Y} + \sum_{i \in r_{u^*}^B} (c_{ui}^B - 1) \mathbf{y}_i \mathbf{y}_i^T + \lambda \mathbf{I}, \quad (10)$$

$$\mathbf{b} = \sum_{i \in r_{u^*}^B} c_{ui}^B \cdot 1 \cdot \mathbf{y}_i. \quad (11)$$

Then, using the conjugate gradient method [9] to iteratively solve the equation, we can obtain the final  $\mathbf{x}_u^B$ . Following the method above, we can solve the latent vector of each user under each behavior, and finally get the user latent matrix under each behavior. Likewise, the attribute latent matrix can be obtained. Now an iteration is over. After iterating several times following the above process until the algorithm converges, we get the final user latent matrices and the attribute latent matrix.

**Parallelization.** In the MMF method, when using the conjugate gradient accelerated alternating least squares method to solve the latent vector of a specific user, we do not need to use other users' latent vectors; when solving the latent vector of a specific attribute, we do not need to use other attributes' latent vectors, which creates conditions for the parallelization of the MMF method.

To make full use of the advantages of multi-core processors, we adopt a multi-process method to implement MMF to accelerate the computation. Specifically, suppose that the number of processes is  $k$ . In each iteration, before solving the user latent matrix under a behavior, we divide the user set  $U$  into  $k$  disjoint subsets  $\{U_1, U_2, \dots, U_k\}$ , and then let process  $j$  ( $j = 1, 2, \dots, k$ ) calculate the latent vector of each user in the set  $U_j$ . At the end of the execution of all the processes, we merge the calculation results of  $k$  processes to obtain the user latent matrix under the current behavior. When calculating the attribute latent matrix, we also first divide the attribute set  $V$  into  $k$  disjoint subsets  $\{V_1, V_2, \dots, V_k\}$ , and let the process  $j$  ( $j = 1, 2, \dots, k$ ) be responsible for calculating the latent vector of each attribute in the set  $V_j$ . After all processes finish their execution, we merge the calculation results to obtain the attribute latent matrix.

### 3 Application Example of MMF

In this section, we show an application example, i.e., applying the user profile constructed by the MMF method to the cold-start recommendation of videos and then observing the performance.

**Datasets.** We collect information from the online video service MX Player, obtaining the information about 863,107 users of Android App watching videos

from June 1, 2020 to June 30, 2020, and form the sample dataset. We also collect attribute information (including video language, genre, etc.) of 9477 videos.

We filter out the users in the sample dataset who browse less than ten videos or whose behavior types are less than three or whose watching duration is less than 3 min. Considering that some users might watch the same video several times, we combine the duration as well as behaviors of these users. Based on the above data, we construct the set of user-item attribute matrices under multiple types of user behaviors, denoted by  $\{\mathbf{R}^{B_0}, \mathbf{R}^{B_1}, \mathbf{R}^{B_2}, \mathbf{R}^{B_3}, \mathbf{R}^{B_4}\}$ . Next, we randomly select 30% of users from  $\mathbf{R}^{B_0}$  to be users in the test set (denoted as  $U_t$ ). For each selected user  $u$ , we randomly extract one of the non-zero elements and 100 zero elements from  $\mathbf{R}^{B_0}(u, :)$  (denoted as  $a_u^0, a_u^1, \dots, a_u^{100}$ , respectively) to constitute the test data for this user (denoted as  $A_u$ ). Using this method, we obtain the test set for  $U_t$ . Then, the updated  $\mathbf{R}^{B_0}$  (i.e., removing the test data and normalizing again) and  $\mathbf{R}^{B_1}, \mathbf{R}^{B_2}, \mathbf{R}^{B_3}, \mathbf{R}^{B_4}$  are used as the training set.

**Video Cold-start Recommendation.** We randomly select three videos, each of which is watched at least 5000 times with more than five minutes per watching. Given a video  $m$  and a user  $u$ , we filter out all the watching records of video  $m$  from the log data of user  $u$ . Then, we perform the user-item attribute matrix completion by adopting the MMF method and get the profile of user  $u$ . We take the sum of the user  $u$ 's score of the attribute values of video  $m$  as the score of video  $m$ . If the score of video  $m$  is higher than the average score of other videos he/she has watched, it is considered that the MMF method successfully captures user  $u$ 's interest in video  $m$ , and the recall is increased by one.

**Table 1.** Effect of video cold-start recommendation

Video ID	#User	#User recalled	Recall
750	5876	4328	0.7366
899	9981	7681	0.7696
4599	5436	4099	0.7540

The experimental results are shown in Table 1, where the total number of users refers to the number of audiences who watch the current video for more than 5 min. From Table 1, we can see that the resulting recalls of three videos cold-start recommendation are all above 0.7.

## References

1. Abel, F., Gao, Q., Houben, G.-J., Tao, K.: Analyzing user modeling on twitter for personalized news recommendations. In: Konstan, J.A., Conejo, R., Marzo, J.L., Oliver, N. (eds.) UMAP 2011. LNCS, vol. 6787, pp. 1–12. Springer, Heidelberg (2011). [https://doi.org/10.1007/978-3-642-22362-4\\_1](https://doi.org/10.1007/978-3-642-22362-4_1)



2. Bennett, P.N., et al.: Modeling the impact of short- and long-term behavior on search personalization. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 185–194 (2012)
3. Gao, C., et al.: Neural multi-task recommendation from multi-behavior data. In: 2019 IEEE 35th International Conference on Data Engineering (ICDE), pp. 1554–1557. IEEE (2019)
4. Harvey, M., Crestani, F., Carman, M.J.: Building user profiles from topic models for personalised search. In: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, pp. 2309–2314 (2013)
5. Jin, B., Gao, C., He, X., Jin, D., Li, Y.: Multi-behavior recommendation with graph convolutional networks. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 659–668 (2020)
6. Majumder, A., Shrivastava, N.: Know your personalization: learning topic level personalization in online services. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 873–884 (2013)
7. Matthijs, N., Radlinski, F.: Personalizing web search using long term browsing history. In: Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, pp. 25–34 (2011)
8. Shmueli, E., Kagian, A., Koren, Y., Lempel, R.: Care to comment? recommendations for commenting on news stories. In: Proceedings of the 21st International Conference on World Wide Web, pp. 429–438 (2012)
9. Takács, G., Pilászy, I., Tikk, D.: Applications of the conjugate gradient method for implicit feedback collaborative filtering. In: Proceedings of the fifth ACM Conference on Recommender Systems, pp. 297–300 (2011)
10. Wang, B., et al.: Whom to mention: expand the diffusion of tweets by@ recommendation on micro-blogging systems. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 1331–1340 (2013)
11. Xia, L., Huang, C., Xu, Y., Dai, P., Zhang, B., Bo, L.: Multiplex behavioral relation learning for recommendation via memory augmented transformer network. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2397–2406 (2020)
12. Xia, L., et al.: Knowledge-enhanced hierarchical graph transformer network for multi-behavior recommendation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 4486–4493 (2021)
13. Yan, H., Yang, C., Yu, D., Li, Y., Jin, D., Chiu, D.M.: Multi-site user behavior modeling and its application in video recommendation. *IEEE Trans. Knowl. Data Eng.* **33**(1), 180–193 (2019)
14. Zhang, W., Mao, J., Cao, Y., Xu, C.: Multiplex graph neural networks for multi-behavior recommendation. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, pp. 2313–2316 (2020)
15. Zhao, Z., Cheng, Z., Hong, L., Chi, E.H.: Improving user topic interest profiles by behavior factorization. In: Proceedings of the 24th International Conference on World Wide Web, pp. 1406–1416 (2015)
16. Zhou, C., Bai, J., Song, J., Liu, X., Zhao, Z., Chen, X., Gao, J.: Atrank: an attention-based user behavior modeling framework for recommendation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
17. Zhuo, W., et al.: A behavior-aware graph convolution network model for video recommendation. APWeb-WAIM 2021, The 5th Asia Pacific Web and Web-Age Information Management Joint International Conference on Web and Big Data. arXiv preprint [arXiv:2106.15402](https://arxiv.org/abs/2106.15402) (2021)



# HACK: A Hierarchical Model for Fake News Detection

Yanqi Li<sup>1,2</sup>, Ke Ji<sup>1,2(✉)</sup>, Kun Ma<sup>1,2</sup>, Zhenxiang Chen<sup>1,2</sup>, Jun Wu<sup>3</sup>, Yidong Li<sup>3</sup>,  
and Guandong Xu<sup>4</sup>

<sup>1</sup> School of Information Science and Engineering,  
University of Jinan, Jinan 250022, China  
ise\_jik@ujn.edu.cn

<sup>2</sup> Shandong Provincial Key Laboratory of Network Based Intelligent Computing,  
University of Jinan, Jinan 250022, China

<sup>3</sup> School of Computer and Information Technology, Beijing Jiaotong University,  
Beijing 100044, China

<sup>4</sup> Data Science and Machine Intelligence Lab, Advanced Analytics Institute,  
University of Technology Sydney, Ultimo, Australia

**Abstract.** Online social media sites have become the most powerful platform to share news nowadays. However, all kinds of unauthenticated news that are released online without strict limits may lead to the spread of fake news, which has become a synonym for social and political threats. The existing solutions to the fake news issue are mostly trying to construct a social graph network by integrating the news content and social context of the news, which may be restricted when lacking social context information. In this paper, we propose a model for text only, regardless of contextual information, and named it HACK (**H**ier**A**rchical dete**C**tion for fa**K**e news), which can construct high-level combined features of spatial capsule vectors from low-level character features and phrase features by fusing a pre-trained language model and convolution network. The experimental results on real-life data show that the classification accuracy is significantly improved by our method comparing with the state-of-the-art methods.

**Keywords:** Fake news · Hierarchical framework · Feature extraction · Pre-trained LM · CapsNet

## 1 Introduction

With the development of the Internet, people tend to deliver news on online social media sites, which allow anyone to express themselves and convey the news to others. However, most of the sites have no strict limits on user behavior, and so all kinds of unauthenticated news are heavily released, among them there is probably fake news as well.

As social media has widely permeated all aspects of our life, the spread of fake news may mislead popular belief and poses a serious threat to public

security. Several approaches have been proposed for solving the problem. They can be broadly divided into three categories: content-based methods and social context-based methods. In the above research work, machine learning methods and deep learning methods are employed to improve classification performance. Though existing work has overcome this issue to a certain extent, there are some restrictions for practical use. For example, many times the publishers of fake news may just be newcomers, only have little user context information, even without social network behavior. In this case, it's impossible to detect fake news only relying on construct propagation network based on social context information.

Many content-based classification methods like [18] have been implemented to the fake news issue, particularly some deep learning frameworks via CNN [7] or RNN [2], which can encode the text to represent content. However, most of the frameworks extract features from an overall perspective, seriously losing structural information about spatial patterns in different positions of the text. In addition, not all content features are useful and isolated features may not do much for recognizing fake news. Clearly, the combination of local features in different positions is more beneficial to distinguish fake news.

## 2 Related Work

### 2.1 Fake News Detection

The existing methods can be divided into two categories, content-based and social context-based methods, and they are often used in combination in actual operation.

News content is the most intuitive discriminant basis. Because fake news often has exaggerated or radical emotional overtones [16], there has been some research work on conflict viewpoints [8] and stance factors [19]. Apart from the content itself, social contextual information related to the news and publishers [1] (e.g., number of posts, age of the account, number of friends/followers) can present a clear description of when the news appears, which is used to give a verified status for the credibility. Furthermore, one of the main hazards of fake news lies in its powerful dissemination ability in a social context. As in [8], the authors propose a credibility propagation network by taking advantage of the conflicting viewpoints of users. In [13], the authors propose a model that can capture changes in user characteristics along the propagation path based on recurrent and convolutional networks. Furthermore, the proposed model in [14] utilizes a four-layer Graph CNN to fuse content, user profile and activity, social graph, and news propagation, which has a better performance than using convolutional networks. However, there are some restrictions for context-based methods when social context information collected from the publishers is sparse or limited.

## 2.2 Text Classification

In essence, fake news detection can be abstracted to the text classification issue (more specifically, binary classification). Therefore, many NLP-based machine learning approaches can be applied to solve the fake news issue.

The core of text classification is how to get a text representation that contains more information. The traditional methods such as K-nearest Neighbor and Naive Bayes are influenced greatly by the sparsity of features. In recent years, the neural network is introduced into NLP, particularly the pre-trained language model, which can construct better-distributed feature representation with contextual information and model complex relationships within sparse data. Representative methods are Word2Vec and GloVe. However, both of them can only generate static word embeddings, which leads to the polysemy problem. To solve this problem, pre-trained language models such as ELMO [15], GPT [17], and BERT [3] are designed to dynamically adjust word embedding according to text context. In addition, due to the powerful effect of CNN and RNN in processing sequence data, many text classification methods [4, 5] are proposed. The existing classifiers with good performance are all based on CNN and RNN. We will use them as the baseline approaches and discuss them in Results Analysis.

## 3 Our Model

### 3.1 Character Feature Representation

The pre-trained language model can not only integrate contextual information into each character, the smallest Chinese language unit, through the self-attention mechanism but also bring prior knowledge from other corpora into the current task, overcoming the problem of insufficient data. We choose Bert [3] as the first level of our framework to get the embedding representation for the characters.

The input is the sum of  $k$ -dimensional embeddings: token  $E_i^t$ , segmentation  $E_i^s$ , and position  $E_i^p$ :

$$x_i = E_i^t \oplus E_i^s \oplus E_i^p \quad (1)$$

Given news  $x_i$  containing  $L$  characters, the pre-training Bert can output the embedding vectors  $T = [\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_L]$  corresponding to the characters.

### 3.2 Phrase Feature Representation

Character is the basic language unit, several adjacent characters can compose a local feature of the text, particularly for Chinese text, only the word or phrase with multiple characters has expressive meaning. We set a sliding window across the vectors  $\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3, \dots$

Assume window size is set to  $K_1$  and  $j$ -th filter for the convolution operation is  $W_j \in \mathbb{R}^{K_1 * k}$ . For example, a new feature  $c_{i,j}$  is generated from the adjacent characters in the window:

$$c_{i,j} = f(W_j \circ \mathbf{t}_{i:i+K_1+1} + b), \quad (2)$$

where  $\mathbf{t}_{i:i+K_1-1}$  means the characters from  $i$  to  $i + K_1 - 1$ ,  $f$  is a nonlinear function,  $\circ$  is element-wise multiplication and  $b$  is a bias term. If we set  $B$  filters, the similar operations in the same position can generate  $B$  new features that are arranged as a vector:

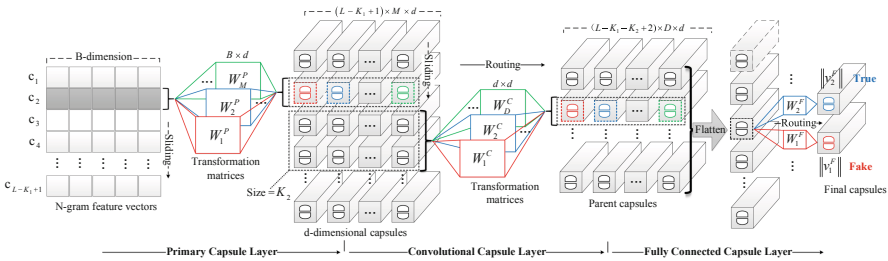
$$\mathbf{c}_i = (c_{i,1}, c_{i,2}, c_{i,3}, \dots, c_{i,B}) \tag{3}$$

The sliding window from front to back can produce  $L - K_1 + 1$  vectors, which can be arranged as matrix  $C \in \mathbb{R}^{(L-K_1+1) \times B}$ .

$$C = [\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3, \dots, \mathbf{c}_{L-K_1+1}] \tag{4}$$

### 3.3 Sentence Feature Representation

Because the kernel size of the convolution filter is usually small and only focuses on the local N-gram, the pooling operation will lead to the loss of position information. The combination of some local features in different positions will be possible for finding significant semantic features or spatial patterns to distinguish them. Inspired by CapsNet [6], we construct high-level combined feature representations based on it. The architecture is shown in Fig. 1, including three layers:



**Fig. 1.** The third level of our framework, which uses capsule network to construct high-level combined feature representation with capsule vectors.

**Primary Capsule Layer** converts every vector to some  $d$ -dimensional capsules by  $M$  transformation matrices  $W_1^P, W_2^P, \dots, W_M^P \in \mathbb{R}^{B \times d}$ ,  $j$ -th capsule  $\mathbf{p}_{i,j}$  derived from vector  $\mathbf{c}_i$  is computed as:

$$\mathbf{p}_{i,j} = g(W_j^P \mathbf{c}_i + \mathbf{b}_i) \tag{5}$$

where  $g$  is the nonlinear squash function. The same operation on all vectors of  $C$  will collect  $(L - K_1 + 1) \times M$  capsules as tensor  $P$ .

**Convolutional Capsule Layer** sets a  $K_2 \times M$  window on  $P$ , and learns  $D$  parent capsules based on  $K_2 \times M$  child capsules in the window. Suppose shared transformation matrices  $W_1^C, W_2^C, \dots, W_D^C \in \mathbb{R}^{d \times d}$ . Given a child capsule  $\mathbf{u}_i$ , the prediction vector  $\hat{\mathbf{u}}_{j|i}$  for potential parent capsule  $\mathbf{v}_j$  is computed as:

$$\hat{\mathbf{u}}_{j|i} = W_j^C \mathbf{u}_i + \hat{\mathbf{b}}_{j|i}, \tag{6}$$

where  $\hat{\mathbf{b}}_{j_i}$  is the capsule bias term. With the predictions, the dynamic routing algorithm is used to optimize iteratively how each child is sent to an appropriate parent. After the window slides to the end, we can have  $(L - K_1 - K_2 + 2) \times D$  parent capsules. See the paper [6] for details of the dynamic routing algorithm.

**Fully Connected Capsule Layer** flats the parent capsules below into a list  $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{(L-K_1-K_2+2) \times D}$ , which are fed into  $H$  final capsules  $\mathbf{v}_1^F, \mathbf{v}_2^F, \dots, \mathbf{v}_H^F$  by a fully connected layer. The process is similar to the steps above approach. Suppose  $H$  shared transformation matrices  $W_1^F, W_2^F, \dots, W_H^F \in \mathbb{R}^{d \times d}$ . The dynamic routing algorithm is used to optimize iteratively how each capsule  $\mathbf{v}_i$  is sent to the final capsule  $\mathbf{v}_j^F$ . Since recognizing fake news is binary classification,  $H$  is set to 2, corresponding to 2 final capsules. Because of adding squash function, the length of the output capsule is compressed from 0 to 1, representing the probability of a category.

Note that as the squash function is used to compress the capsules into (0,1) interval in dynamic routing, we use the length of the final capsule  $\mathbf{v}_i^F$  to represent the probability of a category.

## 4 Experiments

### 4.1 Datasets

We conduct extensive experiments on 2 benchmark news datasets in Chinese: BiendataFake and CHECKED. The basic statistics of the dataset are summarized in Table 1.

**BiendataFake:** This dataset is released by ICT (Institute of Computer Technology, Chinese Academy of Sciences) and BAAI (Beijing Academy of Artificial Intelligence), which are published in the competition platform - biendata<sup>1</sup>.

**CHECKED:** This dataset is the first Chinese COVID-19 fake news dataset [20] based on the Weibo platform and we adopt the dataset from on github<sup>2</sup>.

**Table 1.** Analysis of statistical information to the news

Dataset	Language	# Item			# Category
BiendataFake	Chinese	Fake: 19,285	Real: 19,378	Total: 38,663	2
CHECKED	Chinese	Fake: 344	Real: 1,759	Total: 2,103	2

### 4.2 Baselines

In order to comprehensively evaluate the effectiveness of our model, we compare it with the following baselines:

<sup>1</sup> <https://www.biendata.xyz/competition/falsenews/>.

<sup>2</sup> <https://github.com/cyang03/checked>.

**TextCNN** [10] is a CNN-based neural network model, the core point of CNN is that it can capture the local correlation of information.

**TextRNN** [12] is an RNN-based neural network model, due to the ability of memory function of RNN, TextRNN can tackle long text better.

**TextRCNN** [11] replaces the convolutional layer with a bidirectional RNN on the basis of TextCNN.

**DPCNN** [9] uses the structure of deep word-level CNN for text classification.

**Bert** [3] is a language model built on a bi-directional Transformer, the first token of every sequence is used as the feature representation for classification by a fully connected softmax layer.

**BertCNN** takes Bert as an embedder and is followed by CNN as a classifier.

**BertCNN\*** is an enhancement we have done based on BertCNN, not only taking account of max-pooling but also average-pooling. It uses a combination of the feature representations derived from two kinds of pooling operations for classification.

### 4.3 Experimental Results and Analysis

In this section, we investigate whether the classification performance can be improved by implementing our method. Table 2 shows the experimental results.

As shown in Table 1, the CHECKED has the problem of data imbalance, so the effect of HACK on CHECKED is not as obvious as that on BiendataFake. Because the amount of data of biendataFake is much larger than that of CHECKED, we divide biendataFake in the ratio of 0.1: 0.6: 1, where the division ratio 0.1 is to compare the experimental effect when the amount of data of biendataFake and CHECKED is similar, 0.6 and 1 are to test whether the model effect will be affected as the size of data increases.

Table 2 shows that CNN-based models perform better than RNN-based models on the whole in the fake news issue. The results show that the performances of RNN-based models are inferior to that of the CNN-based models, which verifies that CNN-based models are better suited for this issue. Bert, as a transformer-based model, which is good at integrating text context by self-attention mechanism, when takes it as an embedder and combines it with CNN, it can perform better. For the structure, BertCNN is similar to TextCNN, but BertCNN makes significant improvements over the latter, proving that Bert can construct better feature representations than the usual word embedding model does. It also performs better than Bert, proving the advantage of N-gram convolution and Max-pooling on feature extraction. BertCNN\* further improves BertCNN, illustrating that Average-Max pooling can tie up with Max-pooling to help abstract the features. At last, we observe that HACK outperforms all of the above methods. The comparison results show that instead of the output of a pre-trained language model or CNN-based feature extraction, a capsule network in the third level of our framework can better use a combination of local features in different positions to classify the fake news while preserving spatial information to the maximum.

**Table 2.** Test metrics results of different models on datasets. The best results are bold and the second-best results are underlined to easily compare.

Models	BiendataFake								
	p			r			f1		
	10%	60%	100%	10%	60%	100%	10%	60%	100%
TextCNN	0.7835	0.8757	0.9057	0.7773	0.8751	0.9043	0.7757	0.8751	0.9043
TextRNN	0.8399	0.8561	0.8860	0.8342	0.8560	0.8816	0.8183	0.8560	0.8814
TextRCNN	0.8415	0.8680	0.8863	0.8360	0.8661	0.8863	0.8206	0.8661	0.8863
DPCNN	0.7992	0.8987	<u>0.9177</u>	0.7962	0.8920	<u>0.9173</u>	0.7959	0.8916	<u>0.9172</u>
Bert	0.8569	0.8724	0.8818	0.8553	0.8656	0.8723	0.8560	0.8689	0.8770
BertCNN	0.8754	0.8809	0.8981	0.8719	0.8755	0.8875	0.8736	0.8781	0.8927
BertCNN*	<u>0.8818</u>	<u>0.8992</u>	0.9130	<u>0.8723</u>	<u>0.8890</u>	0.9038	<u>0.8770</u>	<u>0.8940</u>	0.9083
HACK	<b>0.9261</b>	<b>0.9310</b>	<b>0.9455</b>	<b>0.9179</b>	<b>0.9228</b>	<b>0.9396</b>	<b>0.9219</b>	<b>0.9268</b>	<b>0.9425</b>
Models	CHECKED								
	p			r			f1		
TextRNN	0.8648			<u>0.8385</u>			0.7721		
TextRCNN	0.6901			0.8307			0.7539		
DPCNN	0.6944			0.8333			0.7575		
Bert	<u>0.8993</u>			<b>0.8854</b>			<b>0.8598</b>		
BertCNN	0.7206			0.7142			0.7174		
HACK	<b>0.9117</b>			0.8258			<u>0.8452</u>		

## 5 Conclusion

In this paper, we solve the fake news issue by a hierarchical extraction framework, which can construct high-level features from low-level features. The novelty of our method is that with the capsule network, the local features can keep their spatial information by replacing neural nodes with capsule vectors, and the combination of pre-trained language model and capsule network can generate the sentence-level macroscopic feature representations by encoding spatial patterns.

**Acknowledgement.** This work is supported by National Science Foundation of China No. 61702216, 61772231, and Higher Educational Science and Technology Program of Jinan City under Grant with No. 2020GXRC057, 2018GXRC002.

## References

1. Chen, Z., Freire, J.: Proactive discovery of fake news domains from real-time social media feeds. In: Companion Proceedings of the Web Conference 2020, pp. 584–592 (2020)
2. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint [arXiv:1406.1078](https://arxiv.org/abs/1406.1078) (2014)
3. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding (2018)
4. Dos Santos, C., Gatti, M.: Deep convolutional neural networks for sentiment analysis of short texts. In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pp. 69–78 (2014)



5. Guo, L., Zhang, D., Wang, L., Wang, H., Cui, B., et al.: CRAN: a hybrid CNN-RNN attention-based model for text classification. In: Trujillo, J.C. (ed.) ER 2018. LNCS, vol. 11157, pp. 571–585. Springer, Cham (2018). [https://doi.org/10.1007/978-3-030-00847-5\\_42](https://doi.org/10.1007/978-3-030-00847-5_42)
6. Hinton, G.E., Sabour, S., Frosst, N.: Matrix capsules with EM routing (2018)
7. Jacovi, A., Shalom, O.S., Goldberg, Y.: Understanding convolutional neural networks for text classification. arXiv preprint [arXiv:1809.08037](https://arxiv.org/abs/1809.08037) (2018)
8. Jin, Z., Cao, J., Zhang, Y., Luo, J.: News verification by exploiting conflicting social viewpoints in microblogs. In: Thirtieth AAAI Conference on Artificial Intelligence (2016)
9. Johnson, R., Zhang, T.: Deep pyramid convolutional neural networks for text categorization. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 562–570 (2017)
10. Kim, Y.: Convolutional neural networks for sentence classification. Eprint Arxiv (2014)
11. Lai, S., Xu, L., Liu, K., Zhao, J.: Recurrent convolutional neural networks for text classification. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 29 (2015)
12. Liu, P., Qiu, X., Huang, X.: Recurrent neural network for text classification with multi-task learning. arXiv preprint [arXiv:1605.05101](https://arxiv.org/abs/1605.05101) (2016)
13. Liu, Y., Wu, Y.F.: Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
14. Monti, F., Frasca, F., Eynard, D., Mannion, D., Bronstein, M.M.: Fake news detection on social media using geometric deep learning. arXiv preprint [arXiv:1902.06673](https://arxiv.org/abs/1902.06673) (2019)
15. Peters, M.E., et al.: Deep contextualized word representations. arXiv preprint [arXiv:1802.05365](https://arxiv.org/abs/1802.05365) (2018)
16. Przybyla, P.: Capturing the style of fake news. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 490–497 (2020)
17. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding by generative pre-training (2018). <https://s3-us-west-2.amazonaws.com/openai-assets/researchcovers/languageunsupervised/languageunderstandingpaper.pdf>
18. Rubin, V.L., Conroy, N., Chen, Y., Cornwell, S.: Fake news or truth? Using satirical cues to detect potentially misleading news. In: Proceedings of the Second Workshop on Computational Approaches to Deception Detection, pp. 7–17 (2016)
19. Xu, C., Paris, C., Nepal, S., Sparks, R.: Cross-target stance classification with self-attention networks. arXiv preprint [arXiv:1805.06593](https://arxiv.org/abs/1805.06593) (2018)
20. Yang, C., Zhou, X., Zafarani, R.: Checked: Chinese COVID-19 fake news dataset. arXiv preprint [arXiv:2010.09029](https://arxiv.org/abs/2010.09029) (2020)

# **Spatial and Temporal Data Analysis**



# An Efficient Approach for Spatial Trajectory Anonymization

Yuetian Wang<sup>1</sup>, Wen Hua<sup>2(✉)</sup>, Fengmei Jin<sup>2</sup>, Jing Qiu<sup>1</sup>, and Xiaofang Zhou<sup>3</sup>

<sup>1</sup> Guangzhou University, Guangzhou 510006, China  
yuetian.w@e.gzhu.edu.cn, qiujing@gzhu.edu.cn

<sup>2</sup> The University of Queensland, Brisbane, QLD 4072, Australia  
{w.hua, fengmei.jin}@uq.edu.au

<sup>3</sup> The Hong Kong University of Science and Technology, Kowloon, Hong Kong  
zxf@cse.ust.hk

**Abstract.** Spatial trajectories are being extensively collected and utilized nowadays. When publishing trajectory datasets that contain identifiable information about individuals, it is critically important to protect user privacy against linking attack. Although k-anonymity has been proven as a powerful tool to tackle trajectory re-identification, there still exists a significant gap in model efficiency, which severely impacts the feasibility of existing approaches for large-scale trajectory data. In this paper, we propose *Gindex*, a highly scalable solution for trajectory k-anonymization. It utilizes a hierarchical grid index and various optimization techniques to speed up k-clustering and trajectory merging. Extensive experiments on a real-life trajectory dataset verify the efficiency and scalability of *Gindex* which outperforms existing k-anonymity models by several orders of magnitude.

**Keywords:** Trajectory privacy · Linking attack · K-anonymity · Hierarchical grid index · K-clustering · Trajectory merging

## 1 Introduction

Trajectory data is abundant yet vulnerable. Recent studies [4, 11, 12] have demonstrated that individuals can be identified with a sufficiently high success rate (>80%) by exploring their personalized movement patterns. To address this issue, various privacy protection techniques on trajectory data publishing have been developed including ad-hoc models (e.g., Mixzone [15] and Dummy [14]) and formal models (i.e., k-anonymity, l-diversity, t-closeness, and differential privacy). K-anonymization (e.g., W4M [2], GLOVE [7], KLT [19]) has been proven to be the most effective in tackling individual re-identification (or linking attack) [10] by hiding each trajectory within a group of  $k$  others. Nevertheless, a common bottleneck of existing trajectory k-anonymization models is the efficiency issue. Theoretically, k-anonymization needs to group trajectories into k-clusters while minimizing the overall utility loss (merging trajectories in a k-cluster to generate

the anonymized trajectory naturally results in spatial stretch, i.e., utility loss). This is NP-hard, and even the greedy algorithm would require time complexity of  $O(N^2n^2)$  where  $N = |D|$  is the dataset size and  $n = |\tau|$  is the average trajectory length, due to the inevitable pairwise computation of trajectory merge cost. Practically, our experiments on a real-world trajectory dataset T-drive [20] demonstrate that the most advanced k-anonymity algorithms (i.e., GLOVE and KLT) cannot scale up to large datasets, taking more than 2 days to anonymize 2000+ objects with trajectories for one-week period.

Obviously, such a full calculation of trajectory-wise merge cost is too time-consuming. In fact, real-life trajectories are usually localized and merging trajectories that are far away leads to huge utility loss (e.g., spatial stretch), which indicates that merge costs only need to be computed among nearby trajectories. In this paper, we take advantage of trajectory “locality” by designing effective indexing and pruning techniques to avoid unnecessary computation of merge cost between trajectories, so as to speed up the whole procedure of k-anonymization. Overall, our major contributions can be summarized as follows:

- We devise a hierarchical grid-based spatial encoding and indexing mechanism *Gindex* to reduce the time complexity of trajectory k-clustering from  $O(N^2n^2)$  to  $O(\frac{N \log N}{k} + Nm^2)$  ( $m \ll n$  is the average length of grid sequences) with a slight sacrifice in utility loss.
- We introduce z-ordering and length-based pruning to further improve *Gindex*, achieving a final time complexity of  $O(\frac{N \log N}{k} + Nm)$  for k-clustering.
- We utilize grid index to prune unnecessary calculation of point-level merge loss, which effectively speeds up trajectory merging.
- Experiments on the T-drive dataset verifies the superiority of *Gindex* which outperforms existing k-anonymity models by several orders of magnitude.

The remaining of the paper is organized as follows: Sect. 2 provides a summarization of related works; Sect. 3 formally defines the problem of trajectory k-anonymization; Sect. 4 introduces the details of our proposed *Gindex* algorithm and optimization techniques; Experimental results are reported in Sect. 5, followed by a brief conclusion in Sect. 6.

## 2 Related Work

**Privacy Protection in Relational Data.** In order to protect users’ privacy, many anonymization models have been proposed for relational data publishing including k-anonymity, l-diversity, t-closeness, and differential privacy. Basically, k-anonymity [18] requires that each record in the dataset cannot be distinguished from at least another  $k-1$  records. It is a fundamental standard against record linkage attacks. L-diversity [16] demands that an anonymity group contains at least  $l$  well-represented values for each sensitive attribute. T-closeness [13] requires the value distribution of a sensitive attribute in any anonymity group to be close to the distribution of the attribute in the overall dataset, i.e., the distance between distributions is smaller than  $t$ . Differential privacy [5, 6, 21]

restricts that the query result of any two neighboring datasets should be indistinguishable. It is usually achieved by adding randomized noise into the original dataset.

**Privacy Protection in Trajectory Data.** Ad-hoc models have been designed specifically for trajectory publishing. Mixzone [15] avoids moving objects being tracked by placing mixzone regions and swapping the pseudonym of objects passing through the mixzone. Dummy [14] releases original trajectories along with fictitious ones generated by resembling individuals' movement in free space. Although ad-hoc models are effective against certain attacks, they cannot provide theoretical privacy guarantee. Formal models extend the privacy guarantee achieved in relational database to trajectory data. In particular, NWA [1] is the first application of  $k$ -anonymity to trajectory publishing. It introduces  $(k, \delta)$ -anonymity by defining each trajectory as a cylinder of radius  $\delta$  due to positioning inaccuracy and then grouping  $k$  co-localization trajectories. W4M [2] extends NWA by using spatiotemporal edit distance as the trajectory similarity measure, to overcome the limitations of Euclidean distance. GLOVE [7] pays attention to the mobile traffic dataset and achieves  $k$ -anonymity by merging  $k$  trajectories for generalization. In order to minimize the resolution loss caused by generalization, it iteratively merges two trajectories with the minimum spatial stretch until the whole dataset is anonymized. KLT [19], based on GLOVE, further considers the semantic information (i.e., POIs) as the sensitive attribute of trajectory data. It introduces  $l$ -diversity and  $t$ -closeness to ensure that each location in a trajectory is mapped to heterogeneous semantic classes of POIs, thus hiding the semantics of individuals' movement. DPT [9] is a typical algorithm achieving differential privacy in trajectories. It models users' moving patterns with a hierarchical reference system, and utilizes the Laplacian mechanism to introduce random noise in the published synthetic dataset. Among these privacy models,  $k$ -anonymity achieves the best protection against linking attacks [10].

### 3 Preliminaries

**Definition 1 (Moving Object and Trajectory).** *A moving object  $o$  is an entity that changes position over time. Each object  $o$  is associated with a trajectory  $\tau_o$  that records its entire movement history. Specifically, trajectory  $\tau$  is a sequence of spatial points organized chronologically, denoted as  $\tau = \langle p_1, p_2, \dots, p_n \rangle$ , where each point  $p = (x, y)$  represents a spatial location  $(x, y)$  (e.g., latitude and longitude) and  $|\tau| = n$  is the trajectory length. A trajectory dataset is represented as  $D = \{\tau_1, \tau_2, \dots, \tau_N\}$ , where  $N$  denotes the total number of objects/trajectories (Table 1).*

In this work, we apply  $k$ -anonymity model [18], with its proven effectiveness in relational databases, to cope with linking attack in trajectory data. Generally speaking, a dataset satisfies  $k$ -anonymity if each record is indistinguishable among at least  $k$  other records. Hence,  $k$ -anonymity can naturally protect against re-identification of moving objects by hiding them within a group of  $k$  objects.

**Table 1.** Summary of notations.

Notation	Definition
$o$	Moving object
$\tau_o, \tau'_o$	The original and anonymized trajectories of moving object $o$
$p_i$	The $i$ -th point $p_i = (x, y)$ of trajectory $\tau$
$G_\tau$	The grid sequence of trajectory $\tau$
$g_i$	The $i$ -th grid in the grid sequence of trajectory $\tau$
$BG_\tau$	The bounding grid of trajectory $\tau$
$\phi(p_a, p_b)$	The point-level merge loss
$Loss(\tau_i, \tau_j)$	The trajectory-level merge loss
$Sim(\tau_i, \tau_j)$	The similarity between trajectories $\tau_i$ and $\tau_j$
$T_k$	The trajectory $k$ -cluster $\{\tau_1, \tau_2, \dots, \tau_k\}$

**Definition 2 (K-cluster and Representative Trajectory).** A  $k$ -cluster is a group of  $k$  similar trajectories, denoted as  $T_k = \{\tau_1, \tau_2, \dots, \tau_k\}$ , and is associated with a representative trajectory  $\tau'$  merged from the  $k$  trajectories.  $\tau'$  denotes the anonymized trajectory where the exact locations of those  $k$  objects are unknown.

Although  $k$ -clustering can provide privacy guarantee of  $k$ -anonymity, merging trajectories in the  $k$ -cluster naturally causes spatial stretch, i.e., utility loss (formalized in Sect. 4.1). Hence, we define trajectory  $k$ -anonymization below:

**Definition 3 (Trajectory K-anonymization).** Given a trajectory dataset  $D = \{\tau_1, \tau_2, \dots, \tau_N\}$ ,  $k$ -anonymization aims to generate an anonymized dataset  $D'$  consisting of representative trajectories of  $k$ -clusters, i.e.,  $D' = \{\tau'_1, \tau'_2, \dots, \tau'_M\}$  and  $M = \lfloor N/k \rfloor$ , such that the overall utility loss is minimized.

## 4 Methodology

$K$ -anonymization consists of two steps: 1) grouping the trajectory dataset  $D$  into  $k$ -clusters and 2) generating the representative trajectory  $\tau'$  for each  $k$ -cluster. The  $k$  merged trajectories should be “similar” to each other, in order to minimize utility loss. Therefore, trajectory similarity used for clustering can be defined based on the merge loss. In the following, we first introduce our definition of merge loss in Sect. 4.1, and then propose *Gindex* for efficient trajectory  $k$ -anonymization, as explained in Sect. 4.2.

### 4.1 Merge Loss

We merge trajectories in a point-by-point manner. Given two points (i.e.,  $p_a = (x_a, y_a)$  from trajectory  $\tau_i$  and  $p_b = (x_b, y_b)$  from trajectory  $\tau_j$ ), they naturally bound a rectangle denoted as  $p_c = (\underline{x}_c, \bar{x}_c, \underline{y}_c, \bar{y}_c)$ , where  $(\underline{x}_c, \underline{y}_c)$  (resp.  $(\bar{x}_c, \bar{y}_c)$ )

is the bottom-left (resp. top-right) position of the rectangle. Note that each original point  $p = (x, y)$  can also be represented as a rectangle format where  $\underline{x} = \bar{x} = x$  and  $\underline{y} = \bar{y} = y$ . In this way, the bounding rectangle can be formally calculated as below:

$$\begin{cases} \underline{x}_c = \min(\underline{x}_a, \underline{x}_b), \bar{x}_c = \max(\bar{x}_a, \bar{x}_b) \\ \underline{y}_c = \min(\underline{y}_a, \underline{y}_b), \bar{y}_c = \max(\bar{y}_a, \bar{y}_b) \end{cases} \quad (1)$$

We merge the two points  $p_a$  and  $p_b$  by replacing them with the bounding rectangle  $p_c$  (or the rectangle centre), which makes it unable to infer the exact object location within that rectangle. However, this actually results in a loss of spatial resolution (or utility loss). We define the point-level merge loss as follows:

$$\phi(p_a, p_b) = \frac{(l_a + r_a) + (l_b + r_b)}{2} \quad (2)$$

Here,  $l_a$  and  $r_a$  (resp.  $l_b$  and  $r_b$ ) quantify the left and right stretch in space when merging point  $p_a$  (resp.  $p_b$ ) into rectangle  $p_c$ :

$$\begin{cases} l_a = (\underline{x}_a - \underline{x}_c) + (\underline{y}_a - \underline{y}_c) \\ r_a = (\bar{x}_c - \bar{x}_a) + (\bar{y}_c - \bar{y}_a) \end{cases} \quad (3)$$

Finally, we normalize the point-level merge loss  $\phi(p_a, p_b)$  to  $[0,1]$  by setting the threshold  $\phi^m = 20$  km in the following equation:

$$\phi(p_a, p_b) = \min\left(\frac{\phi(p_a, p_b)}{\phi^m}, 1\right) \quad (4)$$

Given any two trajectories  $\tau_i$  and  $\tau_j$  from a k-cluster (say  $n_i > n_j$ , i.e.,  $\tau_i$  is longer than  $\tau_j$ ), we combine them by merging  $\tau_i$  into  $\tau_j$ . Specifically, for each point  $p_a$  in  $\tau_i$ , we merge it with the “closest” point  $p_b$  in  $\tau_j$  (i.e., with the smallest merge loss) so as to maximize utility preservation in the anonymized dataset. Thus, the trajectory-level merge loss  $Loss(i, j)$  can be formulated as below:

$$Loss(i, j) = \begin{cases} \frac{1}{n_i} \sum_{a=1}^{n_i} \min_{b=1, \dots, n_j} \phi(p_a, p_b), n_i > n_j \\ \frac{1}{n_j} \sum_{a=1, \dots, n_i} \min_{b=1}^{n_j} \phi(p_a, p_b), n_i \leq n_j \end{cases} \quad (5)$$

### 4.2 Gindex for Efficient Trajectory K-anonymization

The best set of k-clusters should minimize the overall dataset-level merge loss. Hence, we formalize the optimal trajectory k-anonymization problem as follows:

$$\arg \min_{X(i,j)} \sum_{i=1}^N \sum_{j=1}^N Loss(i, j) \cdot X(i, j)$$

$$s.t. \begin{cases} \forall i \forall j, X(i, j) \in \{0, 1\} \\ \forall i \sum_{j=1}^N X(i, j) = k \\ \forall j \sum_{i=1}^N X(i, j) = k \end{cases}$$

Here,  $Loss$  is the merge cost matrix calculated using Eq. 5 with  $Loss(i, j) = Loss(j, i)$  and  $Loss(i, i) = 0$ .  $X(i, j) = 1$  if trajectories  $\tau_i$  and  $\tau_j$  belong to the same  $k$ -cluster;  $X(i, j) = 0$  otherwise. Apparently,  $X(i, j) = X(j, i)$  and  $X(i, i) = 1$ . More importantly, each cluster contains  $k$  trajectories, namely  $\sum_{i=1}^N X(i, j) = k$  and  $\sum_{j=1}^N X(i, j) = k$ . We aim to find the best  $X(i, j)$  minimizing the overall merge cost. However, this is a typical NP-hard problem [17], and we resort to a greedy algorithm instead. In particular, we iteratively choose one unclustered trajectory, and calculate its merge loss with all the remaining candidates to find the top- $k$  with the minimum cost. The time complexity of this greedy algorithm is  $O(N^2n^2)$  where  $N$  is the total number of trajectories in the dataset and  $n$  is the average trajectory length. As observed, such method is still quite inefficient: 1) The trajectory-level merge loss is aggregated from the point-level cost, as in Eq. 5, with a time complexity of  $O(n^2)$ . Such inevitable point-by-point calculation is time-consuming especially when the trajectory length  $n$  is large. 2) Real-world trajectory datasets are usually large-scale, containing millions of objects/trajectories. Hence, computing the pairwise trajectory-level merge cost  $O(N^2)$  would take extensive time, making existing  $k$ -anonymization models [7, 19] and the above greed algorithm infeasible in practice. Therefore, in this work, we introduce an efficient solution *Gindex* for trajectory  $k$ -anonymization.

**Gindex.** Trajectories have the “locality” property, i.e., moving objects usually transit within certain areas. It is naturally unnecessary to compare trajectories that are far apart, since it would incur a huge loss of spatial resolution. In other words, the more geographically closer two trajectories are, the smaller the merge loss. Hence, we introduce a grid-based spatial encoding to organize trajectories, and define trajectory similarity as the extent of grid matching to efficiently filter out unpromising candidates during trajectory  $k$ -clustering. Specially, we convert each trajectory from a point sequence  $\tau = \langle p_1, p_2, \dots, p_n \rangle$  into a grid sequence  $G_\tau = \langle g_1, g_2, \dots, g_m \rangle$  by grouping points into grids ( $m \ll n$  in practice).

**Definition 4 (Grid Match).** Given two trajectories  $\tau_i$  and  $\tau_j$  with their grid sequences  $G_{\tau_i}$  and  $G_{\tau_j}$  (say  $m_i \leq m_j$ ),  $\tau_i$  and  $\tau_j$  are grid-matched if  $G_{\tau_i} \subseteq G_{\tau_j}$ .

We define  $Sim(\tau_i, \tau_j) = 1$  if  $\tau_i$  and  $\tau_j$  are grid-matched; Otherwise,  $Sim(\tau_i, \tau_j) = 0$ . With a high probability, such grid-matched trajectories are the spatially closest, leading to the smallest merge cost. As shown in Eqs. 1–3, merge loss is caused by spatial stretch when merging two points. If two trajectories are grid-matched (i.e., all pairs of merged points lie in the same grid), the point-level



merge loss in each grid is bounded by  $2d$  where  $d$  denotes the grid size. Note that  $k$ -anonymization must guarantee that each  $k$ -cluster contains at least  $k$  objects/trajectories, while we possibly cannot find  $k$  grid-matched trajectories within one search. This can be addressed using a hierarchical grid structure. As illustrated in Fig. 1(a), we construct a four-level grid with increasing sizes and convert trajectories into their grid sequences  $G_{\tau}^l$  at each level  $l$ . Although trajectories  $\tau_1$  and  $\tau_2$  cannot be matched at levels B or C, a successful match is identified at higher level A, generating enough candidates for the  $k$ -cluster (say  $k = 2$ ). Therefore, our basic *Gindex* algorithm runs in a bottom-up manner: we start searching for grid-matched trajectories at the base level to minimize merge loss, and repeat at higher levels until  $k$  trajectories are identified in a cluster. This algorithm takes  $O(\frac{N^2 m^2}{k})$  time for  $k$ -clustering where  $N$  is the total number of trajectories in the dataset and  $m$  is the average length of grid sequences.

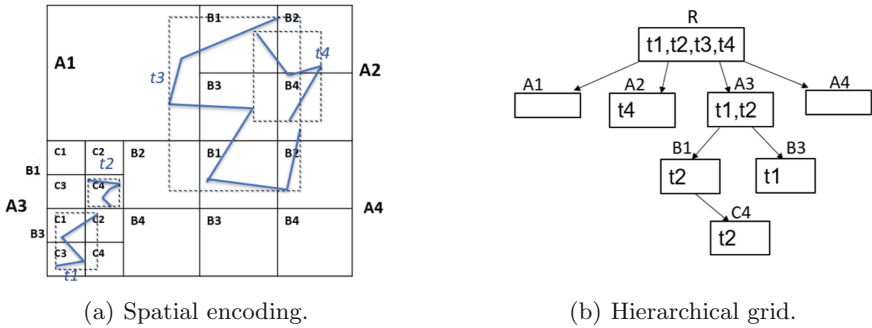


Fig. 1. Gindex.

The basic *Gindex* checks all the remaining trajectories at each level while most of them can only be matched at higher levels. Hence, we redesign *Gindex* as a top-down search, avoiding unnecessary checking of unpromising candidates.

**Definition 5 (Trajectory Bounding Grid).** The bounding grid  $BG_{\tau}$  of a trajectory  $\tau$  is the lowest level of grid that covers the whole trajectory, i.e.,  $\forall p \in BG_{\tau}$  where  $p \in \tau$  or  $\forall g \in BG_{\tau}$  where  $g \in G_{\tau}$ .  $BG_{\tau.l}$  denotes the grid level.

Figure 1(b) illustrates the hierarchical grid index used to support the top-down search, where each grid  $g$  stores a list of trajectories  $T_g$  whose bounding grids are covered by  $g$ . For example,  $BG_{\tau_1} = R.A_3.B_3$ , and thus  $\tau_1$  is contained in all the three grids  $R, R.A_3$  and  $R.A_3.B_3$ .

**Lemma 1.** Given trajectories  $\tau_i$  and  $\tau_j$  with their bounding grids  $BG_{\tau_i}$  and  $BG_{\tau_j}$  (say  $BG_{\tau_i.l} \leq BG_{\tau_j.l}$ ), they can only be matched at level  $l > BG_{\tau_j.l}$  if  $BG_{\tau_i} \notin BG_{\tau_j}$ .

*Proof.* Suppose  $\tau_i$  and  $\tau_j$  can be matched at level  $l \leq BG_{\tau_j}.l$ , i.e.,  $G_{\tau_i}^l \subseteq G_{\tau_j}^l$ . Based on Definition 5,  $BG_{\tau_i}$  covers all the grids in  $G_{\tau_i}^l$  and  $BG_{\tau_j}$  covers all the grids in  $G_{\tau_j}^l$ . Hence,  $G_{\tau_i}^l \subseteq G_{\tau_j}^l$  also indicates that  $BG_{\tau_j}$  covers  $BG_{\tau_i}$  or  $BG_{\tau_i} \in BG_{\tau_j}$ . This causes a contradiction.

Consider trajectories  $\tau_1$  and  $\tau_2$  in Fig. 1(b) where  $BG_{\tau_1} = R.A_3.B_3$  and  $BG_{\tau_2} = R.A_3.B_1.C_4$ .  $\tau_1$  and  $\tau_2$  can only be matched at level  $l = A$  which is higher than  $BG_{\tau_1}.l = B$  and  $BG_{\tau_2}.l = C$ .

**Theorem 1.** *Given a trajectory  $\tau$  and the lowest grid  $g$  in the hierarchical grid index satisfying  $\tau \in T_g$  and  $|T_g| \geq k$ , we can find the best  $k$ -cluster for  $\tau$  in  $T_g$ .*

---

**Algorithm 1.** Gindex

---

**Input:** Original dataset  $D$ , anonymity level  $k$ , hierarchical grid level  $l$   
**Output:** Anonymized dataset  $D'$

- 1:  $G = Index(D, l);$  ▷ Construct the hierarchical grid index
- 2: **for** each  $\tau_i \in D$  **do**
- 3:      $T_k = \{\tau_i\}; D = D - \{\tau_i\};$
- 4:      $g = G.root; \hat{g} = g.child(\tau_i);$  ▷  $\hat{g}$  is  $g$ 's sub-grid containing  $\tau_i$
- 5:     **while**  $|T_{\hat{g}}| \geq k$  **do** ▷ Enough candidates in  $\hat{g}$
- 6:          $g = \hat{g}; \hat{g} = g.child(\tau_i);$
- 7:      $\hat{l} = l;$  ▷ Start grid-matching from the base level
- 8:     **while**  $|T_k| < k$  **do**
- 9:         **for** each  $\tau_j \in T_g$  **do**
- 10:             **if**  $Match(\tau_i, \tau_j, \hat{l})$  **then** ▷  $\tau_i$  and  $\tau_j$  are grid-matched
- 11:                  $T_k = T_k \cup \{\tau_j\}; D = D - \{\tau_j\};$
- 12:              $\hat{l} = \hat{l} - 1;$  ▷ Continue grid-matching at higher level
- 13:              $G = G - T_k;$  ▷ Update the grid index
- 14:              $\tau'_i = Merge(T_k);$  ▷ Generate merged trajectory for  $k$ -cluster
- 15:              $D' = D' \cup \{\tau'_i\};$
- 16: **return**  $D'$ ;

---

*Proof.* Recall that we aim to find the  $k$  grid-matched trajectories at the lowest possible level so as to minimize merge loss. Based on Lemma 1, trajectories in all the neighbouring grids  $\hat{g} \notin g$  can only match  $\tau$  at level  $l > g.l$  while trajectories in  $g$  can match  $\tau$  at level  $l \leq g.l$ . Hence, the top- $k$  matched candidates for  $\tau$  can be found in  $T_g$ .

Based on Theorem 1, for  $\tau_1$  in Fig. 1(b), we can obtain the best grid-matched trajectories  $\tau_1$  and  $\tau_2$  (say  $k = 2$ ) in grid  $R.A_3$ , and ignore checking all the other branches  $R.A_1, R.A_2$ , and  $R.A_4$ .

Algorithm 1 explains the overall process of *Gindex* where we iteratively construct the  $k$ -cluster for each unclustered trajectory  $\tau_i$  (lines 2–12) and generate the anonymized trajectory  $\tau'_i$  by merging trajectories in that  $k$ -cluster (lines 14–15). Given trajectory  $\tau_i$ , we first conduct a top-down search in the hierarchical

grid index  $G$  to find the lowest grid  $g$  containing  $\tau_i$  with  $|T_g| \geq k$  (lines 4–6, time complexity  $O(\log N)$ ), and then check each trajectory  $\tau_j \in T_g$  to find the  $k$ -cluster for  $\tau_i$  (lines 7–12, time complexity  $O(km^2)$ ). In particular, we start the grid-matching from the base level (line 7). When  $\tau_j$  matches  $\tau_i$ , we add it into the  $k$ -cluster  $T_k$  and remove it from the original dataset  $D$  to avoid re-checking (lines 9–11). We then continue searching at the higher level (line 12) if  $k$ -anonymity cannot be satisfied at the current level (i.e.,  $|T_k| < k$ , line 8). Finally, the grid index is updated to remove the clustered trajectories  $T_k$  (line 13).  $M = \lfloor N/k \rfloor$  anonymized trajectories can be generated using *Gindex*, where  $N$  is the number of trajectories in the original dataset (the last  $k$ -cluster may contain more than  $k$  trajectories). The overall time complexity of Algorithm 1 is  $O(\frac{N \log N}{k} + Nm^2)$  for  $k$ -clustering and  $O(Nn^2)$  for trajectory merging.

**Efficient Grid Match.** One important step in Algorithm 1 is  $\text{Match}(\tau_i, \tau_j, \hat{l})$  which checks whether two trajectories  $\tau_i$  and  $\tau_j$  grid-match each other at level  $\hat{l}$  (i.e.,  $G_{\tau_i} \subseteq G_{\tau_j}$  if  $m_i \leq m_j$ ). As a straightforward solution, we check the whole  $G_{\tau_j}$  for each  $g^j \in G_{\tau_i}$  to see whether a matched grid can be found. This incurs  $O(m^2)$  time complexity where  $m$  denotes the average length of grid sequences. In the following, we introduce several techniques to further improve the efficiency of grid matching, as shown in Algorithm 2.

*Z-ordering.* The time cost of  $\text{Match}(\tau_i, \tau_j, \hat{l})$  can be reduced by introducing an overall ordering of all grid cells. We consider Z-ordering [3] in this work: 1) It preserves data “locality”, i.e., spatially adjacent grid cells are also close in the Z-ordering; 2) The Z-order of upper-level grid cells can be obtained directly from the Z-order of lower-level grids. In particular, Z-ordering is a widely-used space filling curve which recursively partitions each unit square in the data space

---

**Algorithm 2.** Grid Match

---

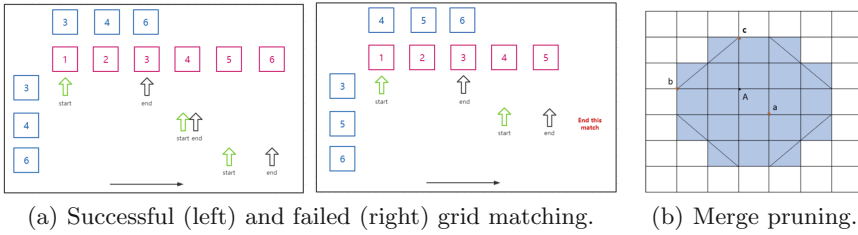
**Input:** Trajectories  $\tau_i$  and  $\tau_j$ , current grid level  $\hat{l}$   
**Output:** Match result *match*

```

1:  $G_{\tau_i} = \text{GetGridSeq}(\tau_i, \hat{l});$                                 ▷ Get Z-ordered grid sequence at level  $\hat{l}$ 
2:  $G_{\tau_j} = \text{GetGridSeq}(\tau_j, \hat{l});$                                 ▷ Get Z-ordered grid sequence at level  $\hat{l}$ 
3:  $match = true; a = b = 1;$ 
4: while  $a \leq m_i$  do
5:   while  $b \leq m_j$  do
6:     if  $g_a^i > g_b^j$  then
7:        $b++;$ 
8:     if  $g_a^i = g_b^j$  then
9:        $a++; b++;$  break;
10:    if  $g_a^i < g_b^j$  then
11:       $match = false;$  break;
12:    if  $m_i - a > m_j - b$  then
13:       $match = false;$  break;
14: return  $match;$ 

```

---



**Fig. 2.** Gindex improvement.

into 4 quadrants of equal size and orders the quadrants in a Z-shaped curve. For each trajectory  $\tau \in D$ , we first transform it into a grid sequence at the base level  $l$  and then sort it in ascending order of the corresponding Z-order values, denoted as  $G_\tau = \{g_1, g_2, \dots, g_m\}$ . The upper-level grid sequence can be obtained directly by dividing each Z-order value by 4, i.e.,  $g_i = \lfloor \frac{g_i}{4} \rfloor, \forall g_i \in G_\tau$ . Given any two trajectories  $\tau_i$  and  $\tau_j$  with their Z-ordered grid sequences  $G_{\tau_i} = \{g_1^i, g_2^i, \dots, g_{m_i}^i\}$  and  $G_{\tau_j} = \{g_1^j, g_2^j, \dots, g_{m_j}^j\}$  (say  $m_i \leq m_j$ ), we check grid-matching efficiently using a linear scan on the grid sequences instead of a pairwise grid comparison (lines 3–11 in Algorithm 2). In particular, we use two cursors  $a$  and  $b$  to enumerate  $G_{\tau_i}$  and  $G_{\tau_j}$  respectively, trying to match between grids  $g_a^i \in G_{\tau_i}$  and  $g_b^j \in G_{\tau_j}$ . Once a grid match is found, the enumeration continues from the current position rather than restart from the beginning of the grid sequence. This reduces the time complexity of  $\text{Match}(\tau_i, \tau_j, \hat{l})$  from  $O(m^2)$  to  $O(m)$ , and the overall cost of k-clustering in Algorithm 1 is  $O(\frac{N \log N}{k} + Nm)$ .

*Length-Based Pruning.* Assume  $m_i \leq m_j$  and the current cursor positions of  $G_{\tau_i}$  and  $G_{\tau_j}$  are at  $a$  and  $b$  respectively. We use  $G_{\tau_i}^{s \rightarrow e}$  to denote the sub-sequence from  $s$  to  $e$ , i.e.,  $G_{\tau_i}^{s \rightarrow e} = \{g_s, g_{s+1}, \dots, g_{e-1}, g_e\}$ . It is worth noting that  $G_{\tau_i}^{a \rightarrow m_i}$  and  $G_{\tau_j}^{b \rightarrow m_j}$  cannot be grid-matched if  $m_i - a > m_j - b$ , which further indicates  $G_{\tau_i} \not\subseteq G_{\tau_j}$ . Figure 2(a) demonstrates the examples of successful and failed grid matching between  $G_{\tau_i}$  (blue sequence) and  $G_{\tau_j}$  (red sequence). When  $a = 3$  and  $b = 5$  in the left example, it is possible to find a successful grid matching from  $G_{\tau_i}^{3 \rightarrow 3} = \{6\}$  to  $G_{\tau_j}^{5 \rightarrow 6} = \{5, 6\}$ . However, when  $a = 1$  and  $b = 4$  in the right example,  $G_{\tau_i}^{1 \rightarrow 3} = \{4, 5, 6\}$  cannot be matched to  $G_{\tau_j}^{4 \rightarrow 5} = \{4, 5\}$  as  $m_i - a > m_j - b$ . Such length-based information can be adopted to support early-stop in the grid matching process (lines 12–13 in Algorithm 2).

**Efficient Trajectory Merge.** Merging trajectories will inevitably lead to the loss of data resolution, causing utility loss. Hence, given trajectories  $\tau_i$  and  $\tau_j$  in a k-cluster  $T_k$ , we aim to merge each point  $p_a \in \tau_i$  to the “closest” point  $p_b \in \tau_j$  with the minimum merge loss defined in Eqs. 1–3. This takes  $O(n^2)$  time where  $n$  is the average trajectory length. Naturally, the grid index can also help to utilize point “locality” and filter out unpromising candidates when searching for the nearest neighbour of each point. In particular, we construct a grid index

at the base level  $l$  with grid size  $d$ . For each trajectory  $\tau$ , grid  $g_\tau$  stores all the covered points, i.e.,  $g_\tau = \{p | p \in \tau \wedge p \in g\}$ .

**Theorem 2.** *Given two trajectories  $\tau_i$  and  $\tau_j$  and a point  $p_a \in \tau_i$  in grid  $g$ , if  $g_{\tau_j} \neq \emptyset$ , then  $p_a$  can be matched to  $p_b \in Neighbour(g)$  with the minimum merge loss.  $Neighbour(g)$  are the grids covered by the rhombus area centred at  $g$  as illustrated in Fig. 2(b).*

*Proof.* Based on Eqs. 1–3, the merge loss between points  $p_a$  and  $p_b$  can be reformulated as  $\phi(p_a, p_b) = \Delta_x + \Delta_y$  where  $\Delta_x = |x_a - x_b|$  and  $\Delta_y = |y_a - y_b|$ . We consider three scenarios: 1)  $\forall p_b \in g, \phi(p_a, p_b) \leq 2d$ ; 2) If  $p_b \in Neighbour(g)$ , then it is also possible that  $\phi(p_a, p_b) \leq 2d$ ; 3)  $\forall p_b \notin Neighbour(g), \phi(p_a, p_b) > 2d$ . Since  $g_{\tau_j} \neq \emptyset$  (i.e.,  $\exists p_b \in g$ ), we can find at least one point  $p_b$  with  $\phi(p_a, p_b) \leq 2d$ . In other words, it is safe to skipping checking all the points outside  $Neighbour(g)$ .

Algorithm 3 shows our approach to trajectory merging. Given each point  $p_a \in \tau_i$  in grid  $g$ , we first check whether  $g$  contains points from  $\tau_j$ . If  $g_{\tau_j} \neq \emptyset$ , we only consider candidate points covered by  $Neighbour(g)$  based on Theorem 2; Otherwise, all the points from  $\tau_j$  will be checked (lines 3–6). We then calculate the point-wise merge loss  $\phi(p_a, p_b)$  for each  $p_b \in P$  and obtain  $p_b$  with the minimum loss (lines 7–11). Finally,  $p_a$  and  $p_b$  are merged into the anonymized trajectory  $\tau'$ . Although the worst case time complexity of Algorithm 3 is still  $O(n^2)$ , a large amount of unpromising points can be pruned from the pairwise checking in practice.

---

**Algorithm 3.** Trajectory Merge

---

```

Input: Trajectories  $\tau_i$  and  $\tau_j$  ( $n_i \geq n_j$ )
Output: Merged trajectory  $\tau'$ 
1:  $\tau' = \emptyset$ 
2: for each  $p_a \in \tau_i$  do
3:    $P = \text{GetPoints}(\tau_j)$ ;
4:   if  $g_{\tau_j} \neq \emptyset$  then                                      $\triangleright p_a \in g$  and  $g$  contains points from  $\tau_j$ 
5:      $Grids = \text{GetNeighbour}(g)$ ;                                $\triangleright$  Get the surrounding grids of  $g$ 
6:      $P = \text{GetPoints}(Grids, \tau_j)$ ;                              $\triangleright$  Get  $\tau_j$ 's points covered in  $Grids$ 
7:      $min = +\infty$ 
8:     for each  $p \in P$  do
9:       if  $\phi(p_a, p) < min$  then                                  $\triangleright$  Point-level merge loss
10:         $min = \phi(p_a, p)$ ;
11:         $p_b = p$ ;
12:         $p' = \text{merge}(p_a, p_b)$ ;
13:         $\tau' = \tau' \cup \{p'\}$ ;
14: return  $\tau'$ ;

```

---

## 5 Experiments

**Dataset.** We adopt a widely-used public dataset, *T-Drive* [20], to systematically compare the trajectory protection models. T-Drive was generated by 10,357 taxis during the period of 2–8 February 2008 within Beijing, China. There are 94,177 raw trajectories consisting of 15 million GPS points. On average, the sampling rate is around 3.1 min per point and the Euclidean distance between two continuous points is about 600 m. All the trajectories for each taxi is concatenated into a single trajectory.

**Compared Methods.** We compare our *Gindex* algorithm with the current state-of-the-art trajectory k-anonymization models: W4M [2], GLOVE [7] and KLT [19] discussed in Sect. 2. By default, we set  $N = |D| = 1000$ ,  $k = 5$  and base grid size =  $64 \times 64$ . All the algorithms are implemented in Java, and evaluated on a server with two Intel(R) Xeon(R) CPU E5-2630, 10 cores/20 threads at 2.2 GHz each, 378 GB memory, and Ubuntu 16.04 operating system.

**Evaluation Metrics.** We evaluate our proposal from various performance criteria including privacy protection, utility preservation and time cost.

- Privacy Metrics: Although k-anonymity models can provide theoretical privacy guarantee with parameter  $k$ , we still need to verify their performance against real attacks. We use the state-of-the-art re-identification algorithm [11, 12] to evaluate the linking accuracy ( $LA$ ), i.e., whether the anonymized dataset can successfully link back to the original one. Higher  $LA$  indicates less privacy protection.
- Utility Metrics: We apply two typical metrics to evaluate utility preservation: diameter error ( $DE$ ) and trip error ( $TE$ ) [8].  $DE$  (resp.  $TE$ ) measures the difference of diameter (resp. trip or origin-destination regions) distribution between the original and anonymized trajectory datasets by Jensen-Shannon divergence. Higher  $DE/TE$  indicates less utility preservation.

### 5.1 Results and Analysis

**Impact of Anonymization Level  $k$ .** Figure 3(a)–3(d) report the time cost (seconds in log scale), privacy protection and utility loss respectively of the compared privacy models when  $k$  ranges in  $\{2, 4, 6, 8, 10\}$ . Formally, larger  $k$  provides a better privacy guarantee in k-anonymization. Overall, it can be observed that  $LA$  drops with the increase of  $k$  while  $DE$  and  $TE$  increases with  $k$ , which demonstrates a trade-off between privacy protection and utility preservation. In particular, W4M is the most robust against  $k$ , achieving the worst anonymization ( $LA \approx 0.85$ ) with the smallest utility loss ( $DE \approx 0.05$  and  $TE \approx 0.33$ ). This is because W4M only slightly modifies trajectories in a k-cluster to make them similar, and hence many original points stay unchanged in the anonymized dataset. Among the three merging-based k-anonymization models, *Gindex* dramatically outperforms GLOVE and KLT in terms of  $LA$  (especially when  $k < 6$ ), with a quite similar utility loss in both  $DE$  and  $TE$ . This showcases the superiority of

our proposed *Gindex* model. More importantly, *Gindex* achieves a significant efficiency improvement over GLOVE and KLT by several orders of magnitude, and is also much faster than the very simple W4M model, as shown in Fig. 3(a). In particular, the computational time of W4M, GLOVE and KLT rises slightly when  $k$  increases. On the contrary, *Gindex* becomes more efficient with larger  $k$ , which is consistent with our theoretical analysis in Sect. 4.2 that the time complexity of *Gindex* is reduced to  $O(\frac{N \log N}{k} + Nm + Nn^2)$ , where  $O(\frac{N \log N}{k})$  for the top-down search in the hierarchical grid index is inversely proportional to the anonymization level  $k$ .

**Scalability and Robustness.** We also evaluate the scalability and robustness of  $k$ -anonymization models by varying the dataset size  $|D|$  (i.e., the total number of objects/trajectories) in the range  $\{500, 1000, 2000, 4000, 6000, 8000, 10000\}$ , as demonstrated in Fig. 3(e)–3(h). For GLOVE and KLT, we only report their performance when  $k \leq 2000$  since it would take more than 2 days to anonymize 2000+ objects/trajectories using GLOVE or KLT. As expected, the running

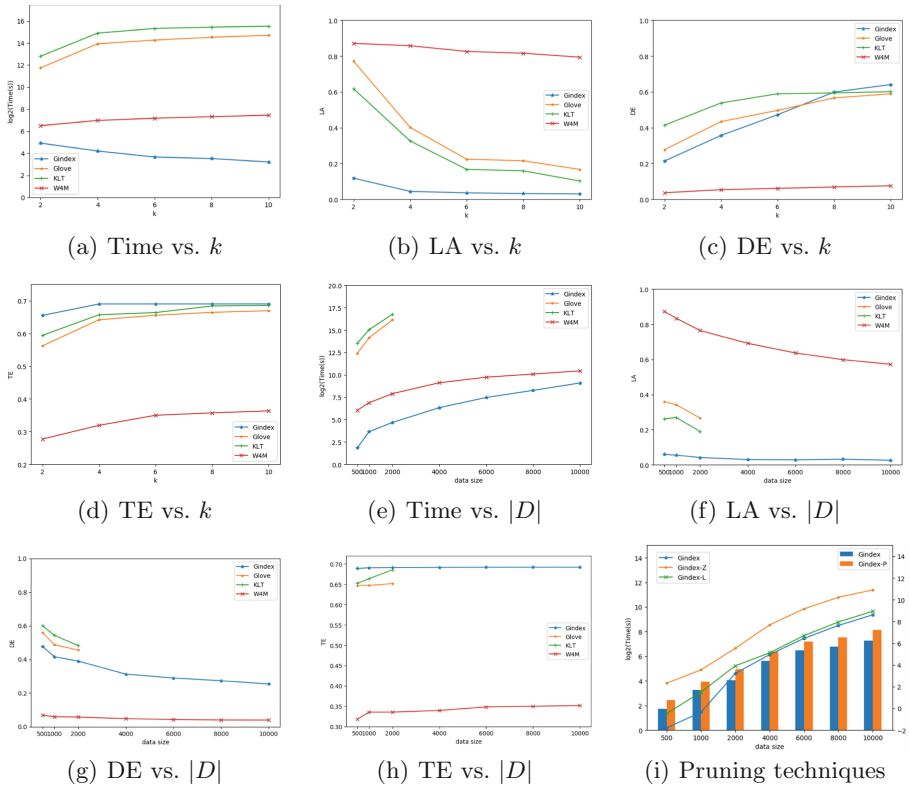


Fig. 3. Experimental results.

time of all these models increases with dataset size  $|D|$ , and such upsurge is more obvious in GLOVE and KLT which are extremely time-consuming and cannot be applied to large-scale datasets. Our *Gindex* model is feasible in practice, taking less than 10 min to anonymize 10,000 taxis with one-week trajectories. This verifies the effectiveness of our grid-based encoding and index mechanism, since it helps to avoid the laborious pairwise trajectory/point comparison and only reserves the promising candidates for trajectory k-clustering/merging. Furthermore, our *Gindex* algorithm is more efficient than W4M on all data scales, and meanwhile achieves a much better privacy protection. As depicted in Fig. 3(f), the objects are hardly re-identifiable when anonymized using *Gindex* ( $LA < 0.05$ ) while the linking accuracy of W4M reaches  $[0.6, 0.9]$ . In other words, 60%–90% taxis can still be identified from the dataset even when W4M anonymization is applied. This is because the trajectory generalization/merging used in *Gindex* makes the  $k$  objects in a  $k$ -cluster indistinguishable, which is quite effective in preventing linking attacks, while W4M still preserves similar trajectory for each of those objects. Given the better privacy protection in larger datasets (i.e.,  $LA$  drops with the increase of  $|D|$  in Fig. 3(f)), higher utility loss would naturally be expected. It is true with  $TE$  shown in Fig. 3(h), which slightly increases with  $|D|$ . In particular, *Gindex* is quite stable and robust with almost no change in  $TE$  when dealing with larger datasets. However,  $DE$  decreases when  $|D|$  rises, indicating better utility preservation. Although the generalization-based  $k$ -anonymization models (i.e., GLOVE, KLT and *Gindex*) merge trajectories in each  $k$ -cluster, the distortion of diameter (i.e., spatial coverage) would become smaller when more objects/trajectories exist in the original dataset.

**Effectiveness of Pruning Techniques.** Finally, we conduct an ablation study to investigate the impact of each pruning technique introduced in Sect. 4.2 for  $k$ -clustering and trajectory merging. In particular, we compare *Gindex*'s  $k$ -clustering time with two variations *Gindex*-Z and *Gindex*-L by removing Z-ordering and length-based pruning respectively from the *Gindex* algorithm, and compare *Gindex*'s merging time against *Gindex*-P which ignores the grid-based pruning during trajectory merging. According to Fig. 3(i) (in log scale), both pruning strategies are quite effective for filtering out unpromising trajectory candidates in the  $k$ -clustering process, and the time difference becomes larger when the dataset size  $|D|$  increases. Specifically, *Gindex*-Z is the most time-consuming among the three models ( $k$ -clustering in *Gindex*-Z takes 1212 s while *Gindex* takes 391 s when  $|D| = 10,000$ ), which indicates the importance of Z-ordering in candidate pruning. Theoretically, Z-ordering also reduces the  $k$ -clustering time complexity of *Gindex* from  $O(\frac{N \log N}{k} + Nm^2)$  to  $O(\frac{N \log N}{k} + Nm)$  where  $N = |D|$  and  $m$  is the average length of grid sequences, conforming to our experimental results in Fig. 3(i). Although the length-based pruning (*Gindex*-L) is less effective compared with Z-ordering (*Gindex*-Z), it can still help to reduce unnecessary comparisons in the trajectory matching process, as demonstrated in Fig. 3(i) (*Gindex*-L takes 488 s when  $|D| = 10,000$ , surpassing *Gindex* by 97 s). Furthermore, trajectory merging with index optimization is obviously more efficient than



that without pruning (*Gindex* takes 154 s for merging while *Gindex-P* takes 276 s when  $|D| = 10,000$ ). This is because the index-based pruning successfully limits the search space for nearest neighbour (with the minimum merge cost) from the entire trajectory to the surrounding grids  $Neighbour(g)$ , although the time complexity for merging is still  $O(Nn^2)$ .

## 6 Conclusion

In this work, we propose an efficient *Gindex* model for trajectory k-anonymization. We design a hierarchical grid-based spatial encoding and indexing mechanism to speed up trajectory k-clustering and merging, and introduce various optimization techniques, including z-ordering, length-based pruning and merge pruning, to further improve the performance of *Gindex*. The overall time complexity is reduced to  $O(\frac{N \log N}{k} + Nm)$  for k-clustering and  $O(Nn^2)$  for trajectory merging. Extensive experiments on the widely-used T-drive dataset demonstrate the superiority of *Gindex*, outperforming state-of-the-art k-anonymity models by several orders of magnitude.

**Acknowledgments.** This work was partially supported by the National Natural Science Foundation of China (NSFC62072125) and the Australian Research Council (DP200103650 and LP180100018).




## References

1. Abul, O., Bonchi, F., Nanni, M.: Never walk alone: Uncertainty for anonymity in moving objects databases. In: ICDE, pp. 376–385 (2008)
2. Abul, O., Bonchi, F., Nanni, M.: Anonymization of moving objects databases by clustering and perturbation. *Inf. Syst.* **35**(8), 884–910 (2010)
3. Böxhm, C., Klump, G., Kriegel, H.P.: XZ-ordering: a space-filling curve for objects with spatial extension. In: International Symposium on Spatial Databases, pp. 75–90 (1999)
4. De Montjoye, Y.A., Hidalgo, C.A., Verleysen, M., Blondel, V.D.: Unique in the crowd: the privacy bounds of human mobility. *Sci. Rep.* **3**, 1376 (2013)
5. Dwork, C.: Differential privacy: a survey of results. In: International Conference on Theory and Applications of Models of Computation, pp. 1–19 (2008)
6. Fung, B.C., Wang, K., Chen, R., Yu, P.S.: Privacy-preserving data publishing: a survey of recent developments. *ACM Comput. Surv.* **42**(4), 1–53 (2010)
7. Gramaglia, M., Fiore, M.: Hiding mobile traffic fingerprints with GLOVE. In: CoNEXT, pp. 26:1–26:13 (2015)
8. Gursoy, M.E., Liu, L., Truex, S., Yu, L.: Differentially private and utility preserving publication of trajectory data. *IEEE Trans. Mob. Comput.* **18**(10), 2315–2329 (2019)
9. He, X., Cormode, G., Machanavajjhala, A., Procopiuc, C.M., Srivastava, D.: DPT: differentially private trajectory synthesis using hierarchical reference systems. *PVLDB* **8**(11), 1154–1165 (2015)
10. Jin, F., Hua, W., Francia, M., Chao, P., Orlowska, M., Zhou, X.: A survey and experimental study on privacy-preserving trajectory data publishing (2021)

11. Jin, F., Hua, W., Xu, J., Zhou, X.: Moving object linking based on historical trace. In: ICDE, pp. 1058–1069 (2019)
12. Jin, F., et al.: Trajectory-based spatiotemporal entity linking. TKDE (2020)
13. Li, N., Li, T., Venkatasubramanian, S.: t-Closeness: privacy beyond k-anonymity and l-diversity. In: ICDE, pp. 106–115 (2007)
14. Liu, X., Chen, J., Xia, X., Zong, C., Zhu, R., Li, J.: Dummy-based trajectory privacy protection against exposure location attacks. In: Ni, W., Wang, X., Song, W., Li, Y. (eds.) WISA 2019. LNCS, vol. 11817, pp. 368–381. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-30952-7\\_37](https://doi.org/10.1007/978-3-030-30952-7_37)
15. Liu, X., Zhao, H., Pan, M., Yue, H., Li, X., Fang, Y.: Traffic-aware multiple mix zone placement for protecting location privacy. In: INFOCOM, pp. 972–980 (2012)
16. Machanavajjhala, A., Kifer, D., Gehrke, J., Venkatasubramanian, M.: l-diversity: privacy beyond k-anonymity. TKDD **1**(1), 3-es (2007)
17. Meyerson, A., Williams, R.: On the complexity of optimal k-anonymity. In: SIGMOD, pp. 223–228 (2004)
18. Sweeney, L.: K-anonymity: a model for protecting privacy. Int. J. Uncertain. Fuzziness Knowl. Based Syst. **10**(5), 557–570 (2002)
19. Tu, Z., Zhao, K., Xu, F., Li, Y., Su, L., Jin, D.: Protecting trajectory from semantic attack considering k-anonymity, l-diversity, and t-closeness. IEEE Trans. Netw. Serv. Manage. **16**(1), 264–278 (2019)
20. Yuan, J., et al.: T-drive: driving directions based on taxi trajectories. In: SIGSPATIAL, pp. 99–108 (2010)
21. Zhu, T., Li, G., Zhou, W., Philip, S.Y.: Differentially private data publishing and analysis: a survey. TKDE **29**(8), 1619–1638 (2017)



# Developing a Deep Learning Based Approach for Anomalies Detection from EEG Data

Ashik Mostafa Alvi<sup>(✉)</sup> , Siuly Siuly , and Hua Wang 

Victoria University, Melbourne, VIC, Australia  
ashik.alvi@live.vu.edu.au, {siuly.siuly,hua.wang}@vu.edu.au

**Abstract.** Electroencephalography (EEG) contribute a leading role in brain studies, mental and brain diseases and disorders diagnosis, and treatments. Traditional Machine Learning (TML) approaches are employed in most of the recent efforts in identifying the anomalies from EEG data. But their shallowed architecture is one of the reasons why they fail to detect correctly and efficiently. Furthermore, these systems need to be fed the discriminant features manually. To overcome these issues, this study aims to develop an EEG data analysis system involving a multi-layer Gated Recurrent Unit (GRU) for anomalies detection. There are four steps to the suggested framework: (1) Collecting Raw EEG Data, (2) Data pre-processing (de-noising, segmenting, and down-sampling), (3) discover hidden significant characteristics of EEG data and classification using GRU based scheme, and (4) model's performance evaluation. Our proposed model is tested on a publicly available EEG dataset and achieved 96.91% of accuracy, 97.95% of sensitivity, 96.16% of specificity and 96.39% of F1 score. This study will guide the future bio-medical researchers and technology experts to have a deep learning based automated anomaly detection system from EEG data.

**Keywords:** Gated Recurrent Unit · EEG · Deep learning · Data mining · Mild cognitive impairment

## 1 Introduction

Artificial intelligence (AI) has received a lot of researcher's attention in recent years. AI, machine learning (ML), and deep learning (DL) appear in a slew of technology focused articles [8]. DL is allowing change and innovation in many aspects of our modern life. It is at the heart of the majority of AI advancements reported in recent tech news [20]. Different AI fields, for example genomics [33], graph theory [32], computer vision [27], natural language processing [10], cloud computing [12], sentiment analysis [11, 31], automation [3, 14], big data [15], and so on are filled with DL applications. It has placed its foot not only in the tech world, but also in agricultural [16], health care [24–26], and business [21] sectors.

Supported by Australian Research Council.

Electroencephalography is a tool to record electrical activity in the brain. Cerebral electrical potentials are measured by electrodes placed on the scalp. EEG is a non-invasive, portable, non-stationary, easy to use and interpret instrument mostly suitable for brain related abnormality detection [4].

Kashefpoor et al. [17] conducted an EEG study with a correlated based pursuit for feature extraction and employed Takagi-Sugeno neurofuzzy (NF) inference system with K-nearest neighbor (KNN). Though this study achieved 100% sensitivity, but the accuracy and specificity fall short of 88.89% and 83.33% respectively. Yin et al. [23, 30] removed the artifacts using the same Stationary Wavelet Transformation (SWT) and introduced Spectral temporal way of feature extraction. Extracted features were evaluated and selected based on 3-D evaluation algorithm. Finally, a Support Vector Machine (SVM) classifier was displayed and achieved 96.94% accuracy. Another effort [13] using the Spectral Analysis and KNN was reported and attained 81.5% accuracy. EEG classification were done using phase locking value (PLV) and achieved 95.9% accuracy. Dictionary learning had introduced in [19] for pre-processing the EEG data. K-means and singular value decomposition (K-SVD), label consistent KSVD (LC-KSVD) and correlation-based LC-KSVD (CLC-KSVD) were three classifiers tried in that study and CLC-KSVD achieved 88.9% accuracy among those three.

There were couple of DL based studies tried to work with EEG classification. Amezquita-Sanchez et al. [6] used Empirical wavelet transform also known as MUSIC-EWT for noise reduction and fractality dimension (FD) from the chaos theory for data compression. Finally, processed EEG data were fed into an enhanced probabilistic neural network (EPNN) and achieved 90.3% accuracy. A four layer convolutional neural network (CNN) was designed for EEG classification along with biological experimental set up to measure the oxygenated haemoglobin changes in [29]. The performance of this CNN model was 90.37% differentiating abnormality in the EEG signals. Chen et al. [7] reported a graph theory based EEG classification where they performed segmentation and created a brain network, connectivity matrix. After completing the pre-processing steps, CNN was brought into play for classifying EEG subjects.  $92.06 \pm 1.5$  % of the time, the CNN classifier was correct. LeCun's general LeNet CNN architecture was selected with an EEG dataset which was filtered with Fast-Fourier Transformation (FFT) to classify EEG normal subjects in [28] and performed very poorly. The accuracy of this work was just 69.23%, but having the specificity was high to 88.89%.

It can be concluded from the reviewed literature that the TML algorithms performed consistently and better compared to a few of the DL based efforts. But TML algorithms are not time effective option when it comes to feature extraction. An extra step is always required when it comes to TML algorithm based study. And having a complex nature of EEG data, it requires more attention for identifying discriminant features. The shallow architecture of the TML methods make it more time costly and hard to handle huge data like EEG.

To overcome the reported research gaps, we have introduced a Gated Recurrent Unit (GRU) based EEG classification framework. Our proposed framework

starts with reducing the noises using the Butterworth filter as it has the potential to reach more linear phase reaction and comprehensive flat response. Then, we have segmented the filtered data and down-sampled it using the Average filter to reduce the computational overhead. Average filter works fine with any dataset when there is no or minimum amount of outliers. Finally, a two layer GRU network has been designed to classify EEG data. The main efforts of this study can be summarised as follows:

- For the first time, we have reported GRU based deep learning study for EEG classification.
- We have investigated average filtering as a down-sampling approach to aid the suggested model’s computational overhead.
- Our proposed model is computationally inexpensive to design and achieve efficient and competitive retrieval results when compared to the existing DL models.

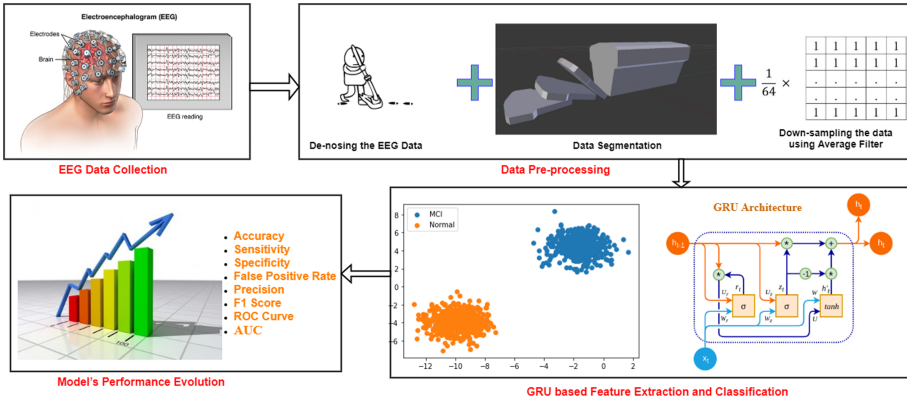
The rest of the article is sorted as follows: Sect. 2 provides a description of the proposed model framework. A detailed result and analysis is given in Sect. 3. A comparative discussion is provided in Sect. 4. Finally, this study finishes by truing to the conclusion in Sect. 5.

## 2 Proposed Framework

The deep learning based anomaly detection framework from EEG data is designed involving 4 sub modules: (1) Collecting Raw Data, (2) Data pre-processing (de-noising, segmenting, and down-sampling), (3) discover hidden significant characteristics of data and classification using GRU based scheme, and (4) model’s performance evaluation. Figure 1 illustrates the overall framework in a nutshell. We have collected a publicly available raw EEG dataset of 27 subjects. Data pre-processing has 3 inner steps: (1) De-noising the raw EEG data, (2) EEG data segmentation, and (3) down-sampling the segmented data using Average filter [2, 5]. Processed data are fed into the GRU classifier for feature extraction and anomaly detection. There are 8 standard evaluation matrixes are used to validate our proposed model’s performance. All the sub-modules are described in details below.

### 2.1 Collecting Raw EEG Data

This proposed anomaly detection framework is tested with a publicly available EEG dataset of 27 subjects where there are 16 normal healthy controls and 11 Mild Cognitive Impairment (MCI) subjects aging between 60 years to 77 years. These EEG data was collected in Sina and Nour Hospitals, Isfahan, Iran [17, 18]. All the participated subjects gave their consent and the deputy of research and technology, Isfahan University of Medical Sciences, Isfahan, Iran ethically approved these EEG data collection. Each of the subjects underwent a



**Fig. 1.** Overview of the GRU based anomaly detection framework from EEG data.

neuropsychiatric interview for MCI illness, according to Peterson’s criteria. The participants were validated using the Mini-mental state examination (MMSE) score. Subjects with an MMSE score of 21 to 26 were treated as MCI, while those with a score of more than 26 were considered as Normal. Head trauma, dementia, and a history of significant mental problems, serious medical condition, or drug abuse were all considered as exclusion criteria.

### 2.2 Data Pre-processing (De-noising, Segmenting, and Down-Sampling)

The raw EEG data often get mixed with unwanted signals and other outliers. And for an individual subject, the recording lasted for 30 min. So, we can understand the size of the data. These huge EEG data needs to be segmented for smooth processing. Our proposed pre-processing sub-module has 3 inner steps: (a) de-noising using Butterworth filter, (b) segmentation, and (c) down-sampling using Average filter to ensure the smooth feature extraction and classification.

**De-noising the Data.** Artifacts or so-called undesirable signals frequently contaminate EEG recordings. Outlier values, electrode-pops, breathing, power supply fluxes and interference (50 Hz), baseline drift, eye blinking, or muscle electrical activity, among other things, are some of the most prevalent causes of EEG recording contamination. As a result, if we want a decent classifier, we must first remove the noise from the recording.

Our proposed study engaged Butterworth filter to de-noise the unwanted signals. The Butterworth filter used in this work to exclude undesirable high frequency signals such as those caused by body movement, electricity grid inference, eye blinks, and heartbeats. It aids in the production of more linear phase reactions while also optimising flat response.

The recordings were viewed and noise was removed using the MATLAB EEGLAB [9] package. To remove artefacts, a low pass 3rd order Butterworth filter with a cut-off frequency 50Hz was employed. Each having a duration of  $60\text{s} \times 30\text{min} \times 256\text{Hz}$ . We digitised each subject's signal, which has 460800 rows and all 19 electrodes representing the columns. The recordings were saved as .mat files.

**Segmentation.** By character, EEG signals are non-stationary, non-periodic, and massive in size. We'll need a lot of computer power to analyse this big dataset of 27 people, and it will take a long time to build a model. To deal with such data type, segmentation is a good option. It also helps to increase the sample size by not losing any data as each of the segment still holds important features.

For our work, we intended to split each of the 30 min of recording and capture 6s for each segment by keeping the sampling frequency same which 256 Hz. Following the segmentation of each subject, we produced 300 additional segments from each subject's recording, all of which are clearly labelled with the appropriate subject's label (MCI/Normal). Finally, we had a total of  $27 \times 300 = 8,100$  individuals, each with 1,536 rows and 19 columns.

**Down-Sampling Using Average Filter.** To ensure an efficient deep learning model, we need to have a large dataset. But we had only 27 subjects. From these 27 EEG recordings, we created 8,100 subjects by segmenting the dataset. Now, we want to resolve the computational power issue as we want to have a cost effective model which can be trained in a regular machine.

To reduce the computational overhead, we down-sampled the data using the Average filter. Average filter works fine when the data is outlier and noise free. So, this filter fits perfectly with our dataset and helped us to reduce the data dimension. Initially, the sampling frequency 256Hz and by using the Average filter we down-sampled it 4Hz. The sliding window method was employed having the window size  $64 \times 1$  for each iteration. While keeping the number of channel constant, we run the window column wise. For each of the channel, the window ran for 24 times giving us a new row number for each of the subject. We saved the data in CSV format after performing all of the pre-processing processes in MATLAB.

### 2.3 Discover Hidden Significant Characteristics of Data and Classification Using GRU Based Scheme

The main goal of this study is to have a simple, computationally cheap deep learning based model which perform efficiently. To investigate further, we introduced a Gated Recurrent Unit (GRU) based classifier in this study to identify the hidden features and differentiate normal EEG signals from abnormal ones. To our best knowledge, this dataset has never been subjected to GRU analysis. We find GRU suitable with this EEG study due to the nature of EEG data and

the characteristics of GRU. As we know, EEG is a no-stationary huge chaotic data and it suits well with deep learning based methods. It is for suitable any recurrent network based methods. As Long Short-term Memory (LSTM) which is a variant of recurrent network has a tendency of holding the memory for long time, it is not suitable if we want a cost effective model. When it comes with GRU particularly, it becomes a perfect match as it has a forget gate inside its architecture which helps to set free the memory.

To build up our proposed GRU network, we used Jupyter notebook as an IDE and Python as the programming language as it has good support for deep learning providing libraries like Keras [22], SciPy, Pandas, NumPy, and Scikit Learn.

With the aid of the Pandas library, all pre-processed CSV files were read into the system. The NumPy library was then used to transform all of these data frames into two-dimensional arrays. At the same time, each subject's category label was stored in a separate NumPy array. Then, using the NumPy library's reshape function, we decreased the input data's dimension. With the Scikit-Learn package, the dataset was then randomly split into the training and validation sets. In our validation set, we had included 20% of the data. Finally, the Keras library with Tensor Flow backend [1] was used to construct the proposed GRU network.

Our proposed GRU network had two deep hidden layers and a dense layer. The dense layer had a single neuron and the 1st hidden layer of the GRU network had 1024 neurons and the 2nd one had 512 neurons in it. We had used 'tanh' as the activation function in both of the deep hidden layers and 'sigmoid' in the dense layer. There was a flatten layer employed just before the dense layer. 'binary\_crossentropy' had used as the loss function as it is a binary classification problem that we were trying to solve. We had picked the 'adam' optimiser for our work.

To ensure our proposed GRU network did not over-fit, we deposited early stopping by keeping a check pointer throughout the training process. Check pointer was set on the validation loss and validation accuracy having the min\_delta set to 0.001 and patience to 10. For fitting the data, we set the batch size to 300 and epoch to 100. After every check point, it sorted the best model in "hdf5" format which was later used for further model evaluation.

## 2.4 Model's Performance Evaluation

Our proposed GRU based model has been tested with eight standard evaluation matrixes. Accuracy, Sensitivity, Specificity, False Positive Rate, Precision, F1 Score, Receiver Operating Characteristic (ROC) curve, and Area under the ROC curve (AUC) are the performance measure matrixes used to justify our proposed study. The plots of Sensitivity (true positive rate) vs false positive rates are shown on the ROC curve in Fig. 4.



### 3 Result and Analysis

All the raw EEG data was stored in European Data Format (EDF). For the pre-processing steps we have used MATLAB as a tool. Using the Butterworth filter we have de-noised the raw EEG data and saved as MATLAB formatted data (.mat). After that, all of the artifact-free data was segmented, and each segment was down-sampled using the Average filter. Comma Separated Value (CSV) files were created for each of the filtered segments. Finally, these segments are inputted into the GRU network for feature extraction and classification. All the experiments are performed on a MacBook Pro with a 2.6 GHz Intel core i7 CPU, 16 GB RAM and 4 GB Graphics Card.

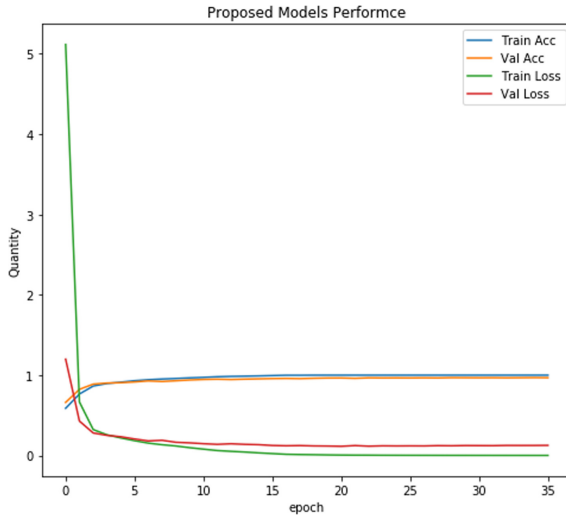


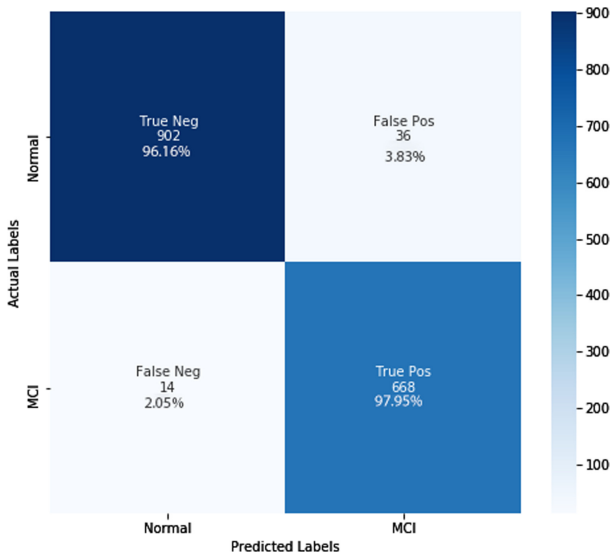
Fig. 2. Proposed model’s accuracy and Loss over epochs.

There were 100 epochs with a batch size of 300 while training the model. To ensure our model does not over-fit, we have set the early stopping on validation loss and accuracy. The min\_delta value was set to 0.001 and patience to 10. Finally, our model achieved 96.91% accuracy within only 36 epochs. Figure 2 portrays both the training and validation accuracy and loss over epochs. In the beginning of the training process, the training loss was so high and it dropped drastically just after 1st epoch. The validation loss decreased 42.82% to 12.59% over the 36 epochs. The validation accuracy improved 66.05% to 96.91% over the training process. The overall performance of our proposed model is reported in Table 1. The model has achieved 96.91% accuracy, 97.95% sensitivity, 96.16% specificity, 3.84% false positive rate, 94.89% precision, 96.39% F1 score, and 97.05% AUC value.

**Table 1.** Overall performance of the proposed GRU model.

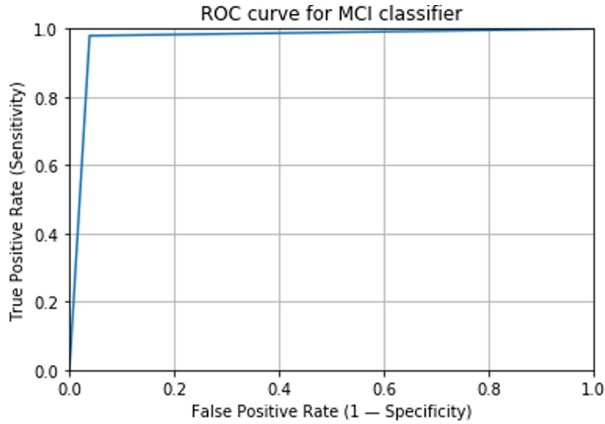
Evaluation matrixes	Predicted outcome
Accuracy	96.91%
Sensitivity	97.95%
Specificity	96.16%
False positive rate	3.84%
Precision	94.89%
F1 Score	96.39%
AUC	97.05%

Figure 3 shows the confusion matrix of our GRU based classifier. There were 938 Normal and 682 MCI testing samples to measure the performance. Among the normal testing samples 902 (96.16%) samples were correctly identified as normal subjects and 36 (3.83%) samples were misclassified as MCI. Again, among the MCI testing samples, 668 (97.95%) samples were correctly recognized as MCI and 14 (2.05%) samples were misclassified as Normal. It can be said that the misclassification rate is very minimum compared to the properly identification rate. And the false alarm rate tends to decrease if we increase the number of testing samples.



**Fig. 3.** Confusion matrix of the GRU based model.

Receiver operating characteristic (ROC) curve is a measure to recognise a classifier's performance. It demonstrates the relationship between sensitivity and specificity. The ROC curve is shown by the x-axis, which depicts  $1 - \text{specificity} = \text{FP}/(\text{FP} + \text{TN})$ , and the y-axis, which depicts  $\text{sensitivity} = \text{TP}/(\text{TP} + \text{FN})$ . The blue curve in Fig. 4 illustrates the ROC curve and the area under the ROC curve is known as the AUC value. The bigger the value, better the model is. The AUC value always stays between 0 and 1. In our case, after multiplying by 100, the AUC value we have got is 97.05% which proves the efficiency of our proposed model.



**Fig. 4.** ROC curve showing the efficiency of the proposed model.

## 4 Discussion

This study is committed to differentiate normal EEG signals from irregular EEG signals. For that purpose, we have chosen 27 EEG recordings (16 normal controls and 11 MCIs). The framework we have designed, performed really well differentiating normal EEG samples from the abnormal ones with an accuracy of 96.91%. It is observed that the false alarm rate is 9.84% and it is very low. The reason behind it could be a smaller dataset and artifacts affecting the EEG data recording. To our best knowledge, GRU model has never been set up with this dataset and it has shown promising performance. Though previous attempts to classify EEG data with TML methods had shown promising performance, but the computational expense to build the shallow architectures and feature extraction methods make it costly. An overall comparison with previous attempts including both TML and DL methods along side with our proposed method is reported in Table 2.

Previous studies having a shallow architecture failed to have cost effective and efficient model. From the literature, it can be said that very few studies have

**Table 2.** Comparison with previous efforts.

Studies	Accuracy	Sensitivity	Specificity
Kashefpoor et al. [17]	88.89%	100%	83.33%
Yin et al. [30]	96.94%	96.89%	96.99%
Hadiyoso et al. [13]	81.5%	81.82%	81.25%
Kashefpoor et al. [19]	88.9%	83.3%	100%
Chen et al. [7]	92.06%	Not reported	Not reported
G. Vrbancic and V. Podgorelec [28]	69.23%	25%	88.89%
Amezquita-Sanchez et al. [6]	90.3%	92.1%	87.9%
Yang et al. [29]	90.37%	Not reported	Not reported
<b>Proposed method</b>	<b>96.91%</b>	<b>97.95%</b>	<b>96.16%</b>

done using deep learning based models to find out the regular EEG signals out of the abnormal ones. Most of the previous attempts evolved with traditional ML algorithms like SVM, KNN, Logistic Regression (LR) etc. which do not have the power to extract the features for themselves. They need to be fed manually extracted features by different feature extraction methods. It is one of the main reason of missing important features which has impact on the classifier’s performance. And due to this extra step of feature extraction, the cost of the model increases whereas the deep learning based model can do this feature extraction by themselves. But it is often led to a non-efficient model if the sample size is too small.

Our proposed GRU model having 2 deep hidden layers and a dense layer is too simple to design. It has 1024, 512 and 1 neurons respectively. The hidden layers have ‘tanh’ whereas dense layer has ‘sigmoid’ as activation functions. With such simple design it achieve very good accuracy. It took 36 epochs to reach such efficiency and on average per epoch last for only 1120s.

## 5 Conclusion

Our proposed GRU based anomaly identification framework has proven its capability in differentiating normal EEG data. GRU has never introduced with this publicly available EEG dataset to separate the anomalies. It is a big challenge to work with smaller sample size, particularly when we use a deep learning based model as a classifier. So, we had to pre-process the raw data with some denoising, segmentation and down sampling. De-noising helped us to remove the unwanted signals from the raw data and segmentation helped us to increase the sample size. As we wanted to have a simpler model which can be trained on a regular machine with a simple configuration, we down-sampled the segmented data. This down-sample helped us to reduce the data dimension and while doing this we have to sacrifice some of the data. That is one of the main reason of not having

a perfect model. But while down-sampling, we tried our best not to lose much of the data that may cause us a not efficient model.

In the end, we can say that our proposed GRU based anomaly detection framework is a balanced model having a simpler architecture with not much cost to design and high performance in identifying normal EEG signals from the irregular one. The future study should focus on having a larger dataset and investigating this GRU architecture with different amount of hidden layers and activation functions. We believe that our study will guide the future EEG researchers a pathway towards a perfect normal EEG identification model.

## References

1. Abadi, M., et al.: TensorFlow: a system for large-scale machine learning. In: 12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16), pp. 265–283 (2016)
2. Alvi, A.M., Basher, S.F., Himel, A.H., Sikder, T., Islam, M., Rahman, R.M.: An adaptive grayscale image de-noising technique by fuzzy inference system. In: 2017 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), pp. 1301–1308. IEEE (2017)
3. Alvi, A.M., Shaon, M.F.I., Das, P.R., Mustafa, M., Bari, M.R.: Automated course management system. In: 2017 12th International Conference for Internet Technology and Secured Transactions (ICITST), pp. 161–166. IEEE (2017)
4. Alvi, A.M., Siuly, S., Wang, H.: Neurological abnormality detection from electroencephalography data: a review. *Artif. Intell. Rev.* (2021, early access). <https://doi.org/10.1007/s10462-021-10062-8>
5. Alvi, A.M., Siuly, S., Wang, H., Sun, L., Cao, J.: An adaptive image smoothing technique based on localization. In: Developments of Artificial Intelligence Technologies in Computation and Robotics: Proceedings of the 14th International FLINS Conference (FLINS 2020), pp. 866–873. World Scientific (2020)
6. Amezcuita-Sanchez, J.P., Mammone, N., Morabito, F.C., Marino, S., Adeli, H.: A novel methodology for automated differential diagnosis of mild cognitive impairment and the Alzheimer’s disease using EEG signals. *J. Neurosci. Meth.* **322**, 88–95 (2019)
7. Chen, H., Song, Y., Li, X.: A deep learning framework for identifying children with ADHD using an EEG-based brain network. *Neurocomputing* **356**, 83–96 (2019)
8. Chollet, F.: Deep Learning with Python. Simon and Schuster, Manhattan (2017)
9. Delorme, A., et al.: EEGLAB, SIFT, NFT, BCILAB, and ERICA: new tools for advanced EEG processing. *Comput. Intell. Neurosci.* **2011**, 130714 (2011)
10. Deng, L., Liu, Y.: Deep Learning in Natural Language Processing. Springer, Singapore (2018). <https://doi.org/10.1007/978-981-10-5209-5>
11. Du, J., Michalska, S., Subramani, S., Wang, H., Zhang, Y.: Neural attention with character embeddings for hay fever detection from twitter. *Health Inf. Sci. Syst.* **7**(1), 1–7 (2019). <https://doi.org/10.1007/s13755-019-0084-2>
12. Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., Bennamoun, M.: Deep learning for 3D point clouds: a survey. *IEEE Trans. Pattern Anal. Mach. Intell.* (2020, early access)
13. Hadiyoso, S., La Febry, A., Mengko, T.L.E., Zakaria, H.: Early detection of mild cognitive impairment using quantitative analysis of EEG signals. In: 2019 2nd International Conference on Bioinformatics, Biotechnology and Biomedical Engineering (BioMIC)-Bioinformatics and Biomedical Engineering, vol. 1, pp. 1–5. IEEE (2019)

14. He, J., Rong, J., Sun, L., Wang, H., Zhang, Y., Ma, J.: A framework for cardiac arrhythmia detection from IoT-based ECGs. *World Wide Web* **23**(5), 2835–2850 (2020)
15. Jiang, H., Zhou, R., Zhang, L., Wang, H., Zhang, Y.: Sentence level topic models for associated topics extraction. *World Wide Web* **22**(6), 2545–2560 (2018). <https://doi.org/10.1007/s11280-018-0639-1>
16. Kamilaris, A., Prenafeta-Boldú, F.X.: Deep learning in agriculture: a survey. *Comput. Electron. Agric.* **147**, 70–90 (2018)
17. Kashefpoor, M., Rabbani, H., Barekattain, M.: Automatic diagnosis of mild cognitive impairment using electroencephalogram spectral features. *J. Med. Sig. Sens.* **6**(1), 25 (2016)
18. Kashefpoor, M., Rabbani, H., Barekattain, M.: EEG signals from normal and mci (mild cognitive impairment) cases (2016). <http://ww25.biosigdata.com/?download=egsignals-from-normal-and-mci-cases>. Accessed 2 Nov 2019
19. Kashefpoor, M., Rabbani, H., Barekattain, M.: Supervised dictionary learning of EEG signals for mild cognitive impairment diagnosis. *Biomed. Sig. Process. Control* **53**, 101559 (2019)
20. Kelleher, J.D.: *Deep Learning*. MIT Press, Cambridge (2019)
21. Kraus, M., Feuerriegel, S., Oztekin, A.: Deep learning in business analytics and operations research: models, applications and managerial implications. *Eur. J. Oper. Res.* **281**(3), 628–641 (2020)
22. Ois, C.F.: Keras (2015). <https://github.com/fchollet/keras>. Accessed 18 Mar 2021
23. Siuly, S., et al.: A new framework for automatic detection of patients with mild cognitive impairment using resting-state EEG signals. *IEEE Trans. Neural Syst. Rehabil. Eng.* **28**(9), 1966–1976 (2020)
24. Siuly, S., Li, Y., Zhang, Y.: EEG signal analysis and classification. *IEEE Trans. Neural Syst. Rehabil. Eng.* **11**, 141–4 (2016)
25. Supriya, S., Siuly, S., Wang, H., Zhang, Y.: Automated epilepsy detection techniques from electroencephalogram signals: a review study. *Health Inf. Sci. Syst.* **8**(1), 1–15 (2020). <https://doi.org/10.1007/s13755-020-00129-1>
26. Tawhid, M.N.A., Siuly, S., Wang, H., Whittaker, F., Wang, K., Zhang, Y.: A spectrogram image based intelligent technique for automatic detection of autism spectrum disorder from EEG. *PLoS ONE* **16**(6), e0253094 (2021)
27. Voulodimos, A., Doulamis, N., Doulamis, A., Protopapadakis, E.: Deep learning for computer vision: a brief review. *Comput. Intell. Neurosci.* **2018**, 1–13 (2018)
28. Vrbancic, G., Podgorelec, V.: Automatic classification of motor impairment neural disorders from EEG signals using deep convolutional neural networks. *Elektronika ir Elektrotechnika* **24**(4), 3–7 (2018)
29. Yang, D., et al.: Detection of mild cognitive impairment using convolutional neural network: temporal-feature maps of functional near-infrared spectroscopy. *Front. Aging Neurosci.* **12**, 141 (2020)
30. Yin, J., Cao, J., Siuly, S., Wang, H.: An integrated mci detection framework based on spectral-temporal analysis. *Int. J. Autom. Comput.* **16**(6), 786–799 (2019)
31. Zhang, L., Wang, S., Liu, B.: Deep learning for sentiment analysis: a survey. *Wiley Interdiscip. Rev.: Data Min. Knowl. Disc.* **8**(4), e1253 (2018)
32. Zhang, Z., Cui, P., Zhu, W.: Deep learning on graphs: a survey. *IEEE Trans. Knowl. Data Eng.* (2020, early access)
33. Zou, J., Huss, M., Abid, A., Mohammadi, P., Torkamani, A., Telenti, A.: A primer on deep learning in genomics. *Nat. Genet.* **51**(1), 12–18 (2019)



# Dynamic Transit Flow Graph Prediction in Spatial-Temporal Network

Liying Jiang<sup>1</sup>, Yongxuan Lai<sup>1</sup>(✉), Quan Chen<sup>2</sup>, Wenhua Zeng<sup>1</sup>, Fan Yang<sup>3</sup>,  
Fan Yi<sup>4</sup>, and Qisheng Liao<sup>5</sup>

<sup>1</sup> School of Informatics/Shenzhen Research Institute, Xiamen University,  
Shenzhen, China

jiangliying@stu.xmu.edu.cn, {laiyx,whzeng}@xmu.edu.cn

<sup>2</sup> Department of Computer Science, Xiamen University Malaysia, Sepang, Malaysia

<sup>3</sup> Department of Automation, Xiamen University, Xiamen, China

yang@xmu.edu.cn

<sup>4</sup> School of Mathematics and Statistics, Key Laboratory of Complex Systems and  
Intelligent Computing, Qiannan Normal University for Nationalities, Duniyun, China

<sup>5</sup> New York University, New York, USA

q.liao@nyu.edu

**Abstract.** Traffic flow prediction is of great importance for traffic management. However, most existing researches only focus on region flow or road segment flow (vertex value) prediction, and the transit flow (edge weight) prediction is largely untouched. Compared to region flow and road segment flow prediction, transit flow prediction is more challenging in that 1) the transit flow between pairs of regions has complex spatial-temporal dependencies, and 2) it has larger changes over time due to the large number of region pairs. To address these issues, in this paper we define the transit flow as edges in directed graphs and formulate the transit flow prediction problem as a dynamic weighted link prediction problem. We propose a deep learning based method called Spatial-Temporal Network (STN) to make an accurate prediction of the transit flow. The STN model combines graph convolutional network (GCN) and long short-term memory (LSTM) to capture the dynamic spatial-temporal correlations. To capture the static topological structure, the neighborhood relation graph is adopted as an auxiliary graph to improve the prediction accuracy, and a two-stage-skip strategy is adopted to allow edge features reused which makes the STN focus more on the edge values compared to simple GCN modeling. We conduct the proposed STN model and verify its effectiveness in transit flow prediction on two real-world taxi datasets. Experiments demonstrate that our model reduces the prediction RMSE error by approximately 15.88%–52.48% on real-world datasets compared to state-of-the-art methods.

---

This work was supported in part by the Natural Science Foundation of Guangdong under Grant 2021A1515011578, Natural Science Foundation of China under Grant 61672441 and Grant 61673324, Natural Science Foundation of Fujian under Grant 2018J01097, Shenzhen Basic Research Program under Grant JCYJ20170818141325209 and Grant JCYJ20190809161603551.

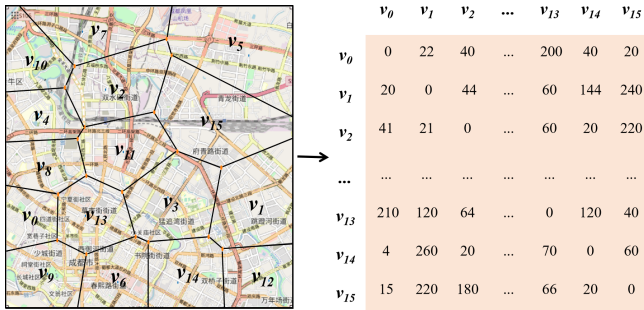
© Springer Nature Switzerland AG 2021

W. Zhang et al. (Eds.): WISE 2021, LNCS 13080, pp. 603–618, 2021.

[https://doi.org/10.1007/978-3-030-90888-1\\_46](https://doi.org/10.1007/978-3-030-90888-1_46)

**Keywords:** Transit flow prediction · Spatio-temporal network · LSTM · GCN

# 1 Introduction



**Fig. 1.** Example of a transit flow graph in Chengdu City, China. The city map is divided into irregular regions, each region can be described as a node and the transit flow between a pair of regions can be described as an edge. The transit flow graph is represented by a square matrix, where each element of the matrix indicates the transit flow between a pair of regions.

Intelligent Transport System (ITS) plays an important role in improving our mode of traveling and enhancing the safety of transportation, among which traffic flow prediction is of great importance for traffic management. It is critical for traffic administration to analyze the traffic condition and propose some preventive measures in advance. Predicting traffic flow is also helpful for ride-sourcing systems to address supply-demand imbalance across space and time [12]. For example, if a system can predict the number of passengers who will leave region A for region B, it can reallocate idle vehicles to region A accordingly so that there are enough vehicles for the rides, which also shortens the waiting time.

Most of the existing researches on traffic flow prediction only focus on the traffic change of flow of a region or a road segment (vertex value). But they do not address the problem of change of flow between pairs of regions or road segments (edge weight). As illustrated in Fig. 1, the city map of Chengdu City, China is divided into 16 non-overlap regions, and each region can be described as a vertex and the transit flow between a pair of regions can be described as an edge. The transit flow graph is represented by a square matrix, where each element of the matrix indicates the traffic flow between a pair of regions. For example,  $(v_1, v_2) = 44$  indicates that there are 44 taxis (or passengers) leaving region 1 for region 2. So the prediction of the traffic flow between pairs of regions is denoted as the *transit flow prediction* problem, which is more difficult than the region flow prediction problem. The major challenges lie in three folds: 1)



the transit flow has more complex spatial-temporal dependencies in nature; 2) the transit flow between regions has larger changes over time than the region flow; 3) the transit flow data is usually very sparse. The transit flow between far-away regions is usually close to or equal to 0.

To tackle the aforementioned challenges, we formulate the transit flow prediction problem as a dynamic weighted link prediction problem [14], where the transit flow among all regions at a certain timeslot can be described as a directed graph. We aim to make an accurate prediction for the transit flow graph at the next timeslot when given the transit flow graphs of previous timeslots. In this paper, we propose a deep-learning approach called STN (Spatial-Temporal Network) to make an accurate prediction of the transit traffic flow. It integrates the GCN [13] and LSTM [10] to capture the dynamic spatial-temporal characteristics of the traffic flow in the city, and adopts the neighborhood relation graph as an auxiliary graph to enhance the model of transit flow prediction. The main contributions of this paper are summarized as follows:

1. We partition a city into irregular non-overlap regions based on the K-Means clustering algorithm and represent the transit flow as directed graphs, where nodes represent regions and edges represent transit flow of pairs of regions. Then we formulate the transit flow graph prediction problem as a dynamic weighted link prediction problem.
2. We propose a Spatial-Temporal Network (STN) model to predict the transit traffic flow in the graph. The model combines GCN and LSTM to model the dynamic spatial dependencies and temporal dependencies respectively. We use the neighborhood relation graph as an auxiliary graph to capture the static spatial dependencies and use a two-stage-skip strategy to allow edge feature reusing.
3. We evaluate our STN model on two real-world taxi datasets. The results show that our method reduces the prediction RMSE error by approximately 15.88%–52.48% on TaxiXM and TaxiCD<sup>1</sup> compared to state-of-the-art methods, which demonstrates the effectiveness of our model in the transit flow graph prediction.

The remainder of this paper is organized as follows. Section 2 describes the related works. Section 3 defines several key concepts and introduces our problem. Section 4 shows the details of our method. Section 5 introduces the datasets and experimental settings. Section 6 presents the performance of the proposed model and compares it with other methods in two real-world datasets. Section 7 concludes the paper and outlines some future works.

## 2 Related Work

Traffic flow prediction has received considerable attention in recent years. The existing traffic flow prediction methods can be divided into two categories: 1)

<sup>1</sup> <https://gaia.didichuxing.com>.

traditional methods; 2) deep learning based methods. Traditional methods consist of statistical based methods and machine learning methods. Statistical based methods, such as ARIMA (Autoregressive Integrated Moving Average model) [1], can only work on steady-state traffic conditions. Furthermore, such methods require comprehensive prior knowledge which is hard for most researchers. Compare with statistical based methods, machine learning methods, such as SVM (Support Vector Machine) [22], and Random Forest [7], have a stronger ability to model the complex traffic data. But these methods still cannot model the spatial correlations and temporal correlations at the same time.

In recent years, with the rapid development of deep learning, deep neural network models have been used to predict the traffic flow because of its strong ability to capture the dynamic characteristics of traffic data [9, 23]. Deep learning based methods usually consist of several components to capture spatial dependencies and temporal dependencies. Researchers usually apply RNN and its variants (e.g., LSTM, GRU) [8, 15], 1D CNN [17] to model the temporal correlations, use CNN to model the spatial correlations in euclidean space [18] and use GCN to capture the spatial correlations in non-euclidean space [23]. For example, Ma et al. [15] proposed an LSTM based model to predict traffic speed/flow in road segments and it showed more superior performance in capturing temporal dependencies than traditional methods. Zhang et al. [19] proposed a CNN based model DeepST to predict the region flow. It was the first time to partition a city into  $I \times J$  regular regions so that CNN can be applied to model the spatial correlations. Zhao et al. [21] proposed the T-GCN model which combined GCN and GRU to capture the spatial correlations and temporal correlations of the traffic data on the road network simultaneously. The result showed that the combined model T-GCN had superiority in traffic forecasting than single GCN and single GRU.

The researches mentioned above mainly focus on region or road segment flow (vertex value) prediction. They do not address the more challenging transit flow (edge weight) prediction problem. Region flow and road segment flow prediction address the changes of node features (i.e., the total in-flow and out-flow of a region or a road segment) and don't concern about the changes between nodes (i.e., a flow leave a region for another region). To tackle this problem, Zhang et al. [20] first divided the city into regular grids then proposed a fully connected and CNN based method to predict region flow (edge vertex) and in/out flow (edge weight) of regions simultaneously. It used fully convolutional and external factors to capture temporal correlations and use CNN to capture spatial correlations. Same as it [20], in this paper, we focus on in flow predictions (our model could be applied to predictions of out flow). Different from it, first we divided the city into non-regular grids based on K-Means. Then we used LSTM and GCN to capture temporal-temporal correlations. Compared to CNN, GCN are more suitable for non-euclidean spatial correlations modeling.

For traffic flow prediction researches, the existing GCN based methods [9, 13] usually assumed the relations among regions to be static and usually used fixed matrices to represent them. In our problem, the transit flow between regions is

dynamic and we aim to predict it in future timeslots. In region flow prediction problem, the input of model is dynamic node feature and several static adjacency matrix. But in transit flow prediction problem, the input of model is dynamic transit matrix and dynamic node feature. The transit flow graph prediction problem can be formulated as a dynamic weighted link prediction problem. But different from the researches on conventional link prediction, which only focus on predicting the existence of links, in this paper we also predict the weights of links. Furthermore, conventional link prediction methods usually focus on one relation between nodes in the modeling phase and predicting phase. But in our approach, we adopt the neighborhood relation graph as an auxiliary graph for the transit flow prediction, which takes advantage of static and dynamic relationships among nodes.

### 3 Preliminaries

**Table 1.** Description of notations

Symbol	Description
$v_i$	i-th node, i.e., i-th region
$t$	t-th timeslot
$e_t(v_i, v_j)$	The relation of node $v_i$ and $v_j$ at t-th timeslot, i.e., the total transit flow from region $v_i$ to region $v_j$ during t-th time interval
$V$	Node set
$E_t$	The relations among all nodes at t-th timeslot
$G_t = (V, E_t)$	Transit flow graph at t-th timeslot
$t_j$	Timestamp of j-th trajectory point
$p_{j, t_j}$	j-th trajectory point
$x_{i, t}$	i-th node feature at t-th timeslot, i.e., the total region flow of region $v_i$ at t-th timeslot
$X_t$	Node features of all nodes at t-th timeslot
$A$	The adjacency matrix
$G_{t':t}$	Transit flow graph from t'-th timeslot to t-th timeslot
$X_{t':t}$	Node features of all nodes from t'-th timeslot to t-th timeslot

Some traffic flow prediction researches divided cities into various regular regions [18, 19]. These partitions cannot work well under the complex administrative and functional properties of cities. There are also a few researches divided cities into irregular regions according to the road networks [11]. However, it is hard to divide cities well due to the complexity of road networks. In this paper, we divide cities into irregular regions based on K-Means [16], a simple clustering algorithm. First,

we mine the origin-destination pairs from taxi trajectories. Second, we use K-Means to get the cluster centroids from origin-destination pairs. Finally, we use Voronoi tessellation to define the Voronoi cells based on the K-Means centroids. As Fig. 1 shows, Voronoi tessellation divides the city into non-overlap irregular regions. Compared to regular partition and road network based partition, K-Means based method has two advantages; 1) origin-destination pairs reflect the characteristics of the resident trip so it can partition a city well; 2) we can obtain origin-destination data from taxi datasets and no additional data is needed.

In the rest of this section, we first define several key concepts then formulate the research problem. Table 1 lists the notations used in this paper.

### 3.1 Transit Flow Graph

We divide a day into several uniform intervals (e.g., 1 h). Then we represent the transit flow at  $t$ -th timeslot as a weighted graph  $G_t = \{V, E_t\}$ , whose nodes are regions and edges are transit flow among regions.  $v_i \in V$  denotes the  $i$ -th region, and  $e_t(v_i, v_j)$  denotes a traffic flow from region  $v_i$  to region  $v_j$  during  $t$ -th time interval. The example of a transit flow graph is shown in Fig. 1. The time-order transit graphs can be described as:

$$G = \{G_1, G_2, \dots, G_t\} \quad (1)$$

where  $G_t$  is the transit flow graph of  $t$ -th timeslot.

### 3.2 Node Flow

We define trajectory point as a historical GPS point. Each GPS point  $p_j$  contains: the region number  $v_i$  and timestamp  $t_k$ . A region can be described as a node, for a node  $v_i$ , the node flow during the time interval  $t$  is defined as

$$x_{i,t} = \{p_{j,t_k} \in v_i \wedge t_k \in t\} \quad (2)$$

where  $p_{j,t_k} \in v_i$  means the trajectory point  $p_{j,t_k}$  lies within the region  $v_i$  and  $t_k \in t$  means the timestamp  $t_k$  is in the time interval  $t$ . We use  $X_t$  to denote the node flow of all nodes at  $t$ -th timeslot. Then we use  $X_t$  as the dynamic node features during the GCN stage.

### 3.3 Neighborhood Relation Graph

Transit flow between adjacent regions is likely to be larger than that between non-adjacent regions. So we use the neighborhood relation graph to enhance the relations of adjacent regions. We define an adjacent matrix to indicate whether two regions are adjacent.

$$A = \begin{cases} 1 & \text{region } v_i \text{ and region } v_j \text{ are adjacent} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

For example, as Fig. 1 shows,  $A_{v_0, v_8} = 1$ ,  $A_{v_0, v_4} = 0$ .

### 3.4 Problem Statement

Transit flow graph prediction problem aims to learn a function  $f$  that is able to forecast the next timeslot transit flow graph while given  $t'$  historical transit flow graphs, node features and a neighborhood relation graph  $A$ , it can be formulated as:

$$[G_{t-t'+1:t}, X_{t-t'+1:t}, A] \xrightarrow{f} [G_{t+1}] \tag{4}$$

The transit flow between regions changes over time but the neighborhood relation between regions is always constant. Therefore, we use neighborhood relation graph  $A$  as an auxiliary graph to capture the static spatial dependencies of regions.

## 4 Methodology

### 4.1 Background Technologies

**Graph Convolutional Networks (GCN).** The traditional convolutional neural network (CNN) can obtain local spatial features, but it can only be used in euclidean space and is not applicable for general graphs. To address this issue, Bruna et al. [4] proposed graph convolutional networks (GCN) which redefine convolution operators to capture the spatial features for non-euclidean data. GCN is defined over a graph  $G = (V, A)$ , where  $V$  is the set of vertices and  $A \in R^{|V| \times |V|}$  is the adjacency matrix whose entries represent the relation between vertices. A 1-layer GCN operation is defined as:

$$f(X, A) = \sigma(\widehat{A}XW) \tag{5}$$

where  $X$  represents the node feature,  $\widehat{A} = \widetilde{D}^{-1/2}\widetilde{A}\widetilde{D}^{-1/2}$  denotes the graph Laplacian matrix,  $I$  is an identity matrix,  $\widetilde{A} = A + I_N$  is a matrix with self-connection structure,  $D$  is the degree matrix,  $W$  represents the learnable weights,  $\sigma(\cdot)$  represents the activation function.

**Long Short-Term Memory (LSTM).** LSTM is a variant of recurrent neural network (RNN), which is RNN with learned gating mechanisms. LSTM has 3 more gates (Input, Forget and Output) than simple RNN, which mitigates the vanishing gradient problem and allow the model to learn longer-term dependencies. For the input  $x_t$ , LSTM follows these manners:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{6}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{7}$$

$$\widetilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{8}$$

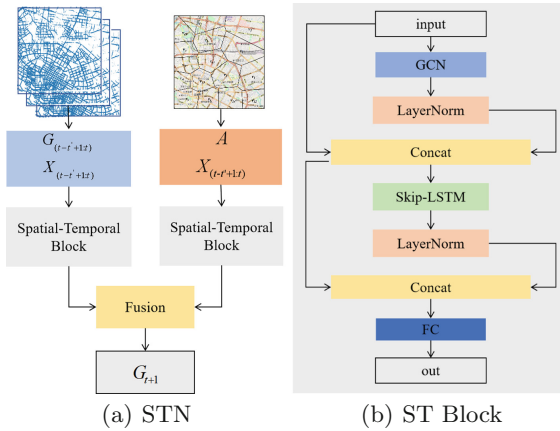
$$C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t \tag{9}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{10}$$

$$h_t = o_t * \tanh(C_t) \tag{11}$$

### 4.2 Spatial-Temporal Network

Figure 2(a) illustrates the framework of our proposed Spatial-Temporal Network (STN), which has two Spatial-Temporal Blocks (ST Block) and a fusion block. In ST block, we use a two-stage-skip strategy to improve the prediction accuracy. The two ST blocks have the same structure, the left one is used to get dynamic spatial dependencies from the transit flow graph (we called dynamic ST block) and the right one is used to get the static spatial dependencies from the neighborhood relation graph (we called static ST block). In dynamic ST block, we first convert the trajectories data along time into transit flow graphs  $G = \{G_1, G_2, \dots, G_t\}$ , which is a time-ordered sequence of graphs. And then the transit flow graphs  $G$  and node features  $X$  are fed into the ST block. In static ST block, we first get the adjacent matrix  $A$  from the regions' neighborhood relation. Then the adjacent matrix  $A$  and node features  $X$  are fed into the ST block. The static ST block is an auxiliary part to predict the transit flow graph. Finally, we use a matrix fusion block to fuse the representations get from these two ST blocks to get the future transit flow graph  $G_{t+1}$ . In the rest of this section, we will elaborate these two components of STN in detail.



**Fig. 2.** The architecture of proposed model Spatial-Temporal Network (STN) and Spatial-Temporal Block (ST Block)

**Spatial-Temporal Block (ST Block).** The structure of ST block is illustrated in Fig. 2(b). This component mainly consists of two parts. The first part is GCN, and the second part is Skip-LSTM (Fig. 3) [3]. Assuming that the input transit flow graph is  $G$  and the node feature is  $X$ . First, we utilize GCN to explore the topology characteristics of each graph and use LayerNorm [2] to normalize the representation of each node. Then we get a new representation  $G'$ . We concatenate  $G$  and  $G'$ , called first-stage-skip, and pass it to a Skip-LSTM network to model the dynamic evolution of graphs along time. Figure 3 illustrates

the structure of Skip-LSTM. Different from the conventional LSTM network, we concatenate the input  $x_t$  and hidden layer's output  $h_t$  of an LSTM cell together as the final hidden layer's output, then pass it to the next LSTM cell. We also use LayerNorm to normalize the output of Skip-LSTM. Assuming that the output of Skip-LSTM is  $G'_T$ , we concatenate  $G'$  and  $G'_T$ , called second-stage-skip, and pass it to a fully connected layer to create a final representation  $G'_{ST}$ .

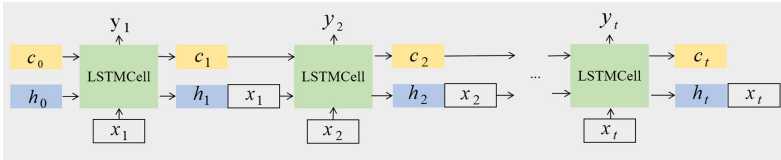
The propagation rule of the Spatial-Temporal block can be summarized as follows:

$$G' = \text{LayerNorm}(\text{GCN}(G, X)) \quad (12)$$

$$G'_T = \text{LayerNorm}(\text{Skip-LSTM}(G || G')) \quad (13)$$

$$G'_{ST} = \text{FC}(G' || G'_T) \quad (14)$$

where  $||$  represents the concatenation operation.



**Fig. 3.** The architecture of Skip-LSTM

**Fusion Block.** As Sect. 3 mentioned, the transit flow between adjacent regions is likely to be larger than that in non-adjacent regions. The spatial properties of transit flow are affected by the dynamic topological structure and static topological structure by varying degrees. Inspired by this, we use a parametric-matrix-based fusion method to fuse the two ST blocks. Assuming that the output of the dynamic ST block is  $G^d_{t+1}$  and the output of the static ST block is  $G^s_{t+1}$ , the fusion method can be defined as:

$$G_{t+1} = W_d \circ G^d_{t+1} + W_s \circ G^s_{t+1} \quad (15)$$

where  $\circ$  is Hadamard product,  $W_d$  and  $W_s$  are learnable parameters that adjust the degrees affected by the dynamic and static spatial correlations respectively.

## 5 Experimental Settings

### 5.1 Datasets

To evaluate the effectiveness of our model, we carried out comparative experiments on two real-world taxi datasets as shown in Table 2, detailed as follows:

**TaxiXM.** The trajectory data is taxi GPS data of Xiamen city, China from 1st Jul. 2014 to 31st Jul. 2014. We select Xiamen island as the study area, with an area of approximately 132.5 km<sup>2</sup>.

**TaxiCD.** The trajectory data is taxi GPS data of Chengdu City, China from 1st Oct. 2016 to 30th Nov. 2016. We select an area that lies between  $30.65^\circ E$  to  $30.72^\circ E$  latitude and  $104.04^\circ N$  to  $104.12^\circ N$  longitude as the study area, with an area of approximately  $65 \text{ km}^2$ .

In the experiments, we partitioned the Xiamen city into 32 regions and Chengdu City into 16 regions, each region with an average area of approximately  $4 \text{ km}^2$ , and divided a day into 72 timeslots, each slot is 20 min. We used Min-Max normalization to normalize the input data. We used 80% of the data for training, 20% for testing.

**Table 2.** Details of datasets

Dataset	TaxiXM	TaxiCD
Data type	Taxi GPS	Taxi GPS
Location	Xiamen	Chengdu
Area	$132.5 \text{ km}^2$	$65 \text{ km}^2$
Time Span	7/1/2014–7/31/2014	10/1/2016–11/30/2016
Time interval	20 min	20 min
Total time slot	2232	4392
Region number	32	16

## 5.2 Hyperparameters

Recalling that the task is to learn a function  $f : [G_{(t-t'+1:t)}, X_{(t-t'+1:t)}, A] \rightarrow [G_{t+1}]$ , we aim at forecasting the next timeslot transit flow graph given previous  $t'$  transit flow graphs. In our experiment, we set  $t' = 6$  (we set  $t' = 3$  to 12 then select the best result, in Sect. 6, we show the impact of input sequence length). We implemented the STN model based on PyTorch 1.5.1, a widely used Deep Learning Python library. In our model, we set graph convolution kernels as 64 and LSTM units as 256. During the training phase, we set batch size as 64 and the learning rate as 0.001.

## 5.3 Metrics

We evaluate the performance of our model by three widely used metrics, i.e., Root Mean Square Error (RMSE), Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE). They are defined as follows:

$$RMSE = \sqrt{\frac{1}{n} * \sum_{i=1}^n (G_{t+1}^i - \tilde{G}_{t+1}^i) / (|V| * |V|)} \quad (16)$$

$$MAE = \frac{1}{n} * \left| \sum_{i=1}^n (G_{t+1}^i - \tilde{G}_{t+1}^i) / (|V| * |V|) \right| \quad (17)$$



$$MAPE = \frac{100\%}{n} * \left| \sum_{i=1}^n \frac{(G_{t+1}^i - \tilde{G}_{t+1}^i)}{G_{t+1}^i} \right| / (|V| * |V|) \quad (18)$$

where  $G_{t+1}$  is the actual transit flow graph,  $\tilde{G}_{t+1}^i$  is the predicted transit flow graph and  $|V|$  is the total region number.

## 5.4 Compared Algorithms

We compare our model with the following 4 methods:

**HA.** Historical Average model uses the average of historical values in corresponding timeslots.

**XGBoost** [6]. Xtreme Gradient Boosting is a powerful ensemble method. In our experiment, we predicted the value of each edge in the transit flow graph separately. For every edge, we predict it according to its previous  $t'$  values.

**E-lstm-d** [5]. E-lstm-d is an LSTM based encoder-decoder model for conventional dynamic link prediction. Our research problem needs to predict the weights of links, so we replace the activation function Sigmoid in the model with ReLU which is more suitable for the regression problem.

**GCN-GAN** [14]. GCN-GAN is a non-linear model for weighted dynamic link prediction. GCN-GAN model combines GCN, LSTM and GAN and shows good performance in weighted temporal link prediction task.

# 6 Experimental Results

## 6.1 Performance Comparison

**Table 3.** Performance comparison of different approaches for transit flow graph prediction on TaxiXM and TaxiCD

Method	TaxiXM			TaxiCD		
	RMSE	MAE	MAPE(%)	RMSE	MAE	MAPE(%)
HA	10.93	5.57	0.36	13.32	6.34	0.39
XGBoost	7.54	4.21	0.40	8.86	4.66	0.41
E-lstm-d	10.72	6.05	0.63	11.37	5.83	0.55
GCN-GAN	6.99	4.03	0.37	7.56	4.21	0.35
STN(ours)	<b>5.88</b>	<b>3.35</b>	<b>0.27</b>	<b>6.33</b>	<b>3.52</b>	<b>0.25</b>

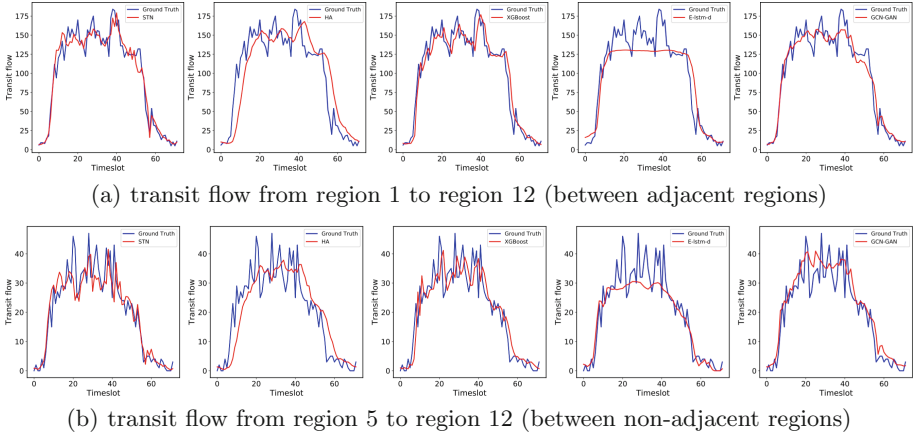
Table 3 shows the STN model and other baseline methods for transit flow graph prediction 20 min ahead on TaxiXM and TaxiCD. It can be seen that the STN model obtains the best prediction performance under all metrics. From Fig. 4, we can see that statistical based methods HA and machine learning method XGBoost have time lags and LSTM-based method E-lstm-d performs badly in

some timeslots. Therefore, they perform worse than GCN-based methods GCN-GAN and STN. On TaxiXM, the RMSE error is reduced approximately 15.88%–46.20%, the MAE error is reduced approximately 16.87%–44.63% and the MAPE error is reduced approximately 25.00%–57.14% compared to other methods. On TaxiCD datasets, the RMSE error is reduced approximately 16.27%–52.48%, the MAE error is reduced approximately 16.39%–44.48% and the MAPE error is reduced approximately 28.57%–54.54% compared to other methods. The results prove the effectiveness of the STN model in transit flow graph prediction.

To explore the performance of these methods in adjacent regions and non-adjacent regions, we select a pair of adjacent regions and a pair of non-adjacent regions on Chengdu as a study case. Figure 4(a) is transit flow prediction between adjacent regions and Fig. 4(b) is transit flow prediction between non-adjacent regions. From these pictures, we observe that: 1) Statistical based method HA and machine learning method XGBoost have time lags in both situations, which leads to worse prediction precision than deep learning based methods GCN-GAN and STN. 2) LSTM based method E-lstm-d performs badly in modeling the time series with frequent changes and no fixed patterns (e.g., timeslot 10 to 50). Thus, in our experiment, E-lstm-d shows worse than XGBoost. 3) GCN based methods GCN-GAN and STN perform well in both adjacent regions and non-adjacent regions. And STN performs better during timeslot 10 to 50 while transit flow changes frequently. Table 4 shows the predicted result of GCN-GAN and STN in all adjacent regions and non-adjacent regions. Take GCN-GAN as a baseline, STN has a higher improvement in adjacent regions than in non-adjacent regions. For example, the MAPE error is reduced 10.52% in adjacent regions and reduced 3.44% in non-adjacent regions on TaxiCD. The reason may be that the neighborhood relation graph enables the model to learn the spatial dependencies from both dynamic topological and static topological and strengthen the relations of adjacent regions.

**Table 4.** Performance comparison of GCN-GAN and STN in adjacent regions and non-adjacent regions on TaxiXM and TaxiCD

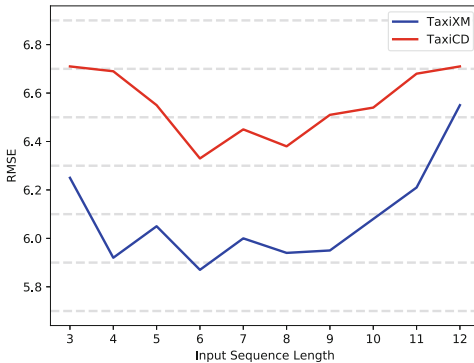
		TaxiXM			TaxiCD		
		RMSE	MAE	MAPE(%)	RMSE	MAE	MAPE
Adjacent regions	GCN-GAN	13.40	9.42	0.18	13.04	9.01	0.19
	STN	11.88	8.34	0.16	12.03	8.38	0.17
		↓11.34%	↓11.46%	↓11.11%	↓7.74%	↓6.99%	↓10.52%
Non-adjacent regions	GCN-GAN	4.81	2.74	0.31	4.24	2.32	0.29
	STN	4.32	2.53	0.30	4.00	2.23	0.28
		↓10.18%	↓7.66%	↓3.22%	↓5.66%	↓3.87%	↓3.44%



**Fig. 4.** The results of transit flow prediction 20 min ahead on Chengdu City.

### 6.2 Effect of Input Sequence Length

In our method, we hope to mine the pattern and development trend of the transit flow from the input sequence. Hence, if the input sequence length is too short, the prediction results may not be satisfactory, but if the input sequence length is too long, the training data will contain some noise information which will mislead experiment results.



**Fig. 5.** Results of different lengths of input sequence on TaxiXM and TaxiCD.

To test how long input sequence length is needed to achieve a good performance, we vary the length of input sequence and see how prediction error changes. Figure 5 shows the results in TaxiXM and TaxiCD. In TaxiCD dataset, we can find that if the length of input sequence is shorter than 4, our model does not perform very well. By increasing the length of input sequence beyond

10, the model also has bad performance. With these results, we suggest that for robust prediction accuracy, the training data is desired to be last for least one and a half hours and no longer than three hours.

### 6.3 Effect of Each Component

To investigate the effect of each component in our model, we evaluate two variants by removing the auxiliary graph and the skip connection strategy separately, named STN-NA and STN-NS. Table 5 demonstrates the effect of different components on the final experiment performance. We find that the skip connection strategy affects the prediction results significantly. The RMSE error is reduced by approximately 1.85%–5.52% on TaxiXM and TaxiCD. This may be because the skip connection strategy allows feature reusing and ensures the spatial features learned by the GCN layer are not fewer than the original graph. Therefore, GCN with skip connection strategy focuses more on edge values than simple GCN. The auxiliary graph also improves the prediction result. The RMSE error is reduced approximately 0.34%–1.40%, the MAPE error is reduced approximately 3.57%–7.40% and the MAE error is reduced approximately 1.18%–2.49%. These results confirm the effectiveness of the skip connection strategy and auxiliary graph.

**Table 5.** Effect of different components on TaxiXM and TaxiCD

	TaxiXM			TaxiCD		
	RMSE	MAE	MAPE(%)	RMSE	MAE	MAPE(%)
STN-NA	5.90	3.39	0.28	6.42	3.61	0.27
STN-NS	6.18	3.47	0.28	6.70	3.70	0.26
STN	<b>5.88</b>	<b>3.35</b>	<b>0.27</b>	<b>6.33</b>	<b>3.52</b>	<b>0.25</b>

**Table 6.** Effect of city partition methods on TaxiXM and TaxiCD

	TaxiXM			TaxiCD		
	RMSE	MAE	MAPE(%)	RMSE	MAE	MAPE(%)
Regular	5.97	3.41	0.27	6.82	3.93	0.26
K-Means	5.88	3.35	0.27	6.33	3.52	0.25

### 6.4 Effect of City Partition Methods

To explore the effect of different city partition methods in transit flow predicting. We also divided the city into non-overlap regular regions, the number of which is equals to irregular regions. Then we predict the transit flow between these regular regions. Table 6 show the result of the effect on different city partition

methods. We can see that the model performs better in K-Means based partition regions both in TaxiXM and TaxiCD datasets. The result improves that origin-destination pairs reflect the characteristics of the resident trip, so it can partition a city well.

## 7 Conclusion

In this paper, we formulate the transit flow prediction problem as a dynamic weighted link prediction problem and propose a spatial-temporal network called STN to predict the transit traffic flow. Our model combines the GCN and Skip-LSTM, where GCN is to model the spatial correlations and Skip-LSTM to model the temporal correlations. The neighborhood relation graph is also adopted as an auxiliary graph to highlight the correlations between adjacent regions and a two-stage-skip strategy is used to reuse the edge features. The experiment results show the effectiveness of these two components. Compared with the HA, XGBoost, E-lstm-d and GCN-GAN models, our STN model achieves the best prediction results under different metrics on two real-world datasets.

Actually, the traffic flow is affected by many external factors, there are still many issues to investigate in future work. For example, we would like to involve more knowledge or data as auxiliary graphs, e.g. functional similarity graph and interaction graph of regions that could be extracted from historical taxi transition records, and embed them into the proposed framework to improve the prediction accuracy.

## References

1. Ahmed, M.S., Cook, A.R.: Analysis of freeway traffic time-series data by using Box-Jenkins techniques, vol. 722 (1979)
2. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint [arXiv:1607.06450](https://arxiv.org/abs/1607.06450) (2016)
3. Bonner, S., et al.: Temporal neighbourhood aggregation: Predicting future links in temporal graphs via recurrent variational graph convolutions. In: 2019 IEEE International Conference on Big Data (Big Data), pp. 5336–5345. IEEE (2019)
4. Bruna, J., Zaremba, W., Szlam, A., LeCun, Y.: Spectral networks and locally connected networks on graphs. arXiv preprint [arXiv:1312.6203](https://arxiv.org/abs/1312.6203) (2013)
5. Chen, J., et al.: E-LSTM-D: a deep learning framework for dynamic network link prediction. *IEEE Trans. Syst. Man Cybern. Syst.* **51**(6), 3699–3712 (2019)
6. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794 (2016)
7. Chen, Z., Ling, X., Feng, X., Zheng, H., Xu, Y.: Short-term traffic state prediction approach based on FCM and random forest. *J. Electron. Inf. Technol.* **40**(8), 1879–1886 (2018)
8. Fu, R., Zhang, Z., Li, L.: Using LSTM and GRU neural network methods for traffic flow prediction. In: 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC), pp. 324–328. IEEE (2016)

9. 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)
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
11. Hu, J., Guo, C., Yang, B., Jensen, C.S., Chen, L.: Recurrent multi-graph neural networks for travel cost prediction. arXiv preprint [arXiv:1811.05157](https://arxiv.org/abs/1811.05157) (2018)
12. Ke, J., Qin, X., Yang, H., Zheng, Z., Zhu, Z., Ye, J.: Predicting origin-destination ride-sourcing demand with a spatio-temporal encoder-decoder residual multi-graph convolutional network. arXiv preprint [arXiv:1910.09103](https://arxiv.org/abs/1910.09103) (2019)
13. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint [arXiv:1609.02907](https://arxiv.org/abs/1609.02907) (2016)
14. Lei, K., Qin, M., Bai, B., Zhang, G., Yang, M.: GCN-GAN: a non-linear temporal link prediction model for weighted dynamic networks. In: IEEE INFOCOM 2019-IEEE Conference on Computer Communications, pp. 388–396. IEEE (2019)
15. Ma, X., Tao, Z., Wang, Y., Yu, H., Wang, Y.: Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. *Transp. Res. Part C Emerg. Technol.* **54**, 187–197 (2015)
16. 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, Oakland, CA, USA (1967)
17. Wu, Y., Tan, H.: Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework. arXiv preprint [arXiv:1612.01022](https://arxiv.org/abs/1612.01022) (2016)
18. Zhang, J., Zheng, Y., Qi, D.: Deep spatio-temporal residual networks for citywide crowd flows prediction. arXiv preprint [arXiv:1610.00081](https://arxiv.org/abs/1610.00081) (2016)
19. Zhang, J., Zheng, Y., Qi, D., Li, R., Yi, X.: DNN-based prediction model for spatio-temporal data. In: Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, pp. 1–4 (2016)
20. Zhang, J., Zheng, Y., Sun, J., Qi, D.: Flow prediction in spatio-temporal networks based on multitask deep learning. *IEEE Comput. Archit. Lett.* **32**(03), 468–478 (2020)
21. Zhao, L., et al.: T-GCN: a temporal graph convolutional network for traffic prediction (2018)
22. Zhao-sheng, Y., Yuan, W., Qing, G.: Short-term traffic flow prediction method based on SVM. *J. Jilin Univ. (Eng. Technol. Ed.)* **6**, 009 (2006)
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)



# Disatra: A Real-Time Distributed Abstract Trajectory Clustering

Liang Chen<sup>1</sup>, Pingfu Chao<sup>2</sup>, Junhua Fang<sup>1</sup>(✉), Wei Chen<sup>1</sup>, Jiajie Xu<sup>1</sup>,  
and Lei Zhao<sup>1</sup>

<sup>1</sup> Department of Computer Science and Technology, Soochow University,  
Suzhou, China

20195227061@stu.suda.edu.cn, {jhfang, robertchen, xujj, zhaol}@suda.edu.cn

<sup>2</sup> The University of Queensland, Brisbane, Australia

**Abstract.** Trajectory clustering is regarded as the building block of many applications in trajectory data mining. Nowadays, the ubiquity of positioning devices generates massive trajectory data continuously, which enables various real-time applications, like traffic congestion analysis, accident detection, and traveling group detection. However, these applications rely on an efficient distributed algorithm to cluster trajectory streams in real time. In this paper, we propose a real-time distributed trajectory clustering algorithm to solve this problem. The algorithm starts with a trajectory abstraction process, which compresses trajectories of arbitrary lengths into uniform data structures, named as abstract trajectories, to address the data skewness. Then, a Geohash-based indexing strategy is proposed to partition the abstract trajectories so that clustering can be performed locally with no cross-node interaction. Finally, we design a density-based clustering algorithm on abstract trajectory which achieves a similar accuracy compared with existing clustering methods applied on original trajectories, but with much higher efficiency. Extensive experiments conducted on a real-world dataset show that our approach generates similar clustering results with significantly higher throughput and lower latency, which enable the online clustering.

**Keywords:** Distributed computing · Trajectory clustering · Stream processing · Data abstraction

## 1 Introduction

Nowadays, with the ubiquity of GPS-equipped devices, like mobile phones, vehicles and roadside cameras, massive trajectory data is being generated continuously, which enables many applications that help construct the smart city, such as driving anomaly detection [16, 20] and traffic prediction [3]. For example, by clustering trajectory data streams in real-time, we can discover the patterns of vehicle congestion propagation and crowd gathering to control the traffic flow proactively. As the prerequisite of many trajectory applications, trajectory clustering [22] has been studied since the early 2000s. According to different definitions, it can be applied to different scenarios, like finding object groups that

travel together [6, 8, 12, 19] through the detection of continuously moving clusters or identifying traffic flows [11, 13, 15] by clustering trajectories that follow the same direction. Due to the high complexity, the clustering process was usually done in offline mode. However, the recent advance in distributed computing architecture, as well as the increasing demand for real-time clustering, boost the development of distributed clustering algorithms for real-time trajectories, which is the main focus of our paper.

Most of the existing trajectory clustering algorithms [11–13] are designed for offline clustering, which takes entire trajectories as input and generates clusters according to their respective definitions. The process is done in a centralized environment, and their major differences come from the unique definitions of a cluster, which serve various user applications, as well as the corresponding algorithms. Meanwhile, most algorithms [8, 11–13, 19] follow a two-phase process: they first perform clustering on each timestamp, then they merge adjacent clustered snapshots into final results. Since the efficiency of those methods heavily relies on the time span of the dataset, they are hardly scalable to online scenario despite some recent attempts [15].

On the other hand, it is also challenging to extend the existing algorithms to a distributed environment. As it is difficult for distributed nodes to perceive each other’s content, the data may not be clustered properly when a cluster reaches the margin of a partition. To extend the clustering algorithms to a real-time distributed environment, we need a good partitioning strategy to divide the data properly, such that points in the same cluster are mostly distributed to the same node while the load balancing is still maintained.

In this paper, we propose a real-time distributed abstract trajectory (Dis-*atra*) clustering algorithm. To avoid performing clustering at every timestamp, we propose a trajectory abstraction method, which compresses a trajectory from arbitrary length to a fixed-sized data structure, termed as abstract trajectory, summarizing its spatial characteristics. The abstraction is done in real-time by incrementally merging the incoming points of the same trajectory to the existing abstract. Subsequently, Geohash indexing [18] is introduced to partition and distribute the abstract trajectories so that the clustering can be performed locally. Eventually, following the idea of DBSCAN [5], we propose a density-based clustering algorithm that takes abstract trajectory as input for clustering. Overall, our solution greatly accelerates the trajectory clustering process by getting rid of the timestamp factor and data transmission cost, as well as solving the data skewness [4] problem. The major contribution of our paper is as follows:

- We propose a real-time distributed trajectory clustering algorithm, which greatly reduces the complexity of traditional trajectory clustering methods to enable their ability on clustering large-scale real-time trajectories.
- We propose a trajectory abstraction method that greatly compresses the original trajectory to a concise structure, which solves the data skewness problem under distributed environment.
- Extensive experiments on a real-world taxi dataset show that our method achieves both high performance and high accuracy.



The rest of this paper is organized as follows. Section 2 describes the related works. We then define the problem of trajectory clustering in Sect. 3 and illustrates the proposed solution in Sect. 4. Section 5 shows the experiment results of this paper. Finally, we conclude the paper in Sect. 6.

## 2 Related Works

Our goal is to achieve real-time trajectory stream clustering in a distributed environment. Therefore, in this section, we will review existing works on trajectory clustering, real-time data stream clustering and distributed clustering, which lay the foundation of our work.

**Trajectory Clustering.** Since trajectories are usually different in both geographic length and temporal duration, it is hard to calculate their similarity directly. Hence, many trajectory clustering methods conduct the clustering based on segmented sub-trajectories [9–11]. Instead of segmenting a trajectory by connecting every two consecutive trajectory points into a sub-trajectory, some research works focus on segmenting trajectories based on semantic meanings, like using the minimum description length (MDL) [11, 13] to divide a trajectory when moving direction is changed. In addition to different segmentation strategies, various clustering algorithms perform clustering according to different spatio-temporal features. TraClus [11] emphasizes on spatial dimension while ignoring the temporal rules, but it cannot handle the incremental data stream. TCMM [13] proposes efficient algorithms for maintaining and updating the clusters when new trajectories are received. However, it still ignores the temporal features of the trajectories. Moreover, despite the fact that TCMM supports incremental updates, the trajectory data need to be cached for segmentation due to the application premise of MDL. Hence, the real-time performance is not optimistic.

Swarm [12] is a purpose-oriented pattern detection that focuses on identifying traveling groups. It is a type of trajectory clustering and is known for its fast processing speed. Swarm’s specific operation is to check whether  $k$  time points are satisfied within a period  $t$ , and more than  $m$  neighbors are moving together in the range of  $\epsilon$ , which forms a Swarm. Since Swarm does not strictly require the spatial similarity of the trajectory, it only requires to meet the similarity at a particular time within a certain period. This phenomenon is consistent with driving on a different route, but the final destination is always the same, so Swarm is very suitable for trajectory clustering.

**Real-Time Data Stream Clustering.** The research on real-time stream clustering generally adopts two-phase clustering to better extract cluster features. Aggarwal et al. [1] introduce an online-offline clustering approach, which consists of an online phase of maintaining statistics over the stream and an offline phase of clustering on these statistics. Such an approach provides more flexibility in data stream exploration as multiple clusters can be extracted from the stream by selecting different time windows from the summaries. CluStream [1] follows

the online-offline stream clustering approach. In the online phase, CluStream continually maintains a set of micro-clusters in the data stream. In the offline phase, it performs k-means to cluster based on micro-clusters. CluStream analyzes clusters' evolution by using additional property to extract information of micro-clusters during a specific time range.

**Distributed Clustering.** The method commonly used in distributed clustering computing nowadays is to divide and conquer, and make a data copy to solve the problem of clusters across partitions.

RP-DBSCAN [17] finds out approximated DBSCAN clusters by using a cell-based random partitioning strategy. SDBC [7] first selects local representative features and then performs global clustering based on these local symbols. Both of these two works adopt the idea of divide and conquer, which divides big data into independent and irrelevant small data partitions and then performing cluster analysis independently. However, data partitioning usually causes cross-partition problems. Specifically, no cluster can be generated across multiple partitions as points from different partitions are processed separately. The existing solutions are to make a copy of the data around the margin of a partition to find clusters across partitions. Nevertheless, this method reduces the efficiency of clustering due to the data redundancy. Therefore, we adopt a new data structure (we call it the Stateful Center) to solve the cross-partition problem.

### 3 Problem Definition

In this section, we will introduce the definition of our trajectory stream clustering problem. To begin with, we first list all notations frequently used in our paper, shown in Table 1:

**Table 1.** Summary of notations

Notation	Description
$\mathbb{S}$	Moving object set
$\epsilon$	Distance threshold between segments
$\rho$	Minimum number of segments in an $\epsilon$ -neighborhood
$NH_\epsilon(l_i)$	$\epsilon$ -neighborhood of trajectory segment $l_i$
$C_S$	Density-based cluster set of object set $\mathbb{S}$
$t_{ws}$	Start time of a time window
$t_{we}$	End time of a time window
$\delta t$	Step size of a time window

As the clustering is performed on trajectories, we first give the formal definition of a trajectory:

**Definition 1 (Trajectory).** A trajectory  $Tr_i$  is a series of chronologically ordered points  $Tr_i = \langle p_1 \rightarrow p_2 \rightarrow \dots \rightarrow p_n \rangle$ , representing the trace of an moving object. Each point  $p_j = \langle x, y, t \rangle$  denotes the object location at time  $p_j.t$ , and a line connecting two adjacent points  $\overline{p_k p_{k+1}}$  is defined as a trajectory segment  $l_j$ .

At present, there are many kinds of trajectory distance metrics, such as Dynamic Time Warping (DTW), distance based on Longest Common Subsequence (LCSS), and Edit Distance with Real Penalty (ERP). These algorithms are usually based on dynamic programming, and their time complexity is the square of the time series length. For this reason, we consider whether the trajectory data can be incrementally expressed as an abstract line segment (a simplified trajectory), which lower time overhead by eliminating the excessively long period of the trajectory. Therefore, in this paper, we focus on the clustering of trajectory segments instead of points. Our segment-based clustering requires a line segment distance function [11] to measure the segment distance, which is defined below:

**Definition 2 (Distance Function).** Given two trajectory segments  $l_i$  and  $l_j$  where  $|l_i| \geq |l_j|$ , the distance between two segments is defined as  $dist(l_i, l_j) = \omega_{\perp} \cdot d_{\perp}(l_i, l_j) + \omega_{\parallel} \cdot d_{\parallel}(l_i, l_j) + \omega_{\theta} \cdot d_{\theta}(l_i, l_j)$ ;

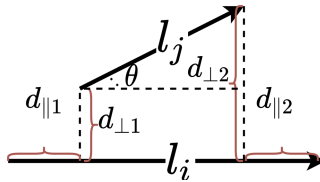


Fig. 1. Distance function.

Figure 1 demonstrates the idea of the distance function. As defined in Definition 2, there are three factors contributing to the distance measure, namely the angle distance  $d_{\theta}$ , the vertical distance  $d_{\perp}$  and the horizontal length difference  $d_{\parallel}$ . The rationale behind these three factors are as follows:

- The angle distance aims to measure the directional difference between two segments, which can avoid clustering segments from opposing directions to one group despite their spatial similarity.
- The vertical distance measures the spatial closeness between two segments. It is crucial to trajectory clustering as it is commonly measured as point-wise distances in point-based clustering methods.
- The horizontal length difference mainly focuses on measuring the speed difference. As in our method, we only cluster the segments simplified from original trajectories (details explained in Sect. 4), so the segments to be compared are derived from the objects running at the same time period, which makes the segment length proportionate to the object’s moving speed.

Note that both the vertical distance and horizontal length difference are combined results of multiple values, i.e.  $d_{\perp} = f(d_{\perp 1}, d_{\perp 2})$  and  $d_{\parallel} = g(d_{\parallel 1}, d_{\parallel 2})$ . The details of the functions  $f(x, y)$  and  $g(x, y)$  will be introduced in Sect. 4.4. Besides, as explained in Sect. 2, the density-based clustering methods are more suitable for identifying real-world trajectory clusters. Therefore, in our work, we perform the trajectory clustering using density-based clustering methods on trajectory segments. Similar to the definition of DBSCAN [5] on points, we introduce the density-reachable and density-connected concepts on trajectory segments. Firstly, given a set of trajectory segment  $L$ , the  $\epsilon$ -neighborhood of a trajectory segment  $l_p$  is defined as  $NH_{\epsilon}(l_p) = \{l_q \in L \mid dist(l_i, l_j) \leq \epsilon\}$ , therefore:

**Definition 3 (Density-reachable).** *Given a distance threshold  $\epsilon$  and an integer  $\rho$ , a trajectory segment  $l_p$  is said to be directly density-reachable to  $l_q$  if  $l_q \in NH_{\epsilon}(l_p)$  and  $|NH_{\epsilon}(l_p)| \geq \rho$ . Then a trajectory segment  $l_p$  is density-reachable from segment  $l_q$  with respect to  $\epsilon$  and  $\rho$  if there exists a chain of segments  $l_1, l_2, \dots, l_n$  from set  $L$  such that  $l_1 = l_p$ ,  $l_n = l_q$  and  $l_{i+1}$  is directly density-reachable from  $l_i$ .*

Likewise, the density-connected is defined as follows:

**Definition 4 (Density-connected).** *Given a trajectory segment set  $L$ , a segment  $l_p \in L$  is density-connected to another segment  $l_q$  with respect to  $\epsilon$  and  $\rho$  if there exists a segment  $l_x \in L$  such that  $l_p$  and  $l_q$  are density-reachable from  $l_x$ .*

Besides, since we target the trajectory clustering in an online scenario, the trajectories are not arrived in full before clustering, and our clustering process is done continuously as new trajectory points come. Meanwhile, in this paper, we apply the idea of *abstract trajectory* to abstract a trajectory  $Tr_i$  into a compact data format, denoted by  $AT_i$ , which can be regarded as a trajectory segment with additional attributes. Then, we perform clustering on abstract trajectories instead of original trajectories to achieve similar clustering result with faster processing speed in a distributed environment, which is achieved by our load-balancing and data skew-free design. Overall, our clustering problem is defined as follows:

**Definition 5 (Online Abstract Trajectory Clustering).** *Given a moving object set  $\mathbb{S}$  and a timestamp  $t$ , the online abstract trajectory clustering generates a collection of object clusters  $C_{\mathbb{S}} = \{c_1, c_2, \dots, c_n\}$ , such that (1)  $\forall O_i \in \mathbb{S}$ , its abstract trajectory  $AT_i(t)$  is in one of the cluster in  $C_{\mathbb{S}}$ ; (2)  $\forall c_u, c_v \in C_{\mathbb{S}}$ ,  $c_u \cap c_v = \emptyset$ ; (3)  $\forall AT_i(t), AT_j(t) \in c_u$ ,  $AT_i(t)$  and  $AT_j(t)$  are density-connected.*

Here, the  $AT_i(t)$  represents the abstract of the object  $O_i$ 's trajectory till the time  $t$ . According to the definition, the trajectories are continuously clustered for every timestamp. In practice, we do it periodically to reduce the system overhead. Specifically, we start a batch processing as long as all trajectory data generated within a time window  $[t_{ws}, t_{we}]$  are collected, and the duration of a time window is set as  $\delta t$ .

## 4 Real-Time Distributed Abstract Trajectory Clustering

In this section, we will first overview the framework of our proposed Disatra algorithm. Then, we will introduce the major techniques applied in each phase, respectively, including trajectory abstraction, Geohash indexing and Density-based Line Clustering (DLC).

### 4.1 Framework Overview

The Disatra algorithm is designed to achieve real-time trajectory clustering on the distributed streaming environment where the trajectory data keeps flowing into the system infinitely. Overall, the Disatra consists of three phases: data abstraction, data partitioning, and local clustering, depicted in Fig. 2.

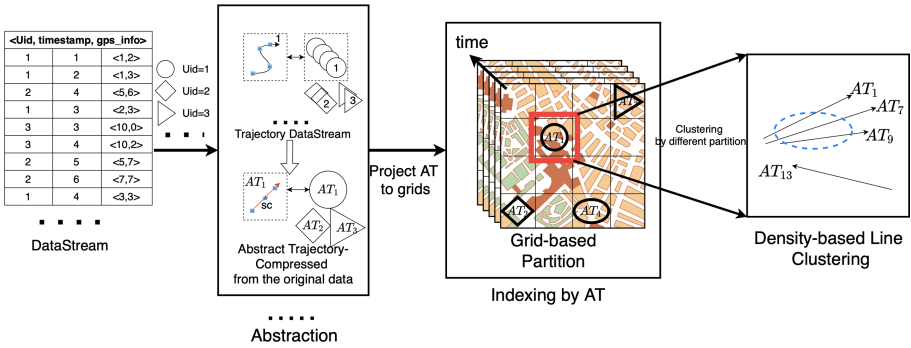


Fig. 2. Framework of Disatra

In data abstraction, the main objective is to compress trajectories with arbitrary lengths to uniform-size data structures, which we name as abstract trajectory. Each abstract trajectory summarizes the features of a trajectory till the current time, and it keeps updating as new trajectory points are received. Specifically, we first group the incoming points according to object id and append to the existing points of the same object, respectively. As shown in Fig. 2, the objects are depicted as different shapes (circles, diamonds, triangles, etc.), and the points of the same shape are grouped into a sequence. Then, in the trajectory abstraction process, the incoming sequences are either compressed into new abstract trajectories  $AT_i$  ( $i = 1, 2, 3, \dots$ ) (if no previous  $AT_i$  of the same object exists) or combined with existing  $AT_i$  and perform an update. Note that the abstraction process is done periodically according to the time window size  $\delta t$  to maximize the system throughput. Overall, by converting heterogeneous trajectories into uniform-size abstract trajectories, the trajectories can then be evenly distributed to processing nodes, which helps to solve the data skewness problem and maintain the load balancing in a distributed system.

In data partitioning, our goal is to distribute the abstract trajectories evenly to every node, in the mean time, the partitioning strategy should also make the best effort to avoid cross-partition clusters. Therefore, we introduce the Geohash method to index the abstract trajectories. With the help of the hash coding, the abstract trajectories can be evenly distributed to the processing nodes. More importantly, the abstract trajectories in each partition can be clustered separately without the need for cross-partition clustering.

Finally, in local clustering, we propose a density-based line clustering method that performs clustering on abstract trajectories in parallel. With the our partitioning strategy, the clustering is only done locally for each distributed node. Besides, the process is done periodically according to the size of the time windows. In the rest of this session, we will introduce the above techniques, respectively.

### 4.2 Trajectory Abstraction

Note that previous abstraction methods generate abstract for trajectory points from different objects in the same time period. Such solution is not conducive to distributed processing as the adjacent abstracts are related to each other and is hard for partitioning. Therefore, we focus on trajectory-level abstraction, which can be distributed easily.

Our trajectory abstraction aims to compress the input trajectory so as to process streaming data efficiently and ensure the load balance. Meanwhile, the compressed trajectory, termed as abstract trajectory, should capture the profile of the trajectory as much as possible so that the clustering on abstract trajectory can represent the clustering on original ones without loss of accuracy. Consider that a trajectory usually travels in a particular area at a certain speed and heading for a while, we propose the abstract trajectory which is a data tuple comprised of five attributes, i.e.  $AT_i = \{sc, \theta, bl, tr, t\}$ . Specifically, the  $sc$  is a calculated position which represents the center of the trajectory,  $\theta$  shows the overall heading,  $bl$  and  $tr$  encloses the trajectory moving area, and  $t$  records the current timestamp to divide the time window.

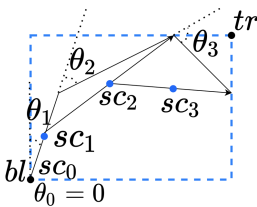


Fig. 3. Abstract trajectory.

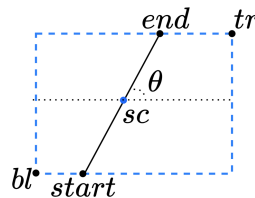


Fig. 4.  $bl$ & $tr$  value update process.

**Data Structure of Abstract Trajectory.** One AT is generated for each object in our method and keeps updating once new trajectory points of the same object

are received. An example of calculating the AT is shown in Figs. 3 and 4, initially, the  $sc$  in  $AT_i$  is set as the trajectory starting point  $sc_0$ . Each time a new point is received, the midpoint position  $(sc_1, sc_2, sc_3)$  is calculated by connecting it with its predecessor. The historical location  $sc$  is updated to the latest midpoint position, and the update is repeated iteratively. We call this historical location the Stateful Center. We also count the deflection angle  $(\theta_0, \theta_1, \theta_2, \theta_3, \dots, \theta_N)$  and add to  $\theta(\theta = \sum_{i=1}^N \theta_i)$ , which represents the abstract of angle. In general, a larger angle means the trajectory travels a circular route. Besides, we use the blue dashed rectangular box to enclose the trajectory. This rectangular box is called the Minimum Area Bounding Rectangle (MBR [23]), which is saved by its bottom-left ( $bl$ ) and its top-right ( $tr$ ) coordinates. Finally,  $t$  stores the timestamp of the latest point.

**Generation of Abstract Trajectory.** The  $\theta$  and  $sc$  provides an approximation of the object’s movement. However, it is not accurate enough, and a data structure with five attributes is too fractured to be clustered. Therefore, we need to integrate and output a specific abstract trajectory line segment composed of start and end endpoints for subsequent line-based clustering. Specifically, as shown in Fig. 4, we first find the position of  $sc$ , draw a line at the angle of  $\theta$  between this position and the horizontal line, and intersect with the MBR at  $start$  and  $end$ . If  $2k\pi + 0.5\pi \leq \theta < 2k\pi + 1.5\pi, k \in \mathbb{N}^+$ , we swap the values of  $start$  and  $end$ , and keep them unchanged otherwise. Then the values of  $bl$  and  $tr$  are updated to the values of  $start$  and  $end$ . After the process, we can output a line segment based on  $bl$  and  $tr$ . In addition to the line segment, the  $sc$  monitors the moving trend of the trajectory, which will be utilized in the subsequent partitioning process to ensure the isolation of partitions concerning clustering, while  $t$  is used to associate the AT to the correct time window.

In terms of the AT update, according to the definition of AT tuple, it is clear that the update does not require a re-summation of the existing AT. Instead, by continuing the calculation of  $sc$  and the summation of  $\theta$ , an incremental update can be easily achieved.

We use this incremental update to maintain the trajectory status information of each object. For  $M$  objects, the average trajectory of each object is composed of  $N$  coordinate points (uniform sampling). Then the complexity we need to deal with is  $O((MN)^2)$ , and when we simplify the representation of each object as an abstract trajectory, our complexity is lower to  $O(M^2)$ . We can say that it is much more efficient than the traditional non-compressed processing method.

### 4.3 GeoHash Indexing

As the  $sc$  from each AT represents the moving trend of a trajectory, it can be used as the key for indexing. Also, by representing an AT using an  $sc$  point, it can be partitioned without overlap. Thanks to the fact that the value of  $sc$  is derived from the historical state, the movement’s central area can be found easily. Here, we use GeoHash to partition and index the AT data.

The essence of GeoHash is grid-based indexing, which divides data into different grid partitions. GeoHash is a longitude and latitude coordinate encoding method, which can encode coordinate points into a string of characters so that the spatial relationship between the indexed items can be converted to the distance between strings. GeoHash can be used in technologies such as map indexing and range searching.

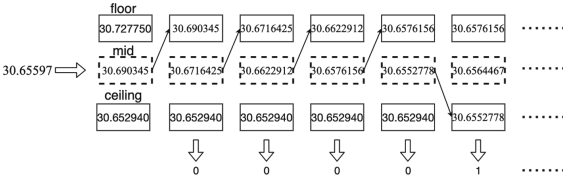


Fig. 5. GeoHash encoding.

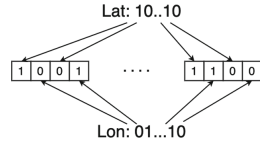


Fig. 6. Latitude and longitude coding.

As exemplified in Fig. 5, Geohash encoding first calculates two parameters *floor* and *ceiling* to determine the geographic boundary of the hash. Taking latitude as an example, we assume that  $floor = 30.727750$  and  $ceiling = 30.652940$ . Now given the value  $lat = 30.65597$ , we compare it with the median value of *floor* and *ceiling*, which is  $mid = 30.690345$ . Since  $lat < mid$ , we mark the first-bit code as 0 (otherwise, the output is 1), and then we update  $floor = mid$ . We repeat the above operations until the code reaches the specified digit length  $n$ , ( $n = 5$  in our example). We use this encoding method to encode longitude and latitude into binary strings of length  $n$ , respectively. Then, we interleave the longitude and latitude codes into a  $2n$  sequence, as shown in Fig. 6. Finally, we use Base32 to translate the encoding result into a string. A GeoHash-encoded string represents the geographic area where the geographic object exists. The length of the string represents the accuracy of the specific geographic location. By intercepting the character string’s prefix, the parent area to which it belongs can be obtained. The more digits it represents, the more detailed the area location is, and the more accurate range search can be achieved.

Our partitioning index via Geohash encoding is built based on *sc* values from ATs. Since the *sc* values are represented as geographical points, it can well-separate the trajectories into isolated partitions. Therefore, our index produces no cross-partition data, so we will not face the cross-node data transmission problem in the distributed/multi-threaded environment.

#### 4.4 Density-Based Line Clustering (DLC)

Before clustering, we first prune out partitions that are irrelevant to the clustering process. Specifically, we first perform density statistics on each grid partition.



Only data with sufficient density (according to the number of AT entries) are selected to participate in the clustering process.

As most of the areas formed by traffic jams are irregularly shaped, which motivates us to mine clusters of arbitrary shapes, we require a density-based clustering. Similar to the idea of DBSCAN, the core of density-based clustering is to find directly density-reachable clusters concerning the radius  $\epsilon$  and the density threshold  $\rho$ . The number of points within the threshold distance of a starting point reflects the density within the cluster. However, as we cluster the line segments instead of points, in this paper, we propose a Density-based Line Clustering (DLC) which consists of a line segment distance function.

The line segment distance measurement is mainly composed of three factors: the angle distance  $d_\theta$ , the vertical distance  $d_\perp$  and the horizontal length difference  $d_\parallel$ . The measurement result is a weighted linear combination of the three factors. As depicted in Fig. 1 where  $|l_j| \leq |l_i|$ , the formula is shown as follows (the shorter line is  $l_j$ ):

$$dist(l_i, l_j) = \omega_\perp \cdot d_\perp(l_i, l_j) + \omega_\parallel \cdot d_\parallel(l_i, l_j) + \omega_\theta \cdot d_\theta(l_i, l_j) \quad (1)$$

$$d_\perp = \frac{d_{\perp 1}^2 + d_{\perp 2}^2}{d_{\perp 1} + d_{\perp 2}} \quad (2)$$

$$d_\parallel = MIN(d_{\parallel 1}, d_{\parallel 2}) \quad (3)$$

$$d_\theta = \| l_j \| \times \sin(\theta) \quad (4)$$

Here,  $d_\perp$  represents the distance between the line segments, which can be understood as the judgment of the difference in the area where the line segment is located;  $d_\parallel$  represents the difference in travel distance, as the trajectories are captured during the same time period, such difference can reflect the difference in movement speed;  $d_\theta$  represents the difference in the direction of movement, if the angle is greater than  $90^\circ$ ,  $d_\theta$  will return a negative value, which means that two trajectories travel in the opposite directions, so the difference is significant.  $\omega_\perp$ ,  $\omega_\parallel$  and  $\omega_\theta$  are weights that are decided based on applications. As we treat each factor's importance equally, the values are set as 1 in our case.

The DLC algorithm requires the user to input two parameters: the radius threshold ( $\epsilon$ ) and density threshold ( $\rho$ ), which means that a cluster is a largest collection of density-connected ( $\epsilon$ ) line segments with sufficiently high density ( $\rho$ ) in line segment candidate space. Here we specify the line segment candidate space as line segments that exist in the same time window. To perform the clustering, given a data set  $L = \{l_1, l_2, \dots, l_n\}$  in one partition, we first randomly visit an unvisited arbitrary line  $l_i$  and mark it as visited. Then we calculate its distance to the other line segments according to the distance function, the density-reachable line segments found are then marked as visited and are included in the  $\epsilon$ -neighborhood set  $NH_\epsilon(l_i)$ . We continue to expand the line segments in the  $\epsilon$ -neighborhood set until a boundary is found and the cluster is finalized. Then we repeat the same process in the remaining unvisited line segments until all line segments are visited, which ends the clustering process.

## 5 Experiment

We implement our distributed abstract trajectory clustering in real-time streaming system. Our evaluations mainly focus on answering the following two questions.

- 1) How effective is the compression strategy in trajectory abstraction, in terms of efficiency and accuracy?
- 2) What is the overall throughput and latency of our framework?

### 5.1 Experiment Settings

**System Settings.** Our experiments are conducted on a local cluster, which we set five degrees of parallelism in most distributed scenarios. Specifically, the cluster runs CentOS 7.4 operating system and is equipped with 128 processors (Intel(R) Xeon(R) CPU E7-8860 v3 @ 2.20 GHz). Overall, our cluster provides 120 computing nodes and a 512-core environment. All algorithms are based on Flink<sup>1</sup> 1.9.0 and Java 8. We set the parallelism of the algorithm to 5, the Geohash code length is set to 10 bits, then the grid partition is about  $32 \times 32$  (a total of 1024 grids). The specific number of grid divisions and coding accuracy can be defined according to user needs.

**Dataset Settings.** Our dataset is a commercial dataset<sup>2</sup>, which contains about 3.73 GB of taxi GPS trajectory data from Didi in Chengdu on November 1, 2016. The longest trajectory is about 11.8 km, and the sampling interval is 3 s. The data analyzed in this paper is randomly taken from one hour of a day for real-time analysis (the maximum volume is about 250 MB). We sampled the data set multiple times to generate 1k, 10k, 100k, 1M and all data record samples for experiment. Since streaming data clustering mainly focuses on high throughput in real-time scenario, the time span of the dataset is not essential in this experiment. What we need to focus on is the efficiency of data processing at every moment.

**Baseline Method.** We choose TraClus [11], ST-DBSCAN [2], Swarm [12], Distributed-DBSCAN [21] (We abbreviate it as D2BSCAN) and Topology [21] as our baseline methods, respectively. TraClus is a partition-and-group framework for trajectory clustering. ST-DBSCAN is a classic algorithm for clustering spatio-temporal data which is similar to ours. Swarm is currently the simplest and most efficient mining mode in trajectory movement mode mining, so it is used for speed comparison with our algorithm. Distributed-DBSCAN and Topology are two specific application algorithms for trajectory aggregation detection. To clarify our trajectory compression effect, we calculated the abstraction/compression performance and visualized the results for verification by using different sample data. We also inspected the clustering latency and throughput to verify the clustering performance. We also use a swarm algorithm, a two-stage clustering for the

<sup>1</sup> <https://flink.apache.org>.

<sup>2</sup> <https://outreach.didichuxing.com/research/opendata/>.

spatial-temporal dimension attributes, and make the comparison concerned with the latency and throughput to prove the superiority of our algorithm. Besides, Silhouette Coefficient [14] is a metric used to judge the quality of clustering, which will become our standard for testing the clustering quality. The algorithm calculates the compactness of the same cluster and the interval between different clusters to judge whether the cluster is good or bad.

### 5.2 Results and Analysis

We first depict the clustering results of our method and ST-DBSCAN in Fig. 7 and Fig. 10. In the figures, the trajectories are represented as polylines, and the clusters are identified by different colors. The results show that our clustering results are clean with clear boundaries. Besides, we can find that Disatra’s is reliable in differentiating the objects with different movement trend. The trend divides the categories in the same geographic location into categories that show various sports trends, something ST-DBSCAN cannot do. Therefore, Disatra converts clusters formed by ST-DBSCAN into sub-categories representing different motion trends. Regarding the performance, we use silhouette coefficient to evaluate the quality of clustering results. The Disatra’s silhouette coefficient is 0.3 whereas the best silhouette coefficient that we can achieve on ST-DBSCAN is 0.37 (score range is  $-1$  to  $1$ , the higher the better). Overall, comparing with ST-DBSCAN, we sacrifice little correctness to obtain nearly a hundred times more efficiency in terms of time and space consumptions, which is worthwhile.

**Abstraction Performance and Memory Cost.** In this section, we process the data in each period (one hour a period) and compare the changes in the data volume before and after abstraction to show the advantages of abstraction for real-time data processing.

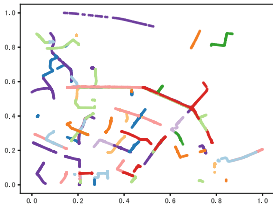


Fig. 7. Disatra

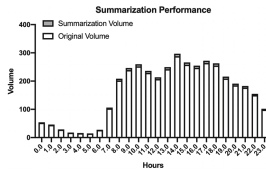


Fig. 8. Abstraction performance

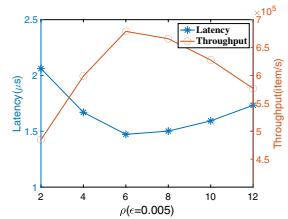


Fig. 9. The impact of  $\rho$  on time cost

Figure 8 shows the change in data capacity of different data volumes in different time periods before and after abstraction, in units of 10,000. We can clearly

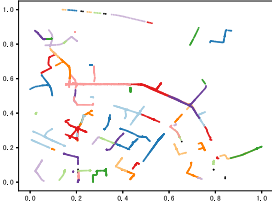


Fig. 10. ST-DBSCAN

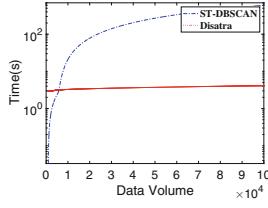


Fig. 11. Time cost comparison

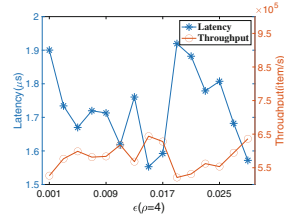


Fig. 12. The impact of  $\epsilon$  on time cost

find that the two data lines are fitted together, which means that our compression capacity is stabilized at 30 times. With this technology, we can stably compress the data volume by 30 times, which is undoubtedly a great help for real-time processing. Figure 11 illustrates the running time comparison on data of different magnitudes. We can find that the time overhead of ST-DBSCAN increases exponentially with the increase in the amount of data. The overall time overhead of Disatra doesn't scale with the data volume and tends to be stable. Disatra's time overhead is bigger than ST-DBSCAN when the input is small (within about 10,000), limited by the window mechanism of the data stream. However, the increase in data volume has minimal impact on Disatra's performance from the overall operation perspective.

From Table 2, we can see the experiments done under data of different magnitudes. Our framework's memory consumption is much smaller than that of ST-DBSCAN. The memory consumption of ST-DBSCAN is 189.6 GB when processing 100,000 pieces of data, and the consumption when processing 10,000 pieces of data is 2 GB, this increment can be described as an explosive rise in demand. When Disatra processes over 2 million pieces of data, only 812 MB of memory is used.

Table 2. ST-DBSCAN v.s. Disatra

Method	Data volume	Time cost(s)	Memory cost
ST-DBSCAN	97 KB	0.033	71.2 MB
	970 KB	5.277	2 GB
	9.7 MB	600.418	189.6 GB
Disatra	97 KB	0.322	123 MB
	970 KB	0.557	238 MB
	9.7 MB	1.134	334 MB
	206.7 MB	5.01	812 MB

**Throughput and Time Cost.** As show in Fig. 9 and Fig. 12, We chose Traclus as the trajectory clustering’s baseline and we also compared the distributed processing of swarm [12] by using a random partition strategy with this experiment since the processing time of swarm is extremely fast. We also adopted Distributed-DBSCAN and Topology which are both used for real-time traffic jam detection in [21]. Obviously, our algorithm has remarkable characteristics of high throughput and low latency in trajectory clustering. From Table 3, it can be seen that our algorithm is extremely faster than our counterpart-Distra only cost 1.72  $\mu$ s. Furthermore, we have deleted too long motion trajectory data from the user dimension since we are talking about compressing individual user trajectories into an abstract tuple. Therefore, we solve the data skew in the user dimension. The specific situation is if the start-end position of the two users is the same, but one of the users detours but does not deviate from the main route. At this time, the data comparison between the two parties may cause a great join operator overhead due to the inconsistent data length. The trajectory compression in this technology can solve this user-dimensional data skew.

**Table 3.** Performance comparison

Method	Throughput (items/sec)	Latency ( $\mu$ s)
D2BSCAN	1.2k	830
Swarm	5k	175
Traclus	8k	120
Topology	10k	50
Disatra	579k	1.72

## 6 Conclusion and Future Work

This paper studied the real-time distributed clustering of massive trajectory data at the city level. We proposed a new framework-Disatra, which is a distributed real-time computing clustering framework. Through the experimental performance analysis of real data sets and the comparison with existing trajectory stream clustering methods, we can see the advantages in real-time performance by abstracting trajectories, load balancing, and distributed deployment. Besides, we can find through the silhouette coefficient that the clustering quality of Disatra is still very high when only local related data is considered in a distributed environment. Therefore, we can say that Disatra is a high-precision real-time distributed trajectory clustering method suitable for real-time calculation of massive trajectory data. Our follow-up work expects to combine Disatra with motion pattern recognition in order to study crowd behaviour.

**Acknowledgments.** This work was supported by National Natural Science Foundation of China under grant (No. 61802273), Postdoctoral Science Foundation of China (No. 2020M681529), a project funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).

## References

1. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for clustering evolving data streams. In: VLDB, pp. 81–92 (2003)
2. Birant, D., Kut, A.: ST-DBSCAN: an algorithm for clustering spatial-temporal data. *Data Knowl. Eng.* **60**(1), 208–221 (2007)
3. Chen, M., Liu, Y., Yu, X.: Predicting next locations with object clustering and trajectory clustering. In: Cao, T., Lim, E.-P., Zhou, Z.-H., Ho, T.-B., Cheung, D., Motoda, H. (eds.) PAKDD 2015, Part II. LNCS (LNAI), vol. 9078, pp. 344–356. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-18032-8\\_27](https://doi.org/10.1007/978-3-319-18032-8_27)
4. Cheung, D.W., Xiao, Y.: Effect of data skewness in parallel mining of association rules. In: Wu, X., Kotagiri, R., Korb, K.B. (eds.) PAKDD 1998. LNCS, vol. 1394, pp. 48–60. Springer, Heidelberg (1998). [https://doi.org/10.1007/3-540-64383-4\\_5](https://doi.org/10.1007/3-540-64383-4_5)
5. Ester, M., Kriegel, H., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD, pp. 226–231 (1996)
6. Gudmundsson, J., van Kreveld, M.J.: Computing longest duration flocks in trajectory data. In: ACM-GIS, pp. 35–42 (2006)
7. Januzaj, E., Kriegel, H.-P., Pfeifle, M.: Scalable density-based distributed clustering. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) PKDD 2004. LNCS (LNAI), vol. 3202, pp. 231–244. Springer, Heidelberg (2004). [https://doi.org/10.1007/978-3-540-30116-5\\_23](https://doi.org/10.1007/978-3-540-30116-5_23)
8. Jeung, H., Yiu, M.L., Zhou, X., Jensen, C.S., Shen, H.T.: Discovery of convoys in trajectory databases. VLDB **1**(1), 1068–1080 (2008)
9. Khaing, H.S., Thein, T.: An efficient clustering algorithm for moving object trajectories. In: ICCTAI, pp. 11–12 (2014)
10. Lee, J., Han, J., Li, X., Gonzalez, H.: TraClass: trajectory classification using hierarchical region-based and trajectory-based clustering. VLDB **1**(1), 1081–1094 (2008)
11. Lee, J., Han, J., Whang, K.: Trajectory clustering: a partition-and-group framework. In: SIGMOD, pp. 593–604 (2007)
12. Li, Z., Ding, B., Han, J., Kays, R.: Swarm: mining relaxed temporal moving object clusters. VLDB **3**(1), 723–734 (2010)
13. Li, Z., Lee, J.-G., Li, X., Han, J.: Incremental clustering for trajectories. In: Kitagawa, H., Ishikawa, Y., Li, Q., Watanabe, C. (eds.) DASFAA 2010, Part II. LNCS, vol. 5982, pp. 32–46. Springer, Heidelberg (2010). [https://doi.org/10.1007/978-3-642-12098-5\\_3](https://doi.org/10.1007/978-3-642-12098-5_3)
14. Lian-Jiang, Z., Bing-Xian, M.A., Xue-Quan, Z.: Clustering validity analysis based on Silhouette coefficient. *J. Comput. Appl.* **30**, 139–141 (2010)
15. Mao, J., Song, Q., Jin, C., Zhang, Z., Zhou, A.: TScluWin: trajectory stream clustering over sliding window. In: Navathe, S.B., Wu, W., Shekhar, S., Du, X., Wang, X.S., Xiong, H. (eds.) DASFAA 2016, Part II. LNCS, vol. 9643, pp. 133–148. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-32049-6\\_9](https://doi.org/10.1007/978-3-319-32049-6_9)
16. Pan, B., Zheng, Y., Wilkie, D., Shahabi, C.: Crowd sensing of traffic anomalies based on human mobility and social media. In: SIGSPATIAL, pp. 334–343 (2013)

17. Song, H., Lee, J.: RP-DBSCAN: a superfast parallel DBSCAN algorithm based on random partitioning. In: SIGMOD, pp. 1173–1187 (2018)
18. Suwardi, I.S., Dharma, D., Satya, D.P., Lestari, D.P.: Geohash index based spatial data model for corporate. In: ICEEL, pp. 478–483. IEEE (2015)
19. Wang, Y., Lim, E., Hwang, S.: Efficient mining of group patterns from user movement data. *Data Knowl. Eng.* **57**(3), 240–282 (2006)
20. Wang, Y., Qin, K., Chen, Y., Zhao, P.: Detecting anomalous trajectories and behavior patterns using hierarchical clustering from taxi GPS data. *ISPRS* **7**(1), 25 (2018)
21. Yang, Q., Yue, Z., Chen, R., Zhang, J., Hu, X., Zhou, Y.: Real-time detection of traffic congestion based on trajectory data. *J. Eng.* **2019**(11), 8251–8256 (2019)
22. Zheng, Y.: Trajectory data mining: an overview. *ACM Trans. Intell. Syst. Technol.* **6**(3), 291–2941 (2015)
23. Zheng, Y., Zhou, X. (eds.): *Computing with Spatial Trajectories*. Springer, Heidelberg (2011). <https://doi.org/10.1007/978-1-4614-1629-6>



# Extra-Budget Aware Task Assignment in Spatial Crowdsourcing

Shuhan Wan, Detian Zhang<sup>(✉)</sup>, An Liu, and Junhua Fang

Institute of Artificial Intelligence, School of Computer Science and Technology,  
Soochow University, Suzhou, China  
20195227042@stu.suda.edu.cn, {detian, anliu, jhfang}@suda.edu.cn

**Abstract.** With the prevalence of sharing economy and mobile Internet, spatial crowdsourcing (SC) has been receiving increased attentions recently. A core issue in SC is task assignment, which aims to assign tasks to suitable workers. As workers need to reach the corresponding locations to complete the tasks, they prefer tasks nearby to save travel cost. Therefore, most of the existing solutions for task assignment give workers a fixed range constraint. However, those solutions do not consider the tasks that out of the range, which may make these remote tasks never been completed. Therefore, in this paper, we propose a new problem called extra-budget aware task assignment (EBATA) in spatial crowdsourcing, where extra budget is provided to subsidize the over cost of workers to ensure that the remote tasks have a chance to be accomplished. To address the EBATA problem, two baseline algorithms and two improved greedy algorithms are devised in the paper. The two improved greedy algorithms can heavily reduce the computational time and keep most of the number of matched pairs with the optimal one. Extensive experiments on real dataset verify the effectiveness and efficiency of the proposed methods.

**Keywords:** Spatial crowdsourcing · Task assignment · Extra budget

## 1 Introduction

With the development of mobile devices and wireless networks, spatial crowdsourcing (SC) has attracted more and more attentions from industry and academia. Typical spatial crowdsourcing services include online car-hailing services (e.g., Uber and Didi), take-out services (e.g., Grubhub and Ele.me), citizen sensing services (e.g., Waze), and so on. Figure 1 shows a typical example of spatial crowdsourcing system. The SC platform receives taxi-calling tasks from passengers, and then assigns these tasks to the current available driver workers based on a predefined task assignment algorithm. After that, the drivers need to pick up the assigned passengers to complete the tasks.

Task assignment, which is a core issue in spatial crowdsourcing, aims to assign tasks to the suitable workers based on given constraints, e.g., spatial, temporal



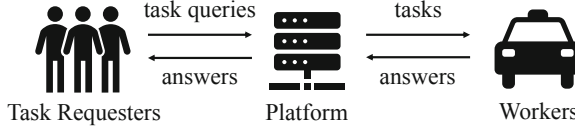


Fig. 1. Spatial crowdsourcing system

and other constraints [14]. The most widely-used one is range constraint, which is one type of spatial constraints. As workers need to reach the corresponding locations to complete the tasks, they prefer tasks nearby to save travel cost. Therefore, most of the existing task assignment algorithms give workers a fixed range constraint [3, 7, 11, 13], i.e., only the tasks within the range have the chance to be processed by the worker. However, these methods do not consider the tasks that out of the range, which makes the remote tasks may never be processed.

Therefore, in this paper, we investigate the task assignment of spatial crowdsourcing under such a problem setting, namely Extra-Budget Aware Task Assignment (EBATA). More specifically, two kinds of range constraint are considered in our setting, i.e., the fixed range constraint which is set by the platform and the same to all of the tasks, and the extra range constraint which depends on the extra budget paid by the task requesters and is different based on different tasks. Extra budget is used for subsidizing the over cost of workers to ensure that the remote tasks have a chance to be accomplished. Taking online taxi-calling service for example, a passenger in a remote place can provide an extra budget to attract a driver worker to pick up him.

Because EBATA problem has two range constraints, i.e., a fixed one and an extra one, instead of only one constraint as existing task assignment problem, most of existing algorithms can not be directly utilized in our problem. Therefore, in this paper, we propose two baseline algorithms and two effective improved greedy algorithms to address the EBATA problem. Extensive experiment results show the efficiency and effectiveness of our approach.

## 2 Problem Definitions

**Definition 1 (Task).** A task is denoted by  $t = \langle l_t, p_t, r_t, b_t \rangle$ , where  $l_t$  is the location of  $t$  in a 2D space at timestamp  $p_t$  and  $r_t$  is the radius of  $t$  centered in  $l_t$  which is the fixed range constraint of  $t$ , and  $b_t$  is the extra budget of  $t$ .

**Definition 2 (Worker).** A worker is denoted by  $w = \langle l_w, p_w \rangle$ .  $l_w$  is the location of  $w$  in a 2D space at timestamp  $p_w$ .

**Definition 3 (Travel cost).** The travel cost, denoted by  $\text{cost}(t, w)$ , is the cost to travel from  $l_w$  to  $l_t$ .

**Definition 4 (Extra travel cost).** The extra travel cost, which is the part that exceeds the fixed range constraint of travel cost, is denoted by  $e_t = \text{cost}(t, w) - r_t$ , where  $\text{cost}(t, w)$  is the Euclidean distance from  $l_w$  to  $l_t$ .

**Definition 5** (*Extra-Budget Aware Task Assignment (EBATA) problem*). A set of tasks  $T$  and a set of workers  $W$  are given in this scenario. The matched pair set  $M$  represents a series of arrays containing task-worker pairs. The EBATA problem is to find an  $M$  to minimize the total extra travel cost, i.e.  $\text{Minimum}(E) = \sum_{t \in T} e_t$  under the premise of maximizing the total number of matched pairs, i.e.  $\text{Maximum}(M)$ , where  $E$  is the total extra travel cost.

### 3 Baseline Algorithms

#### 3.1 The Optimal Solution

In this section, we use the minimum-cost maximum-flow algorithm [2] to solve the EBATA problem.

**Theorem 1.** *The Optimal solution for the EBATA problem is reducible to the minimum-cost maximum-flow problem.*

*Proof.*  $T = \{t_1, t_2, \dots\}$  and  $W = \{w_1, w_2, \dots\}$  are given as the set of tasks and workers respectively. We conduct a flow network graph  $G = (V, S)$ . The set  $V$  contains  $|T| + |W| + 2$  vertices including vertices of task  $t_i$  marked by  $V_i$ , vertices of worker  $w_j$  marked by  $V_{|T|+j}$ , source vertex marked by  $V_0$  and destination vertex marked by  $V_{|T|+|W|+1}$ . The set  $S$  contains  $|T| + |W| + m$  edges and the capacity of each edge is set to 1. The cost of each edge which connects  $V_0$  to  $V_i$  and  $V_{|T|+j}$  to  $V_{|T|+|W|+1}$  would be 0. We assume edges connecting tasks and workers within the total range constraint ( $r_t + b_t$ ) amount to  $m$ , with the cost of each edge is set to their extra travel cost. Thus, we can run the minimum-cost maximum-flow algorithm [2] in  $G$  to obtain the result.

According to Theorem 1, We first utilize the information of tasks and workers to build a bipartition graph. Then we run the traditional minimum-cost maximum-flow algorithm on the graph to get the result.

**Complexity Analysis.** The minimum-cost maximum-flow algorithm spends  $O(|V|^2)$  time in each call to find a new flow from  $s$  to  $d$ . Thus, the time complexity of the Optimal algorithm is  $O(V^2 * |M|)$ , where  $V = |T| + |W| + 2$  and  $|M|$  is the total flows obtained by the minimum-cost maximum-flow algorithm.

#### 3.2 A Simple Greedy Algorithm

The Optimal algorithm can attain the best results. However, the solution bears a high computational cost as it needs to run the Dijkstra algorithm on a huge graph. A low-cost method is to use a greedy algorithm, which greedily assigns a given task to the worker within the total range constraint who incurs the smallest travel cost. To solve EBATA more efficiently and accurately, we will propose two improved greedy algorithms in the next section.

## 4 The Improved Greedy Algorithms

### 4.1 The Greedy Algorithm with Fewer Workers First

The workers in the total range constraint of a task are labeled as candidate workers for this task. If a task has several candidate workers, it is highly likely to be assigned with a suitable worker. On the contrary, those tasks have a few candidate workers are less likely to be matched with suitable workers, who are instead assigned to other nearby tasks. Consequently, the tasks with fewer workers around should be assigned first to increase the overall quantity of matched pairs. Based on this point, we propose the Greedy Algorithm with Fewer Workers First (G-FWF). The tasks are sorted in ascending order according to the number of their candidate workers and the higher ranked tasks have higher priority in the task selection process.

---

**Algorithm 1** Greedy Algorithm with Fewer Workers First

---

```

Input: a set of tasks  $T$ , a set of workers  $W$ 
Output: the matched pair set  $M$ , the unassigned task set  $T'$ , the unassigned worker set  $W'$ 
1:  $M \leftarrow \emptyset$ 
2: for each task  $t_i \in T$  do
3:   for each worker  $w_j \in W$  do
4:     if  $w_j$  is in the total range constraint of  $t_i$  then
5:       total number of candidate workers  $c_i \leftarrow c_i + 1$ 
6: sorting all the tasks in ascending order based on  $c_i$ 
7: for each task  $t_i \in T$  according to the obtained order do
8:   select the worker  $w_j$  with the smallest travel cost
9:   if  $cost(t_i, w_j) < r_t + b_t$  then
10:     $M \leftarrow M + \langle t_i, w_j \rangle$ ,  $E \leftarrow E + e_t$ 
11:    remove  $t_i$  from  $T$ , remove  $w_j$  from  $W$ 
12: return  $M, E, T', W'$ 

```

---

Algorithm 1 illustrates the procedure of G-FWF. In lines 1–5, total number of candidate workers  $c_i$  of each tasks are calculated. In line 6, we sort all tasks in an ascending order based on the number of their candidate workers. In lines 7–11, the algorithm goes through the candidate worker set to get the nearest worker for each task. In line 12, the algorithm returns the result.

**Complexity Analysis.** The nested loop cost is  $O(|T| * |W|)$ , where  $|T|$  is the number of tasks and  $|W|$  is the number of workers. The sorting cost is  $O(|T|)$ . The time complexity of this algorithm is  $O(|T|^2 * |W|)$ .

### 4.2 The Greedy Algorithm with Incremental Search

The matched pairs found by G-FWF algorithm are more than those found by G-Simple. However, the extra travel cost could be rather large. This is because in G-FWF, these tasks that would have been able to be matched to the nearby workers are matched to distant workers based on the proposed priority. Therefore, we propose the Greedy Algorithm with Incremental Search (G-IncSearch) which has two steps. It aims to overcome the shortcoming of G-FWF.

The main idea of this algorithm is to first assign as many tasks as possible, and then reduce the extra travel cost. To reduce the number of tasks that may incur extra travel cost, the tasks within the fixed range constraint are assigned first. In the first step of this algorithm, we use the idea of G-FWF. More specifically, we replace the total range constraint with the fixed range constraint in line 4 and 9 of Algorithm 1. Then, in the second step, we consider the extra budget of remaining tasks which have not been assigned. We increase the extra range constraint incrementally in each round. The overall maximum extra range constraint is  $b^*$ , and the overall extra range constraint for each round is  $d = \frac{b^*}{n} \times k$ , where  $n$  is the total number of rounds given by the platform and  $k$  is the index of round.

---

**Algorithm 2** The second step of G-IncSearch: incremental extra range constraint

---

**Input:** a set of unassigned tasks  $T'$ , a set of unassigned workers  $W'$ , the matched pair set  $M$

**Output:** the matched pair set  $M$ , the total extra travel cost  $E$

- 1:  $E \leftarrow 0$
  - 2: **for** the overall extra range constraint  $d$  increases linearly from  $\frac{b^*}{n}$  to  $b^*$  **do**
  - 3:     **for** each task  $t_i \in T'$  **do**
  - 4:         **if**  $d < b_{t_i}$  **then**
  - 5:             select the worker  $w_j$  with the smallest travel cost
  - 6:             **if**  $cost(t_i, w_j) < (r_{t_i} + d)$  **then**
  - 7:                  $M \leftarrow M + \langle t_i, w_j \rangle$ ,  $E \leftarrow E + e_t$
  - 8:                 remove  $t_i$  from  $T'$ , remove  $w_j$  from  $W'$
  - 9: **return**  $M, E$
- 

Algorithm 2 illustrates the second step of G-IncSearch. In lines 1–5, the overall extra range constraint is incrementally added to the unassigned tasks. If the overall extra range constraint of this round is smaller than the extra range constraint of the unassigned task, the algorithm seeks for the nearest unassigned worker for the task by going through the unassigned worker set. In lines 6–8, the algorithm will check the pairs are valid or not for each round based on the travel cost between the task and worker. In line 9, the algorithm returns the result.

**Complexity Analysis.** The time complexity of the second step of G-IncSearch is  $O(n * |T| * |W|)$ , where  $|T|$  and  $|W|$  are the number of unassigned tasks and

workers in the worst case, and  $n$  is the total number of rounds. The overall time complexity of G-IncSearch is  $O(\max(|T|^2 * |W|, n * |T| * |W|))$ .

## 5 Experiments

### 5.1 Experimental Setup

**Datasets.** We use real dataset (i.e., Didi Chuxing dataset from [1]) to evaluate our proposed EBATA approaches. We use the pick-up location and the start time of billing as the task location and calling-time respectively. Since the worker becomes available again after the passenger getting off the car, the drop off location and the end billing time is used as the worker location and available time.

**Compared Algorithms.** We take Optimal solution (OPT) and Simple Greedy Algorithm (G-Simple) as baselines, and compare them with the Greedy Algorithm with Fewer Workers First (G-FWF), the Greedy Algorithm with Incremental Search (G-IncSearch) in terms of running time, number of matched pairs, average travel cost and average extra travel cost.

### 5.2 Results on Real Dataset

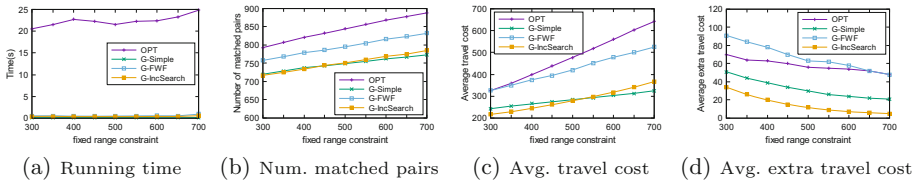


Fig. 2. Effect of the fixed range constraint on real dataset

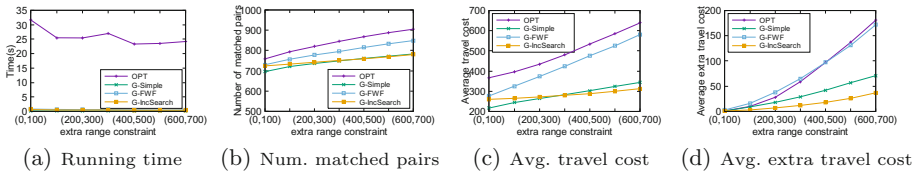


Fig. 3. Effect of the extra range constraint on real dataset

**Effect of the Fixed Range Constraint.** In Fig. 2(a), we can observe that with the increase of the fixed range constraint, the running time of all the algorithms increases accordingly. The reason is that more valid task-worker pairs need to be processed. We can see OPT has the longest running time, because OPT needs much time on the repeated running of Dijkstra algorithm. As shown in Fig. 2(b), the number of matched pairs of all the algorithms increases when the fixed range constraint increases. This is because a task can be reached by more candidate workers. When the fixed range constraint increases, the average travel cost increases accordingly as shown in Fig. 2(c). The reason for this is that the workers need to reach tasks located far away. G-IncSearch outperforms other algorithms. As shown in Fig. 2(d), the average extra travel cost of all the algorithms decreases along with the increase of the fixed range constraint, because more tasks can be completed within the range. Similarly, G-IncSearch outperforms other algorithms.

**Effect of the Extra Range Constraint.** In Fig. 3(a), the running time of all the algorithms increases. As shown in Fig. 3(b), the number of matched pairs of all the algorithms increases when the extra range constraint increases. This is because the larger the extra range constraint is, the more workers a task can be reached by. In Fig. 3(c), with the increase of extra range constraint, the average travel cost of all algorithms increases correspondingly. This is because more tasks are assigned to workers far away. In Fig. 3(d), as the extra range constraint increases, the average extra travel cost of all algorithms increases. This is because more tasks can be assigned to farther workers within the extra range constraint. We can see G-IncSearch has the smallest extra travel cost, which proves the effectiveness of our proposed algorithm.

## 6 Related Work

Task assignment aims to assign tasks to workers based on different objectives. One objective is to maximize the total number of completed tasks [5, 9, 12]. Kazemi et al. [5] transfer the bipartite graph into the maximum flow problem [2] and use the Ford-Fulkerson algorithm [4] to obtain the exact result. Some studies focus on maximizing the total expected payoff [10], which depends on the payoff of tasks to workers and the success ratio. Tong et al. [10] develop the Greedy-RT algorithm [8] by randomly choosing a threshold and adding an edge among the ones whose weights are no less than the threshold. With the purpose of minimizing the total travel distance of workers [5, 10], Tong et al. [10] greedily assign each new task to nearest unmatched worker.

In real world, the supply and the demand often vary in space and time. Developed by Uber, an effective solution is surge pricing. The unit fare is changed by an incentive mechanism to attract more workers [14]. Tong et al. use bipartite graphs to model the Global Dynamic Pricing (GDP) problem [11]. Some studies take workers' reputation and willingness into consideration. Yu et al. [15] and Miao et al. [6] model the quality of workers with their reputation. The worker

with higher reputation will obtain higher reward. However, these works mainly study determining the reward for workers before the assignment, while our work focuses on the task assignment with tasks having different budget to reward workers.

## 7 Conclusion

In this paper, we formalize the problem of extra-budget aware task assignment (EBATA) problem, in which tasks with extra budget are assigned to right workers. We propose two baseline algorithms and two improved greedy algorithms. Extensive experiments have been conducted to show the efficiency and effectiveness of our proposed algorithms on a real dataset.

**Acknowledgments.** This work was supported in part by the National Natural Science Foundation of China under Project 61702227 and 61802273, in part by a project funded by the Postdoctoral Science Foundation of China (No.2020M681529), and in part by a project funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions.

## References

1. Didi chuxing data. <https://gaia.didichuxing.com>
2. Ahuja, R.K., Magnanti, T.L., Orlin, J.B., Weihe, K.: Network flows: theory, algorithms and applications. *ZOR-Methods Models Oper. Res.* **41**(3), 252–254 (1995)
3. Cheng, P., Lian, X., Chen, L., Han, J., Zhao, J.: Task assignment on multi-skill oriented spatial crowdsourcing. *TKDE* **28**(8), 2201–2215 (2016)
4. Ford, L.R., Fulkerson, D.R.: Maximal flow through a network. *Can. J. Math.* **8**, 399–404 (1956)
5. Kazemi, L., Shahabi, C.: Geocrowd: enabling query answering with spatial crowdsourcing. In: *SIGSPATIAL*, pp. 189–198 (2012)
6. Miao, C., Yu, H., Shen, Z., Leung, C.: Balancing quality and budget considerations in mobile crowdsourcing. *Decis. Support Syst.* **90**, 56–64 (2016)
7. Sun, D., et al.: Online delivery route recommendation in spatial crowdsourcing. *World Wide Web* **22**(5), 2083–2104 (2018). <https://doi.org/10.1007/s11280-018-0563-4>
8. Ting, H.F., Xiang, X.: Near optimal algorithms for online maximum edge-weighted b-matching and two-sided vertex-weighted b-matching. *TCS* **607**, 247–256 (2015)
9. Tong, Y., She, J., Ding, B., Chen, L., Wo, T., Xu, K.: Online minimum matching in real-time spatial data: experiments and analysis. *PVLDB* **9**(12), 1053–1064 (2016)
10. Tong, Y., She, J., Ding, B., Wang, L., Chen, L.: Online mobile micro-task allocation in spatial crowdsourcing. In: *ICDE*, pp. 49–60. *IEEE* (2016)
11. Tong, Y., Wang, L., Zhou, Z., Chen, L., Du, B., Ye, J.: Dynamic pricing in spatial crowdsourcing: a matching-based approach. In: *SIGMOD*, pp. 773–788 (2018)
12. Tong, Y., et al.: Flexible online task assignment in real-time spatial data. *PVLDB* **10**(11), 1334–1345 (2017)
13. Tong, Y., Zhou, Z.: Dynamic task assignment in spatial crowdsourcing. *SIGSPATIAL Spec.* **10**(2), 18–25 (2018)

14. Tong, Y., Zhou, Z., Zeng, Y., Chen, L., Shahabi, C.: Spatial crowdsourcing: a survey. *VLDBJ* **29**(1), 217–250 (2020)
15. Yu, H., Miao, C., Shen, Z., Leung, C.: Quality and budget aware task allocation for spatial crowdsourcing. In: *AAMAS*, pp. 1689–1690 (2015)





# Expert Recommendations with Temporal Dynamics of User Interest in CQA

Xiaoqi Lv<sup>1,2</sup>, Ke Ji<sup>1,2</sup>(✉), Zhenxiang Chen<sup>1,2</sup>, Kun Ma<sup>1,2</sup>, Jun Wu<sup>3</sup>,  
Yidong Li<sup>3</sup>, and Guandong Xu<sup>4</sup>

<sup>1</sup> School of Information Science and Engineering, University of Jinan,  
Jinan 250022, China

<sup>2</sup> Shandong Provincial Key Laboratory of Network Based Intelligent Computing,  
University of Jinan, Jinan 250022, China

[ise\\_jik@ujn.edu.cn](mailto:ise_jik@ujn.edu.cn)

<sup>3</sup> School of Computer and Information Technology, Beijing Jiaotong University,  
Beijing 100044, China

<sup>4</sup> Data Science and Machine Intelligence Lab, Advanced Analytics Institute,  
University of Technology Sydney, Ultimo, Australia

**Abstract.** Community question answering (CQA) has become an essential service of promoting knowledge sharing in social platforms. To make question answering more efficient, several expert recommendation methods for CQA have been proposed, but most of them focus on the similarity matching between user interest and question content while ignoring the temporal dynamics of user interest, whose changes may decrease the quality of recommendation results. In this paper, a long and short term-based expert recommendation model (LSTERM) via attention mechanism-based CNN and Bi-GRU, which considers not only user interest but also user expertise, is proposed. The model can learn the embedded user/question feature representation from various content information by using attention mechanism-based CNN and then track the change of user interest and expertise over time by using Bi-GRU. Experiment results on real data demonstrate that with temporal dynamics, the recommendation accuracy is substantially improved compared with other state-of-the-art methods.

**Keywords:** Expert recommendation · Community question answering · Deep learning · Attention mechanism · Recurrent neural network

## 1 Introduction

Knowledge sharing services have become the main tool for people to obtain information. Numerous community question answering (CQA) platforms represented by Zhihu have emerged to improve the efficiency of Internet knowledge sharing [1]. However, CQA has accumulated a large number and various types of questions in a short time. How to recommend questions effectively to users who

may answer them and to determine questions that users are interested in are the problems that need to be solved urgently. Thus, several expert recommendation methods for CQA have been proposed to improve the efficiency of answering [2].

Most methods are content-based recommendations [3] using profile similarity and topic feature similarity. The matching effect depends on the quality of artificially constructed features. In recent years, deep learning represented by convolutional neural network (CNN) and attention mechanism have been successfully applied to text mining. Compared with traditional methods, deep models can learn more expressive deep, complex semantic features, which greatly improve the accuracy [4–6]. Although some progress has been made in the above work, they are all based on the user’s long-term following topics and answer history to learn user interests. Users focus and the questions they are good at answering are likely to change. These dynamic changes will seriously affect the effectiveness of the recommendation. In recent years, the recurrent neural network (RNN) has been widely used to process sequence data [7]. This type of method can be combined with CNN to process question content sequence data, mine dynamic interest from answering behaviors.

In this paper, an expert recommendation model, namely, long and short term-based expert recommendation model (LSTERM) for CQA is proposed. The proposed method mainly includes two parts: question encoder and user encoder. The question encoder is based on the pretrained word embedding model, which can realize the semantic embedding representation of the question text and topic tags, and combines the CNN model with the attention mechanism to generate deep feature encodings of different distance contexts. The user encoder uses double-layer Bi-GRU to process the encoded question sequence by the question encoder according to the user’s answering history sequence and combines the user topic tag and user expertise embedding vector to construct the user dynamic vector. The question encoder and user encoder are used to generate deep features and perform recommendation results calculation. The results of comparative experiments on the Zhihu dataset show that LSTERM performs considerably better than state-of-the-art deep learning expert recommendation methods.

## 2 Related Works

### 2.1 Expert Recommendation

Expert recommendation is a special application of recommendation system, which is used to find expert users who can solve problems. Researchers conducted a series of studies on expert recommendation in CQA. Many works proposed content-based recommendation methods, in which the recommendation results are mainly generated based on the matching degree between user interests and question content [8]. In recent years, deep learning has been applied to recommendation systems, and several neural network methods have emerged, which can learn more expressive semantic features and greatly improve the accuracy of traditional recommendation methods, such as DeepFM [4], XDeepFM [5], and CNN-DSSM [6].

## 2.2 Deep Neural Network

Deep learning is one of the most popular research in machine learning. It can combine low-level features through deep network structures to form high-level semantic features. Typical deep learning models include CNN and RNN. CNN can implicitly learn local weight sharing. RNN is suitable for processing sequence data and mining dynamic behavior. More distinctive bidirectional variant models Bi-LSTM and Bi-GRU [9] are gradually derived, which can flexibly control long and short distance dependent information. In recent years, composite neural networks based on RNN and CNN [7] have been widely used. The attention mechanism is also used in deep learning [10], which can flexibly capture global and local connections; thus, the model has different attention to features.

## 3 Our Approach

### 3.1 Question Encoder

**Question Title and Content Representation** Figure 1 shows the structure of question encoder. The first step is to segment the title and use word embedding. The title is converted into a word vector representation in the implicit semantic space. The word sequence in the title  $t$  is denoted as  $t = [t_1, t_2, t_3, \dots, t_M]$ . The transformed word sequence is represented, as shown in (1).

$$T = \epsilon(t) = [T_1, T_2, T_3, \dots, T_M] \tag{1}$$

Similarly, question content can be described as  $d$ , then the word embedding sequence  $D = \epsilon(d) = [D_1, D_2, D_3, \dots, D_N]$  can be obtained.

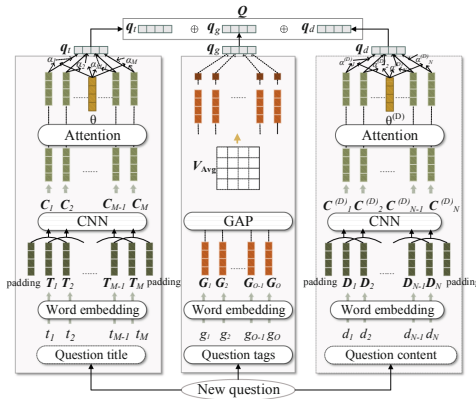


Fig. 1. Structure of question encoder

In the second step, two parallel CNNs are used, taking the embedded title and content as input and learning the contextual representations by capturing local

context information to optimize the word vector further. Taking the training of the question title as an example, the contextual representation of  $T_i$  is denoted as  $C_i$ , and the calculation is shown as (2).

$$C_i = ReLU(\beta \times T_{[i-W:i+W]} + b) \quad (2)$$

In the third step, our method applies the attention mechanism to assign different weights to each word because different words have different importance to the title and content. For the  $i$ -th word in the question title, its attention weight  $A_i$  is calculated as follows:

$$\alpha_i = \tanh(\theta \times C_i + r) \quad (3)$$

$$A_i = \frac{\exp(\alpha_i)}{\sum_{j=1}^M \exp(\alpha_j)} \quad (4)$$

The final representation of the question title is the summation of its contextual word representation and the weight measured by its attention mechanism.

$$q_t = \sum_{i=1}^M A_i C_i \quad (5)$$

Similarly, the question content representation  $q_d = \sum_{i=1}^N A_i^{(D)} C_i^{(D)}$  can also be derived by using (2)–(4).

**Question Tag Representation.** Suppose that all tags of the given question form a set  $g = [g_1, g_2, g_3, \dots, g_O]$ , where  $O$  is the number of tags. First, (1) is used for word embedding,  $G = [G_1, G_2, G_3, \dots, G_O]$  is obtained, the global average pooling(GAP) is used to learn the tag representation  $q_g$ , as shown in (6).

$$q_g = V_{Avg} G \quad (6)$$

Where  $V_{Avg}$  is the GAP parameter matrix.

**Final Question Representation.** The final representation of a new question is a concatenation of title, content, and tag representations. Given the  $i$ -th question,  $Q_i$  can be obtained by (7).

$$Q_i = [q_t, q_d, q_g] \quad (7)$$

### 3.2 User Encoder

The time series of users answering questions reflects changes in user interests and user expertise. Users also select several tags to show their interests. These tags will not change for a long time and can reflect the long-term fixed interests of users. The user encoder learns features from sequence behaviors and user tags to generate the final user representation. The architecture is shown in Fig. 2.

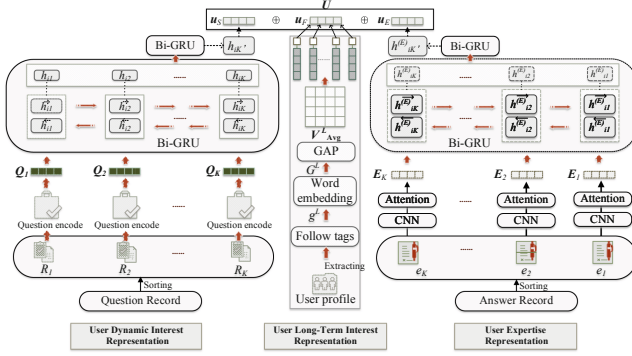


Fig. 2. Structure of user encoder

**User Dynamic Interest Representation.** Assuming a sequence of chronologically ordered historical questions answered by the current user  $I = [I_1, I_2 \dots I_K]$ , the question encoder is used to encode each question to obtain their representation  $Q = [Q_1, Q_2 \dots Q_K]$ . The double-layer Bi-GRU network is used to capture the dynamic changes of user interests. Bi-GRU consists of two GRUs: forward GRU  $\vec{h}_s$  and backward GRU  $\overleftarrow{h}_s$ ; thus, the output at the current moment can be linked to the state of the previous moment and the state of the next moment. First,  $Q$  is input, the hidden layer state of the first Bi-GRU layer is weighted by the forward hidden layer state  $\vec{h}_{is}$  and the backward hidden layer state  $\overleftarrow{h}_{is}$ , as shown in (8)–(10).

$$\overleftarrow{h}_{is} = \overleftarrow{GRU}(Q_s), s \in [1, K] \tag{8}$$

$$\vec{h}_{is} = \vec{GRU}(Q_s), s \in [1, K] \tag{9}$$

$$h_{is} = [\vec{h}_{is} \oplus \overleftarrow{h}_{is}] \tag{10}$$

Second, all hidden layer state outputs  $h_{is}$  of the first Bi-GRU layer are input into the second Bi-GRU layer to obtain a fine-grained dynamic information representation. The second layer only outputs the most representative state, that is, the last hidden layer state  $h'_{ik}$ . Then, our final user representation  $u_S = h'_{ik}$  is obtained.

**User Long-Term Interest Representation.** Topic tags that are extracted and denoted as  $g^L$ . First, (1) is used to embed the sequence. Then, (6) is used to determine the user long-term interest representation  $u_F = V_{Avg}^L G^L$ .

**User Expertise Representation.** First, the answers are extracted and sorted as  $e = [e_1, e_2 \dots e_K]$ . Second, the generation is placed into CNN and attention mechanism network to capture important information  $E = [E_1, E_2 \dots E_K]$ . Third,  $E$  is placed into double-layer Bi-GRU to capture the dynamic changes of user expertise. Finally, user expertise representation  $u_E$  is determined.

**Final User Representation.**  $U$  is computed as follows:

$$U = [u_S, u_F, u_E] \quad (11)$$

### 3.3 Prediction and Training

Given a question  $q_x$  and a user  $u_x$ , the similarity result between the question representation  $Q_x$  of  $q_x$  and the user representation  $U_x$  of  $u_x$  is calculated to determine whether the current user will become the expert. The similarity calculation is based on the dot product  $U_x^T Q_x$  to generate expert score, as shown in (12).

$$S_{core}(u_x, q_x) = Sigmoid(\psi(U_x^T Q_x)) \quad (12)$$

where  $\psi$  is the fully connected layer function.

## 4 Experiments

### 4.1 Dataset

Our dataset is released by BAAI (Beijing Academy of Artificial Intelligence) and Zhihu, which are published in the competition platform - biendata<sup>1</sup>. Zhihu<sup>2</sup> is a well-known CQA platform on the Chinese Internet. We collect 4,513,735 answers, 1,931,654 users, 1,829,900 questions, and 500,000 Q-U pair. The dataset is divided into training set and test set according to a certain proportion, and each experiment uses five-fold cross-validation.

### 4.2 Baseline

Our method LSTERM compares its performance with the following baseline expert recommendation methods:

- DeepFM [4] combines deep neural network with FM. It could extract low-level and high-level features, and predict user behavior through learning implicit feature interaction.
- XDeepFM [5] is an improvement of DeepFM adding the CIN structure. It can learn high-level feature interaction through a combination of explicit and implicit methods, and focus on the effects of feature intersection.
- CNN-DSSM [6] could extract global contextual information through the CNN convolutional layer and pooling layer, and generate semantic vectors to predict user behavior.

Three variants of the LSTERM are also set up:

- LSTERM-OS: Only the dynamic interest representation part is used.
- LSTERM-RL: Only user dynamic interest and user expertise are considered.
- LSTERM-RE: Only the user dynamic interest and long-term interest parts are used.

<sup>1</sup> <https://www.biendata.xyz/competition/falsenews/>.

<sup>2</sup> <https://www.zhihu.com/>.

### 4.3 Evaluation and Implementation

The area under the ROC curve (AUC) and accuracy (ACC) are used to evaluate the recommendation effect.

The experimental environment is as follows: Intel(R) Core(TM) i7-9750H CPU@2.60 GHz + 8 GB, Windows10 x64-bit operating system, deep learning framework: Anaconda Python3.0 + Tenserflow + Keras.

### 4.4 Parameter Setting

The embedding of question text, answer text, and tag information is placed in the same vector space, and the embedding dimension is set to 64. Bi-GRU structure has two layers, and the size of units is set to 128 and 32. For the output vectors of the question encoder and the user encoder, two layers of fully connected structures are set, and the size of the units are 1024 and 568.

### 4.5 Results Analysis

Table 1 shows the experimental results of all methods on 30%, 70%, and 100% training data. When 30% of the training data are used, our method can still achieve certain accuracy, indicating that using pretrained word embedding may ensure the stability and overcome data sparsity. In addition, the performance of our approach continues to improve with the increase of training data.

The DeepFM method has the worst effect probably because the manual feature selection method is less efficient and will lose several important features. The XDeepFm method is better than DeepFM, reflecting that high-level feature interactions can better realize feature representation. The CNN-DSSM method is better than XDeepFM mainly because its sliding window allows more contextual information to be retained. Our method LSTERM and its three variants are remarkably better than the above baselines. Revealing time series information helps us discover user dynamic interests and expertise, and LSTERM is better

**Table 1.** Comparison of the experimental results of all algorithms on 30%, 70% and 100% training data

Method	30% $D_{train}$		70% $D_{train}$		100% $D_{train}$	
	AUC	ACC	AUC	ACC	AUC	ACC
DeepFM	0.5223	0.7249	0.5301	0.7318	0.5323	0.7410
XDeepFM	0.5741	0.7540	0.5760	0.7660	0.5847	0.7886
CNN-DSSM	0.5848	0.7976	0.5995	0.8051	0.6017	0.8102
LSTERM-OS	0.6322	0.8076	0.6404	0.8107	0.6470	0.8176
LSTERM-RL	0.6366	0.8109	0.6417	0.8114	0.6523	0.8192
LSTERM-RE	0.6390	0.8188	0.6435	0.8193	0.6536	0.8209
LSTERM	<b>0.6407</b>	<b>0.8208</b>	<b>0.6495</b>	<b>0.8216</b>	<b>0.6592</b>	<b>0.8230</b>

than LSTERM-OS, LSTERM-RL, and LSTERM-RE possibly because the combination of dynamic interest and long-term fixed interest could better express user interest, and dynamic information and long-term fixed information could better mine expert users.

## 5 Conclusion

The large number of questions and the low answering efficiency are serious challenges for CQA. This paper proposes an expert recommendation method (LSTERM) based on CNN, attention mechanism, and recurrent neural network Bi-GRU to solve the problem. The method contains two core parts: question encoder and user encoder. The question encoder realizes the deep feature coding of the question context. The user encoder captures the dynamic interest and expertise in the time series of the user's historical answering questions and combines the fixed tag information to represent the user interest.

**Acknowledgement.** This work is supported by National Science Foundation of China No. 61702216, 61772231, and Higher Educational Science and Technology Program of Jinan City under Grant with No. 2020GXRC057, 2018GXRC002.

## References

1. Daud, A., Ahmad, M., Malik, M.S.I., Che, D.: Using machine learning techniques for rising star prediction in co-author network. *Scientometrics* **102**(2), 1687–1711 (2014). <https://doi.org/10.1007/s11192-014-1455-8>
2. Li, B., King, I.: Routing questions to appropriate answerers in community question answering services. In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management (2010)
3. Shen, Y., et al.: Question/answer matching for CQA system via combining lexical and sequential information. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 29, no. 1 (2015)
4. Guo, H., et al.: DeepFM: a factorization-machine based neural network for CTR prediction. arXiv preprint [arXiv:1703.04247](https://arxiv.org/abs/1703.04247) (2017)
5. Lian, J., et al.: XDeepFM: combining explicit and implicit feature interactions for recommender systems. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (2018)
6. Shen, Y., et al.: A latent semantic model with convolutional-pooling structure for information retrieval. In: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (2014)
7. Zhang, S., et al.: Deep learning based recommender system: a survey and new perspectives. *ACM Comput. Surv. (CSUR)* **52**(1), 1–38 (2019)
8. Lavrenko, V., Bruce Croft, W.: Relevance-based language models. In: ACM SIGIR Forum, vol. 51, no. 2. ACM, New York (2017)
9. Jiao, Z., Sun, S., Sun, K.: Chinese lexical analysis with deep Bi-GRU-CRF network. arXiv preprint [arXiv:1807.01882](https://arxiv.org/abs/1807.01882) (2018)
10. Zheng, Z., et al.: Answer selection in community question answering by normalizing support answers. In: Huang, X., Jiang, J., Zhao, D., Feng, Y., Hong, Yu. (eds.) NLPCC 2017. LNCS (LNAI), vol. 10619, pp. 672–682. Springer, Cham (2018). [https://doi.org/10.1007/978-3-319-73618-1\\_57](https://doi.org/10.1007/978-3-319-73618-1_57)



## Author Index

- Al-Banna, Mortada I-3  
Alhazmi, Ahoud II-480  
Ali, Esraa I-207  
Aljubairy, Abdulwahab II-237, II-480  
Alvi, Ashik Mostafa I-591  
Azé, Jérôme I-339
- Baghernezhad-Tabasi, Shadi I-321  
Bagozi, Ada I-59, II-447  
Bai, Quan I-542  
Balasubramaniam, Thirunavukarasu II-377  
Balcerzak, Bartłomiej II-420  
Balogh, Zoltán II-435  
Banerjee, Suman I-485  
Barukh, Moshe C. I-3  
Ben Charrada, Faouzi II-463  
Benatallah, Boualem I-3, I-46, I-129  
Benslimane, Samy I-339  
Bertino, Elisa I-3  
Bi, Tianyuan I-355  
Bianchini, Devis I-59, II-447  
Bringay, Sandra I-339
- Cai, Xinyi I-158  
Cao, Jinli I-143  
Cao, Xiaobo II-518  
Caputo, Annalina I-207  
Chai, Yanfeng I-101  
Chai, Yunpeng I-101  
Chang, Chen II-386  
Chao, Pingfu I-619  
Chen, Bofeng I-242  
Chen, Chen I-453  
Chen, Hongxu II-491  
Chen, Jiaxin I-527  
Chen, Jinjun I-158  
Chen, Lei I-512  
Chen, Liang I-619  
Chen, Quan I-603  
Chen, Wei I-619  
Chen, Weixue I-258  
Chen, Xiaocong II-122  
Chen, Yu I-385
- Chen, Zhenxiang I-565, I-645  
Chen, Zhigang I-305, II-347  
Cheng, Jiajun I-385  
Chennakesavan, Srinivas Kondalsamy II-497  
Cheung, Yiu-ming I-173  
Choudhary, Gautam II-336  
Choudhury, Farhana Murtaza I-274  
Chu, Victor W. I-189  
Conlan, Owen I-207  
Cui, Shuangshuang I-119  
Cui, Zhiming II-34
- Dai, Bo I-111  
De Antonellis, Valeria I-59  
Deng, Ke II-94  
Dong, Jing I-305  
Dong, Xinzhou I-557, II-221  
Dou, Wanchun I-87  
Druette, Loïc I-321  
Duan, Chengyuan I-385  
Duan, Yitao II-275
- Elangovan, Soman II-497  
Emma Zhang, Wei II-363
- Fan, Shuai II-79  
Fang, Junhua I-619, I-636, II-177  
Fang, Yuan I-370  
Feng, Huan II-137  
Feng, Ling I-227  
Fodor, Kristián II-435  
Frasincar, Flavius II-291
- Gao, Liqun I-394, II-67  
Gao, Qiang I-385  
Gao, Wang I-370  
Gao, Yunjun II-94  
Garda, Massimiliano I-59  
Ge, Jiake I-101  
Ge, Ningchao II-527  
Ghedass, Fedia II-463  
Gu, Binbin I-305  
Gu, Haiyao I-119

- Guan, Huhao II-501  
 Guo, Lei I-111  
 Guo, Na I-158  
 Guo, Teng I-497  
 Guo, Ye II-50  
 Gururajan, Raj II-497  
  
 Hamad, Salma II-237  
 Han, Kai I-355  
 Hao, Yongjing II-34  
 Hochstenbach, Ron II-291  
 Hong, Zhenhou II-394  
 Hou, Lei I-14, I-227  
 Hu, Shuaijun II-137  
 Hu, Zhiqiang I-403  
 Hua, Wen I-575  
 Huang, Chuanyi II-137  
 Huang, Guangyan II-321  
 Huang, Junzhou I-423  
 Huang, Wenbing I-423  
 Huang, Yuancai II-275  
 Huang, Zhangcheng I-216  
  
 Ji, Ke I-565, I-645  
 Ji, Wendi II-147, II-209  
 Jia, Yan I-394, II-67  
 Jiang, Chenting I-542  
 Jiang, Liying I-603  
 Jin, Beihong I-557, II-221  
 Jin, Fengmei I-575  
 Jouanot, Fabrice I-321  
  
 Kanfade, Anirudh II-405  
 Kanhere, Salil I-3  
 Kato, Makoto P. II-252  
 Khoa, Nguyen Lu Dang II-237  
 Khritankov, Anton II-267  
 Kong, Deyu II-501  
 Kong, Lingwei I-216  
 Kou, Yue II-518  
 Kwon, Junbum I-189  
  
 Lai, Yongxuan I-603  
 Lawless, Séamus I-207  
 Lee, Roy Ka-Wei I-403  
 Leong, Hong Va I-512  
 Li, Beibei II-221  
 Li, Chengjiang I-227  
 Li, Denghao I-216  
 Li, Guangyu II-137  
 Li, Hui II-79  
 Li, Jianxin I-274, I-497, II-321  
 Li, Jingdong I-242  
 Li, Jinliang I-290  
 Li, Juanzi I-14, I-227  
 Li, Jun I-512  
 Li, Kangwei II-518  
 Li, Lin I-370  
 Li, Mengwei II-518  
 Li, Mingdao II-527  
 Li, Qi II-501  
 Li, Qing I-512, II-18, II-94  
 Li, Ruiyuan II-536  
 Li, Shunyang II-510  
 Li, Sizhuo I-258  
 Li, Weihua I-542  
 Li, Yanqi I-565  
 Li, Yepeng II-193  
 Li, Yicong II-491  
 Li, Yidong I-565, I-645  
 Li, Yuan-Fang I-258  
 Li, Yuefeng II-497  
 Li, Zhixu I-290, I-305, II-347  
 Liao, Qisheng I-603  
 Lin, Chen II-79  
 Lin, Chuangwei I-87  
 Lin, Xuemin I-453, II-510  
 Lin, Zehang I-512  
 Liu, An I-305, I-636, II-177, II-347  
 Liu, Baozhu I-258, I-274  
 Liu, Bowen I-87  
 Liu, Chengfei I-30  
 Liu, Fang II-306  
 Liu, Haobing II-3  
 Liu, Haoyu I-290  
 Liu, Honglin II-137  
 Liu, Jianbo I-403  
 Liu, Jun II-536  
 Liu, Pengkai I-258  
 Liu, Qing I-30  
 Liu, Wei II-162  
 Liu, Wenyin I-527, II-18  
 Liu, Xijuan I-453  
 Liu, Yanchi II-193  
 Long, Chao II-3  
 Lou, Jian I-173  
 Lu, Xingjian I-242  
 Luo, Di II-18  
 Lv, Xiaoqi I-645  
 Lv, Xin I-227

- Lv, Xingyi II-50  
 Lv, Yimin I-557
- Ma, Kun I-565, I-645  
 Mahmood, Adnan II-363  
 Mao, Jiali II-50  
 Mao, Yiming I-14  
 Maruta, Atsuki II-252  
 Maurya, Anurag II-405  
 Maurya, Nitish II-336  
 Melchiori, Michele I-59  
 Meurger, Celine I-321  
 Mi, Xianya I-385  
 Miao, Yuan I-143  
 Modani, Natwar II-336, II-405  
 Mohotti, Watsala Anupama II-377  
 Mollevi, Caroline I-339  
 Morzy, Mikołaj II-420
- Nabożny, Aleksandra II-420  
 Nair, Inderjeet II-405  
 Nayak, Richi II-377  
 Nie, Feiping I-129  
 Nie, Tiezheng II-518
- Pal, Bithika I-485  
 Pang, Chaoyi II-94  
 Park, Eunkyung I-189  
 Patil, Vaidehi II-405  
 Peng, Guozheng I-227  
 Peng, Min II-306  
 Peng, Peng II-527  
 Pilkevich, Anton II-267
- Qi, Yan I-227  
 Qin, Zheng II-527  
 Qiu, Jing I-575  
 Qu, Jianfeng I-290, I-305, II-177, II-347  
 Qu, Xiaoyang II-394
- Rao, Yizhuo I-385  
 Ren, Xiaoguang I-385  
 Rong, Yu I-423  
 Rousset, Marie-Christine I-321  
 Rula, Anisa II-447
- Servajean, Maximilien I-339  
 Sha, Chaofeng I-242  
 Shao, Jie I-403  
 Shen, Cheng I-355
- Shen, Derong II-518  
 Sheng, Quan Z. II-237, II-363, II-480  
 Sheng, Victor S. II-34, II-193  
 Si, Jiarui II-137  
 Si, Shijing II-394  
 Siuly, Siuly I-198, I-591  
 Song, Xiangyu II-321  
 Song, Xin I-394, II-67  
 Song, Zihan II-147  
 Sui, Yuan II-536
- Tang, Jie I-14  
 Tang, Juncheng II-536  
 Tang, Tao I-497  
 Tang, Yu II-275  
 Tao, Tao II-18  
 Tao, Xiaohui I-370, II-497  
 Tawhid, Md. Nurul Ahad I-198  
 Tian, Jia II-518  
 Tian, Jiao I-158  
 Tran, Dai Hoang II-237, II-480  
 Tran, Nguyen H. II-237  
 Truşcă, Maria Mihaela II-291
- Venkataraman, Revathi II-497  
 Verma, Gaurav II-405
- Wan, Shuhan I-636  
 Wang, Aopeng II-94  
 Wang, Fu Lee I-527  
 Wang, Haiming II-162  
 Wang, Haiyang I-394, II-67  
 Wang, Hongzhi I-119  
 Wang, Hua I-143, I-198, I-591  
 Wang, Jianzong I-216, II-394  
 Wang, Kate I-198  
 Wang, Xianzhi I-129  
 Wang, Xiaoling I-242, II-147, II-209  
 Wang, Xiaoyang I-453  
 Wang, Xin I-101, I-258, I-274, II-137  
 Wang, Xu II-275, II-536  
 Wang, Xuesong I-129  
 Wang, Yanpeng I-14  
 Wang, Ye I-394, II-67  
 Wang, Yu II-177  
 Wang, Yuetian I-575  
 Wang, Yufei II-363  
 Wang, Yuquan I-14  
 Wei, Xiao I-385  
 Wen, Shiting II-94

- Wierzbicki, Adam II-420  
 Wong, Raymond K. I-189  
 Wu, Jun I-565, I-645  
 Wu, Shiqing I-542  
 Wu, Yanhui II-107  
 Wu, Zekai I-527  
  
 Xia, Feng I-497  
 Xian, Xuefeng II-34, II-193  
 Xiao, Hongwang I-158  
 Xiao, Jing I-216, II-394  
 Xiao, Xiaokui I-438  
 Xie, Haoran I-527  
 Xie, Xike II-501  
 Xie, Yuntian I-119  
 Xing, Jiao I-274  
 Xu, Chen I-71  
 Xu, Guandong I-565, I-645  
 Xu, Jiajie I-30, I-619  
 Xu, Liangliang II-18  
 Xu, Minzhang II-306  
 Xu, Shuangyan I-453  
 Xu, Yueyue I-87  
  
 Yaghoub-Zadeh-Fard, Mohammad-Ali I-46  
 Yan, Zhihan I-512  
 Yang, Fan I-603  
 Yang, Haoran II-491  
 Yang, Qiang I-290, II-347  
 Yang, Renchi I-438  
 Yang, Shuiqiao II-321  
 Yang, Yi I-71  
 Yang, Yun I-158  
 Yang, Zhenguo I-512, I-527, II-18  
 Yang, Zhengyi II-510  
 Yang, Zhenhua I-71  
 Yao, Lina I-129, II-122  
 Yi, Fan I-603  
 Yin, Hui II-321  
 Yin, Jian II-162  
 Yin, Jiao I-143  
 You, Hongliang I-385  
 You, Mingshan I-143  
 Yu, Feng I-173  
 Yu, Jeffrey Xu I-423  
 Yu, Jiadi II-3  
 Yu, Jifan I-14  
 Yu, Ke I-158  
 Yu, Nancy Xiaonan I-512  
 Yu, Yanpeng I-469  
  
 Yuan, Jiahao II-147, II-209  
 Yuan, Minzheng II-18  
 Yuen, Chau II-377  
  
 Zaib, Munazza II-237  
 Zeng, Kaisheng I-14, I-227  
 Zeng, Wenhua I-603  
 Zeng, Zhixian I-385  
 Zhan, Siyu I-111  
 Zhang, Detian I-636  
 Zhang, Dongyu I-497  
 Zhang, Fan I-469  
 Zhang, Guipeng II-18  
 Zhang, Hao I-423  
 Zhang, Hong II-137  
 Zhang, Ji I-242  
 Zhang, Jiasheng I-403  
 Zhang, Kai I-158  
 Zhang, Li I-290  
 Zhang, Limeng I-30  
 Zhang, Meng II-107  
 Zhang, Mengqi I-453  
 Zhang, Senhui II-209  
 Zhang, Wei Emma II-237, II-480  
 Zhang, Wenjie II-510  
 Zhang, Xianhang II-510  
 Zhang, Xuyun I-87  
 Zhang, Yang II-363  
 Zhang, Yu II-122  
 Zhang, Yulong I-290  
 Zhao, BoXuan I-101  
 Zhao, Kangfei I-423  
 Zhao, Lei I-305, I-619, II-177, II-347  
 Zhao, Pengpeng II-34, II-193  
 Zhao, Rongying II-363  
 Zhao, Wei II-50  
 Zhao, Weinan II-275  
 Zhao, Xiaokai II-275  
 Zhao, Ying I-158  
 Zheng, Shenghao II-34  
 Zhou, Aoying II-50  
 Zhou, Bin I-394, II-67  
 Zhou, Rui I-30  
 Zhou, Wei I-87  
 Zhou, Weiqing I-111  
 Zhou, Xiaofang I-575  
 Zhou, Xujuan II-497  
 Zhu, Jingdan I-412  
 Zhu, Xinghua II-394

Zhu, Yanmin [II-3](#)  
Zhuo, Wei [I-557](#), [II-221](#)  
Zou, Haohan [II-137](#)  
Zou, Kai [II-501](#)

Zou, Lei [I-469](#)  
Zou, Meng [II-347](#)  
Zou, Quan [II-79](#)