

Albert Bifet · Jesse Davis ·
Tomas Krilavičius · Meelis Kull ·
Eirini Ntoutsi · Indrė Žliobaitė (Eds.)

LNAI 14943

Machine Learning and Knowledge Discovery in Databases

Research Track

European Conference, ECML PKDD 2024
Vilnius, Lithuania, September 9–13, 2024
Proceedings, Part III

3
Part III

ECML
PKDD
2024

 Springer

MOREMEDIA 

Lecture Notes in Computer Science

Lecture Notes in Artificial Intelligence

14943

Founding Editor

Jörg Siekmann

Series Editors

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

Wolfgang Wahlster, *DFKI, Berlin, Germany*

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

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

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

Albert Bifet · Jesse Davis · Tomas Krilavičius ·
Meelis Kull · Eirini Ntoutsi · Indrė Žliobaitė
Editors

Machine Learning and Knowledge Discovery in Databases


Research Track

European Conference, ECML PKDD 2024
Vilnius, Lithuania, September 9–13, 2024
Proceedings, Part III

 Springer

Editors

Albert Bifet 
LTCI
Télécom Paris
Palaiseau Cedex, France

Tomas Krilavičius 
Faculty of Informatics
Vytautas Magnus University
Akademija, Lithuania

Eirini Ntoutsis 
Department of Computer Science
Bundeswehr University Munich
Munich, Germany

Jesse Davis 
KU Leuven
Leuven, Belgium

Meelis Kull 
Institute of Computer Science
University of Tartu
Tartu, Estonia

Indrė Žliobaitė 
Department of Computer Science
University of Helsinki
Helsinki, Finland

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Artificial Intelligence
ISBN 978-3-031-70351-5 ISBN 978-3-031-70352-2 (eBook)
<https://doi.org/10.1007/978-3-031-70352-2>

LNCS Sublibrary: SL7 – Artificial Intelligence

© The Editor(s) (if applicable) and The Author(s), under exclusive license
to Springer Nature Switzerland AG 2024

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

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

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

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

If disposing of this product, please recycle the paper.

Preface

The 2024 edition of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2024) was held in Vilnius, Lithuania, from September 9 to 13, 2024.

The annual ECML PKDD conference acts as a world-wide platform showcasing the latest advancements in machine learning and knowledge discovery in databases. Held jointly since 2001, ECML PKDD has established itself as the leading European Machine Learning and Data Mining conference. It offers researchers and practitioners an unparalleled opportunity to exchange knowledge and ideas about the latest technical advancements in these disciplines. Moreover, the conference appreciates the synergy between foundational advances and groundbreaking data science and hence strongly welcomes contributions about how Machine Learning and Data Mining is being employed to solve real-world challenges.

The conference continues to evolve reflecting evolving technological developments and societal needs. For example, in the Research Track this year there has been an increase in submissions on generative AI, especially LLMs, and various aspects of responsible AI.

We received 826 submissions for the Research Track and 224 for the Applied Data Science Track. The Research track accepted 202 papers (out of 826, 24.5%) and the Applied Data Science Track accepted 56 (out of 224, 24.5%). In addition, 31 papers from the Journal Track (accepted out of 65 submissions) and 14 Demo Track papers (accepted out of 30 submissions).

The papers presented over the three main conference days were organized into five distinct tracks:

Research Track: This track featured research and methodology papers spanning all branches within Machine Learning, Knowledge Discovery, and Data Mining.

Applied Data Science Track: Papers in this track focused on novel applications of machine learning, data mining, and knowledge discovery to address real-world challenges, aiming to bridge the gap between theory and practical implementation.

Journal Track: This track included papers that had been published in special issues of the journals *Machine Learning* and *Data Mining and Knowledge Discovery*.

Demo Track: Short papers in this track introduced new prototypes or fully operational systems that leverage data science techniques, demonstrated through working prototypes.

Nectar Track: Concise presentations of recent scientific advances published in related conferences or journals. It aimed to disseminate important research findings to a broader audience within the ECML PKDD community.

The conference featured five keynote talks on diverse topics, reflecting emerging needs like benchmarking and resource-awareness, as well as theoretical understanding and industrial needs.

- Gintarė Karolina Džiugaitė (Google DeepMind): *The Dynamics of Memorization and Unlearning*.
- Moritz Hardt (Max Planck Institute for Intelligent Systems): *The Emerging Science of Benchmarks*.
- Mounia Lalmas-Roelleke (Spotify): *Enhancing User Experience with AI-Powered Search and Recommendations at Spotify*.
- Patrick Lucey (Stats Perform): *How to Utilize (and Generate) Player Tracking Data in Sport*.
- Katharina Morik (TU Dortmund University): *Resource-Aware Machine Learning — a User-Oriented Approach*.

The ECML PKDD 2024 Organizing Committee supported Diversity and Inclusion by awarding some grants that enable early career researchers to attend the conference, present their research activities, and become part of the ECML PKDD community. We provided a total of 3 scholarships of €1000 to individuals that come from the developing countries and/or communities which are underrepresented in science and technology. The scholarships could be used for travel and accommodation. In addition 3 grants covering all of the registration fees were awarded to individuals who belong to underrepresented communities, based on gender and role/position, to attend the conference and present their research activities. The Diversity and Inclusion action also included the Women Networking event and Diversity and Inclusion Panel discussion. The Women Networking event aimed to create a safe and inclusive space for networking and reflecting on the experience of women in science. The event included a structured brainstorm/reflection on the role and experience of women in science and technology, which will be published in the conference newsletter. The Diversity and Inclusion Panel aimed to reach a wider audience and encourage the discussion on the need for diversity in tech, and challenges and solutions in achieving it.

We want to thank the authors, workshop and tutorial organizers, and participants whose scientific contributions make this such an exciting event. Moreover, putting together an outstanding conference program would also not be possible without the dedication and (substantial) time investments of the area chairs, program committee, and organizing committee. The event would not run smoothly without the many volunteers and sessions chairs. Finally, we want to extend a special thanks to all the local organizers – they dealt with all the little details that are needed to make the conference a memorable event.

We want to extend our heartfelt gratitude to our wonderful sponsors for their generous financial support. We also want to thank Springer for their continuous support and Microsoft for allowing us to use their CMT software for conference management and providing help throughout. We very much appreciate the advice and guidance provided

by the ECML PKDD Steering Committee over the past two years. Finally, we thank the organizing institution, the Artificial Intelligence Association of Lithuania.

September 2024

Albert Bifet
Tomas Krilavičius
Eirini Ntoutsi
Indrė Žliobaitė
Jesse Davis
Meelis Kull
Ioanna Miliou
Slawomir Nowaczyk

Organization

General Chairs

Albert Bifet	IP Paris, France/University of Waikato, New Zealand
Tomas Krilavičius	Vytautas Magnus University, Lithuania

Research Track Program Chairs

Indrė Žliobaitė	University of Helsinki, Finland
Meelis Kull	University of Tartu, Estonia
Jesse Davis	KU Leuven, Belgium
Eirini Ntoutsi	University of the Bundeswehr Munich, Germany

Applied Data Science Track Program Chairs

Slawomir Nowaczyk	Halmstad University, Sweden
Ioanna Miliou	Stockholm University, Sweden

Journal Track Chairs

Panagiotis Papapetrou	Stockholm University, Sweden
Rita Ribeiro	University of Porto/LIAAD, Portugal
Myra Spiliopoulou	Otto-von-Guericke University Magdeburg, Germany
Šarūnas Girdzijauskas	KTH Royal Institute of Technology, Sweden

Local Chair

Linus Petkevičius	Vilnius University, Lithuania
-------------------	-------------------------------

Workshop and Tutorial Chairs

Mantas Lukoševičius
Mykola Pechenizkiy

Kaunas University of Technology, Lithuania
Technische Universiteit Eindhoven,
the Netherlands

Demo Chairs

Povilas Daniušis
Kai Puolamäki

Vytautas Magnus University, Lithuania
University of Helsinki, Finland

Proceedings Chairs

Wouter Duivesteyn

Technische Universiteit Eindhoven,
the Netherlands

Rianne Schouten

Technische Universiteit Eindhoven,
the Netherlands

PhD Forum Chairs

Virginijus Marcinkevičius
Simona Ramanauskaitė

Vilnius University, Lithuania
Vilnius Tech, Lithuania

Discovery Track Chairs

Peter van der Putten
Jan N. van Rijn

Universiteit Leiden, the Netherlands
Universiteit Leiden, the Netherlands

Workshop Proceedings Chairs

Danguole Kalinauskaite
Kristina Štutiene

Vytautas Magnus University, Lithuania
Kaunas Technology University, Lithuania

Social Media and Web Chairs

Julija Vaitonytė

Kamilė Dementavičiūtė

Tilburg University, the Netherlands

Vilnius University, Lithuania

Sponsorship Chairs

Mariam Barry

Dalia Breskuvienė

Daniele Apiletti

BNP Paribas, France

Vilnius University, Lithuania

Politecnico di Torino, Italy

Diversity and Inclusion Chair

Rūta Binkytė-Sadauskienė

Inria, France

Industry Track Chairs

Pieter Van Hertum

Bjoern Bringmann

ASML, the Netherlands

Deloitte, Germany

Nectar Track Chairs

Heitor Murilo Gomes

Jesse Read

Victoria University of Wellington, New Zealand

École Polytechnique, France

Awards Chairs

Michele Sebag

João Gama

CNRS, France

University of Porto, Portugal

ECML PKDD Steering Committee

Tijl De Bie

Francesco Bonchi

Albert Bifet

Ghent University, Belgium

ISI Foundation, Italy

Télécom ParisTech, France

Andrea Passerini	University of Trento, Italy
Katharina Morik	TU Dortmund, Germany
Arno Siebes	Utrecht University, the Netherlands
Sašo Džeroski	Jožef Stefan Institute, Slovenia
Robert Jan van Wijk	ASML, the Netherlands
Ilaria Bordino	UniCredit, Italy
Siegfried Nijssen	Université catholique de Louvain, Belgium
Albrecht Zimmermann	University of Caen - Normandie, France
Annalisa Appice	University of Bari 'Aldo Moro', Italy
Tania Cerquitelli	Politecnico di Torino, Italy
Alípio Jorge	University of Porto, Portugal
Fernando Perez-Cruz	ETH Zurich, Switzerland
Massih-Reza Amini	University Grenoble Alpes, France
Peggy Cellier	INSA Rennes, IRISA, France
Tias Guns	KU Leuven, Belgium
Grigorios Tsoumakas	Aristote University of Thessaloniki, Greece
Elena Baralis	Politecnico di Torino, Italy
Claudia Plant	Universität Wien, Austria
Manuel Gomez Rodriguez	Max Planck Institute for Software Systems, Germany

Program Committees

Guest Editorial Board, Journal Track

Richard Allmendinger	University of Manchester, UK
Marie Anastacio	Leiden University, the Netherlands
Giuseppina Andresini	Università degli Studi di Bari 'Aldo Moro', Italy
Annalisa Appice	Università degli Studi di Bari 'Aldo Moro', Italy
Jaume Bacardit	Newcastle University, UK
Maria Bampa	Stockholm University, Sweden
Mitra Baratchi	LIACS - University of Leiden, the Netherlands
Szymon Bobek	Jagiellonian University, Poland
Claudio Borile	CENTAI Institute, Italy
Falko Brause	University of Vienna, Austria
Barbara Catania	University of Genoa, Italy
Michelangelo Ceci	University of Bari, Italy
Loïc Cerf	Universidade Federal de Minas Gerais, Brazil
Tianyi Chen	Boston University, USA
Filip Cornell	KTH Royal Institute of Technology, Sweden

Marco Cotogni	University of Pavia, Italy
Claudia Diamantini	Università Politecnica delle Marche, Italy
Sebastien Destercke	UTC, France
César Ferri	Universitat Politècnica València, Spain
Olga Fink	EPFL, Switzerland
Esther Galbrun	University of Eastern Finland, Finland
Joao Gama	INESC TEC - LIAAD, Portugal
Jose A. Gamez	Universidad de Castilla-La Mancha, Spain
Paolo Garza	Politecnico di Torino, Italy
Carolina Geiersbach	Weierstrass Institute Berlin, Germany
Riccardo Guidotti	University of Pisa, Italy
Francesco Gullo	UniCredit, Italy
Martin Holena	Institute of Computer Science, Czechia
Dino Ienco	INRAE, France
Georgiana Ifrim	University College Dublin, Ireland
Felix Iglesias	Technical University of Vienna, Austria
Angelo Impedovo	University of Bari 'Aldo Moro', Italy
Matthias Jacobs	Technical University Dortmund, Germany
Szymon Jaroszewicz	Polish Academy of Sciences, Poland
Yifei Jin	Ericsson Research/KTH Royal Institute of Technology, Sweden
Panagiotis Karras	University of Copenhagen, Denmark
Mehdi Kaytoue	Infologic R&D, France
Dragi Kocev	Josef Stefan Institute, Slovenia
Helge Langseth	Norwegian Univ of Science and Technology, Norway
Thien Le	MIT, USA
Hsuan-Tien Lin	National Taiwan University, Taiwan
Marco Lippi	University of Modena and Reggio Emilia, Italy
Corrado Loglisci	Università degli Studi di Bari 'Aldo Moro', Italy
Brian Mac Namee	University College Dublin, Ireland
Sindri Magnusson	Stockholm University, Sweden
Giuseppe Manco	ICAR-CNR, Italy
Michael Mathioudakis	University of Helsinki, Finland
Ioanna Miliou	Stockholm University, Sweden
Olof Mogren	RISE Research Institutes, Sweden
Nuno Moniz	University of Notre Dame, France
Anna Monreale	University of Pisa, Italy
Alberto Montresor	University of Trento, Italy
Katharina Morik	Technical University Dortmund, Germany
Lia Morra	Politecnico di Torino, Italy
Amedeo Napoli	LORIA, Nancy, France

Andrea Paudice	University of Milan, Italy
Benjamin Noack	Otto-von-Guericke University Magdeburg, Germany
Slawomir Nowaczyk	Halmstad University, Sweden
Vincenzo Pasquabisceglie	Università degli Studi di Bari 'Aldo Moro', Italy
Ruggero G. Pensa	University of Turin, Italy
Linas Petkevicius	Vilnius University, Finland
Marc Plantevit	EPITA, France
Kai Puolamäki	University of Helsinki, Finland
Jan Ramon	Inria, France
Matteo Riondato	Amherst College, USA
Isak Samsten	Stockholm University, Sweden
Shinichi Shirakawa	Yokohama National University, Japan
Amira Soliman	Halmstad University, Sweden
Fabian Spaeh	Boston University, USA
Gerasimos Spanakis	Maastricht University, the Netherlands
Mahito Sugiyama	National Institute of Informatics, Japan
Nikolaj Tatti	Helsinki University, Finland
Josephine Thomas	University of Kassel, Germany
Sebastian Stober	Otto-von-Guericke University Magdeburg, Germany
Genoveva Vargas-Solar	CNRS LIRIS, France
Bruno Veloso	University of Porto, Portugal
Pascal Welke	Technical University of Vienna, Austria
Marcel Wever	Ludwig-Maximilian-University Munich, Germany
Ye Zhu	Deakin University, Australia
Albrecht Zimmermann	Université de Caen Normandie, France
Blaz Zupan	University of Ljubljana, Slovenia

Area Chairs, Research Track

Leman Akoglu	CMU, USA
Anthony Bagnall	University of Southampton, UK
Gustavo Batista	UNSW, Australia
Jessa Bekker	KU Leuven, Belgium
Bettina Berendt	TU Berlin, Germany
Hendrik Blockeel	KU Leuven, Belgium
Henrik Bostrom	KTH Royal Institute of Technology, Sweden
Zied Bouraoui	CRIL CNRS & Univ Artois, France
Ulf Brefeld	Leuphana, Germany

Toon Calders	Universiteit Antwerpen, Belgium
Michelangelo Ceci	University of Bari, Italy
Fabrizio Costa	Exeter University, UK
Tijl De Bie	Ghent University, Belgium
Tom Diethé	AstraZeneca, UK
Kurt Driessens	Maastricht University, the Netherlands
Wouter Duivesteyn	TU Eindhoven, the Netherlands
Sebastijan Dumancic	TU Delft, the Netherlands
Tapio Elomaa	Tampere University, Finland
Stefano Ferilli	University of Bari, Italy
Cèsar Ferri	Universitat Politècnica València, Spain
Peter Flach	University of Bristol, UK
Elisa Fromont	Université Rennes 1, IRISA/Inria rba, France
Johannes Fürnkranz	JKU Linz, Austria
Esther Galbrun	University of Eastern Finland, Finland
Joao Gama	INESC TEC - LIAAD, Portugal
Aristides Gionis	KTH Royal Institute of Technology, Sweden
Bart Goethals	Universiteit Antwerpen, Belgium
Chen Gong	Nanjing University of Science and Technology, China
Dimitrios Gunopulos	University of Athens, Greece
Tias Guns	KU Leuven, Belgium
Barbara Hammer	CITEC, Bielefeld University, Germany
José Hernández-Orallo	Universitat Politècnica de València, Spain
Sibylle Hess	TU Eindhoven, the Netherlands
Andreas Hotho	University of Wuerzburg, Germany
Eyke Hüllermeier	University of Munich, Germany
Georgiana Ifrim	University College Dublin, Ireland
Manfred Jaeger	Aalborg University, Denmark
Szymon Jaroszewicz	Polish Academy of Sciences, Poland
George Karypis	University of Minnesota, Twin Cities, USA
Ioannis Katakis	University of Nicosia, Cyprus
Marius Kloft	TU Kaiserslautern, Germany
Dragi Kocev	Jožef Stefan Institute, Slovenia
Parisa Kordjamshidi	Michigan State University, USA
Lars Kotthoff	University of Wyoming, USA
Petra Kralj Novak	Central European University, Austria
Georg Krempel	Utrecht University, the Netherlands
Peer Kröger	Christian-Albrechts-Universität Kiel, Germany
Leo Lahti	University of Turku, Finland
Mark Last	Ben-Gurion University of the Negev, Israel
Jefrey Lijffijt	Ghent University, Belgium

Jessica Lin	George Mason University, USA
Michele Lombardi	University of Bologna, Italy
Donato Malerba	Università degli Studi di Bari 'Aldo Moro', Italy
Fragkiskos Malliaros	CentraleSupélec, France
Giuseppe Marra	KU Leuven, Belgium
Wannes Meert	KU Leuven, Belgium
Ernestina Menasalvas	Universidad Politécnica de Madrid, Spain
Pauli Miettinen	University of Eastern Finland, Finland
Dunja Mladenic	Jozef Stefan Institute, Slovenia
Emmanuel Müller	TU Dortmund, Germany
Siegfried Nijssen	Université catholique de Louvain, Belgium
Symeon Papadopoulos	Information Technologies Institute/Centre for Research & Technology - Hellas, Greece
Evangelos Papalexakis	UC Riverside, USA
Andrea Passerini	University of Trento, Italy
Jaakko Peltonen	Tampere University, Finland
Bernhard Pfahringer	University of Waikato, New Zealand
Claudia Plant	University of Vienna, Austria
Ricardo Prudencio	Universidade Federal de Pernambuco, Brazil
Milos Radovanovic	U. Novi Sad, Serbia
Chedy Raissi	Inria, France
Jesse Read	Ecole Polytechnique, France
Celine Robardet	INSA Lyon, France
Salvatore Ruggieri	University of Pisa, Italy
Steven Schockaert	Cardiff University, Wales, UK
Matthias Schubert	Ludwig-Maximilians-Universität München, Germany
Thomas Seidl	LMU Munich, Germany
Arno Siebes	Universiteit Utrecht, the Netherlands
Fabrizio Silvestri	Sapienza University of Rome, Italy
Jerzy Stefanowski	Poznan University of Technology, Poland
Nikolaj Tatti	Helsinki University, Finland
Evimaria Terzi	Boston University, USA
Grigorios Tsoumakas	Aristotle University of Thessaloniki, Greece
Charalampos Tsourakakis	Boston University, USA
Matthijs van Leeuwen	Leiden University, the Netherlands
Jan Van Rijn	LIACS, Leiden University, the Netherlands
Celine Vens	KU Leuven, Belgium
Jilles Vreeken	CISPA Helmholtz Center for Information Security, Germany
Willem Waegeman	Universiteit Gent, Belgium
Wei Ye	Tongji University, China

Wenbin Zhang	Florida International University, USA
Arthur Zimek	University of Southern Denmark, Denmark
Albrecht Zimmermann	Université de Caen Normandie, France

Area Chairs, Applied Data Science Track

Annalisa Appice	University of Bari ‘Aldo Moro’, Italy
Sahar Asadi	King (Microsoft), Sweden
Martin Atzmueller	Osnabrück University & DFKI, Germany
Michael R. Berthold	KNIME, Germany
Michelangelo Ceci	University of Bari, Italy
Peggy Cellier	INSA Rennes, IRISA, France
Nicolas Courty	IRISA, Université Bretagne-Sud, France
Bruno Cremilleux	Université de Caen Normandie, France
Tom Diethe	AstraZeneca, UK
Dejing Dou	BCG, USA
Olga Fink	EPFL, Switzerland
Elisa Fromont	Université Rennes 1, IRISA/Inria rba, France
Johannes Fürnkranz	JKU Linz, Austria
Sreenivas Gollapudi	Google, USA
Andreas Hotho	University of Wuerzburg, Germany
Alipio M. G. Jorge	INESC TEC/University of Porto, Portugal
George Karypis	University of Minnesota, Minneapolis, USA
Yun Sing Koh	University of Auckland, New Zealand
Parisa Kordjamshidi	Michigan State University, USA
Niklas Lavesson	Blekinge Institute of Technology, Sweden
Chuan Lei	Amazon, USA
Thomas Liebig	TU Dortmund Artificial Intelligence Unit, Germany
Tony Lindgren	Stockholm University, Sweden
Patrick Loiseau	Inria, France
Giuseppe Manco	ICAR-CNR, Italy
Gabor Melli	PredictionWorks, USA
Ioanna Miliou	Stockholm University, Sweden
Anna Monreale	University of Pisa, Italy
Luis Moreira-Matias	sennder, Germany
Jian Pei	Simon Fraser University, Canada
Fabio Pinelli	IMT Lucca, Italy
Zhiwei (Tony) Qin	Lyft, USA
Visvanathan Ramesh	Independent Researcher, Germany
Fabrizio Silvestri	Sapienza, University of Rome, Italy

Liang Sun	Alibaba Group, China
Jiliang Tang	Michigan State University, USA
Sandeep Tata	Google, USA
Yinglong Xia	Meta, USA
Fuzhen Zhuang	Institute of Artificial Intelligence, Beihang University, China
Albrecht Zimmermann	Université de Caen Normandie, France

Program Committee Members, Research Track

Zahraa Abdallah	University of Bristol, UK
Ziawasch Abedjan	TU Berlin, Germany
Koren Abitbul	Ben-Gurion University, Israel
Timilehin Aderinola	Insight SFI Research Centre for Data Analytics, University College Dublin, Ireland
Homayun Afrabandpey	Nokia Technologies, Finland
Reza Akbarinia	Inria, France
Esra Akbas	Georgia State University, USA
Cuneyt Akcora	University of Central Florida, USA
Youhei Akimoto	University of Tsukuba/RIKEN AIP, Japan
Ozge Alacam	University of Bielefeld, Germany
Amr Alkhatib	KTH Royal Institute of Technology, Sweden
Mari-Liis Allikivi	University of Tartu, Estonia
Ranya Almohsen	West Virginia University, USA
Jose Alvarez	Scuola Normale Superiore, Italy
Ehsan Aminian	INESC TEC, Portugal
Christos Anagnostopoulos	University of Glasgow, UK
James Anderson	Columbia University, USA
Thiago Andrade	INESC TEC/University of Porto, Portugal
Jean-Marc Andreoli	Naverlabs Europe, France
Giuseppina Andresini	University of Bari 'Aldo Moro', Italy
Simone Angarano	Politecnico di Torino, Italy
Akash Anil	Cardiff University, Wales, UK
Ekaterina Antonenko	Mines Paris - PSL, France
Alessandro Antonucci	IDSIA, Switzerland
Edward Apeh	Bournemouth University, UK
Nikhilanand Arya	Indian Institute of Technology, Patna, India
Saeed Asadi Bagloee	University of Melbourne, Australia
Ali Ayadi	University of Strasbourg, France
Steve Azzolin	University of Trento, Italy
Lilian Berton	Universidade Federal de Sao Paulo, Brazil

Florian Babl	Universität der Bundeswehr München, Germany
Michael Bain	University of New South Wales, Australia
Chandrajit Bajaj	University of Texas, Austin, USA
Bunil Balabantaray	NIT Meghalaya, India
Federico Baldo	University of Bologna, Italy
Georgia Baltsov	Information Technologies Institute/Centre for Research & Technology - Hellas, Greece
Hubert Baniecki	University of Warsaw, Poland
Mitra Baratchi	LIACS - University of Leiden, the Netherlands
Francesco Bariatti	Univ Rennes, CNRS, IRISA, France
Franka Bause	University of Vienna, Austria
Florian Beck	JKU Linz, Austria
Jacob Beck	LMU Munich, Germany
Rita Beigaitė	VTT, Finland
Michael Beigl	Karlsruhe Institute of Technology, Germany
Diana Benavides Prado	University of Auckland, New Zealand
Andreas Bender	LMU Munich, Germany
Idir Benouaret	Epita Research Laboratory, France
Gilberto Bernardes	INESC TEC & University of Porto, Faculty of Engineering, Portugal
Jolita Bernatavičienė	Vilnius University, Lithuania
Cuissart Bertrand	University of Caen, France
Eva Besada-Portas	Universidad Complutense de Madrid, Spain
Jalaj Bhandari	Columbia University, USA
Monowar Bhuyan	Umea University, Sweden
Manuele Bicego	University of Verona, Italy
Przemyslaw Biecek	Warsaw University of Technology, Poland
Albert Bifet	Telecom Paris, France
Livio Bioglio	University of Turin, Italy
Anton Björklund	University of Helsinki, Finland
Szymon Bobek	Jagiellonian University, Poland
Ludovico Boratto	University of Cagliari, Italy
Stefano Bortoli	Huawei Research Center
Annelot Bosman	Universiteit Leiden, the Netherlands
Tassadit Bouadi	Université de Rennes, France
Hamid Bouchachia	Bournemouth University, UK
Jannis Brugger	TU Darmstadt, Germany
Dariusz Brzezinski	Poznan University of Technology, Poland
Maria Sofia Bucarelli	Sapienza University of Rome, Italy
Mirko Bunse	TU Dortmund University, Germany
Tomasz Burzykowski	Hasselt University, Belgium

Sebastian Buschjäger	TU Dortmund Artificial Intelligence Unit, Germany
Maarten Buyl	Ghent University, Belgium
Zaineb Chelly Dagdia	UVSQ, Paris-Saclay, France
Huaming Chen	University of Sydney, Australia
Xiaojun Chen	Institute of Information Engineering, CAS, China
Tobias Callies	Universität der Bundeswehr München, Germany
Xiaofeng Cao	University of Technology Sydney, Australia
Cécile Capponi	Aix-Marseille University, France
Lorenzo Cascioli	KU Leuven, Belgium
Guilherme Cassales	University of Waikato, New Zealand
Giovanna Castellano	University of Bari 'Aldo Moro', Italy
Andrea Cavallo	Delft University of Technology, the Netherlands
Remy Cazabet	Lyon, France
Antanas Čenys	Vilnius Gediminas Technical University, Lithuania
Mattia Cerrato	JGU Mainz, Germany
Ricardo Cerri	Federal University of Sao Carlos, Brazil
Prithwish Chakraborty	IBM Corporation
Harry Kai-Ho Chan	University of Sheffield, UK
Laetitia Chapel	IRISA, France
Victor Charpenay	Mines Saint-Etienne, France
Arthur Charpentier	UQAM, Canada
Chunchun Chen	Tongji University, China
Huiping Chen	University of Birmingham, UK
Jin Chen	Hong Kong University of Science and Technology, China
Kuan-Hsun Chen	University of Twente, the Netherlands
Lingwei Chen	Wright State University, USA
Minyu Chen	Shanghai Jiaotong University, China
Xuefeng Chen	Chongqing University, China
Ying Chen	RMIT University, Australia
Zheng Chen	Osaka University, Japan
Zhong Chen	Southern Illinois University, USA
Ziheng Chen	Walmart, USA
Zehua Cheng	University of Oxford, UK
Hua Chu	Xidian University, China
Oana Cocarascu	King's College London, UK
Johanne Cohen	LISN-CNRS, France
Lidia Contreras-Ochando	Universitat Politècnica de València, Spain
Denis Coquenat	IRISA, France
Luca Corbucci	University of Pisa, Italy

Roberto Corizzo	American University, USA
Nathan Cornille	KU Leuven, Belgium
Baris Coskunuzer	University of Texas at Dallas, USA
Andrea Cossu	University of Pisa, Italy
Tiago Cunha	Expedia Group, Portugal
Florence d'Alché-Buc	Télécom Paris, France
Sebastian Dalleiger	KTH Royal Institute of Technology, Sweden
Robertas Damaševičius	Vytautas Magnus University, Lithuania
Xuan-Hong Dang	IBM T.J. Watson Research Center, USA
Thi-Bich-Hanh Dao	University of Orleans, France
Paul Davidsson	Malmö University, Sweden
Jasper de Boer	KU Leuven, Belgium
Andre de Carvalho	USP, Brazil
Graziella De Martino	University of Bari 'Aldo Moro', Italy
Lennert De Smet	KU Leuven, Belgium
Marcilio de Souto	LIFO/Univ. Orleans, France
Julien Delaunay	Inria, France
Emanuele Della Valle	Politecnico di Milano, Italy
Pieter Delobelle	KU Leuven, Belgium
Vincent Derkinderen	KU Leuven, Belgium
Guillaume Derval	UCLouvain - ICTEAM, Belgium
Sebastien Destercke	UTC, France
Laurens Devos	KU Leuven, Belgium
Bhaskar Dhariyal	University College Dublin, Ireland
Davide Di Pierro	Università degli Studi di Bari, Italy
Yiqun Diao	National University of Singapore, Singapore
Lucile Dierckx	Université catholique de Louvain, Belgium
Anastasia Dimou	KU Leuven, Belgium
Jingtao Ding	Tsinghua University, China
Zifeng Ding	LMU Munich, Germany
Lamine Diop	EPITA, France
Christos Diou	Harokopio University of Athens, Greece
Alexander Dockhorn	Leibniz University Hannover, Germany
Stephan Doerfel	Kiel University of Applied Sciences, Germany
Hang Dong	University of Oxford, UK
Nanqing Dong	Shanghai Artificial Intelligence Laboratory, China
Emilio Dorigatti	LMU Munich, Germany
Haizhou Du	Shanghai University of Electric Power, China
Stefan Duffner	University of Lyon, France
Inês Dutra	University of Porto, Portugal
Anany Dwivedi	University of Waikato, New Zealand
Sofiane Ennadir	KTH Royal Institute of Technology, Sweden

Mark Eastwood	University of Warwick, UK
Vasilis Efthymiou	Harokopio University of Athens, Greece
Rémi Emonet	Université Saint-Etienne, France
Dominik Endres	Philipps-Universität Marburg, Germany
Eshant English	Hasso Plattner Institute, Germany
Bojan Evkoski	Central European University, Austria
Zipei Fan	University of Tokyo, Japan
Hadi Fanaee-T	Halmstad University, Germany
Fabio Fassetti	Universita della Calabria, Italy
Ad Feelders	Universiteit Utrecht, the Netherlands
Wenjie Feng	National University of Singapore, Singapore
Len Feremans	Universiteit Antwerpen, Belgium
Luca Ferragina	University of Calabria, Italy
Carlos Ferreira	INESC TEC, Portugal
Julien Ferry	LAAS-CNRS, France
Michele Fontana	Università di Pisa, Italy
Germain Forestier	University of Haute Alsace, France
Edouard Fouché	Karlsruhe Institute of Technology (KIT), Germany
Matteo Francobaldi	University of Bologna, Italy
Christian Frey	Fraunhofer IIS, Germany
Holger Froening	University of Heidelberg, Germany
Benoît Frénay	University of Namur, Belgium
Fabio Fumarola	Prometeia, Italy
Shanqing Guo	Shandong University, China
Claudio Gallicchio	University of Pisa, Italy
Shengxiang Gao	Kunming University of Science and Technology, China
Yifeng Gao	University of Texas Rio Grande Valley, USA
Manuel Garcia-Piqueras	Universidad de Castilla-La Mancha, Spain
Dario Garigliotti	University of Bergen, Norway
Damien Garreau	Université Côte d'Azur, France
Dominique Gay	Université de La Réunion, France
Alborz Geramifard	Meta, USA
Pierre Geurts	Montefiore Institute, University of Liège, Belgium
Alireza Gharahighehi	KU Leuven, Belgium
Siamak Ghodsi	Leibniz University of Hannover Free University Berlin, Germany
Shreya Ghosh	Penn State, USA
Vasilis Gkolemis	ATHENA RC, Greece
Dorota Glowacka	University of Helsinki, Finland
Heitor Gomes	Victoria University of Wellington, New Zealand

Wenwen Gong	Tsinghua University, China
Adam Goodge	I2R, A*STAR, Singapore
Anastasios Gounaris	Aristotle University of Thessaloniki, Greece
Brandon Gower-Winter	Utrecht University, the Netherlands
Michael Granitzer	University of Passau, Germany
Xinyu Guan	Xian Jiaotong University, China
Massimo Guarascio	ICAR-CNR, Italy
Riccardo Guidotti	University of Pisa, Italy
Dominique Guillot	University of Delaware, USA
Nuwan Gunasekara	AI Institute, University of Waikato, New Zealand
Thomas Guyet	Inria, Centre de Lyon, France
Vanessa Gómez-Verdejo	Universidad Carlos III de Madrid, Spain
Huong Ha	RMIT University, Australia
Benjamin Halstead	University of Auckland, New Zealand
Marwan Hassani	TU Eindhoven, the Netherlands
Yujiang He	University of Kassel, Germany
Edith Heiter	Ghent University, Belgium
Lars Hillebrand	Fraunhofer IAIS and University of Bonn, Germany
Martin Holena	Institute of Computer Science, Czechia
Mike Holenderski	Eindhoven University of Technology, the Netherlands
Hongsheng Hu	Data 61, CSIRO, Australia
Chao Huang	University of Hong Kong, China
Denis Huseljic	University of Kassel, Germany
Julian Höllig	University of the Bundeswehr Munich, Germany
Dimitrios Iliadis	UGENT, Belgium
Dino Ienco	INRAE, France
Roberto Interdonato	CIRAD, France
Omid Isfahani Alamdari	University of Pisa, Italy
Elvin Isufi	TU Delft, the Netherlands
Giulio Jacucci	University of Helsinki, Finland
Kuk Jin Jang	University of Pennsylvania, USA
Inigo Jauregi Unanue	University of Technology Sydney, Australia
Renhe Jiang	University of Tokyo, Japan
Pengfei Jiao	Hangzhou Dianzi University, China
Yilun Jin	Hong Kong University of Science and Technology, China
Rūta Juozaitienė	Vytautas Magnus University, Lithuania
Joonas Jälkö	University of Helsinki, Finland
Mira Jürgens	Ghent University, Belgium

Vana Kalogeraki	Athens University of Economics and Business, Greece
Toshihiro Kamishima	Independent Researcher, Japan
Nikos Kanakaris	University of Southern California, USA
Sevvandi Kandanaarachchi	CSIRO, Australia
Bo Kang	Ghent University, Belgium
Jurgita Kapočiūtė-Dzikienė	Tilde SIA, University of Latvia, Tilde IT, Vytautas Magnus University, Lithuania
Maiju Karjalainen	University of Eastern Finland, Finland
Panagiotis Karras	University of Copenhagen, Denmark
Gjergji Kasneci	TU Munich, Germany
Panagiotis Kasnesis	University of West Attica, Greece
Dimitrios Katsaros	University of Thessaly, Greece
Natthawut Kertkeidkachorn	Japan Advanced Institute of Science and Technology (JAIST), Japan
Stefan Kesselheim	Forschungszentrum Jülich, Germany
Jaleed Khan	University of Oxford, UK
Adem Kikaj	KU Leuven, Belgium
Nadja Klein	University Alliance Ruhr and TU Dortmund, Germany
Tomas Kliegr	University of Economics Prague, Czechia
Astrid Klipfel	CRIL - UMR 8188, France
Simon Koop	Technische Universiteit Eindhoven, the Netherlands
Frederic Koriche	Univ. d'Artois, CRIL CNRS UMR 8188, France
Grazina Korvel	Vilnius University, Lithuania
Ana Kostovska	Jožef Stefan Institute, Slovenia
Stefan Kramer	Johannes Gutenberg University Mainz, Germany
Emmanouil Krasanakis	CERTH, Greece
Anna Krause	Universität Würzburg, Germany
Nils Kriege	University of Vienna, Austria
Ričardas Krikštolaitis	Vytautas Magnus University, Lithuania
Amer Krivosija	TU Dortmund, Germany
Paweł Ksieniewicz	Wrocław University of Science and Technology, Poland
Janne Kujala	University of Turku, Finland
Nitesh Kumar	Cardiff University, UK
Vivek Kumar	Universität der Bundeswehr München, Germany
Olga Kurasova	Vilnius University, Institute of Data Science and Digital Technologies, Lithuania
Marius Köppel	Johannes Gutenberg University Mainz, Germany
Antti Laaksonen	University of Helsinki, Finland
Ville Laitinen	University of Turku, Finland

Carlos Lamuela Orta	University of Helsinki, Finland
Johannes Langguth	Simula Research Laboratory, Norway
Helge Langseth	Norwegian University of Science and Technology, Norway
Martha Larson	Radboud University, the Netherlands
Anton Lautrup	University of Southern Denmark, Denmark
Aonghus Lawlor	University College Dublin, Ireland
Tuan Le	New Mexico State University, USA
Erwan Le Merrer	Inria, France
Thach Le Nguyen	University College Dublin, Ireland
Tai Le Quy	IU International University of Applied Sciences, Germany
Mustapha Lebbah	Paris Saclay University-Versailles, France
Yeon-Chang Lee	Ulsan National Institute of Science and Technology (UNIST), South Korea
Zed Lee	Stockholm University, Sweden
Mathieu Lefort	Univ. Lyon, France
Vincent Lemaire	Orange Innovation
Daniel Lemire	University of Quebec (TELUQ), Canada
Florian Lemmerich	University of Passau, Germany
Daphne Lenders	University of Antwerp, Belgium
Carson Leung	University of Manitoba, Canada
Dan Li	Sun Yat-Sen University, China
Gang Li	Deakin University, Australia
Mark Junjie Li	Shenzhen University, China
Mingxio Li	KU Leuven, Belgium
Nian Li	Tsinghua University, China
Peiyan Li	Ludwig Maximilian University of Munich, Germany
Shuai Li	University of Cambridge, UK and University of Tokyo, Japan and Tsinghua University, China
Tong Li	HKUST, China
Xiang Li	East China Normal University, China
Yinsheng Li	Fudan University, China
Yong Li	Huawei European Research Center, Germany
Zhixin Li	Guangxi Normal University, China
Zhuoqun Li	Louisiana State University, USA
Yuxuan Liang	Hong Kong University of Science and Technology, China
Nick Lim	University of Waikato, New Zealand
Jason Lines	Independent Researcher, UK
Piotr Lipinski	Institute of Computer Science, University of Wroclaw, Poland

Arunas Lipnickas	Kaunas University of Technology, Lithuania
Marco Lippi	University of Florence, Italy
Bin Liu	Chongqing University of Posts and Telecommunications, China
Fenglin Liu	University of Oxford, UK
Junze Liu	University of California, Irvine, USA
Li Liu	Chongqing University, China
Xu Liu	National University of Singapore, Singapore
Zihan Liu	Zhejiang University & Westlake University, China
Corrado Loglisci	Università degli Studi di Bari 'Aldo Moro', Italy
Antonio Longa	University of Trento, Italy
Marco Loog	Radboud University, the Netherlands
Ana Carolina Lorena	ITA, Brazil
Beatriz López	University of Girona, Spain
Tuwe Löfström	Jönköping University, Sweden
Pingchuan Ma	HKUST, China
Ziqiao Ma	University of Michigan, USA
Henryk Maciejewski	Wrocław University of Science and Technology, Poland
Michael Madden	National University of Ireland Galway, Ireland
Sindri Magnusson	Stockholm University, Sweden
Ajay Mahimkar	AT&T, USA
Cedric Malherbe	AstraZeneca, UK
Giuseppe Manco	ICAR-CNR, Italy
Domenico Mandaglio	DIMES Dept., University of Calabria, Italy
Justina Mandravickaitė	Vytautas Magnus University, Lithuania
Silviu Maniu	Université Grenoble Alpes, France
Naresh Manwani	International Institute of Information Technology, Hyderabad, India
Alexandru Mara	Ghent University, Belgium
Virginijus Marcinkevičius	Vilnius University, Lithuania
Timo Martens	KU Leuven, Belgium
Linas Martišauskas	Vytautas Magnus University, Lithuania
Fernando Martínez-Plumed	Universitat Politècnica de València, Spain
Koji Maruhashi	Fujitsu Research, Fujitsu Limited
Rytis Maskeliūnas	PolSl, Poland
Florent Masegla	Inria, France
Antonio Mastropietro	Università di Pisa, Italy
Sarah Masud	LCS2, IIIT-D, India
Dalius Matuzevicius	Vilnius Gediminas Technical University, Lithuania
Chandresh Maurya	IBM Research, India

Wolfgang Mayer	University of South Australia, Australia
Giacomo Medda	University of Cagliari, Italy
Nida Meddouri	LRE-EPITA, France
Stefano Melacci	University of Siena, Italy
Alessandro Melchiorre	Johannes Kepler University Linz, Austria
Marco Mellia	Politecnico di Torino, Italy
Joao Mendes-Moreira	University of Porto, Portugal
Engelbert Mephu Nguifo	Université Clermont Auvergne, CNRS, LIMOS, France
Fabio Mercurio	University of Milan-Bicocca, Italy
Henning Meyerhenke	Humboldt-Universität zu Berlin, Germany
Matthew Middlehurst	University of Southampton, UK
Jan Mielniczuk	Polish Academy of Sciences, Poland
Paolo Mignone	University of Bari ‘Aldo Moro’, Italy
Matej Mihelčić	University of Zagreb, Croatia
Tsunenori Mine	Kyushu University, Japan
Pierre Monnin	Université Côte d’Azur, Inria, CNRS, I3S, France
Carlos Monserrat-Aranda	Universitat Politècnica de València, Spain
Raha Moraffah	Arizona State University, USA
Thomas Mortier	Ghent University, Belgium
Frank Mtumbuka	Cardiff University, Wales, UK
Koyel Mukherjee	Adobe Research, India
Mario Andrés Muñoz	University of Melbourne, Australia
Nikolaos Mylonas	Aristotle University of Thessaloniki, Greece
Tommi Mäklin	University of Helsinki, Finland
Felipe Kenji Nakano	KU Leuven, Belgium
Géraldin Nanfack	University of Concordia, Canada
Mirco Nanni	CNR-ISTI Pisa, Italy
Francesca Naretto	University of Pisa, Italy
Fateme Nateghi Haredasht	Stanford University, USA
Benjamin Negrevergne	Université PSL – Paris Dauphine, France
Matti Nelimarkka	University of Helsinki, Finland
Kim Thang Nguyen	LIG, University Grenoble-Alpes, France
Shiwen Ni	Shenzhen Institute of Advanced Technology (SIAT), Chinese Academy of Sciences, China
Mikko Niemi	City of Helsinki, Finland
Nikolaos Nikolaou	University College London, UK
Simona Nisticò	University of Calabria, Italy
Hao Niu	KDDI Research, Inc., Japan
Andreas Nuernberger	Magdeburg University, Germany
Claire Nédellec	INRAE, MaIAGE, France
Barry O’Sullivan	University College Cork, Ireland

Makoto Onizuka	Osaka University, Japan
Jose Oramas	University of Antwerp, IMEC-IDLab, Belgium
Luis Ortega Andrés	Autonomous University of Madrid, Spain
Latifa Oukhellou	IFSTTAR, France
Agne Paulauskaite-Taraseviciene	KTU, Artificial Intelligence Centre, Lithuania
Massimo Piccardi	University of Technology Sydney, Australia
Marc Plantevit	EPITA, France
Andrei Paleyes	University of Cambridge, UK
Emmanouil Panagiotou	Freie Universität Berlin, Germany
George Panagopoulos	University of Luxembourg
Pance Panov	Jozef Stefan Institute, Slovenia
Apostolos Papadopoulos	Aristotle University of Thessaloniki, Greece
Panagiotis Papapetrou	Stockholm University, Sweden
Francesco Parisi	University of Calabria, Italy
Abigail Parker	University of Helsinki, Finland
Antonio Parmezan	University of São Paulo, Brazil
Vincenzo Pasquadibisceglie	University of Bari 'Aldo Moro', Italy
Tatiana Passali	Aristotle University of Thessaloniki, Greece
Eliana Pastor	Politecnico di Torino, Italy
Anand Paul	Louisiana State University HSC, USA
Mykola Pechenizkiy	TU Eindhoven, the Netherlands
Yulong Pei	TU Eindhoven, the Netherlands
Nikos Pelekis	University of Piraeus, Greece
Leonardo Pellegrina	University of Padova, Italy
Charlotte Pelletier	Université de Bretagne du Sud, France
Antonio Pellicani	Università degli Studi di Bari 'Aldo Moro', Italy
Frédéric Pennerath	CentraleSupélec - LORIA, France
Ruggero Pensa	University of Torino, Italy
Lucas Pereira	Interactive Technologies Institute, LARSyS, Técnico Lisboa, Portugal
Pedro Pereira Rodrigues	University of Porto, Portugal
Miquel Perello-Nieto	University of Bristol, UK
Lorenzo Perini	KU Leuven, Belgium
Linus Petkevicius	Vilnius University, Lithuania
Ninh Pham	University of Auckland, New Zealand
Nico Piatkowski	Fraunhofer IAIS, Germany
Francesco Piccialli	Independent Researcher, Italy
Martin Pilát	Charles University, Czechia
Gianvito Pio	University of Bari, Italy
Darius Plonis	Vilnius Gediminas Technical University, Lithuania
Marco Podda	University of Pisa, Italy

Mirko Polato	University of Turin, Italy
Marco Polignano	Università di Bari, Italy
Giovanni Ponti	ENEA, Italy
Alexandru Popa	University of Bucharest, Romania
Fabrice Popineau	CentraleSupélec/LISN, France
Cedric Pradalier	GeorgiaTech Lorraine, France
Paul Prasse	University of Potsdam, Germany
Mahardhika Pratama	University of South Australia, Australia
Bardh Prenkaj	Sapienza University of Rome, Italy
Steven Prestwich	University College Cork, Ireland
Giulia Preti	CENTAI, Italy
Philippe Preux	Inria, France
Danil Provodin	TU Eindhoven, the Netherlands
Chiara Pugliese	ISTI Institute of National Research Council University of Pisa, Italy
Simon Puglisi	University of Helsinki, Finland
Andrea Pugnana	University of Pisa, Italy
Erasmus Purificato	Otto von Guericke University Magdeburg, Germany
Peter van der Putten	Leiden University, the Netherlands
Abdulhakim Qahtan	Utrecht University, the Netherlands
Kun Qian	Amazon, USA
Kallol Roy	University of Tartu, Estonia
Dimitrios Rafailidis	University of Thessaly, Greece
Muhammad Rajabinasab	University of Southern Denmark, Denmark
Chang Rajani	University of Helsinki, Finland
Simona Ramanauskaitė	Vilnius Gediminas Technical University, Lithuania
Jan Ramon	Inria, France
M. José Ramírez-Quintana	Technical University of Valencia, Spain
Rajeev Rastogi	Amazon, USA
Domenico Redavid	University of Bari, Italy
Luis Rei	Jožef Stefan Institute, Slovenia
Christoph Reinders	Leibniz University Hannover, Germany
Qianqian Ren	Heilongjiang University, China
Mina Rezaei	LMU Munich, Germany
Rita Ribeiro	Porto, Portugal
Matteo Riondato	Amherst College, USA
Simon Rittel	University of Vienna, Austria
Giuseppe Rizzo	Niuma s.r.l, Italy
Pieter Robberechts	KU Leuven, Belgium

Christophe Rodrigues	DVRC pôle universitaire Léonard de Vinci, France
Federica Rollo	UNIMORE, Italy
Luca Romeo	University of Macerata, Italy
Nicolas Roque dos Santos	University of São Paulo, Brazil
Céline Rouveirol	LIPN Univ. Sorbonne Paris Nord, France
Arjun Roy	Freie Universität Berlin, Germany
Krzysztof Rudaś	Institute of Computer Science, Polish Academy of Sciences, Poland
Allou Same	Université Gustave Eiffel, France
Oswaldo Solarte-Pabon	Universidad del Valle, Spain
Amal Saadallah	TU Dortmund, Germany
Matthia Sabatelli	University of Groningen, the Netherlands
Chafik Samir	CNRS-UCA, France
Ramses Sanchez	University of Bonn, Germany
Ioannis Sarridis	Information Technologies Institute/Centre for Research & Technology - Hellas, Greece
Milos Savic	University of Novi Sad, Serbia
Nripsuta Saxena	University of Southern California, USA
Alexander Schiendorfer	Technische Hochschule Ingolstadt, Germany
Christian Schlauch	Humboldt-Universität zu Berlin, Germany
Rainer Schlosser	Hasso Plattner Institute, Germany
Johannes Schneider	University of Liechtenstein, Liechtenstein
Rianne Schouten	Technische Universiteit Eindhoven, the Netherlands
Andreas Schwung	Fachhochschule Südwestfalen, Germany
Patrick Schäfer	Humboldt-Universität zu Berlin, Germany
Kristen Scott	KU Leuven, Belgium
Marian Scuturici	LIRIS, France
Raquel Sebastião	ESTGV-IPV & IEETA-UA
Nina Seemann	University of the Bundeswehr, Germany
Artūras Serackis	Vilnius Tech, Lithuania
Giuseppe Serra	Goethe University Frankfurt, Germany
Mattia Setzu	University of Pisa, Italy
Manali Sharma	Samsung, USA
Shubhranshu Shekhar	Brandeis University, USA
Qiang Sheng	Institute of Computing Technology, Chinese Academy of Sciences, China
John Sheppard	Montana State University, USA
Bin Shi	Xi'an Jiaotong University, China
Jimeng Shi	Florida International University, USA
Paula Silva	INESC TEC - LIAAD, Portugal

Telmo Silva Filho	University of Bristol, UK
Esther-Lydia Silva-Ramírez	Universidad de Cádiz, Spain
Raivydas Šimėnas	Vilnius University, Lithuania
Kuldeep Singh	Cerence GmbH, Germany
Andrzej Skowron	University of Warsaw, Poland
Carlos Soares	University of Porto, Portugal
Dennis Soemers	Maastricht University, the Netherlands
Andy Song	RMIT University, Australia
Liyan Song	Harbin Institute of Technology, China
Zixing Song	Chinese University of Hong Kong, China
Sucheta Soundarajan	Syracuse University, USA
Fabian Spaeh	Boston University, USA
Myra Spiliopoulou	Otto-von-Guericke-University Magdeburg, Germany
Dimitri Stauffer	TU Berlin, Germany
Kostas Stefanidis	Tampere University, Finland
Pavel Stefanovič	Vilnius Tech, Lithuania
Julian Stier	University of Passau, Germany
Giovanni Stilo	Università of L'Aquila, Italy
Michiel Stock	Ghent University, Belgium
Luca Stradiotti	KU Leuven, Belgium
Lukas Struppek	Technical University of Darmstadt, Germany
Maximilian Stubbemann	University of Hildesheim, Germany
Nikolaos Stylianou	Information Technologies Institute, Greece
Jinyan Su	University of Electronic Science and Technology of China, China
Peijie Sun	Tsinghua University, China
Weiwei Sun	Shandong University, China
Swati Swati	Universität der Bundeswehr München, Germany
Panagiotis Symeonidis	University of the Aegean, Greece
Maryam Tabar	University of Texas at San Antonio, USA
Shazia Tabassum	INESC TEC, Portugal
Andrea Tagarelli	DIMES - UNICAL, Italy
Martin Takac	Mohamed bin Zayed University of Artificial Intelligence, UAE
Acar Tamersoy	NortonLifeLock Research Group, USA
Chang Wei Tan	Monash University, Australia
Xing Tang	Tencent, China
Enzo Tartaglione	Télécom Paris - Institut Polytechnique de Paris, France
Romain Tavenard	Univ. Rennes, LETG/IRISA, France
Gustaf Tegnér	KTH Royal Institute of Technology, Lithuania

Paweł Teisseyre	Warsaw University of Technology, Poland
Alexandre Termier	Université Rennes, France
Stefano Teso	University of Trento, Italy
Surendrabikram Thapa	Virginia Tech, USA
Martin Theobald	University of Luxembourg, Luxembourg
Maximilian Thiessen	TU Wien, Austria
Steffen Thoma	FZI Research Center for Information Technology, Germany
Matteo Tiezzi	University of Siena, Italy
Matteo Tiezzi	SAILab, DIISM, University of Siena, Italy
Gabriele Tolomei	Sapienza University of Rome, Italy
Paulina Tomaszewska	Warsaw University of Technology, Poland
Dinh Tran	King Fahd University of Petroleum & Minerals, Saudi Arabia
Isaac Triguero	Nottingham University, UK
Andre Tättar	University of Tartu, Estonia
Evaldas Vaičiukynas	Kaunas University of Technology, Lithuania
Jente Van Belle	KU Leuven, Belgium
Fabio Vandin	University of Padova, Italy
Aparna S. Varde	Montclair State University, USA
Bruno Veloso	INESC TEC & FEP-UP, Portugal
Dmytro Velychko	University of Oldenburg, Germany
Sreekanth Vempati	Myntra, India
Gabriele Venturato	KU Leuven, Belgium
Michela Venturini	KU Leuven, ITEC, Belgium
Mathias Verbeke	KU Leuven, Belgium
Théo Verhelst	Université libre de Bruxelles, Belgium
Rosana Veroneze	LBiC, UK
Gennaro Vessio	University of Bari 'Aldo Moro', Italy
Paul Viallard	Inria Rennes, France
Herna Viktor	University of Ottawa, Canada
Joao Vinagre	Joint Research Centre - European Commission, Spain
Jean-Noël Vittaut	Sorbonne Université, CNRS, LIP6, France
Maximilian von Zastrow	Southern Denmark University, Denmark
Tomasz Walkowiak	Wrocław University of Science and Technology, Poland
Beilun Wang	Southeast University, China
Huandong Wang	Tsinghua University, China
Hui (Wendy) Wang	Stevens Institute of Technology, USA
Jianwu Wang	University of Maryland, Baltimore County, USA
Jiaqi Wang	Penn State University, USA

Suhang Wang	Pennsylvania State University, USA
Yanhao Wang	East China Normal University, China
Yimu Wang	University of Waterloo, Canada
Yue Wang	Microsoft Research
Zhaonan Wang	University of Illinois Urbana-Champaign, USA
Zichong Wang	Florida International University, USA
Zifu Wang	KU Leuven, Belgium
Zijie J. Wang	Georgia Tech, USA
Roger Wattenhofer	ETH Zurich, Germany
Tonio Weidler	Maastricht University, the Netherlands
Jörg Wicker	University of Auckland, New Zealand
Alicja Wieczorkowska	Polish-Japanese Academy of Information Technology, Poland
Michael Wilbur	Vanderbilt University, USA
David Winkel	LMU Munich, Germany
Moritz Wohlstein	Leuphana Universität Lüneburg, Germany
Szymon Wojciechowski	Wrocław University of Science and Technology, Poland
Bin Wu	Zhengzhou University, China
Chenwang Wu	University of Science and Technology of China, China
Di Wu	Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, China
Wei Wu	Ben Gurion University of the Negev, Israel
Yongkai Wu	Clemson University, USA
Zhiwen Xiao	Southwest Jiaotong University, China
Cheng Xie	Yunnan University, China
Yaqi Xie	Carnegie Mellon University, USA
Huanlai Xing	Southwest Jiaotong University, China
Xing Xing	Tongji University, China
Ning Xu	Southeast University, China
Weifeng Xu	Weifeng Xu, USA
Ziqi Xu	CSIRO, Australia
Yexiang Xue	Purdue University, USA
Yan Yan	Carleton University, Canada
Yu Yan	School of Information and Cyber Security, People's Public Security University of China, China
Lincen Yang	Leiden University, the Netherlands
Shaofu Yang	Southeast University, China
Muchao Ye	Pennsylvania State University, USA
Kalidas Yeturu	Indian Institute of Technology Tirupati, India

Jaemin Yoo	KAIST, South Korea
Kristina Yordanova	University of Greifswald, Germany
Hang Yu	Shanghai University, China
Jidong Yuan	Beijing Jiaotong University, China
Xiaoyong Yuan	Clemson University, USA
Klim Zaporojets	Aarhus University, Denmark
Claudius Zelenka	Kiel University, Germany
Akka Zemhari	Univ. Bordeaux, France
Guoxi Zhang	Beijing Institute of General Artificial Intelligence, China
Hao Zhang	Fudan University, China
Teng Zhang	Huazhong University of Science and Technology, China
Tianlin Zhang	University of Manchester, UK
Xiang Zhang	National University of Defense Technology, China
Xiao Zhang	Shandong University, China
Xiaoming Zhang	Beihang University, China
Yaqian Zhang	University of Waikato, New Zealand
Yin Zhang	University of Electronic Science and Technology of China
Zhiwen Zhang	University of Tokyo, Japan
Lingxiao Zhao	Carnegie Mellon University, USA
Tongya Zheng	Hangzhou City University, China
Wenhao Zheng	Shopee, Singapore
Yu Zheng	Tsinghua University, China
Yujia Zheng	CMU, USA
Zhengyang Zhou	University of Science and Technology of China, China
Jing Zhu	University of Michigan, Ann Arbor, USA
Ye Zhu	Deakin University, Australia
Yichen Zhu	Midea Group, China
Zirui Zhuang	Beijing University of Posts and Telecommunications, China
Tommaso Zoppi	University of Florence, Italy
Pedro Zuidberg Dos Martires	Örebro University, Sweden
Meiyun Zuo	Renmin University of China, China

Program Committee Members, Applied Data Science Track

Ziawasch Abedjan	TU Berlin, Germany
Shahrooz Abghari	Blekinge Institute of Technology, Sweden
Christian M. Adriano	Hasso-Plattner Institute, Germany
Haluk Akay	KTH Royal Institute of Technology, Lithuania
Fahed Alkhabbas	Malmö University, Sweden
Mohammed Ghaith Altarabichi	Högskolan i Halmstad, Sweden
Evelin Amorim	INESC TEC, Portugal
Giuseppina Andresini	University of Bari 'Aldo Moro', Italy
Sunil Aryal	Deakin University, New Zealand
Awais Ashfaq	Region Halland, Sweden
Asma Atamna	Ruhr-University Bochum, Germany
Berkay Aydin	Georgia State University, USA
Mehdi Bahrami	Fujitsu Research of America, USA
Hareesh Bahuleyan	Zalando, Sweden
Michael Bain	University of New South Wales, Australia
Hubert Baniecki	University of Warsaw, Poland
Enda Barrett	University of Galway, Ireland
Michele Bernardini	Università Politecnica delle Marche, Ancona, Italy
Lilian Berton	Universidade Federal de Sao Paulo, Brazil
Antonio Bevilacqua	Meetecho, Italy
Szymon Bobek	Jagiellonian University, Poland
Veselka Boeva	Blekinge Institute of Technology, Sweden
Martin Boldt	Blekinge Institute of Technology, Sweden
Anton Borg	Blekinge Institute of Technology, Sweden
Cecile Bothorel	IMT Atlantique, France
Mohamed Reda Bouadjenek	Deakin University, New Zealand
Axel Brando	Barcelona Supercomputing Center (BSC) and Universitat de Barcelona (UB), Spain
Stefan Byttner	Halmstad University, Sweden
Ece Calikus	KTH Royal Institute of Technology, Lithuania
Shilei Cao	Tencent, China
Yixuan Cao	Institute of Computing Technology, CAS, China
Hau Chan	University of Nebraska-Lincoln, USA
Chung-Chi Chen	National Taiwan University, Taiwan
Lei Chen	Hong Kong University of Science and Technology, China
Wei-Peng Chen	Fujitsu Research of America, USA
Zhiyu Chen	Amazon, USA
Dawei Cheng	Tongji University, China

Wei Cheng	NEC Laboratories America
Farhana Choudhury	University of Melbourne, Australia
Linyang Chu	McMaster University, Canada
Zhendong Chu	University of Virginia, USA
Paolo Cintia	Kode srl, Italy
Pablo José Del Moral Pastor	Ekkono.ai, Sweden
Yushun Dong	University of Virginia, USA
Antoine Doucet	La Rochelle Université, France
Farzaneh Etminani	Halmstad University and Region Halland, Sweden
Michael Faerber	KIT, Germany
Yuantao Fan	Halmstad University, Sweden
Yixiang Fang	Chinese University of Hong Kong, China
Damien Fay	INFOR Logicblox, USA
Dayne Freitag	SRI International, USA
Erik Frisk	Linköping University, Sweden
Yanjie Fu	Arizona State University, USA
Ariel Fuxman	Google, USA
Xiaofeng Gao	Shanghai Jiaotong University, China
Yunjun Gao	Zhejiang University, China
Lluis Garcia-Pueyo	Meta, USA
Mariana-Iuliana Georgescu	Helmholtz Munich, Germany
Aakash Goel	Amazon, USA
Markus Götz	Karlsruhe Institute of Technology (KIT), Germany
Håkan Grahñ	Blekinge Institute of Technology, Sweden
Francesco Guerra	University of Modena e Reggio Emilia, Italy
Nuno RPS Guimarães	INESC TEC & University of Porto, Portugal
Huifeng Guo	Huawei Noah's Ark Lab, Canada
Vinayak Gupta	University of Washington Seattle, USA
Jinyoung Han	Sungkyunkwan University, South Korea
Shuchu Han	StellarCyber, USA
Julia Handl	University of Manchester, UK
Atiye Sadat Hashemi	Halmstad University, Sweden
Aron Henriksson	Stockholm University, Sweden
Andreas Holzinger	University of Natural Resources and Life Sciences Vienna, Austria
Sebastian Hönel	Linnaeus University, Sweden
Ping-Chun Hsieh	National Yang Ming Chiao Tung University, Taiwan
Zhengyu Hu	HKUST, China
Chao Huang	University of Notre Dame, USA

Hong Huang	Huazhong University of Science and Technology, China
Yizheng Huang	York University, UK
Yu Huang	University of Florida, USA
Angelo Impedovo	Niuma s.r.l., Italy
Radu Tudor Ionescu	University of Bucharest, Romania
Wei Jin	Emory University, USA
Xiaobo Jin	Xi'an Jiaotong-Liverpool University, China
Xiaolong Jin	Institute of Computing Technology, CAS, China
Pinar Karagoz	Middle East Technical University (METU), Turkey
Saeed Karami Zarandi	Halmstad University, Sweden
Thomas Kober	Zalando, Germany
Elizaveta Kopacheva	LNU, Sweden
Christos Koutras	TU Delft, the Netherlands
Adit Krishnan	University of Illinois at Urbana-Champaign, USA
Rafal Kucharski	Jagiellonian University, Poland
Niraj Kumar	Fujitsu, India
Krzysztof Kutt	Jagiellonian University, Poland
Susana Ladra	University of A Coruña, Spain
Matthieu Latapy	CNRS, France
Niklas Lavesson	Blekinge Institute of Technology, Sweden
Roy Ka-Wei Lee	Singapore University of Technology and Design, Singapore
Alessandro Leite	Inria, France
Daniel Lemire	University of Quebec (TELUQ), Canada
Chang Li	Apple, USA
Daifeng Li	Sun Yat-Sen University, China
Haifang Li	Baidu Inc., China
Junxuan Li	Microsoft, USA
Lei Li	Hong Kong University of Science and Technology, China
Shijun Li	University of Science and Technology of China
Shuai Li	University of Cambridge, UK and University of Tokyo, Japan and Tsinghua University, China
Wei Li	Harbin Engineering University, China
Xiang Lian	Kent State University, USA
Guojun Liang	Halmstad University, Sweden
Zhaohui Liang	National Library of Medicine, NIH, USA
Kwan Hui Lim	Singapore University of Technology and Design, Singapore
Adi Lin	Didi, China

Bang Liu	University of Montreal, Canada
Dugang Liu	Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen University, China
Jingjing Liu	MD Anderson Cancer Center, USA
Li Liu	Chongqing University, China
Qing Liu	Zhejiang University, China
Xueyan Liu	Jilin University, China
Yongchao Liu	Ant Group, China
Andreas Lommatzsch	TU Berlin, Germany
Ping Luo	Chinese Academy of Sciences, China
Guixiang Ma	Intel Labs, USA
Zongyang Ma	York University, UK
Saulo Martiello Mastelini	Volt Robotics, Brazil
Elio Masciari	University of Naples, Italy
Nédra Mellouli	LIASD, France
Zoltan Miklos	University of Rennes, France
Mihaela Mitici	Utrecht University, the Netherlands
Martin Mladenov	Google, Brazil
Ahmed K. Mohamed	Meta, USA
Seung-Hoon Na	Jeonbuk National University, South Korea
Sepideh Nahali	York University, UK
Mirco Nanni	CNR-ISTI Pisa, Italy
Richi Nayak	Queensland University of Technology, Brisbane, Australia
Wee Siong Ng	Institute for Infocomm Research, Singapore
Le Nguyen	University of Oulu, Finland
Thanh Thi Nguyen	Monash University, Australia
Slawomir Nowaczyk	Halmstad University, Sweden
Tomas Olsson	RISE SICS, Sweden
Panagiotis Papadakos	FORTH-ICS, Greece
Manos Papagelis	York University, UK
Panagiotis Papapetrou	Stockholm University, Sweden
Luca Pappalardo	ISTI, Italy
Sepideh Pashami	Halmstad University, Sweden
Vincenzo Pasquadibisceglie	University of Bari 'Aldo Moro', Italy
Leonardo Pellegrina	University of Padova, Italy
Pop Petrica	Technical University of Cluj-Napoca, Romania
Pablo Picazo-Sanchez	Halmstad University, Sweden
Srijith PK	IIT, Hyderabad, India
Buyue Qian	Xi'an Jiaotong University, China
Enayat Rajabi	Halmstad University, Sweden

Yanghui Rao	Sun Yat-sen University, China
Salvatore Rinzivillo	KDDLab - ISTI - CNR, Italy
Riccardo Rosati	Università Politecnica delle Marche, Ancona, Italy
Stefan Rueping	Fraunhofer IAIS, Germany
Snehanshu Saha	BITS Pilani Goa Campus, India
Lou Salaün	Nokia Bell Labs, France
Isak Samsten	Stockholm University, Sweden
Eric Sanjuan	Avignon University, France
Johannes Schneider	University of Liechtenstein, Liechtenstein
Wei Shao	Data61, CSIRO, Australia
Nasrullah Sheikh	IBM Research, USA
Jun Shen	University of Wollongong, Australia
Jingwen Shi	Michigan State University, USA
Yue Shi	Meta, USA
Carlos N. Silla	Pontifical Catholic University of Parana (PUCPR), Brazil
Gianmaria Silvello	University of Padova, Italy
Yang Song	Apple, USA
Shafiullah Soomro	Linnaeus University, Sweden
Efstathios Stamatatos	University of the Aegean, Greece
Ting Su	Imperial College London, UK
Gan Sun	South China University of Technology, China
Munira Syed	Procter & Gamble, USA
Zahra Taghiyarrenani	Halmstad University, Sweden
Liang Tang	Google, USA
Xing Tang	Tencent, China
Junichi Tatemura	Google, USA
Joe Tekli	Lebanese American University, Lebanon
Mingfei Teng	Amazon, USA
Sofia Tolmach	Amazon, USA
Gabriele Tolomei	Sapienza University of Rome, Italy
Ismail Hakki Toroslu	METU, Turkey
Md Zia Ullah	Edinburgh Napier University, UK
Maurice Van Keulen	University of Twente, the Netherlands
Ranga Raju Vatsavai	North Carolina State University, USA
Bruno Veloso	INESC TEC & FEP-UP, Portugal
Chang-Dong Wang	Sun Yat-sen University, China
Chengyu Wang	Alibaba Group, China
Kai Wang	Shanghai Jiao Tong University, China
Pengyuan Wang	University of Georgia, USA
Sen Wang	University of Queensland, USA

Senzhang Wang	Central South University, China
Sheng Wang	Wuhan University, China
Wei Wang	Tsinghua University, China
Wentao Wang	Michigan State University, USA
Xiaoli Wang	Xiamen University, China
Yang Wang	University of Science and Technology of China, China
Yu Wang	Vanderbilt University, USA
Zhibo Wang	Zhejiang University, China
Paweł Wawrzyński	IDEAS NCBR, Poland
Hua Wei	Arizona State University, USA
Shi-ting Wen	Ningbo Tech University, China
Zeyi Wen	Hong Kong University of Science and Technology, China
Avani Wildani	Emory University, USA
Fangzhao Wu	MSRA, China
Jun Wu	University of Illinois at Urbana–Champaign, USA
Wentao Wu	Microsoft Research, USA
Xianchao Wu	NVIDIA, Japan
Haoyi Xiong	Baidu, Inc., China
Guandong Xu	University of Technology Sydney, Australia
Yu Yang	City University of Hong Kong, China
Lina Yao	University of New South Wales, Australia
Fanghua Ye	University College London, UK
Dongxiao Yu	Shandong University, China
Haomin Yu	Aalborg University, Denmark
Ran Yu	DSIS Research Group, University of Bonn, Germany
Erik Zeitler	Stream Analyze, Sweden
Chunhui Zhang	Dartmouth College, USA
Denghui Zhang	Rutgers University, USA
Li Zhang	University of Sheffield, UK
Mengxuan Zhang	Australian National University, Australia
Kaiping Zheng	National University of Singapore
Yucheng Zhou	University of Macau, China
Yuanyuan Zhu	Wuhan University, China
Ziwei Zhu	George Mason University, USA
Vasileios Zografos	sennder, Germany

Program Committee Members, Demo Track

Bijaya Adhikari	University of Iowa, USA
Andrius Budrionis	Norwegian Centre for E-health Research, Norway
Luca Cagliero	Politecnico di Torino, Italy
Tania Cerquitelli	Politecnico di Torino, Italy
Gintautas Daunys	Vilnius University, Lithuania
Katharina Dost	University of Auckland, New Zealand
Sourav Dutta	Huawei Research Centre, Ireland
Françoise Fessant	Orange, France
Christelle Godin	CEA, France
Anil Goyal	Amazon, India
Maciej Grzenda	Warsaw University of Technology, Poland
Marius Gudauskis	Institute of Mechatronics, KTU, Lithuania
Thomas Guyet	Inria, Centre de Lyon, France
Andreas Henelius	Independent Researcher, Finland
Rokas Jurevicius	Scandit AG, Lithuania/Switzerland
Pawan Kumar	IIIT, Hyderabad, India
Olga Kurasova	Vilnius University, Institute of Data Science and Digital Technologies, Lithuania
Moreno La Quatra	Kore University of Enna, Italy
Jan Lemeire	Vrije Universiteit Brussel (VUB), Belgium
Martin Luckner	Warsaw University of Technology, Poland
Hoang Phuc Hau Luu	University of Helsinki, Finland
Jarmo Mäkelä	CSC - IT Center for Science Ltd, Finland
Michael Mathioudakis	University of Helsinki, Finland
Darius Miniotas	Vilnius Gediminas Technical University, Lithuania
Michalis Mountantonakis	FORTH-ICS, and CS Department - University of Crete, Greece
Raj Nath Patel	Huawei Ireland Research Center, Ireland
Darius Plikynas	Vilnius Gediminas Technical University, Lithuania
Alexandre Reiffers	IMT Atlantique, France
Marina Reyboz	Univ. Grenoble Alpes, CEA, LIST, France
Yuya Sasaki	Osaka University, Japan
Ines Sousa	Fraunhofer AICOS, Portugal
Jerzy Stefanowski	Poznan University of Technology, Poland
Guoxin Su	University of Wollongong, Australia
Lu-An Tang	NEC Labs America, USA
Michael C. Thrun	Philipps-Universität Marburg, Germany

Yannis Tzitzikas

FORTH-ICS and Computer Science Department,
University of Crete, Greece

Aleksandras Voicikas

Vilnius University, Lithuania

Jörg Wicker

University of Auckland, New Zealand

Hao Xue

University of New South Wales, Australia

Sponsors



CENTAI



Invited Talks Abstracts

The Dynamics of Memorization and Unlearning

Gintarė Karolina Džiugaitė

Google DeepMind

Abstract. Deep learning models exhibit a complex interplay between memorization and generalization. This talk will begin by exploring the ubiquitous nature of memorization, drawing on prior work on “data diets”, example difficulty, pruning, and other empirical evidence. But is memorization essential for generalization? Our recent theoretical work suggests that eliminating it entirely may not be feasible. Instead, I will discuss strategies to mitigate unwanted memorization by focusing on better data curation and efficient unlearning mechanisms. Additionally, I will examine the potential of pruning techniques to selectively remove memorized examples and explore their impact on factual recall versus in-context learning.

Biography: Gintarė is a senior research scientist at Google DeepMind, based in Toronto, an adjunct professor in the McGill University School of Computer Science, and an associate industry member of Mila, the Quebec AI Institute. Prior to joining Google, Gintarė led the Trustworthy AI program at Element AI/ServiceNow, and obtained her Ph.D. in machine learning from the University of Cambridge, under the supervision of Zoubin Ghahramani. Gintarė was recognized as a Rising Star in Machine Learning by the University of Maryland program in 2019. Her research combines theoretical and empirical approaches to understanding deep learning, with a focus on generalization, memorization, unlearning, and network compression.

The Emerging Science of Benchmarks

Moritz Hardt

Max Planck Institute for Intelligent Systems

Abstract. Benchmarks have played a central role in the progress of machine learning research since the 1980s. Although there's much researchers have done with them, we still know little about how and why benchmarks work. In this talk, I will trace the rudiments of an emerging science of benchmarks through selected empirical and theoretical observations. Looking back at the ImageNet era, I'll discuss what we learned about the validity of model rankings and the role of label errors. Looking ahead, I'll talk about new challenges to benchmarking and evaluation in the era of large language models. The results we'll encounter challenge conventional wisdom and underscore the benefits of developing a science of benchmarks.

Biography: Hardt is a director at the Max Planck Institute for Intelligent Systems, Tübingen. Previously, he was Associate Professor for Electrical Engineering and Computer Sciences at the University of California, Berkeley. His research contributes to the scientific foundations of machine learning and algorithmic decision making with a focus on social questions. He co-authored *Fairness and Machine Learning: Limitations and Opportunities* (MIT Press) and *Patterns, Predictions, and Actions: Foundations of Machine Learning* (Princeton University Press).

Enhancing User Experience with AI-Powered Search and Recommendations at Spotify

Mounia Lalmas-Roelleke

Spotify

Abstract. This talk will explore the pivotal role of search and recommendation systems in enhancing the Spotify user experience. These systems serve as the gateway to Spotify's vast audio catalog, helping users navigate millions of music tracks, podcasts, and audiobooks. Effective search functionality allows users to quickly find specific content, whether it is a favorite song, a trending podcast, or an informative audiobook, while also satisfying broader search needs. Meanwhile, recommendation systems suggest new and relevant content that users might not have thought to search for, while ensuring their current needs for familiar content are met. This encourages exploration and discovery of new artists, genres, and shows, enriching the overall listening experience and keeping users engaged with the platform. Achieving this dual objective of precision and discovery requires sophisticated technology. It involves a deep understanding of representation learning, where both content and user preferences are accurately modeled. Advanced AI techniques, including machine learning and generative AI, play a crucial role in this process. These technologies enable the creation of highly personalized recommendations by understanding complex user behaviors and preferences. Generative AI, for instance, allows us to create personalized playlists, thereby enhancing the user experience with innovative features. This presentation is based on the collective research and publications of numerous contributors at Spotify.

Biography: Mounia is a Senior Director of Research at Spotify and the Head of Tech Research in Personalization, where she leads an interdisciplinary team of research scientists. She also holds an honorary professorship at University College London and serves as a Distinguished Research Fellow at the University of Amsterdam. Previously, Mounia was a Director of Research at Yahoo, overseeing a team focused on advertising quality and collaborating on user engagement projects related to news, search, and user-generated content. Before her tenure at Yahoo, Mounia held a Microsoft Research/RAEng Research Chair at the School of Computing Science, University of Glasgow, and before that was a Professor of Information Retrieval at the Department of Computer Science at Queen Mary, University of London. She is a prominent figure in the research community, regularly serving as a senior program committee member at major conferences such as WSDM, KDD, WWW, and SIGIR. She was also a program

co-chair for SIGIR 2015, WWW 2018, WSDM 2020, and CIKM 2023. Mounia is widely recognized for her contributions as a speaker and author, with over 250 published papers and appearances on platforms like ACM ByteCast and the AI Business Podcasts series. She was nominated for the VentureBeat Women in AI Awards for Research in both 2022 and 2023.

How to Utilize (and Generate) Player Tracking Data in Sport

Patrick Lucey

Stats Perform

Abstract. Even though player tracking data in sports has been around for 25 years, it still poses as one of the most interesting and challenging datasets in machine learning due to its fine-grained, multi-agent, team-based, and adversarial nature. Despite these challenges, it is also extremely valuable as it is (relatively) low-dimensional, interpretable, and interactive, allowing us to measure performance and answer questions we couldn't objectively address before. In this talk, I will first give a brief history of tracking data in sports, then highlight the challenges associated with utilizing it. I will then show that by obtaining a permutation invariant representation, we can not only measure aspects of sports that couldn't be done before, but also interact with and simulate plays akin to a video game via our "visual search" and "ghosting" technology. Finally, I will show how we can use both tracking and event data to create a multimodal foundation model, which enables us to generate player tracking data at scale and achieve our goal of "digitizing every game of professional sport." Throughout the talk, I will utilize examples from top-tier basketball, soccer, and tennis.

Biography: Patrick Lucey is currently the Chief Scientist at sports data giant Stats Perform, leading the AI team with the goal of maximizing the value of the company's extensive sports data. He has studied and worked in the fields of machine learning and computer vision for the past 20 years, holding research positions at Disney Research and the Robotics Institute at Carnegie Mellon University, as well as spending time at IBM's T.J. Watson Research Center while pursuing his Ph.D. Patrick originally hails from Australia, where he received his BEng(EE) from the University of Southern Queensland and his doctorate from Queensland University of Technology, which focused on multimodal speech modeling. He has authored more than 100 peer-reviewed papers and has been a co-author on papers in the MIT Sloan Sports Analytics Conference Best Research Paper Track for 11 of the last 13 years, winning best paper in 2016 and runner-up in 2017 and 2018. Additionally, he has won best paper awards at INTERSPEECH and WACV international conferences. His main research interests are in artificial intelligence and interactive machine learning in sporting domains, as well as AI education. He has recently piloted a course on "AI in Sport," which aims to give students intuition behind AI methods using the interactive and visual nature of sports data.

Website: www.patricklucey.com

Resource-Aware Machine Learning—A User-Oriented Approach

Katharina Morik

TU Dortmund University

Abstract. Machine Learning (ML) has become integrated into several processes, ranging from medicine, manufacturing, logistics, smart cities, sales, recommendations and advertisements to entertainment and many more business and private processes. The applications together consume a considerable amount of energy and emit CO₂. ML research investigates how to make models smaller and faster through pruning and quantization. Also the use of more energy-efficient hardware is an encouraging field. Research on ML under resource constraints is an active field proposing novel algorithms and scenarios. The aim is that for each application a variety of implementations is offered from which customers and the different types of users may choose the most thrifty one. This, in turn, would push tech providers to focus on the production of economical systems. However, if the customers, users, stakeholders do not know which of the models offers the best tradeoff between performance and energy-efficiency, they cannot select the most frugal one. Hence, testing implementations of learning and inference needs to be developed. They should be easy to use, produce visualizations that are mass-tailored for specific user groups. Automatized testing is difficult due to the diversity of models, computing architectures, training and evaluation data, and the fast rate of changes. The talk will illustrate work on resource-aware ML and advocate to pay more attention to the role of users in the development of scenarios, models, and tests.

Biography: Katharina Morik received her doctorate from the University of Hamburg in 1981 and her habilitation from the TU Berlin in 1988. In 1991, she established the chair of Artificial Intelligence at the TU Dortmund. She retired in 2023. She is a pioneer of bringing machine learning and computing architectures together so that machine learning models may be executed or even trained on resource restricted devices. In 2011, she acquired the Collaborative Research Center CRC 876 “Providing Information by Resource-Constrained Data Analysis” consisting of 12 projects and a graduate school. After the longest possible funding period of 12 years, the CRC ended with the publication of 3 books on Resource-Constrained Machine Learning (De Gruyter). She has participated in numerous European research projects and has been the coordinator of one. She was a founding member and Program Chair of the conference series IEEE International Conference on Data Mining (ICDM) and is a member of the steering committee

of ECML PKDD. She is a co-founder of the Lamarr Institute for Machine Learning and Artificial Intelligence. Prof. Morik is a member of the Academy of Technical Sciences and of the North Rhine-Westphalian Academy of Sciences and Arts. She was made a Fellow of the German Society of Computer Science GI e.V. in 2019.

Contents – Part III

Research Track

Interpretable and Generalizable Spatiotemporal Predictive Learning with Disentangled Consistency	3
<i>Jingxuan Wei, Cheng Tan, Zhangyang Gao, Linzhuang Sun, Bihui Yu, Ruifeng Guo, and Stan Li</i>	
Reinventing Node-centric Traffic Forecasting for Improved Accuracy and Efficiency	21
<i>Xu Liu, Yuxuan Liang, Chao Huang, Hengchang Hu, Yushi Cao, Bryan Hooi, and Roger Zimmermann</i>	
Direct-Effect Risk Minimization for Domain Generalization	39
<i>Yuhui Li, Zejia Wu, Chao Zhang, and Hongyang Zhang</i>	
Federated Frank-Wolfe Algorithm	58
<i>Ali Dadras, Sourasekhar Banerjee, Karthik Prakhya, and Alp Yurtsever</i>	
Bootstrap Latents of Nodes and Neighbors for Graph Self-supervised Learning	76
<i>Yunhui Liu, Huaisong Zhang, Tieke He, Tao Zheng, and Jianhua Zhao</i>	
Deep Sketched Output Kernel Regression for Structured Prediction	93
<i>Tamim El Ahmad, Junjie Yang, Pierre Laforgue, and Florence d’Alché-Buc</i>	
Hyperbolic Delaunay Geometric Alignment	111
<i>Aniss Aiman Medbouhi, Giovanni Luca Marchetti, Vladislav Polianskii, Alexander Kravberg, Petra Poklukar, Anastasia Varava, and Danica Kragic</i>	
ApmNet: Toward Generalizable Visual Continuous Control with Pre-trained Image Models	127
<i>Haitao Wang and Hejun Wu</i>	
AdaHAT: Adaptive Hard Attention to the Task in Task-Incremental Learning	143
<i>Pengxiang Wang, Hongbo Bo, Jun Hong, Weiru Liu, and Kedian Mu</i>	

Probabilistic Circuits with Constraints via Convex Optimization	161
<i>Soroush Ghandi, Benjamin Quost, and Cassio de Campos</i>	
FedAR: Addressing Client Unavailability in Federated Learning with Local Update Approximation and Rectification	178
<i>Chutian Jiang, Hansong Zhou, Xiaonan Zhang, and Shayok Chakraborty</i>	
Selecting from Multiple Strategies Improves the Foreseeable Reasoning of Tool-Augmented Large Language Models	197
<i>Yongchao Wu and Aron Henriksson</i>	
Estimating Direct and Indirect Causal Effects of Spatiotemporal Interventions in Presence of Spatial Interference	213
<i>Sahara Ali, Omar Faruque, and Jianwu Wang</i>	
Continuous Geometry-Aware Graph Diffusion via Hyperbolic Neural PDE	231
<i>Jiaxu Liu, Xinping Yi, Sihao Wu, Xiangyu Yin, Tianle Zhang, Xiaowei Huang, and Shi Jin</i>	
SpanGNN: Towards Memory-Efficient Graph Neural Networks via Spanning Subgraph Training	250
<i>Xizhi Gu, Hongzheng Li, Shihong Gao, Xinyan Zhang, Lei Chen, and Yingxia Shao</i>	
AKGNet: Attribute Knowledge Guided Unsupervised Lung-Infected Area Segmentation	267
<i>Qing En and Yuhong Guo</i>	
Diffusion Model in Normal Gathering Latent Space for Time Series Anomaly Detection	284
<i>Jiashu Han, Shanshan Feng, Min Zhou, Xinyu Zhang, Yew Soon Ong, and Xutao Li</i>	
Permutation Dependent Feature Mixing for Multivariate Time Series Forecasting	301
<i>Rikuto Yamazono and Hirotake Hachiya</i>	
Prior Bilinear-Based Models for Knowledge Graph Completion	317
<i>Jiayi Li, Ruilin Luo, Jiaqi Sun, Jing Xiao, and Yujiu Yang</i>	
Thinking Like an Author: A Zero-Shot Learning Approach to Keyphrase Generation with Large Language Model	335
<i>Siyu Wang, Shengran Dai, and Jianhui Jiang</i>	

Molecular Graph Representation Learning via Structural Similarity Information	351
<i>Chengyu Yao, Hong Huang, Hang Gao, Fengge Wu, Haiming Chen, and Junsuo Zhao</i>	
Efficient Privacy-Preserving Truth Discovery and Copy Detection in Crowdsourcing	368
<i>Xiu Susie Fang, Xinyang Du, Hao Chen, Ziqi Wei, Yong Zhan, and Guohao Sun</i>	
FCFL: A Fairness Compensation-Based Federated Learning Scheme with Accumulated Queues	386
<i>Lingfu Wang, Zuobin Xiong, Guangchun Luo, Wei Li, and Aiguo Chen</i>	
MMDL-Based Data Augmentation with Domain Knowledge for Time Series Classification	403
<i>Xiaosheng Li, Yifan Wu, Wei Jiang, Ying Li, and Jianguo Li</i>	
FALCUN: A Simple and Efficient Deep Active Learning Strategy	421
<i>Sandra Gilhuber, Anna Beer, Yunpu Ma, and Thomas Seidl</i>	
Semi-supervised Heterogeneous Domain Adaptation via Disentanglement and Pseudo-labelling	440
<i>Cassio F. Dantas, Raffaele Gaetano, and Dino Ienco</i>	
Author Index	457

Research Track



Interpretable and Generalizable Spatiotemporal Predictive Learning with Disentangled Consistency

Jingxuan Wei^{1,2}, Cheng Tan^{3,4}, Zhangyang Gao^{3,4}, Linzhuang Sun^{1,2},
Bihui Yu^{1,2}(✉), Ruifeng Guo^{1,2}(✉), and Stan Li^{3,4}(✉)

¹ Shenyang Institute of Computing Technology, Chinese Academy of Sciences,
Beijing, China

`weijingxuan20@mails.ucas.edu.cn`, `sunlinzhuang21@mails.ucas.ac.cn`

² University of Chinese Academy of Sciences, Beijing, China
{`yubihui`,`grf`}@`sict.ac.cn`

³ Zhejiang University, Hangzhou, China

⁴ AI Lab, Research Center for Industries of the Future, Westlake University,
Hangzhou, China

{`tancheng`,`gaozhangyang`,`stan.zq.li`}@`westlake.edu.cn`

Abstract. In recent years, significant strides have been made in the field of spatiotemporal predictive learning, a discipline that focuses on accurately forecasting future sequences based on previously observed frames. Despite the impressive capabilities of current leading-edge models, which leverage specialized network architectures to optimize learning in both spatial and temporal domains, these models often fall short in their ability to accurately interpret underlying spatiotemporal dependencies and extend their learnings to unseen data. In this study, we attempt to address these shortcomings by disentangling the context and motion within sequential spatiotemporal data, and then systematically analyzing the relationship between the original and disentangled data. We introduce context-motion disentanglement modules that utilize temporal entropy to segregate the context and motion, and then apply regularization to the disentangled motion to ensure its consistency with the predicted frames produced by conventional spatiotemporal predictive learning. Our proposed methodology can be trained in an end-to-end fashion and serves to improve not just the predictive performance but also the interpretability and generalizability of the model. The efficacy of our proposed method is illustrated through comprehensive quantitative and qualitative assessments.

Keywords: Spatiotemporal predictive learning · self-supervised learning · convolutional neural networks · computer vision applications

1 Introduction

Deep learning has demonstrated considerable success in numerous domains [4, 24–26, 43, 44, 54]. A critical subfield of deep learning is spatiotemporal pre-

J. Wei and C. Tan—Equal Contribution.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024
A. Bifet et al. (Eds.): ECML PKDD 2024, LNAI 14943, pp. 3–20, 2024.
https://doi.org/10.1007/978-3-031-70352-2_1

dictive learning, a self-supervised learning discipline that focuses on forecasting future frames based on past observations. Previous studies have made commendable contributions by developing specialized modules to capture spatial correlations and temporal dependencies based on LSTM [16] and GRU [7]. Though these seminal works have achieved superior results, they face challenges in effectively interpreting the underlying spatiotemporal dependencies and generalizing the insights from disentangled information.

Past research [17, 47] have strived to separate static contexts from dynamic motions, aiming to extract meaningful representations from sequential video data. The primary premise of these studies is that once the model successfully disentangles the context from the motion, it would have effectively learned the spatial correlations and temporal dependencies. Thus, they either build dual networks to separately capture motions and semantic contexts [11] or impose constraints in the latent spaces [17]. However, mediately predicting future frames by fusing the representations of contexts and motions usually performs worse than those directly optimizing for the future frames [12, 52]. The reason to blame may be brute-force disentangling that destroys nonlinear spatiotemporal relations. Moreover, these methods employ disentangling in the latent space, which is difficult to present the actual disentangled contexts and motions explicitly. Their inherent complex architectures even hinder their interpretable ability.

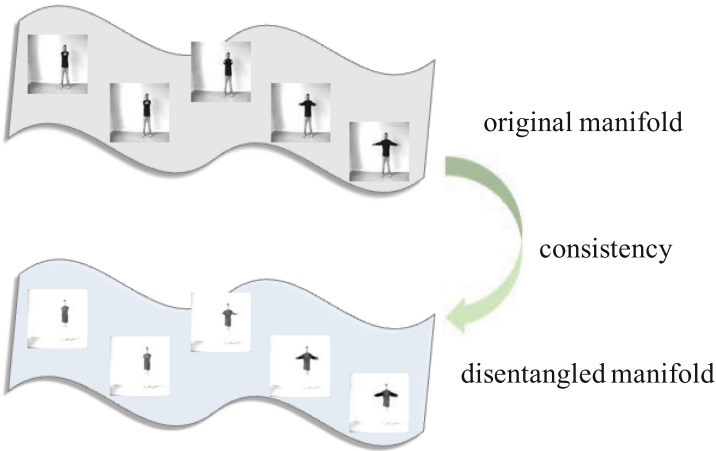


Fig. 1. The consistency between the manifolds of original sequential video data and disentangled representations.

Our study aspires to bridge this gap by fusing standard spatiotemporal learning with disentangled context-motion, creating a framework for interpretable and generalizable spatiotemporal learning. We introduce context-motion disentanglement modules leveraging temporal entropy to separate the context and motion. Based on the principles of manifold learning [27], we hypothesize that the original data and disentangled representations exist on different manifolds with analogous topological spaces. The assumption primarily comes from the basis of the static context and the dynamic motions. While the context is static, we can

regard the context as a constant that is added to the motion. We obtain the disentangled manifold from the original manifold minus a constant so that the manifolds are homeomorphic. As shown in Fig. 1, the disentangled representation containing varying motions should have similar spatiotemporal dependencies to the original data. By imposing a consistency constraint between manifolds, we exploit the disentangled representations in enhancing interpretable and generalizable spatiotemporal predictive learning.

2 Related Works

2.1 Spatiotemporal Predictive Learning

Recent advances in recurrent models [13,30] have provided valuable insights into spatiotemporal predictive learning [1,8,35,41,42,58]. Inspired by recurrent neural networks, VideoModeling [31] adopts language modeling and quantizes the image patches into an extensive dictionary for recurrent units. CompositeLSTM [39] further introduces the LSTM architecture and improves its performance. ConvLSTM [37] leverages convolutional neural networks to model the LSTM architecture. PredNet [29] continually predicts future video frames using deep recurrent convolutional neural networks with bottom-up and top-down connections. PredRNN [50] proposes a Spatiotemporal LSTM unit that simultaneously extracts and memorizes spatial and temporal representations. Its subsequential work PredRNN++ [52] further proposes a gradient highway unit and Casual LSTM adaptively capture temporal dependencies. E3D-LSTM [51] designs eidetic memory transition in recurrent convolutional units. Conv-TT-LSTM [40] employs a higher-order ConvLSTM to predict by combining convolutional features across time. MotionRNN [55] focuses on motion trends and transient variations. LMC-Memory [23] introduces a long-term motion context memory using memory alignment learning. PredCNN [57] and TrajectoryCNN [28] implement convolutional neural networks as the temporal module. SimVP [12] is a seminal work that applies Inception modules with a UNet architecture to learn the temporal evolution. TAU [45] proposes an attention-based temporal module that performs both intra-frame and inter-frame attention for spatiotemporal predictive learning.

2.2 Disentangled Representation

Decomposing the raw sequential video data into disentangled representations is an essential topic in the computer vision. DRNet [11] and MCnet [49] are early works on learning disentangled image representations from video. Their proposed methods aim to learn contexts and motions by two individual networks separately and then fuse the learned static and dynamic features in the latent space. MoCoGAN [47] shares a similar idea but generates video frames conditioned on random vectors. DDPAE [17] performs the video decomposition with multiple objects in addition to disentanglement and designs a specialized framework for

Moving MNIST. MGP-VAE [3] also models the latent space for disentangled representations in video sequences. While the previous studies focus on learning in the latent space, our method aims to explicitly present interpretable and generalizable spatiotemporal predictive learning by a disentangled consistency constraint.

3 Methods

3.1 Preliminaries

We formally define the spatiotemporal predictive learning problem as follows. Given a video sequence $\mathbf{X}^{t,T} = \{\mathbf{x}^i\}_{t-T+1}^t$ at time t with the past T frames, we aim to predict the subsequent T' frames $\mathbf{Y}^{t+1,T'} = \{\mathbf{x}^i\}_{t+1}^{t+T'}$ from time $t + 1$, where $\mathbf{x}^i \in \mathbb{R}^{C \times H \times W}$ is usually an image with channels C , height H , and width W . In practice, we represent the video sequences as tensors, i.e., $\mathbf{X}^{t,T} \in \mathbb{R}^{T \times C \times H \times W}$ and $\mathbf{Y}^{t+1,T'} \in \mathbb{R}^{T' \times C \times H \times W}$.

The model with learnable parameters Θ learns a mapping $\mathcal{F}_\Theta : \mathbf{X}^{t,T} \mapsto \mathbf{Y}^{t+1,T'}$ by exploring both spatial and temporal dependencies. In our case, the mapping \mathcal{F}_Θ is a neural network model trained to minimize the difference between the predicted future frames and the ground-truth future frames. The optimal parameters Θ^* are:

$$\Theta^* = \arg \min_{\Theta} \mathcal{L}(\mathcal{F}_\Theta(\mathbf{X}^{t,T}), \mathbf{Y}^{t+1,T'}), \quad (1)$$

where \mathcal{L} is a loss function that evaluates such differences. By optimizing such a loss function, the model is able to learn the inherent spatiotemporal dependencies and thus accurately predicts future frames.

We recognize context and motion as semantically static and dynamic objects, respectively. The data \mathbf{X} are assumed to consist of the context $\mathbf{c} \in \mathbb{R}^{C \times H \times W}$ and the motion $\mathbf{O} = \{\mathbf{o}_i | \mathbf{o}_i \in \mathbb{R}^{C \times H \times W}\}$. The context and the motion are controlled by the state of movement $\mathbf{S} = \{\mathbf{s}_i | \mathbf{s}_i \in \mathbb{R}^{1 \times H \times W}\}$. For each frame \mathbf{x}^i in \mathcal{X} , the formal representation is:

$$\mathbf{x}^i = \mathbf{o}^i \odot \mathbf{s} + \mathbf{c} \odot (1 - \mathbf{s}), \forall \mathbf{x}_i \in \mathbf{X}, \mathbf{o}_i \in \mathbf{O}, \mathbf{s}_i \in \mathbf{S}, \quad (2)$$

where \odot is the Hadamard product.

In this study, we decouple the context and motion of each frame through explicit context-motion disentanglement mechanism and implicit disentangled consistency for presenting an interpretable and generalizable spatiotemporal predictive learning.

3.2 Context-Motion Disentanglement

We first decompose the desired mapping \mathcal{F} into two submappings:

$$\mathcal{F} \triangleq \mathcal{H} \circ \mathcal{G}, \quad (3)$$

where $\mathcal{H} : \mathbf{X}^{t,T} \mapsto \mathbf{H}^t$, $\mathcal{G} : \mathbf{H}^t \mapsto \mathbf{Y}^{t+1,T'}$, and $\mathbf{H}^t \in \mathbb{R}^{T' \times C \times H \times W}$ is the latent representation at time t that contains information from previous T and following T' . \mathcal{H} can be an arbitrary mapping that aims to explore the underlying spatiotemporal dependencies of the input frames $\mathcal{X}^{t,T}$ and project it into an informative latent space. In contrast to the mapping \mathcal{H} , the latter mapping \mathcal{G} reconstructs the visual imaging and predicts the future frames $\mathbf{Y}^{t+1,T'}$ based on the representation \mathbf{H}^t in the latent space.

For standard spatiotemporal predictive learning methods, \mathcal{G} can be an arbitrary mapping, as well as \mathcal{H} . In this study, we explicitly define the mapping \mathcal{G} for specific context-motion disentanglement:

$$\mathcal{G} \triangleq \mathbf{O} \odot \mathbf{S} + \mathbf{c} \odot (1 - \mathbf{S}), \quad (4)$$

where we practically represent the sets as tensors, i.e., $\mathbf{O} \in \mathbb{R}^{T' \times C \times H \times W}$ and $\mathbf{S} \in \mathbb{R}^{T' \times 1 \times H \times W}$. The context tensor $\mathbf{c} \in \mathbb{R}^{1 \times C \times H \times W}$ is a tensor variation compared to the definition in Sect. 3.1. The motion tensor \mathbf{O} , context tensor \mathbf{c} , and state tensor \mathbf{S} are obtained by mappings $\mathcal{O} : \mathbf{H} \mapsto \mathbf{O}$, $\mathcal{C} : \mathbf{H} \mapsto \mathbf{c}$, and $\mathcal{S} : \mathbf{H} \mapsto \mathbf{S}$, respectively.

Though the \mathcal{G} is specified to decouple the context and motion, directly optimizing the mean square error (MSE) loss alone, as standard spatiotemporal predictive learning does, is unreliable. The MSE loss cannot guide the neural network automatically separate the context and motion. We argue that the key to context-motion disentanglement is to determine the context accurately. Thus, we impose the inductive bias that the pixels in context are likely to be static across the varying time.

To evaluate the inherent uncertainty of video frames, we intuitively borrow the concept of entropy from information theory. Here we refer to $\Delta \mathbf{x}^i$ as a pixel in a specific position of frame \mathbf{x}^i and $\Delta \mathbf{X}$ as the pixel in the same position of all frames in \mathbf{X} . We define the probability of whether this pixel is changing Δw^i as:

$$\Delta w^i = \frac{\Delta \mathbf{x}^i - \Delta \mathbf{x}^0}{\max \Delta \mathbf{x} - \min \Delta \mathbf{x}}, \quad (5)$$

which is normalized in $[0, 1]$ according to the changing scope compared to the initial frame. The uncertainty of whether the pixel belongs to the context is evaluated by its average entropy of w :

$$E(\Delta w) = -\frac{1}{T} \sum_{i=t-T+1}^t p(\Delta w_i) \log p(\Delta w_i), \quad (6)$$

then we obtain a mask $\mathbf{M} \in \{0, 1\}^{1 \times C \times H \times W}$ that should be able to filter *reliable context* by a threshold \bar{w} . For each pixel, if the corresponding E is lower than \bar{w} , we recognize it as the static context, i.e., \mathbf{M} has a value of 1 for this pixel and vice versa.

With the inductive bias of reliable context given by \mathbf{M} , we design the disentanglement loss as:

$$\mathcal{L}_d(\mathbf{X}) = \frac{1}{T'} \sum_{t+1}^{t+T'} \|(c - \mathbf{x}^i) \odot \mathbf{M}\|. \quad (7)$$

This loss guarantees that at least the reliable static context is learned. Taking advantage of the flatness of convolutional networks, the model can disentangle actual context based on the above reliable context.

3.3 Disentangled Consistency

Despite the disentanglement loss \mathcal{L}_d enforcing explicit model discrimination between context and motion, it remains reliant on the inductive bias \mathbf{M} . We contend that the context is intrinsically static in its semantics and that the disentangled frames should exhibit consistency with the actual frames. Consider a manifold \mathcal{M}_x representative of the original data, with the correlated disentangled representations inhabiting another manifold, denoted as \mathcal{M}_o . s **Definition.** We define two topological spaces, denoted as \mathcal{M}_x and \mathcal{M}_o , to be homeomorphic if and only if there exists a bijective mapping function $f : \mathcal{M}_x \mapsto \mathcal{M}_o$ with the following properties: (i) The function f is continuous. (ii) The inverse of f , denoted as f^{-1} , exists and is also continuous.

This definition [10, 15, 32, 38] reveals the relationship between the manifold \mathcal{M}_x and \mathcal{M}_o . According to Eq. 4, we can observe that once the mapping is bijective the disentangled manifold is homeomorphic to the original manifold. In other words, the original manifold \mathcal{M}_x and the disentangled manifold \mathcal{M}_o are topological equivalences.

Theorem. Given a homeomorphism $f(\mathbf{X})$, a mapping that is both smooth and possesses a unique inverse, the mutual information is invariant under such transformation, such that $I(\mathbf{X}, \mathbf{O}) = I(f(\mathbf{X}), \mathbf{O})$.

Proof. First, remember that the entropy of a discrete random variable \mathbf{X} is defined as $H(\mathbf{X}) = -\sum_{x \in \mathbf{X}} p(x) \log p(x)$, where $p(x)$ is the probability mass function of X . For continuous random variables, the entropy is similarly defined but with an integral instead of a sum, and the probability density function instead of the probability mass function.

Now consider a homeomorphism f , and suppose $p_{\mathbf{X}}(x)$ is the probability density function of \mathbf{X} and $p_{\mathbf{O}}(o)$ is the probability density function of \mathbf{O} , which equals to $p_{\mathbf{X}}(f^{-1}(o))$ due to the invariance of probability under the transformation.

The differential entropy $H(\mathbf{O})$ of \mathbf{O} is then:

$$\begin{aligned} H(\mathbf{O}) &= - \int p_{\mathbf{O}}(o) \log p_{\mathbf{O}}(o) do \\ &= - \int p_{\mathbf{X}}(f^{-1}(o)) \log p_{\mathbf{X}}(f^{-1}(o)) do, \end{aligned} \quad (8)$$

By changing the variable from o to $x = f^{-1}(o)$, and remembering that homeomorphisms preserve the measure, the differential entropy $H(\mathbf{O})$ of \mathbf{O} transforms to:

$$H(\mathbf{O}) = - \int p_{\mathbf{X}}(x) \log p_{\mathbf{X}}(x) dx = H(\mathbf{X}). \quad (9)$$

So, the entropy of \mathbf{X} and \mathbf{O} are equal. Since the entropy is invariant under homeomorphisms, the conditional entropy is also invariant. Therefore, mutual information, which is a combination of entropy and conditional entropy, is also invariant under homeomorphisms.

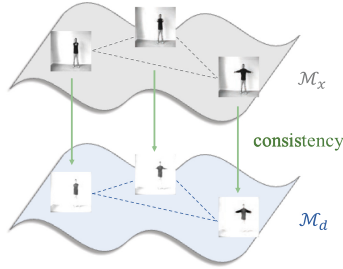


Fig. 2. Characterize the relationship between \mathcal{M}_x and \mathcal{M}_d from the geometric viewpoint and regularize the geometric property to be consistent.

This theorem [9, 21, 46] reveals the connections between \mathcal{M}_x and \mathcal{M}_o . If the mapping f is bijective, their mutual information is:

$$I(\mathbf{X}, \mathbf{O}) = H(\mathbf{X}) + H(\mathbf{O}) - H(\mathbf{X}, \mathbf{O}) \quad (10)$$

is maximized. Based on the above observation, we characterize the relationship between the manifolds \mathcal{M}_x and \mathcal{M}_o from the geometric viewpoint. As shown in Fig. 2, we consider the pairwise distance as the key geometric property and regularize the manifold \mathcal{M}_o to have a similar geometric structure as \mathcal{M}_x . For those limited data points, the mapping f is approaching bijective through preserving this geometric property.

Then, we define the pairwise distances in the two manifolds as follows:

$$d_x = \frac{\|\mathbf{x}^i - \mathbf{x}^j\|}{\sqrt{D}}, \quad d_o = \frac{\|\mathbf{o}^i - \mathbf{o}^j\|}{\sqrt{D}}, \quad (11)$$

where $\|\cdot\|$ is Euclidean distance, $D = C \times H \times W$ is a scale factor for avoiding large magnitude [48], $i, j \in \{t+1, \dots, t+T'\}$, and $i \neq j$. To model the distance in a nonlinear manner and obtain expressive metrics, we project the distance into normal distributions:

$$\begin{aligned} p(d_x) &= \frac{C_x}{\sigma_x \sqrt{2\pi}} \exp\left(-\frac{d_x^2}{2\sigma_x^2}\right), \\ p(d_o) &= \frac{C_o}{\sigma_o \sqrt{2\pi}} \exp\left(-\frac{d_o^2}{2\sigma_o^2}\right), \end{aligned} \quad (12)$$

where C_x, C_o are constants that forces the $p(\cdot) \in [0, 1]$, and σ_x, σ_o are controllable hyperparameters. For the convenience of optimization, we empirically assumes $p(d_g), p(d_o) \sim N(0, \frac{1}{2})$ in the experiments.

The disentangled consistency is formulated as:

$$\begin{aligned} \mathcal{L}_c(\mathbf{X}, \mathbf{O}) = & -p(d_x) \log(p(d_o)) \\ & - (1 - p(d_x)) \log(1 - p(d_o)), \end{aligned} \quad (13)$$

in which the binary cross entropy between $p(d_x)$ and $p(d_o)$ is expected to be minimized.

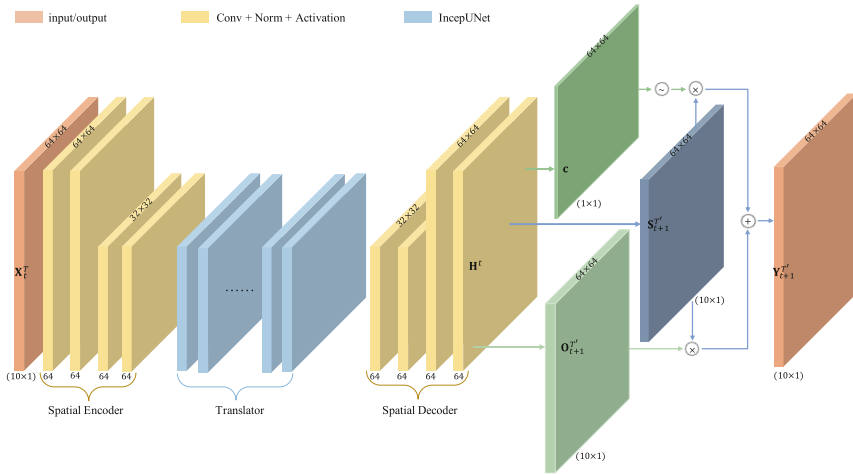


Fig. 3. The model architecture of our proposed method with the input from Moving MNIST. We employ a simple encoder-decoder model as the base architecture. The decoded representation H^t is used to obtain the context c , motion $O_{t+1}^{T'}$ and state $S_{t+1}^{T'}$.

3.4 Practical Implementation

We implement our proposed method by modifying the network of the current state-of-the-art method SimVP [12]. SimVP is a solid baseline in spatiotemporal predictive learning. As shown in Fig. 3, a spatial encoder and a spatial decoder are simple convolutional networks with downsampling and upsampling operations, while a translator network is in the middle for learning the spatiotemporal correlations. In SimVP, the translator network consists of blocks of Inception-UNet (IncepUNet). We remove the last layer of SimVP and employ the output of the penultimate layer as H^t . The mappings $\mathcal{O}, \mathcal{C}, \mathcal{S}$ are implemented by one-layer convolutional networks that project \mathcal{H} to $O_{t+1}^{T'}$, $S_{t+1}^{T'}$, and c , respectively.

The overall loss function is a linear combination of MSE loss, disentanglement loss \mathcal{L}_d , and disentangled consistency loss \mathcal{L}_c :

$$\begin{aligned} \mathcal{L} &= \text{MSE}(\mathcal{F}_\Theta(\mathbf{X}^{t,T}), \mathbf{Y}^{t+1,T'}) + \alpha\mathcal{L}_d + \beta\mathcal{L}_c, \\ &= \|\mathcal{F}_\Theta(\mathbf{X}^{t,T}) - \mathbf{Y}^{t+1,T'}\|^2 + \alpha\mathcal{L}_d + \beta\mathcal{L}_c, \end{aligned} \quad (14)$$

where α, β are weights of loss \mathcal{L}_d and \mathcal{L}_c . We empirically set the values as $\alpha = 1.0, \beta = 0.1$ in default.

It is worth noting that though our proposed method is implemented based on the baseline SimVP, it is also suitable for other spatiotemporal predictive learning baselines.

4 Experiments

We evaluate our method by both quantitative and qualitative validation. We present the interpretability across different experimental settings as follows: (1) standard spatiotemporal predictive learning, (2) generalizing to unknown scenes.

4.1 Standard Spatiotemporal Predictive Learning

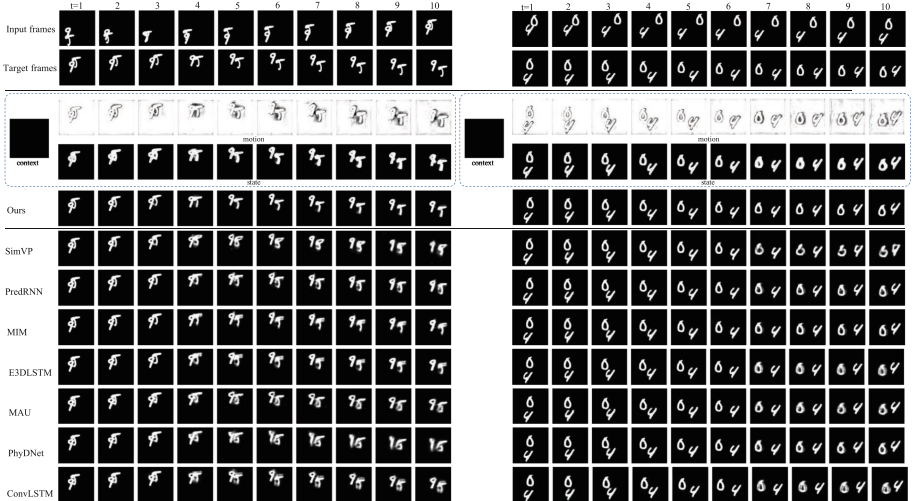


Fig. 4. Qualitative results on the Moving MNIST dataset. We show the disentangled context, motion, and state in the dotted boxes.

Moving MNIST. The Moving MNIST dataset [39], a widely recognized benchmark in standard spatiotemporal predictive learning, is a synthetic compilation. It comprises two individual digits meandering within a 64×64 grid, reacting to boundaries with a bounce-back motion. The task involves predicting the

subsequent 10 frames, given a historical sequence of 10 frames. Our proposed methodology addresses this by explicitly disentangling the complex spatiotemporal dependencies and capitalizing on the ensuing disentangled consistency for improved performance. It is anticipated that our model will demonstrate a high level of proficiency in predicting future frames with precision.

Our experimental setup parallels the one detailed in SimVP [12]. We measure our approach’s performance against formidable benchmarks, including ConvLSTM [37], PredRNN [50], E3D-LSTM [51], MotionGRU [55], CrevNet [59], PhyDNet [14], SimVP [12], and others. We also compare our results with advanced techniques such as PhyDNet [14] and DDPAE [17], which engage in latent space disentanglement. The efficacy of our method is evidenced through quantitative metrics-frame-wise Mean Squared Error (MSE), Mean Absolute Error (MAE), and Structural Similarity Index Measure (SSIM)-and showcased in Table 1. To supplement our quantitative results, we offer a visual representation of our qualitative findings in Fig. 4. It becomes clear that our approach surpasses other state-of-the-art methods in performance, attributing its success to the robust modeling of context and motion. This capability confers our model with a competitive advantage, enabling it to outperform its counterparts.

Table 1. Quantitative results of different methods on the Moving MNIST dataset (10 → 10 frames).

Method	Moving MNIST (2 digits)		
	MSE↓	MAE↓	SSIM↑
ConvLSTM [37]	103.3	182.9	0.707
PredRNN [50]	56.8	126.1	0.867
PredRNN++ [52]	46.5	106.8	0.898
MIM [53]	44.2	101.1	0.910
LMC [23]	41.5	–	0.924
E3D-LSTM [51]	41.3	87.2	0.910
Conv-TT-LSTM [40]	53.0	–	0.915
DDPAE [17]	38.9	90.7	0.922
CrevNet [59]	38.5	–	0.928
MotionGRU [55]	34.3	–	0.928
CMS-LSTM [5]	33.6	73.1	0.931
MAU [6]	27.6	–	0.937
PhyDNet [14]	24.4	70.3	0.947
SimVP [12]	23.8	68.9	0.948
Ours	22.9	68.6	0.949

KTH. The KTH dataset [36], a compendium of human poses, encapsulates 25 individuals performing six distinct actions: walking, jogging, running, boxing, hand waving, and hand clapping. The intricacies of human motion stem from

the stochastic nature of various individuals performing different actions. The KTH dataset, however, is noted for its relatively consistent motion patterns. By studying historical frames, our model—built on the principles of interpretable and generalizable spatiotemporal predictive learning—is engineered to comprehend the dynamics of human motion and, subsequently, to anticipate long-term changes in future poses. Additionally, the prediction of extended sequences accurately is a nuanced problem in conventional spatiotemporal predictive learning. Our method strives to cultivate an interpretable and robust model, harnessing the learned spatiotemporal dependencies to predict long sequences with precision.

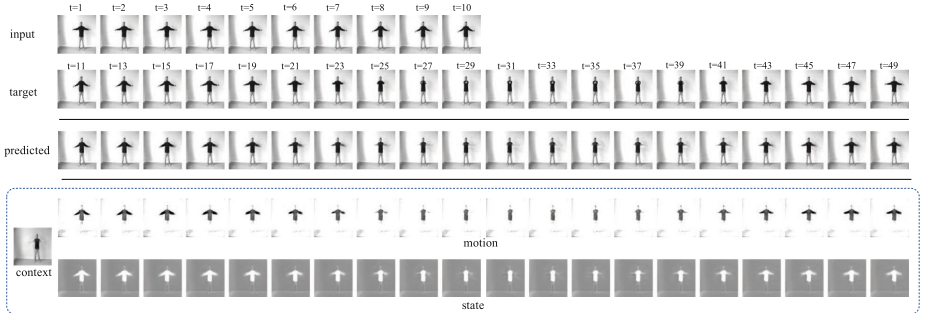


Fig. 5. Qualitative results on the KTH dataset. The example is predicting the next 40 frames based on the given historical 10 frames. The context c , motion O , and state S are shown in the dotted box.

Our experimental framework mirrors the one employed in SimVP [12], with the model being trained for 100 epochs. The evaluation of its performance is carried out using the SSIM and PSNR metrics [34, 51]. Empirically, SSIM tends to focus more on disparities in visual sharpness, while PSNR leans towards pixel-level accuracy. By taking both these metrics into account, we ensure a comprehensive evaluation of the models. We compare the performance under two distinct settings: predicting the next 20 or 40 frames based on the historical ten frames. As depicted in Table 2, our method outperforms state-of-the-art methods on the KTH dataset in both the $10 \rightarrow 20$ and $10 \rightarrow 40$ scenarios. Despite the notable accomplishments of previous baselines, our method still demonstrates superior performance, thereby underscoring the efficacy of delving into context-motion disentanglement and implementing a disentangled consistency strategy.

We visualize an example of the predicted and disentangled results in Fig. 5. It can be seen that the model captures the static part that consists of a scene and a person with blurry arms as the context. The motion captures the details of the arms when it is swung. The result is controlled by the state that determines the proportion of dynamic and static parts. The motion ignores details of the scene and the static legs of the person but clearly delineates the swinging arms.

Table 2. Quantitative results of different methods on the KTH dataset (10 \rightarrow 20 frames and 10 \rightarrow 40 frames).

Method	KTH (10 \rightarrow 20)		KTH (10 \rightarrow 40)	
	SSIM \uparrow	PSNR \uparrow	SSIM \uparrow	PSNR \uparrow
MCnet [49]	0.804	25.95	0.73	23.89
ConvLSTM [37]	0.712	23.58	0.639	22.85
DFN [18]	0.794	27.26	0.652	23.01
fRNN [33]	0.771	26.12	0.678	23.77
SV2Pv [2]	0.838	27.79	0.789	26.12
PredRNN [50]	0.839	27.55	0.703	24.16
VarNet [19]	0.843	28.48	0.739	25.37
SAVP-VAE [22]	0.852	27.77	0.811	26.18
PredRNN++ [52]	0.865	28.47	0.741	25.21
E3d-LSTM [51]	0.879	29.31	0.810	27.24
STMFA Net [20]	0.893	29.85	0.851	27.56
SimVP [12]	0.905	33.72	0.886	32.93
Ours	0.909	33.97	0.890	33.20

4.2 Generalizing to Unknown Scenes

Unknown Object. Our method benefits from the robust modeling of spatiotemporal dependencies that exploits the relationship between context and motion in both explicit and implicit ways. To verify the robustness and generalization ability, we make the Moving Fashion MNIST dataset by replacing digits with objects of Fashion MNIST [56]. We use the models pre-trained on Moving MNIST to evaluate the performance on Moving Fashion MNIST.

Table 3. Quantitative results on the Moving Fashion MNIST dataset (10 \rightarrow 10 frames).

Method	Moving Fashion MNIST (2 objects)		
	MSE \downarrow	MAE \downarrow	SSIM \uparrow
ConvLSTM [37]	96.2	268.1	0.628
PredRNN [50]	90.6	225.6	0.749
PredRNN++ [52]	82.2	204.5	0.782
MIM [53]	80.6	204.7	0.778
E3D-LSTM [51]	78.3	196.9	0.791
PhyDNet [14]	86.7	207.7	0.774
MAU [6]	82.8	201.0	0.781
SimVP [12]	79.1	196.9	0.789
Ours	72.3	178.5	0.810

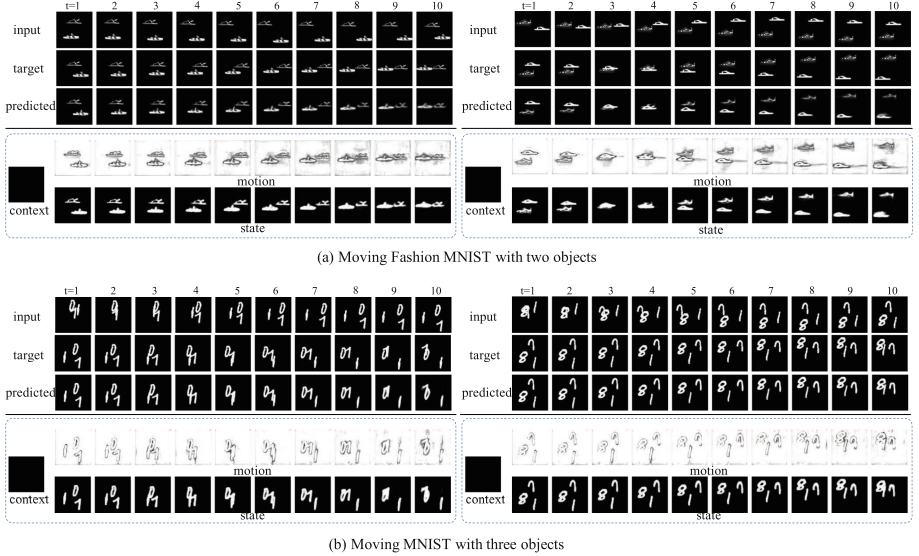


Fig. 6. Qualitative results on unknown scenes. With the model pretrained on the vanilla Moving MNIST dataset, we show the visualization of evaluating Moving Fashion MNIST and three-object Moving MNIST. The disentangled context, motion and state are shown in dotted boxes below the predicted frames.

Table 4. Quantitative results on the Moving MNIST dataset with three digits (10 \rightarrow 10 frames).

Method	Moving MNIST (3 digits)		
	MSE \downarrow	MAE \downarrow	SSIM \uparrow
ConvLSTM [37]	115.4	262.0	0.730
PredRNN [50]	101.8	235.7	0.766
PredRNN++ [52]	84.4	203.3	0.808
MIM [53]	83.1	198.0	0.816
E3D-LSTM [51]	75.6	186.0	0.829
PhyDNet [14]	85.6	179.2	0.833
MAU [6]	71.2	159.0	0.859
SimVP [12]	75.2	165.7	0.849
Ours	68.9	144.1	0.870

We show the quantitative results in Table 3. It can be seen that our model achieves significantly better performance than baseline models. Specifically, our model improves the state-of-the-art SimVP model by about 8.60% in the MSE metric and about 9.34% in the MAE metric, indicating the strong generalization ability of our model. The qualitative results in Fig. 6(a) show that our model captures context and motion well despite the objects have changed.

Unknown Setting. We extend our exploration to test the model’s generalizability in a more complex setting that incorporates three moving digits instead of the customary two. This approach aligns with the strategy delineated in Sect. 4.2, wherein the model, initially trained on the Moving MNIST dataset with two digits, is employed to gauge its performance on a Moving MNIST variant with three digits. To put it differently, we train the model using data containing two digits and evaluate its performance with data featuring three digits. The model is expected to identify the dynamic elements, as opposed to merely recalling the previously observed scenarios.

As represented in Table 4, our model consistently surpasses baseline models by a considerable margin across all metrics. Specifically, our model enhances the state-of-the-art SimVP model by approximately 9.14% in the MSE metric and around 13.03% in the MAE metric. We illustrate a predicted example in Fig. 6(b). Although the context-motion disentanglement mechanism distinctly recognizes the dynamic and static elements, the predicted frames closely resemble the ground-truth frames. These experimental results affirm that our model, by learning the context-motion, exhibits a formidable generalization capacity.

4.3 Ablation Study

A series of ablation studies have been conducted on both Moving MNIST and Moving Fashion MNIST datasets, with the MSE metrics reported in Table 5. Initially, we eliminate the disentangled consistency, a mechanism that implicitly disentangles the context and motion. As a result, we observe a significant deterioration in performance on the Moving Fashion MNIST dataset, underlining the pivotal role disentangled consistency plays in enhancing the model’s generalization capabilities. Subsequently, when we remove the context-motion disentanglement modules, the performance suffers an even more profound degradation.

Table 5. Ablation study of our proposed method. (MSE ↓)

Method	M-MNIST	M-FMNIST
Ours	22.9	72.3
w/o disentangled consistency	23.2	75.4
w/o disentanglement modules	23.8	79.1

5 Limitations

5.1 Reverse Problem

Our model is designed to forecast subsequent sequences given an input sequence $\{\mathbf{x}_i, \mathbf{x}_{i+1}, \dots, \mathbf{x}_{i+T}\}$, yielding a prediction of the form $\{\mathbf{x}_{i+T}, \mathbf{x}_{i+T+1}, \dots, \mathbf{x}_{i+T'}\}$.

An intriguing question that arises is whether the model could perform a "reverse prediction" if the input is rearranged as $\{\mathbf{x}_{i+T}, \mathbf{x}_{i+T-1}, \dots, \mathbf{x}_i\}$, essentially predicting a sequence of the form $\{\mathbf{x}_{i-1}, \mathbf{x}_{i-2}, \dots, \mathbf{x}_{i-T'}\}$. We refer to this scenario as the "reverse prediction problem". The potential of our model, which excels in interpretable and generalizable spatiotemporal predictive learning, to address this challenge presents a fascinating direction for future exploration. This innovative application could provide valuable insights into how prediction models can be utilized in more flexible and versatile ways.

5.2 Handling of Irregularly Sampled Data

The datasets utilized in this research were sampled at consistent time intervals. This approach may not be well-suited for handling irregularly sampled data, such as those with missing values. Our method may struggle to learn from such data and to make predictions for arbitrary timestamps in the future. A plausible solution to this challenge might involve appending timestamp information to the input or hidden feature vectors during the generation phase. Alternatively, neural ordinary differential equations could be employed to model time-continuous data.

5.3 Adaptability to Dynamic Views

The proposed context-motion disentanglement module is likely more compatible with static views, as it operates under the assumption that the background of the same video remains largely unchanged. The extension of our disentanglement strategy to more dynamic views represents an interesting area for future research.

6 Conclusion

In this work, we present an interpretable and generalizable spatiotemporal predictive learning method, which seeks to disentangle the context and the motion from sequential spatiotemporal data. Specifically, we design a context-motion disentanglement mechanism and a disentangled consistency strategy to perform both explicit and implicit context-motion disentanglement. Our experimental results demonstrate that our proposed model adeptly decouples static context from dynamic motion, and further learns the nuanced spatiotemporal dependencies, outshining models that merely rely on rote memorization. Crucially, our model demonstrates robust generalization to previously unseen data. We anticipate that the methodology we have put forth may provide fresh insights and potentially stimulate advancements in the sphere of artificial general intelligence.

Acknowledgements. This work is supported by the Shenyang Science and Technology Plan, grant number 23-407-3-29.

Ethical Statement

Our submission does not involve any ethical issues, including but not limited to privacy, security, etc.

References

1. Acharya, D., Huang, Z., Paudel, D.P., Van Gool, L.: Towards high resolution video generation with progressive growing of sliced wasserstein gans. arXiv preprint [arXiv:1810.02419](https://arxiv.org/abs/1810.02419) (2018)
2. Babaeizadeh, M., et al.: Stochastic variational video prediction. In: ICLR (2018)
3. Bhagat, S., Uppal, S., Yin, Z., Lim, N.: Disentangling multiple features in video sequences using Gaussian processes in variational autoencoders. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12368, pp. 102–117. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58592-1_7
4. Cao, H., et al.: A survey on generative diffusion models. IEEE Trans. Knowl. Data Eng. (2024)
5. Chai, Z., et al.: CMS-LSTM: context embedding and multi-scale spatiotemporal expression LSTM for predictive learning. In: ICME, pp. 01–06 (2022)
6. Chang, Z., et al.: Mau: a motion-aware unit for video prediction and beyond. Adv. NIPS **34** (2021)
7. Cho, K., et al.: On the properties of neural machine translation: encoder–decoder approaches. In: Proceedings of SSST-8, pp. 103–111 (2014)
8. Clark, A., Donahue, J., Simonyan, K.: Adversarial video generation on complex datasets. arXiv preprint [arXiv:1907.06571](https://arxiv.org/abs/1907.06571) (2019)
9. Cover, T.M., Thomas, J.A.: Elements of information theory second edition solutions to problems. Internet Access, 19–20 (2006)
10. Crossley, M.D.: Essential Topology. Springer, Heidelberg (2006). <https://doi.org/10.1007/1-84628-194-6>
11. Denton, E.L., et al.: Unsupervised learning of disentangled representations from video. Adv. NIPS **30** (2017)
12. Gao, Z., Tan, C., Li, S.Z.: Simvp: simpler yet better video prediction. In: Proceedings of CVPR, pp. 3170–3180 (2022)
13. Gehring, J., Auli, M., Grangier, D., Yarats, D., Dauphin, Y.N.: Convolutional sequence to sequence learning. In: International Conference on Machine Learning, pp. 1243–1252. PMLR (2017)
14. Guen, V.L., Thome, N.: Disentangling physical dynamics from unknown factors for unsupervised video prediction. In: Proceedings of CVPR, pp. 11474–11484 (2020)
15. Hawking, S.W., Ellis, G.F.R.: The Large Scale Structure of Space-Time, vol. 1. Cambridge University Press, Cambridge (1973)
16. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
17. Hsieh, J.T., et al.: Learning to decompose and disentangle representations for video prediction. Adv. NIPS **31** (2018)
18. Jia, X., et al.: Dynamic filter networks. Adv. NIPS **29** (2016)
19. Jin, B., et al.: Varnet: exploring variations for unsupervised video prediction. In: IROS, pp. 5801–5806 (2018)
20. Jin, B., et al.: Exploring spatial-temporal multi-frequency analysis for high-fidelity and temporal-consistency video prediction. In: Proceedings of CVPR, pp. 4554–4563 (2020)
21. Kraskov, A., Stögbauer, H., Grassberger, P.: Estimating mutual information. Phys. Rev. E **69**(6), 066138 (2004)
22. Lee, A.X., et al.: Stochastic adversarial video prediction. arXiv preprint [arXiv:1804.01523](https://arxiv.org/abs/1804.01523) (2018)

23. Lee, S., et al.: Video prediction recalling long-term motion context via memory alignment learning. In: Proceedings of CVPR, pp. 3054–3063 (2021)
24. Li, S., et al.: Semireward: a general reward model for semi-supervised learning. arXiv preprint [arXiv:2310.03013](https://arxiv.org/abs/2310.03013) (2023)
25. Li, S., et al.: Moganet: multi-order gated aggregation network. In: The Twelfth International Conference on Learning Representations (2023)
26. Li, S., et al.: Masked modeling for self-supervised representation learning on vision and beyond. arXiv preprint [arXiv:2401.00897](https://arxiv.org/abs/2401.00897) (2023)
27. Lin, T., Zha, H.: Riemannian manifold learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **30**(5), 796–809 (2008)
28. Liu, X., Yin, J., Liu, J., Ding, P., Liu, J., Liu, H.: Trajectorycnn: a new spatio-temporal feature learning network for human motion prediction. *IEEE Trans. Circuits Syst. Video Technol.* **31**(6), 2133–2146 (2020)
29. Lotter, W., Kreiman, G., Cox, D.: Deep predictive coding networks for video prediction and unsupervised learning. In: ICLR (2017)
30. Mahmoud, A., Mohammed, A.: A survey on deep learning for time-series forecasting. In: Hassanien, A.E., Darwish, A. (eds.) *Machine Learning and Big Data Analytics Paradigms: Analysis, Applications and Challenges*. SBD, vol. 77, pp. 365–392. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-59338-4_19
31. Marc’Aurelio Ranzato, A.S., Bruna, J., Mathieu, M., Collobert, R., Chopra, S.: Video (language) modeling: a baseline for generative models of natural videos. *CoRR* [arxiv:1412.6604](https://arxiv.org/abs/1412.6604) (2014)
32. Mendelson, B.: *Introduction to Topology*. Courier Corporation, North Chelmsford (1990)
33. Oliu, M., Selva, J., Escalera, S.: Folded recurrent neural networks for future video prediction. In: Proceedings of ECCV, pp. 716–731 (2018)
34. Oprea, S., et al.: A review on deep learning techniques for video prediction. *IEEE Trans. Pattern Anal. Mach. Intell.* (2020)
35. Patraucean, V., Handa, A., Cipolla, R.: Spatio-temporal video autoencoder with differentiable memory. arXiv preprint [arXiv:1511.06309](https://arxiv.org/abs/1511.06309) (2015)
36. Schudt, C., et al.: Recognizing human actions: a local SVM approach. In: ICPR, vol. 3, pp. 32–36 (2004)
37. Shi, X., et al.: Convolutional LSTM network: a machine learning approach for precipitation nowcasting. *Adv. NIPS* **28** (2015)
38. Simmons, G.F.: *Introduction to topology and modern analysis*, vol. 44. Tokyo (1963)
39. Srivastava, N., Mansimov, E., Salakhudinov, R.: Unsupervised learning of video representations using LSTMs. In: ICML, pp. 843–852 (2015)
40. Su, J., et al.: Convolutional tensor-train LSTM for spatio-temporal learning. *Adv. NIPS* **33**, 13714–13726 (2020)
41. Tan, C., Gao, Z., Li, S., Li, S.Z.: Simvp: towards simple yet powerful spatiotemporal predictive learning. arXiv preprint [arXiv:2211.12509](https://arxiv.org/abs/2211.12509) (2022)
42. Tan, C., et al.: Openstl: a comprehensive benchmark of spatio-temporal predictive learning. *Adv. Neural. Inf. Process. Syst.* **36**, 69819–69831 (2023)
43. Tan, C., et al.: Boosting the power of small multimodal reasoning models to match larger models with self-consistency training. arXiv preprint [arXiv:2311.14109](https://arxiv.org/abs/2311.14109) (2023)
44. Tan, C., Xia, J., Wu, L., Li, S.Z.: Co-learning: learning from noisy labels with self-supervision. In: Proceedings of the 29th ACM International Conference on Multimedia, pp. 1405–1413 (2021)

45. Tan, C., et al.: Temporal attention unit: towards efficient spatiotemporal predictive learning. arXiv preprint [arXiv:2206.12126](https://arxiv.org/abs/2206.12126) (2022)
46. Tschannen, M., Djolonga, J., Rubenstein, P.K., Gelly, S., Lucic, M.: On mutual information maximization for representation learning. In: International Conference on Learning Representations (2019)
47. Tulyakov, et al.: Mocogan: decomposing motion and content for video generation. In: Proceedings of CVPR, pp. 1526–1535 (2018)
48. Vaswani, A., et al.: Attention is all you need. Adv. Neural Inf. Process. Syst. 5998–6008 (2017)
49. Villegas, R., et al.: Decomposing motion and content for natural video sequence prediction. In: ICLR (2017)
50. Wang, Y., et al.: Predrnn: recurrent neural networks for predictive learning using spatiotemporal LSTMs. Adv. NIPS **30** (2017)
51. Wang, Y., et al.: Eidetic 3D LSTM: a model for video prediction and beyond. In: ICLR (2018)
52. Wang, Y., et al.: Predrnn++: towards a resolution of the deep-in-time dilemma in spatiotemporal predictive learning. In: ICML, pp. 5123–5132 (2018)
53. Wang, Y., et al.: Memory in memory: a predictive neural network for learning higher-order non-stationarity from spatiotemporal dynamics. In: Proceedings of CVPR, pp. 9154–9162 (2019)
54. Wei, J., et al.: Enhancing human-like multi-modal reasoning: a new challenging dataset and comprehensive framework. arXiv preprint [arXiv:2307.12626](https://arxiv.org/abs/2307.12626) (2023)
55. Wu, H., et al.: Motionrnn: a flexible model for video prediction with spacetime-varying motions. In: Proceedings of CVPR, pp. 15435–15444 (2021)
56. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint [arXiv:1708.07747](https://arxiv.org/abs/1708.07747) (2017)
57. Xu, Z., Wang, Y., Long, M., Wang, J., KLiss, M.: Predcnn: predictive learning with cascade convolutions. In: IJCAI, pp. 2940–2947 (2018)
58. Yan, W., Zhang, Y., Abbeel, P., Srinivas, A.: Videogpt: video generation using vq-vae and transformers. arXiv preprint [arXiv:2104.10157](https://arxiv.org/abs/2104.10157) (2021)
59. Yu, W., et al.: Efficient and information-preserving future frame prediction and beyond. In: ICLR (2019)



Reinventing Node-centric Traffic Forecasting for Improved Accuracy and Efficiency

Xu Liu¹, Yuxuan Liang²(✉), Chao Huang³, Hengchang Hu¹, Yushi Cao⁴,
Bryan Hooi¹, and Roger Zimmermann¹

¹ National University of Singapore, Singapore, Singapore
{liuxu, huhengc, bhooi, rogerz}@comp.nus.edu.sg

² The Hong Kong University of Science and Technology (Guangzhou),
Guangzhou, China
yuxliang@outlook.com

³ The University of Hong Kong, Hong Kong, China

⁴ Nanyang Technological University, Singapore, Singapore
yushi002@e.ntu.edu.sg

Abstract. Traffic forecasting is a crucial application in smart city efforts. After revisiting the existing literature on deep learning-based traffic forecasting methods, we identify two primary research approaches: node-centric and graph-centric. Node-centric methods focus on constructing spatial features through preprocessing and modeling spatial correlations in the input space. In contrast, graph-centric methods mainly rely on graph neural networks to capture spatial correlations in the latent space. We perform empirical evaluations to identify the pros and cons of each: node methods excel in efficiency while graph methods demonstrate better performance. Based on this, we propose a simple yet effective node-centric framework, named SimST, which overcomes the drawbacks of node-centric methods and enhances their efficiency. Extensive experiments show that SimST achieves performance on par with graph-centric methods while exhibiting up to 39 times inference speedup.

Keywords: Traffic Forecasting · Spatio-Temporal Neural Networks

1 Introduction

Traffic forecasting over road networks plays a pivotal role in enhancing urban planning and traffic management, making it one of the most critical components of Intelligent Transportation Systems [45]. After reviewing the traffic forecasting methods proposed during this decade, we identified two events that have significantly shifted the modeling paradigm in this area. The first occurred with the

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70352-2_2.

breakthrough of deep learning, which enabled the extraction of non-linear patterns in traffic time series. Following this, research in the field gradually moved away from statistical or machine learning models and towards embracing deep learning models [19, 27]. The second shift happened after a series of improvements to the technique of graph neural networks (GNNs) [9, 17]. Since then, Spatio-Temporal Graph Neural Networks (STGNNs) [2, 20, 36] have become the de facto most popular tool for traffic forecasting, in which GNNs are utilized to capture spatial correlations among different sensors and sequential models are applied for modeling temporal dependencies.

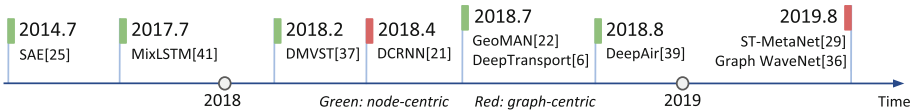


Fig. 1. Representatives between the first and the second modeling paradigm shift.

In contrast to recent work focused on improving STGNNs, in this study we take a step back and investigate how deep traffic forecasting methods modeled spatial correlations after the first, but prior to the second paradigm shift. After studying the related literature of this period (see Fig. 1), we observed that *they focus on modeling spatial correlations in the input space by incorporating the time series of other nodes as a node’s input features. Then the deep models process each node’s features individually.* We refer to this category of methods as **node-centric methods**, in contrast to **graph-centric methods**, e.g., STGNNs, which establish spatial correlations in the latent space (see Fig. 2).

We then conduct a comprehensive evaluation to investigate the advantages and disadvantages of these two types of methods since there are few direct comparisons between them in the literature [11]. Our results indicate that node-centric methods have unique benefits in terms of fast inference time and low GPU memory cost, thanks to their individual node processing nature. However, they require lengthy preprocessing time for incorporating a multitude of nodes’ features and suffer from inferior model performance due to the absence of spatial correlation modeling in the latent space.

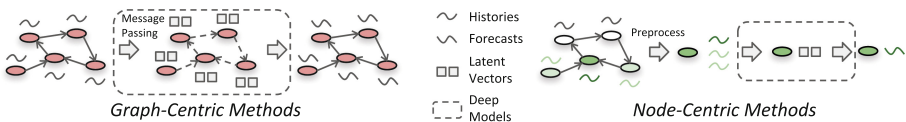


Fig. 2. Illustration of two approaches. Node-centric methods rely solely on the latent vectors of a node itself to predict its future, while graph-centric methods involve message passing among nodes.

In this work, we argue that the efficient nature of node-centric methods is crucial for the applications of traffic forecasting models in the real world, particularly when deploying models in large-scale and real-time forecasting systems that are latency-bound and require fast inference [15]. Thus, we are committed to reinventing conventional node-centric methods by introducing the SimST framework to address their limitations. Specifically, SimST includes a proximity modeling module that restricts the receptive field of each node to its local context, resulting in a significant reduction in preprocessing time. Moreover, SimST adapts the technique of node-specific parameters to the node-centric scenario, allowing for implicit node representation interactions. This is the key component that bridges the performance gap between node-centric and graph-centric methods. Our contribution is fourfold.

- Through revisiting the literature, we categorize existing deep traffic forecasting methods into two classes: node-centric and graph-centric. We identify their strengths and weaknesses via comprehensive empirical evaluations.
- We present a simple yet effective node-centric framework named SimST that overcomes the shortcomings of conventional node-centric methods and enhances their advantages in efficiency.
- Extensive experiments on common benchmarks reveal that compared to graph-centric methods, SimST obtains up to **39** \times inference speedup while performing on par with them. Compared with conventional node-centric methods, SimST achieves up to **26**% performance improvements and up to **83** \times inference speedup.
- We conduct extensive ablation analysis and case studies to promote a better understanding of our method and motivate future research to pay more attention to the node-centric methods.

2 Preliminaries

This section formulates the problem of deep traffic forecasting and defines the graph- and node-centric methods used to address it. Then we describe the representatives in these two research lines.

2.1 Formulations

Deep Traffic Forecasting. We denote the overall traffic readings in a particular time window as $\mathbf{X} \in \mathbb{R}^{N \times T \times F}$, where N is the number of sensors, T is the number of time steps, and F is the feature dimension consisting of a target attribute (e.g., traffic flow) and auxiliary information. We additionally denote the readings of node v as $\mathbf{X}_v \in \mathbb{R}^{T \times F}$. Typically, a directed sensor graph $G = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = N$ nodes and $|\mathcal{E}|$ edges is used to represent the sensors’ correlations [20, 36]. In deep learning based traffic forecasting, a neural network Θ is trained to predict the target attribute in future T_f steps $\mathbf{Y} \in \mathbb{R}^{N \times T_f \times 1}$ based on T_h historical observations $\mathbf{X} \in \mathbb{R}^{N \times T_h \times F}$ over the graph [20, 40].

Graph-Centric Traffic Forecasting. To predict the future readings of a sensor v_i , the model accesses the sensor graph G , the input \mathbf{X} , and the latent vectors (generated from learnable parameters) of all other nodes for spatial correlations modeling.

Node-Centric Traffic Forecasting. To predict the future readings of a sensor v_i , the model accesses the input $f(\mathbf{X}, G)$ and only sensor v_i 's latent vectors, where $f(\mathbf{X}, G)$ is a preprocessing function that builds spatial features for sensor v_i in the input space using G .

2.2 Graph-Centric Approaches

STGNNs are typical graph-centric methods for traffic forecasting, as they utilize stacked GNN layers to establish spatial correlations between nodes and thus require access to the latent vectors of neighboring nodes (see Fig. 2). In recent years, STGNNs have become the most widely used tool for deep traffic forecasting [3, 8, 24]. They generally integrate GNNs with either RNNs or TCNs to capture the complex spatial and temporal dependencies in traffic data. For instance, DCRNN [20], AGCRN [2], and GMSDR [22] incorporated graph convolution networks with RNN and its variants. To improve training speed and enjoy parallel computation, a plethora of approaches such as STGCN [40], GWN [36], and DMSTGCN [12] combined GNNs with dilated causal convolution. Methods that used attention mechanisms for spatial learning, such as GMAN [44] and ASTGCN [11], also fall under the category of graph-centric methods, since attention is performed on the latent vectors of various nodes. Despite achieving great success in accuracy, these graph methods perform explicit node representation interactions via message passing and thus require storing a substantial number of intermediate hidden vectors for each node, making many of them unable to scale to large sensor graphs [15, 25].

2.3 Node-centric Approaches

Before the emergence of GNNs, most deep learning-based traffic forecasting approaches fell into the category of node-centric methods [5, 21, 26, 33, 37, 39, 41]. According to a survey [27], SAE [26] was the earliest deep learning-based method for traffic forecasting. For an anchor node, SAE concatenated observations of all other sensors into a large vector, which served as the node's spatio-temporal features and was passed through an MLP for predictions. To capture the dynamic correlations between sensors, GeoMAN [21] applied an attention mechanism between the histories of the anchor node and all other nodes. The output of the attention module was seen as the spatial features of the anchor, and was then fed into an encoder-decoder LSTM architecture for temporal dependencies modeling. Moreover, DeepTransport [5] searched all possible travel paths that start from an anchor node to its r -order neighbors. The method then used 1D convolution to model the relations within the r th order nodes, and employed LSTMs

and attention mechanisms to capture the relations across the orders. DMVST-Net [37] and DeepAir [39] were also well-known node-centric approaches, which focused on taxi demand prediction and air quality prediction. To summarize, node-centric methods typically involve a preprocessing stage where observations from all or some of the nodes are collected to consider spatial correlations in the input space. Once the constructed input is passed through the neural networks, only the sensor’s own latent vectors are used for making predictions (see Fig. 2). However, as shown in Table 1, these methods necessitate considerable preprocessing time and encounter subpar model performance. We notice that there is one recently proposed node-centric method SGP [8] that utilizes deep echo state networks for data preprocessing. Though this method obtains competitive performance, its preprocessing duration spanned from tens of seconds to several minutes, still lacking the ability to achieve rapid data preprocessing.

3 Empirical Comparisons on Graph-Centric and Node-centric Methods

We experiment with five commonly used traffic datasets (see details in Sect. 5). We predict the next 12 steps traffic status based on the 12-step historical data. Three representative node-centric methods are implemented as baselines: SAE [26], GeoMAN [21] and DeepTransport [5]. Since two of them employ LSTMs for temporal dependencies modeling, we choose RNN-based graph-centric methods for fair comparisons: DCRNN [20] (from the same era as node-centric methods) and GMSDR [22] (a recent published work). We include a variant of the proposed framework here, i.e., SimST-GRU to facilitate a direct comparison. The details of SimST are presented in Sect. 4.

We provide a comprehensive comparison of graph- and node-centric methods from the following criteria: (1) **Accuracy**. We adopt three common metrics in forecasting tasks to evaluate the model performance, including mean absolute error (MAE), root mean squared error (RMSE), and mean absolute percentage error (MAPE). (2) **Time consumption**. We prioritize evaluating inference time (including preprocessing time) over training time, as it is more critical in real-world applications. Training time details can be found in Appendix F. (3) **GPU memory usage**. We measure the GPU memory consumption to reflect the ability of methods to scale to large-scale traffic datasets.

3.1 Results Analysis

We summarize the experimental results in Table 1, with the following observations and conclusions:

- **Graph-centric methods achieve higher accuracy.** As demonstrated in the table, graph-centric methods outperform node-centric methods by a significant margin, except for PeMSD8 where the improvement is less pronounced. This can be attributed to the fact that PeMSD8 has the smallest

Table 1. Due to the page limit, we present four of the five benchmark datasets and only report the MAE metric here. The complete results are available in Appendix F. MAE: averaged test MAE result over all predicted horizons. Pre: preprocessing time (s) before inference. Infer: inference time (s) on the validation set. GPU: GPU memory usage (GB). DeepTrans: DeepTransport. SimST: SimST-GRU, a variant of SimST.

Method	PeMSD8 ($N=170$)				LA ($N=207$)				PeMSD4 ($N=307$)				BAY ($N=325$)			
	MAE	Pre	Infer	GPU	MAE	Pre	Infer	GPU	MAE	Pre	Infer	GPU	MAE	Pre	Infer	GPU
SAE	16.95	6.14	1.90	1.05	3.76	8.97	2.59	1.05	22.22	31.89	5.54	1.07	1.84	32.97	9.20	1.08
GeoMAN	16.83	6.21	4.98	1.42	3.66	8.86	6.54	1.46	23.06	32.54	12.55	1.66	1.85	32.51	20.77	1.68
DeepTrans	17.90	0.05	3.45	3.29	3.65	0.52	3.65	2.84	22.96	0.06	5.97	3.29	1.87	0.81	8.88	2.84
DCRNN	16.62	0	6.72	3.97	3.14	0	7.66	4.71	22.39	0	9.89	6.43	1.63	0	16.36	6.81
GMSDR	16.36	0	7.71	3.65	3.21	0	11.52	5.48	20.49	0	13.20	7.81	1.69	0	22.05	8.21
SimST(our)	14.99	0.01	0.34	2.53	3.16	0.03	0.45	2.65	19.19	0.02	0.55	3.15	1.57	0.07	0.96	3.23

number of nodes, with an average node degree of around 1. In such a simple graph, node-centric methods perform similarly to graph-centric methods in terms of accuracy. However, as the graph becomes more complex, with an increase in either the number of nodes (PeMSD4) or the number of edges (average node degree of 7.3 for LA), modeling spatial correlations in the input space becomes less effective compared to modeling them in the latent space.

- **Graph-centric methods typically do not require preprocessing.** As shown in the table, SAE and GeoMAN require significant preprocessing time as they need to concatenate observations from all other nodes into the input for each node. This process is time-consuming and memory-intensive. Although we used indexing techniques to optimize the time, the results still remain impractical compared to the inference time. DeepTransport has a shorter preprocessing time as it limits its consideration to r -order neighbors. However, it falls short on datasets with a higher number of edges, such as LA and BAY, due to the exponential growth of paths.
- **Node-centric methods generally have faster inference time.** The results show that node-centric methods (SAE and DeepTransport) run significantly faster than graph-centric methods (DCRNN and GMSDR). The reason is that node-centric methods do not necessitate explicit node interactions in the latent space, which is a requisite step in graph-centric methods. For example, STGNNs rely on GNNs for performing the message passing step, which can become significantly inefficient when graphs are large or with dense connections [4, 42]. We note that GeoMAN becomes slower than DCRNN and GMSDR as node number increases. This is due to the use of a global spatial attention module, which is computationally intensive when processing a large number of nodes.
- **Node-centric methods have lower GPU memory usage.** Our results reveal that node-centric methods require less GPU memory than graph-centric methods, and their memory usage remains stable with an increasing number of nodes. We explain the reasons as follows. During the training/inference of a graph-centric method, nodes are strictly bound together and instances within a batch are sampled solely along the temporal axis. We

refer to this as graph-based batch sampling, e.g., a batch with a size of B_g is a four-dimensional tensor $\mathbf{X} \in \mathbb{R}^{B_g \times N \times T_h \times F}$. In contrast, node-centric methods uniformly sample instances across both temporal and spatial dimensions. In this case, nodes belonging to the same time period are viewed as different instances. Consequently, the input to node-centric methods is disproportionate to N and becomes $\mathbf{X} \in \mathbb{R}^{B_n \times T_h \times F}$, where $B_n \ll B_g \times N$. This property plays a key role in allowing node-centric methods to have low GPU costs and admirable scalability. Detailed studies are conducted to reveal the benefits of this node-based batch sampling strategy in Sect. 5.4.

From Table 1, we can also observe that SimST-GRU achieves substantial improvements over node-centric baselines in terms of performance, preprocessing time, and inference time, verifying the versatility of our framework.

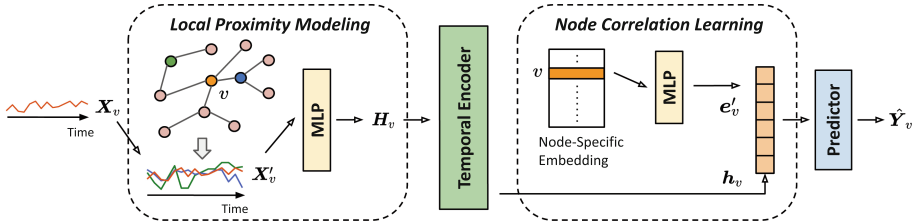


Fig. 3. An illustration of the SimST framework from the perspective of a node v .

4 The Proposed Framework

As illustrated in the preceding section, we have recognized the benefits of node-centric methods, i.e., rapid inference time and reduced GPU memory consumption, as well as their drawbacks, including inferior model performance and lengthy preprocessing time. In this study, we reinvent previous node-centric methods by proposing SimST, a simple yet effective framework that addresses their weaknesses and also enhances their strengths. Specifically, in Sect. 4.1, we detail our approach for modeling local proximity of nodes in the input space, which leads to a notable reduction in preprocessing time. In Sect. 4.2, we present our utilization of node embeddings – a simple yet powerful technique to enable implicit node representation interactions, resulting in significant performance improvements. Lastly, we illustrate temporal encoders for capturing temporal dependencies and a predictor for generating the future in Sect. 4.3. Figure 3 depicts the architecture of SimST.

4.1 Local Proximity Modeling

SAE and GeoMAN require concatenating the histories of all other nodes for each node, leading to considerable computational and memory expenses. DeepTransport searches and records all possible travel paths up to the r -order of neighbors,

which results in an exponential increase in paths for dense graphs. In this study, we also consider spatial correlations in the input space. However, we adopt a more efficient and flexible approach by (1) modeling only the proximity of each node, unlike SAE and GeoMAN, and (2) using h -hop top- k neighbors to describe the local context, which avoids path explosion.

Specifically, we limit the receptive field of each node v by concatenating only its top- k neighbors within one hop and extending the range up to h -hops. Additionally, we take an average of all h -th hop neighbors' histories to allow the anchor node to absorb its context in the h -th hop. The selection of top- k neighbors is based on the weights in the normalized matrix $\tilde{\mathbf{A}}$, where $\tilde{\mathbf{A}} = \tilde{\mathbf{D}}^{-1}(\mathbf{A} + \mathbf{I})$ and $\tilde{\mathbf{D}}$ is the degree matrix of $\mathbf{A} + \mathbf{I}$. The weights in \mathbf{A} are derived from the road network distances between sensors, and the neighbors are ordered based on the weights. Moreover, we consider the neighbors in the reversed direction, i.e., the entries in \mathbf{A}^T , so that the model can perceive the impact from forward and backward neighbors. Let $\mathcal{N}_f^h(v)$ and $\mathcal{N}_b^h(v)$ denote the selected top- k nodes in the h -th hop in two directions, $\mathbf{X}_{avg_f}^h$ and $\mathbf{X}_{avg_b}^h$ represent the average histories of all h -th hop neighbors. We construct the input feature matrix $\mathbf{X}'_v \in \mathbb{R}^{T_h \times (F+h \times (2k+2))}$ of node v during period T_h :

$$\mathbf{X}_v^h = (\{\mathbf{X}_{v_f}^h : v_f \in \mathcal{N}_f^h(v)\}, \{\mathbf{X}_{v_b}^h : v_b \in \mathcal{N}_b^h(v)\}, \mathbf{X}_{avg_f}^h, \mathbf{X}_{avg_b}^h) \quad (1)$$

$$\mathbf{X}'_v = (\mathbf{X}_v \parallel \mathbf{X}_v^1 \parallel \dots \parallel \mathbf{X}_v^h) \quad (2)$$

Then, we consider the second dimension of \mathbf{X}'_v as the feature dimension and apply an MLP to embed it to the latent space $\mathbf{H}_v \in \mathbb{R}^{T_h \times D_m}$, where D_m is the model latent dimension.

4.2 Node Correlation Learning

Representing users with unique embeddings and learning with these embeddings to capture behavioral similarities among users is a fundamental technique in recommender systems [13, 43]. Along a similar line, we argue that the traffic situation of a sensor is largely influenced by its specific position on a road, e.g., on the mainline or on a ramp. This is an intrinsic, time-invariant property of the sensor: hence, we propose to represent it using static sensor embeddings.

Concretely, we randomly initialize a low-dimensional embedding for each sensor, leading to an embedding table $\mathbf{E} \in \mathbb{R}^{|\mathcal{V}| \times D_n}$, where D_n is the embedding size. This sensor embedding is integrated with the latent representations \mathbf{H}_v derived from the sensor's histories, such that the embedding serves as a consistent signal, allowing the model to distinguish different sensors based on the input traffic patterns. Consequently, nodes with inherently similar traffic patterns tend to learn comparable node embeddings. We hypothesize that the spatial relationships between arbitrary sensor pairs can be reflected in their embedding similarities, which enables implicit node representation interactions. The term implicit in this context denotes the achievement of relevant sensors having similar hidden representations, but without performing the message passing step like in GNNs.

We empirically verify our hypothesis in a case study conducted in Sect. 5.5, where we show that the learned sensor embeddings contain rich information for sensor relevance reasoning.

In addition, we notice that some graph-centric methods [2, 36] also employed node-specific parameters. However, these parameters are used in their graph neural networks components, and ours can be viewed as their adaptation in the node-centric scenario.

Complexity Comparison. We compare the complexity of spatial learning modules between SimST and STGNNs. For simplicity we assume the number of features D_m is fixed for all L layers. For STGNNs applying the predefined adjacency matrix [10, 20, 32, 40], the complexity is $O(L|\mathcal{E}|D_m + LND_m^2)$. When using the adaptive adjacency matrix that boosts performance [2, 6, 12, 36], the complexity becomes quadratic: $O(LN^2D_m + LND_m^2)$. SimST has a complexity of $O(h|\mathcal{E}| + ND_nD_m)$, where the first term represents proximity modeling and the second term refers to correlation learning. SimST generally has lower complexity compared to STGNNs in both predefined and adaptive scenarios, when using the common parameter choices such as $L = 3$, $D_m = 32$, $D_n = 20$.

4.3 Temporal Encoder and Predictor

There are several viable options for building the temporal dependencies of a node’s history. To demonstrate that SimST can be generalized to various temporal models, we incorporate three basic and popular backbones in this study: GRU [7], WaveNet [28], and Transformer [35]. Furthermore, we follow WaveNet to introduce causality into the attention mechanism, leading to the design of the Causal Transformer. This ensures that the model cannot violate the temporal order of inputs and such causality can be easily implemented by masking the specific entries in the attention map. Note that GeoMAN and DeepTransport used LSTM as their temporal models, but we have chosen to adopt GRU, which has fewer parameters and faster processing times.

Formally, the input of temporal encoder is the representations $\mathbf{H}_v \in \mathbb{R}^{T_h \times D_m}$. For GRU and WaveNet, they compress the temporal dimension to 1, generating the output $\mathbf{h}_v \in \mathbb{R}^{D_m}$. Note that the resulting representation of Transformer still has the same shape as \mathbf{H}_v and we only take the last time step as the output. Then we concatenate $\mathbf{h}_v \in \mathbb{R}^{D_m}$ of node v with its node embedding $e'_v \in \mathbb{R}^{D_m}$, which is from a non-linear transformation of $e_v \in \mathbb{R}^{D_n}$. Lastly, we implement the predictor as a two-layer MLP for generating the forecasting results of node v : $\hat{\mathbf{Y}}_v = \text{MLP}(\mathbf{h}_v || e'_v) \in \mathbb{R}^{T_f}$. The mean absolute error is used as the loss function.

5 Experiments

5.1 Experimental Setup

Datasets & Baselines. Table 2 presents the experimental datasets, the first five of which are commonly used traffic benchmarks. We also incorporate a recently

Table 2. Dataset statistics.

Dataset	Nodes	Edges	Degree	Instances
PeMSD4 [32]	307	338	1.1	16,992
PeMSD7 [32]	883	865	1.0	28,224
PeMSD8 [32]	170	276	1.6	17,856
LA [20]	207	1,515	7.3	34,272
BAY [20]	325	2,369	7.3	52,116
CA [25]	8,600	201,363	23.4	35,040

published large-scale traffic dataset CA [25] to assess the scalability of our method. The time interval is 15 min for CA and 5 min for other data. Following [6, 20, 32], we use 12-step historical data to predict the next 12 steps, and build adjacency matrices using road network distances with a thresholded Gaussian kernel [31]. See more information in Appendix A. We compare SimST with the following baselines. Historical Average (HA) [29] and Vector Autoregression (VAR) [34] are traditional methods. DCRNN [20], STGCN [40], ASTGCN [11], GWNET [36], STSGCN [32], AGCRN [2] are well-known graph-centric methods. We also include recently published graph-centric methods: STGODE [10], STGNCDE [6], GMSDR [22], and DSTAGNN [18].

Table 3. Performance comparisons. Test MAE, RMSE, and MAPE results are averaged over all predicted time steps. We bold the best results and underline the second best results. * denotes the improvement of a SimST variant over the best baseline is statistically significant at level 0.05. We copy the test results of the best-performing node-centric baseline SAE here for a direct comparison.

Method	PeMSD8 ($N=170$)			LA ($N=207$)			PeMSD4 ($N=307$)			BAY ($N=325$)			PeMSD7 ($N=883$)		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
HA	34.86	59.24	27.88%	4.16	7.80	13.00%	38.03	59.24	27.88%	2.88	5.59	6.80%	45.12	65.64	24.51%
VAR	19.19	29.81	13.10%	5.28	9.06	12.50%	24.54	38.61	17.24%	2.24	3.96	4.83%	50.22	75.63	32.22%
DCRNN	16.62	25.95	10.60%	3.14	6.28	<u>8.65%</u>	22.39	34.93	15.19%	1.63	3.65	3.70%	23.41	36.66	9.98%
STGCN	17.41	26.72	11.78%	3.39	6.79	9.34%	21.59	33.83	15.49%	1.88	4.30	4.28%	26.12	41.43	11.92%
ASTGCN	17.91	27.34	11.36%	3.57	7.19	10.32%	21.58	33.76	14.71%	1.86	4.07	4.27%	25.77	39.41	11.67%
GWNET	14.98	23.75	9.75%	3.06	6.10	8.38%	19.33	30.73	13.18%	<u>1.58</u>	3.54	<u>3.59%</u>	20.65	<u>33.47</u>	8.84%
STSGCN	17.13	26.80	10.96%	3.32	6.66	9.06%	21.19	33.65	13.90%	1.79	3.91	4.06%	24.26	39.03	10.21%
AGCRN	15.95	25.22	10.09%	3.19	6.41	8.84%	19.83	32.26	<u>12.97%</u>	1.62	3.61	3.66%	20.69	34.19	8.86%
STGODE	16.81	25.97	10.62%	4.73	7.60	11.71%	20.84	32.82	13.77%	1.77	3.33	4.02%	22.99	37.54	10.14%
STGNCDE	15.45	24.81	9.92%	3.58	6.84	9.91%	<u>19.21</u>	31.09	12.76%	1.68	3.66	3.80%	<u>20.53</u>	33.84	8.80%
GMSDR	16.36	25.58	10.28%	3.21	6.41	8.76%	20.49	32.13	14.15%	1.69	3.80	3.74%	22.27	34.94	9.86%
DSTAGNN	15.67	24.77	9.94%	3.27	6.57	9.07%	19.30	31.46	12.70%	1.63	3.56	3.73%	21.42	34.51	9.01%
SAE	16.95	26.35	12.34%	3.76	7.40	10.72%	22.22	34.59	15.60%	1.84	4.13	4.16%	24.79	38.61	10.87%
SimST-WN	15.55	24.70	10.17%	3.19	6.35	8.97%	19.56	31.24	13.38%	1.60	3.55	3.66%	20.81	33.94	8.87%
SimST-CT	15.13	<u>24.18</u>	<u>9.73%</u>	<u>3.11</u>	<u>6.19</u>	8.73%	19.32	30.98	13.35%	1.59	3.50	3.61%	20.63	33.71	<u>8.70%</u>
SimST-GRU	<u>14.99</u>	24.26	9.66%	3.16	6.29	8.82%	19.19	<u>30.92</u>	13.13%	1.57	<u>3.47</u>	3.58%	20.14*	33.34	8.46%

Implementation Details. SimST is implemented with PyTorch 1.12. There are three SimST variants based on the applied temporal encoders. We describe the

configurations for variants in Appendix B. The following settings are the same for all variants. We train our method via the Adam optimizer with a learning rate of 0.001 and a weight decay of 0.0001. We set the maximum epochs to 100 and use an early stop strategy with a patience of 20. The batch size B_n is 1,024. The embedding size D_n is 20. The selections of h and k are in Appendix C. For baselines, we run their codes based on the recommended configurations if their accuracy is not known for a dataset. If known, we use their officially reported accuracy. Experiments are conducted on an NVIDIA RTX A6000 GPU. The code is available at: <https://github.com/liuxu77/SimST>.

5.2 Comparisons on Common Benchmarks

We conduct a model comparison between SimST variants and state-of-the-art graph-centric methods on traffic benchmarks. The three variants of SimST are SimST-GRU, SimST-WaveNet (WN), and SimST-Causal Transformer (CT). According to the results in Table 3, all variants of SimST achieve competitive performance on the three evaluation metrics of all datasets, which validates the effective design of SimST and indicates SimST can generalize to various temporal encoders. For comparison among SimST variants: note that for fairness we ensure that the variants have a similar parameter size (around 150k). SimST-GRU generally achieves comparable performance to state-of-the-art graph-centric methods, i.e., GWNEN and STGNCDE. SimST-CT also attains competitive accuracy, with little difference from SimST-GRU and GWNEN. In summary, our results demonstrate that node-centric methods can perform comparable to the graph-centric methods with elaborate design.

Table 4. Efficiency comparisons between best-performing graph-centric baselines and SimST variants. GPU: GPU memory consumption (GB). Infer: inference time (s) on the validation set. We bold the fastest inference time. The Δ column means the inference time improvements of SimST-WN over all other methods.

Method	PeMSD8 ($N=170$)			LA ($N=207$)			PeMSD4 ($N=307$)			BAY ($N=325$)			PeMSD7 ($N=883$)		
	GPU	Infer	Δ	GPU	Infer	Δ	GPU	Infer	Δ	GPU	Infer	Δ	GPU	Infer	Δ
GWNEN	3.15	1.10	3.79×	3.52	1.22	3.30×	4.30	1.81	4.21×	4.45	3.02	3.55×	10.78	10.79	5.65×
AGCRN	2.96	2.07	7.14×	3.46	2.15	5.81×	4.66	2.18	5.07×	4.77	3.58	4.21×	11.74	12.66	6.63×
STGODE	2.96	4.92	16.97×	3.07	4.99	13.49×	3.87	6.11	14.21×	3.91	9.75	11.47×	7.64	24.88	13.03×
STGNCDE	2.68	9.80	33.79×	2.88	13.65	36.89×	3.46	15.55	36.16×	3.57	25.53	30.04×	6.71	73.97	38.73×
GMSDR	3.65	7.71	26.59×	5.48	11.52	31.14×	7.81	13.20	30.70×	8.21	22.05	25.94×	20.68	47.32	24.77×
DSTAGNN	8.82	1.68	5.79×	10.44	1.89	5.11×	15.14	2.83	6.58×	15.98	4.65	5.47×	37.80	22.59	11.82×
SimST-CT	2.10	0.45	1.55×	2.19	0.58	1.57×	2.37	0.74	1.72×	2.40	1.30	1.53×	3.57	3.40	1.78×
SimST-GRU	2.53	0.34	1.17×	2.65	0.45	1.22×	3.15	0.55	1.28×	3.23	0.96	1.13×	5.34	2.48	1.30×
SimST-WN	2.00	0.29	–	2.06	0.37	–	2.19	0.43	–	2.22	0.85	–	2.99	1.91	–

We select the top-performing baselines and compare their efficiency with SimST variants in Table 4. It can be seen that all SimST variants are significantly faster and have lower GPU usage than graph-centric methods, thanks to their simple model architecture and low model complexity. Among the variants,

SimST-WN are the fastest due to WaveNet’s superior parallelism ability. SimST-GRU also achieves good results, which we attribute to an optimized implementation in the PyTorch library. SimST’s preprocessing time for the five datasets in Table 4 is {0.01, 0.03, 0.02, 0.07, 0.09} seconds, respectively (all variants share the same time), which is significantly faster than traditional node-centric methods and can be considered negligible. See Appendix D for more results.

5.3 Comparisons on the CA Dataset

In this part, we conduct an empirical comparison between graph-centric baselines and SimST on the CA dataset to assess the scalability of the models in a vast traffic network. We select SimST-GRU because it exhibits a good balance between performance and efficiency. We reproduce the results reported in [25] due to the usage of different experimental hardware.

Table 5. Performance and efficiency comparisons on the CA dataset. Test MAE, RMSE, and MAPE are averaged over all predicted time steps. Pre: preprocessing time (s) before inference. Infer: inference time (s) on the validation set. GPU usage is not reported, as the baselines typically necessitate the complete occupancy of GPU.

Method	CA ($N=8,600$)					
	MAE	RMSE	MAPE	Pre	Infer	Δ
DCRNN	21.75	34.21	17.06%	0	897	14.70×
STGCN	21.53	36.64	16.11%	0	215	3.52×
GWNET	21.01	33.29	16.71%	0	560	9.18×
STGODE	20.56	36.53	16.07%	0	682	11.18×
SimST-GRU	19.26	32.07	14.78%	1.48	61	–

Table 5 showcases the performance and efficiency evaluation results. The absence of baselines in the table indicates the models incur out-of-memory issue even when configured with a batch size of 4 on a GPU with 48 GB memory. In particular, while STGCN is capable of execution, its speed is prohibitively slow, so we exclude it in this analysis. It can be seen that the CA dataset poses significant challenges for the selected graph-centric baselines in this study, as only four of them are capable of running on it. In contrast, SimST-GRU not only efficiently scales to the CA dataset with fast inference speed, but also exhibits superior model performance. The potential reasons for this discernible performance gap are worth speculating. According to Table 2, the average node degree of the CA dataset is notably higher compared to others. This results in more intricate sensor relationships. Some nodes might have redundant or irrelevant information, and distinguishing between meaningful and noisy spatial correlations becomes more challenging in such scenario. Furthermore, within a sprawling graph, pertinent information might be dispersed across nodes that are

distant from each other. The limited receptive field of GNNs can impede the model’s capacity to capture long-range dependencies.

5.4 Ablation Studies

We next select SimST-GRU and SimST-CT as the representatives (due to their preferable performance) and conduct ablation and case studies on one flow dataset PeMSD4 and one speed dataset LA.

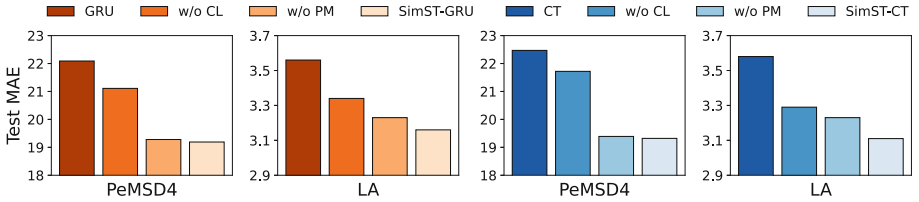


Fig. 4. Effects of correlation learning (CL) and proximity modeling (PM) modules.

Effects of Spatial Learning Modules. We study the effects of two spatial modules in Fig. 4 by considering the following settings: (1) **w/o Correlation Learning (CL)**: we turn off the CL module. (2) **w/o Proximity Modeling (PM)**: for each node, we only input its own historical data. (3) **CT/GRU**: we do not perform spatial learning. First, we find that removing CL leads to significant degradation of MAE for SimST-GRU and SimST-CT on both datasets, revealing the great importance of enabling node interactions in the latent space. Second, the benefit of PM is marginal on PeMSD4 due to the limited neighbor information available, with the average degree of nodes being 1.1. In contrast, LA has an average node degree of 7.3, where PM affects performance more. Third, we notice that removing CL has a greater impact on PeMSD4 than on LA. This is because the CL module can act as a complement to the PM function when neighbor information is scarce. Conversely, when there are more neighbors available, the effect of CL on performance becomes less significant. Moreover, there is a recent research trend in traffic forecasting that focuses on modeling dynamic spatial correlations [18, 30, 38]. To address this, we incorporate a time embedding into our CL module to explore dynamic modeling within node-centric methods. See results in Appendix C.

Effects of Node-Based Batch Sampling. This section shows how the node-based sampling strategy can benefit SimST. First, we show the impact of B_n in the first column of Fig. 5. Generally, SimST-GRU and SimST-CT reach their best performance when setting B_n to 1,024 on the two datasets, verifying that a large B_n reduces the noise in the gradient estimation and leads to a degenerated

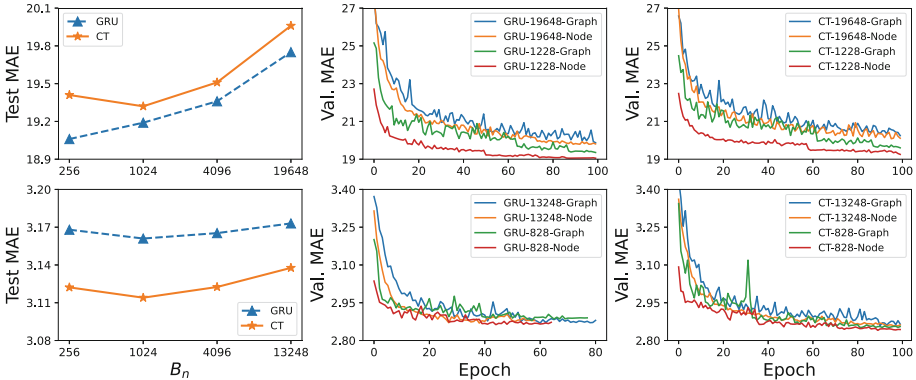


Fig. 5. Effects of the node-based batch sampling method on PeMSD4 (first row) and LA (second row). Here we denote SimST-GRU as GRU and SimST-CT as CT. In the first column, the largest values in the x-axis are computed by $N \times 64$ (the common batch size used in STGNNs [2, 6, 36]). In the second and third columns, the number in the legend denotes B^* .

generalization [16]. Second, we present the influence of different batch-forming methods, i.e., graph-based and node-based sampling by drawing the convergence curves in the second and third columns. Note that we ensure fair comparisons by setting the same actual batch size $B^* = B_g \times N = B_n$ for graph-based and node-based methods. In Fig. 5, it can be seen that node-based batch sampling surpasses the counterpart. Concretely, node-based sampling is marginally superior to graph-based method when using a large B^* (e.g., 19,648/13,248), as the sample diversity in graph-based method is already plentiful at this time. However, decreasing B^* to small values (e.g., 1,228/828), which is essential to boost performance, makes the impact of sample diversity more pronounced, i.e., generalization performance is greatly improved [1], especially on PeMSD4. Also, we find that the curves of node-based sampling are generally more stable than the graph-based method.

5.5 Case Study

We explore further to understand *what exactly the node embeddings have learned* through a case study that makes connections between embedding similarities and real-world sensor locations. We apply SimST-CT on LA and BAY since only these two provide sensors’ coordinates. Concretely, we first compute pairwise cosine similarities between node embeddings, i.e., for $v_i, v_j \in \mathcal{V}$, similarity = $\frac{E_{v_i} \cdot E_{v_j}}{\|E_{v_i}\|_2 \|E_{v_j}\|_2}$, and pairwise geodesic distances between sensors based on their coordinates. Then we calculate the average cosine similarities within the ranges of 0–1, 1–5, 5–10, 10–20, and 20–35 km of all sensors, leading to a statistical overview shown in Fig. 6 (left part). It can be seen that cosine similarity becomes smaller when the geodesic distance gets larger. This result obeys Tobler’s First

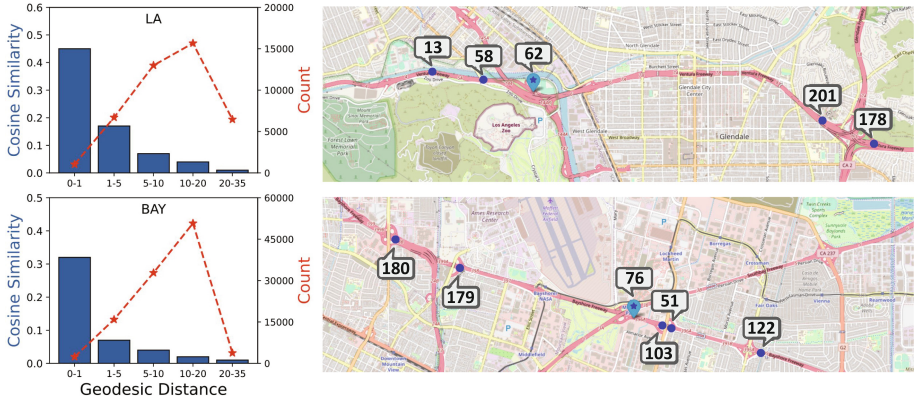


Fig. 6. A case study of learned sensor embeddings. The Count axis on the left part means the number of sensor pairs in the corresponding ranges. The sum of the counts is N^2 , and the trend of count is indicated by red line. On the right part, the star icons are the anchor sensor, the circle icons denote the selected similar sensors, and the roads colored in red are the freeways in California, USA. (Color figure online)

Law of Geography: near things are more related than distant things. Next, we look at an example by selecting sensor 62 in LA as the anchor node, which is located at the intersection of two freeways. We also visualize its top similar neighbors based on cosine similarity in Fig. 6 (upper right). We find that (1) 13 and 58 are the nearby neighbors, which are also included in the adjacency matrix. (2) 201 and 178 are distant nodes for sensor 62. They are considered similar sensors because they are also located at intersections, and thus share similar traffic patterns to the anchor. The situation in the BAY dataset is similar to LA (shown in the lower right of Fig. 6), so we do not elaborate further.

6 Conclusion and Future Work

By reviewing the literature, we classify existing deep traffic forecasting techniques into two categories: node-centric and graph-centric methods. We then compare them through empirical evaluations and propose SimST, a simple yet effective node-centric framework to tackle the shortcomings of existing node-centric methods. Our extensive experiments demonstrate that node-centric methods can attain comparable performance to graph-centric methods, encouraging the community to advance more robust node-centric techniques.

Although SimST demonstrates the ability to generalize to newly added sensors, we note that it still requires fine-tuning for these new nodes and is not fully inductive. As a next step, future research can focus on developing a spatio-temporal model that operates in a completely inductive manner, allowing seamless integration of new nodes without the need for additional fine-tuning. Furthermore, our empirical analysis has specifically focused on traffic forecasting

using STGNNs. Some emerging Transformer-based methods [14, 23] have not been discussed in this study. In the future, we intend to incorporate these methods to broaden the scope of our discussion.

Acknowledgments. This research is supported by Singapore Ministry of Education Academic Research Fund Tier 2 under MOE’s official grant number T2EP20221-0023. This work is also supported by the Guangzhou-HKUST(GZ) Joint Funding Program (No. 2024A03J0620).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Ash, J.T., Zhang, C., Krishnamurthy, A., Langford, J., Agarwal, A.: Deep batch active learning by diverse, uncertain gradient lower bounds. In: ICLR (2020)
2. Bai, L., Yao, L., Li, C., Wang, X., Wang, C.: Adaptive graph convolutional recurrent network for traffic forecasting. In: NeurIPS (2020)
3. Cao, D., et al.: Spectral temporal graph neural network for multivariate time-series forecasting. In: NeurIPS (2020)
4. Chen, T., Sui, Y., Chen, X., Zhang, A., Wang, Z.: A unified lottery ticket hypothesis for graph neural networks. In: ICML (2021)
5. Cheng, X., Zhang, R., Zhou, J., Xu, W.: Deeptransport: learning spatial-temporal dependency for traffic condition forecasting. In: IJCNN (2018)
6. Choi, J., Choi, H., Hwang, J., Park, N.: Graph neural controlled differential equations for traffic forecasting. In: AAAI (2022)
7. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint [arXiv:1412.3555](https://arxiv.org/abs/1412.3555) (2014)
8. Cini, A., Marisca, I., Bianchi, F.M., Alippi, C.: Scalable spatiotemporal graph neural networks. In: AAAI (2023)
9. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: NeurIPS (2016)
10. Fang, Z., Long, Q., Song, G., Xie, K.: Spatial-temporal graph ode networks for traffic flow forecasting. In: KDD (2021)
11. Guo, S., Lin, Y., Feng, N., Song, C., Wan, H.: Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In: AAAI (2019)
12. Han, L., Du, B., Sun, L., Fu, Y., Lv, Y., Xiong, H.: Dynamic and multi-faceted spatio-temporal deep learning for traffic speed forecasting. In: KDD (2021)
13. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: WWW (2017)
14. Jiang, J., Han, C., Zhao, W.X., Wang, J.: Pdformer: propagation delay-aware dynamic long-range transformer for traffic flow prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 4365–4373 (2023)
15. Jin, G., Liang, Y., Fang, Y., Huang, J., Zhang, J., Zheng, Y.: Spatio-temporal graph neural networks for predictive learning in urban computing: a survey. TKDE (2023)
16. Keskar, N.S., Mudigere, D., Nocedal, J., Smelyanskiy, M., Tang, P.T.P.: On large-batch training for deep learning: generalization gap and sharp minima. In: ICLR (2017)

17. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: ICLR (2017)
18. Lan, S., Ma, Y., Huang, W., Wang, W., Yang, H., Li, P.: Dstagnn: dynamic spatial-temporal aware graph neural network for traffic flow forecasting. In: ICML (2022)
19. Lana, I., Del Ser, J., Velez, M., Vlahogianni, E.I.: Road traffic forecasting: recent advances and new challenges. *IEEE Intell. Transport. Syst. Maga.* **10**, 93–109 (2018)
20. Li, Y., Yu, R., Shahabi, C., Liu, Y.: Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In: ICLR (2018)
21. Liang, Y., Ke, S., Zhang, J., Yi, X., Zheng, Y.: Geoman: multi-level attention networks for geo-sensory time series prediction. In: IJCAI (2018)
22. Liu, D., Wang, J., Shang, S., Han, P.: MSDR: multi-step dependency relation networks for spatial temporal forecasting. In: KDD (2022)
23. Liu, H., et al.: Spatio-temporal adaptive embedding makes vanilla transformer sota for traffic forecasting. In: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, pp. 4125–4129 (2023)
24. Liu, X., Liang, Y., Huang, C., Zheng, Y., Hooi, B., Zimmermann, R.: When do contrastive learning signals help spatio-temporal graph forecasting? In: SIGSPATIAL (2022)
25. Liu, X., et al.: Largest: a benchmark dataset for large-scale traffic forecasting. In: NeurIPS (2023)
26. Lv, Y., Duan, Y., Kang, W., Li, Z., Wang, F.Y.: Traffic flow prediction with big data: a deep learning approach. *TITS* **16**, 865–873 (2014)
27. Manibardo, E.L., Laña, I., Del Ser, J.: Deep learning for road traffic forecasting: does it make a difference? *TITS* **23**, 6164–6188 (2021)
28. Oord, A.V.D., et al.: Wavenet: a generative model for raw audio. In: The 9th ISCA Speech Synthesis Workshop (2016)
29. Pan, B., Demiryurek, U., Shahabi, C.: Utilizing real-world transportation data for accurate traffic prediction. In: ICDM (2012)
30. Shao, Z., et al.: Decoupled dynamic spatial-temporal graph neural network for traffic forecasting. In: VLDB (2022)
31. Shuman, D.I., Narang, S.K., Frossard, P., Ortega, A., Vandergheynst, P.: The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Process. Maga.* **30**, 83–98 (2013)
32. Song, C., Lin, Y., Guo, S., Wan, H.: Spatial-temporal synchronous graph convolutional networks: a new framework for spatial-temporal network data forecasting. In: AAAI (2020)
33. Song, X., Kanasugi, H., Shibasaki, R.: Deeptransport: prediction and simulation of human mobility and transportation mode at a citywide level. In: IJCAI (2016)
34. Toda, H.: *Vector Autoregression and Causality*. Yale University, New Haven (1991)
35. Vaswani, A., et al.: Attention is all you need. In: NeurIPS (2017)
36. Wu, Z., Pan, S., Long, G., Jiang, J., Zhang, C.: Graph wavenet for deep spatial-temporal graph modeling. In: IJCAI (2019)
37. Yao, H., et al.: Deep multi-view spatial-temporal network for taxi demand prediction. In: AAAI (2018)
38. Ye, J., et al.: Learning the evolutionary and multi-scale graph structure for multi-variate time series forecasting. In: KDD (2022)
39. Yi, X., Zhang, J., Wang, Z., Li, T., Zheng, Y.: Deep distributed fusion network for air quality prediction. In: KDD (2018)

40. Yu, B., Yin, H., Zhu, Z.: Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. In: IJCAI (2018)
41. Yu, R., Li, Y., Shahabi, C., Demiryurek, U., Liu, Y.: Deep learning: a generic approach for extreme condition traffic forecasting. In: SDM (2017)
42. Zhang, S., Liu, Y., Sun, Y., Shah, N.: Graph-less neural networks: teaching old MLPS new tricks via distillation. In: ICLR (2022)
43. Zhang, S., Yao, L., Sun, A., Tay, Y.: Deep learning based recommender system: a survey and new perspectives. *ACM Comput. Surv.* **52**, 1–38 (2019)
44. Zheng, C., Fan, X., Wang, C., Qi, J.: GMAN: a graph multi-attention network for traffic prediction. In: AAAI (2020)
45. Zheng, Y., Capra, L., Wolfson, O., Yang, H.: Urban computing: concepts, methodologies, and applications. *TIST* **5**, 1–55 (2014)



Direct-Effect Risk Minimization for Domain Generalization

Yuhui Li¹, Zejia Wu², Chao Zhang¹, and Hongyang Zhang²(✉)

¹ Peking University, Beijing, China

yuhui.li@stu.pku.edu.cn, c.zhang@pku.edu.cn

² University of Waterloo, Waterloo, Canada

zejia.woo@gmail.com, hongyang.zhang@uwaterloo.ca

Abstract. We study the problem of out-of-distribution (o.o.d.) generalization where spurious correlations of attributes vary across training and test domains. This is known as the problem of correlation shift and has posed concerns on the reliability of machine learning. In this work, we introduce the framework of direct and indirect effects from causal inference to the domain generalization problem. Models that learn direct effects minimize the worst-case risk across correlation-shifted domains. To eliminate the indirect effects, our algorithm consists of two stages: in the first stage, we learn an indirect-effect representation by minimizing the prediction error of domain labels using the representation and the class labels; in the second stage, we remove the indirect effects learned in the first stage by matching each data with another data of similar indirect-effect representation but of different class labels in the training and validation phase. Our approach is shown to be compatible with existing methods and improve the generalization performance of them on correlation-shifted datasets. Experiments on 5 correlation-shifted datasets and the DomainBed benchmark verify the effectiveness of our approach. Our code is available at <https://github.com/Liyuhui-12/DRMforDG>.

Keywords: Domain generalization · Out-of-distribution generalization · Transfer learning

1 Introduction

Machine learning has achieved huge success in many fields, yet they mostly rely on the independent and identically distributed (i.i.d.) assumption. When it comes to an out-of-distribution (o.o.d.) test domain, machine learning models usually suffer from a sharp performance drop [2, 4, 31]. The o.o.d. data typically come in the form of *correlation shift*, where spurious correlations of attributes vary between training and test domains, or *diversity shift*, where the shifted test

Z. Wu—Work done during an internship at the University of Waterloo.

Y. Li, Z. Wu—Equal contribution.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024
A. Bifet et al. (Eds.): ECML PKDD 2024, LNAI 14943, pp. 39–57, 2024.

https://doi.org/10.1007/978-3-031-70352-2_3

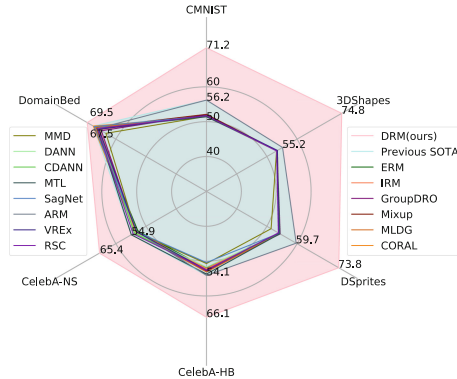


Fig. 1. Test accuracy of o.o.d. algorithms on 5 correlation shift datasets and the DomainBed benchmark (avg). The pink region shows the performance of our method, and the light blue indicates the prior best results implemented by DomainBed using *training-domain validation* (Color figure online).

distribution keeps the semantic content of the data unchanged while altering the data style [50]. The focus of this work is on the former setting known as correlation shift. That is, given stable causality and spurious correlations between attributes, how to disentangle the stable causality and the spurious correlations from the training data. Figure 1 shows the performance gain of our method on the correlation shift datasets.

Much effort has been devoted to learning representations that are invariant across training environments, where many works have introduced the tools from causality to address the o.o.d. generalization problems. When the data are of high dimension and multiple attributes are entangled, it is challenging to identify invariant causality across domains. Many methods have been designed to resolve the issue. Representative methods include incorporating invariance constraints by designing new loss functions [2, 5, 18, 24], learning latent semantic features in causal graphs by VAE [25, 27], and eliminating selection bias by matching [29, 47]. However, these methods, despite their theoretical guarantees, fail to show empirical improvement over Empirical Risk Minimization (ERM) as verified by the DomainBed benchmark [15, 45].

This paper uses the tool of *direct* and *indirect effects* from causal inference to analyze the correlation shift problem. Many existing works indicate that under certain conditions, models that learn stable direct causal effects minimize the worst-case risk across domain-shifted domains. To learn the direct effects, we propose a two-stage approach: in the first stage, we use an extractor to infer the indirect-effect representation \hat{Z} from the data X such that \hat{Z} and the class Y can predict the domain label E through a discriminator head (see the blue box in Fig. 2). In the second stage, we construct *balanced batches* by augmenting the original training batches and the validation set with data of the same indirect-effect representation \hat{Z} but of a different class Y . We test our approach

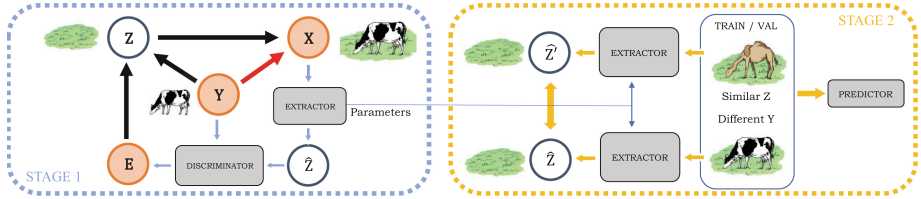


Fig. 2. Description of our two-stage approach. In **Stage 1**, we jointly learn a discriminator and an (indirect-effect) extractor by predicting the domain labels E . In **Stage 2**, the extractor in Stage 1 is used to construct a balanced batch of samples with a similar indirect-effect representation (\hat{Z} and \hat{Z}') but different class labels Y , as well as a balanced validation set which have no access to the test domain. A predictor is trained on the balanced batch and validated on the balanced validation set to predict the class labels. The flow of our method is represented by the blue and the yellow arrows. The red and black arrows form the graphical model of the correlation shift problem (data generation process). The red arrows represent the direct effect and the black arrows represent the indirect effect.

on the DomainBed benchmark. On the correlation shift dataset *Colored MNIST*, our model obtains an average accuracy of 71.2% over three domain generalization problems. While the information-theoretic best accuracy on the *Colored MNIST* dataset is 75%, our method achieves an accuracy as high as 69.7% in the most difficult “-90%” environment. We also demonstrate that our validation balancing approach can overcome the inconsistency of the validation distribution with the test distribution, which was shown to be an important reason for the performance degradation of many existing approaches under the DomainBed protocol. Moreover, we provide evidence that the foundation models can alleviate the diversity shift problem but cannot solve the correlation shift problem well, demonstrating that our approach closes the gap between foundation models and domain generalization to some extent. Our method can be combined with existing domain generalization methods to significantly improve their performance on correlation shift datasets. Our main contributions are as follows:

- We present a framework to analyze the correlation shift problem based on direct/indirect causal effects.
- We propose a new approach to improve o.o.d. generalization by alleviating the correlation shift problem. We recover the indirect-effect representation and eliminate the indirect effect during training and validation. We also show that our model selection method can largely overcome the model selection problem caused by the inconsistency between the validation and the test distribution. Our approach can be easily compatible with other algorithms and substantially improve their performances.

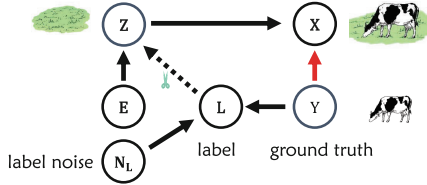


Fig. 3. The causal graph of correlation shift problems. The red arrow represents the stable direct effect, while the $Y \rightarrow L \rightarrow Z \rightarrow X$ pathway represents the indirect effects that changes as the environment changes, which is what we want to remove. (Color figure online)

- Our method outperforms baselines by a large margin on the correlation-shifted datasets. For example, our approach achieves up to 15% absolute improvement on the *Colored MNIST* dataset and up to 11% absolute improvement on the *CelebA* datasets over the state-of-the-art in terms of average accuracy over three domains.

2 Preliminaries

Notations. We use *capital* letters (X, Y, Z) for random variables, *lower-case* (x, y, z) for their realizations, and *hat* (\hat{Z}) for model-inferred variables. The *calligraphic capital* \mathcal{E} signifies the set of environments, while *lower-case* e denotes domain labels. $X \perp\!\!\!\perp Y$ indicates X and Y are independent. \mathcal{D}^e and \mathbb{P}^e represent the distribution and its probability density function (PDF) in environment e . Variables with the superscript e (e.g., x^e) are sampled from the distribution of environment e , and (x_i^e, l_i^e, e) refers to an instance sampled from \mathcal{D}^e , in which l_i^e is the class label that may have been affected by the noise. We denote by \mathcal{H} the hypothesis class of models, by $h : \mathcal{X} \rightarrow \mathcal{Y}$ the predictor, and by $R^e(h)$ its risk in environment e . We use *environment* and *domain* interchangeably.

2.1 Correlation Shift

We consider the correlation shift problem in this paper. In a supervised learning setting, the goal is to learn the labeling function $f : \mathcal{X} \rightarrow \mathcal{Y}$, which is consistent in all environments. However, there often exists a variable set \mathcal{Z} such that there are spurious correlations between $Z \in \mathcal{Z}$ and the ground truth Y . When the spurious correlation changes with the environment, the model that utilizes the spurious correlation may face a performance breakdown in the new test environment. Spurious correlations may originate from the data generation process or selection bias, which is very common in reality. We define the correlation shift as follow, which is consistent with that in [50].

Definition 1 (Correlation shift). Assume that we have a training environment $e_S \in \mathcal{E}_{train}$ and a test environment $e_T \in \mathcal{E}_{test}$, whose probability density

functions are \mathbb{P}^{e_S} and \mathbb{P}^{e_T} , respectively. Assume that $\mathbb{P}^{e_S}(y) = \mathbb{P}^{e_T}(y)$ for every $y \in \mathcal{Y}$, and that e_S and e_T share the same labeling function. Then there exists correlation shift between e_S and e_T if there exists a set \mathcal{Z} such that

$$\sum_{y \in \mathcal{Y}} \int_{\mathcal{Z}} |\mathbb{P}^{e_S}(z|y) - \mathbb{P}^{e_T}(z|y)| dz \neq 0,$$

where $\mathbb{P}^{e_S}(z) \times \mathbb{P}^{e_T}(z) \neq 0$.

By definition, we consider a direct acyclic graph (DAG) describing the data generation process with label noise (see Fig. 3), where the pathways from Y to X are composed of two parts: $Y \rightarrow X$ and $Y \rightarrow L \rightarrow Z \rightarrow X$. In causal inference, the former is referred to as the *direct effect* of Y on X , while the latter is referred to as the *indirect effect*. Z is a child of the environment E , which leads to the indirect effects varying with the environment. In a supervised learning setting, models are trained to learn the reversed process of the data generation process, i.e. $X \rightarrow Y$ or $X \rightarrow Z \rightarrow L \rightarrow Y$. Many existing theoretical works (such as [10]) have demonstrated that to obtain models that can generalize across different domains under correlation shift, we need to cut off the indirect effect pathway and force the model to learn the reversed mapping of the robust direct effects.

It’s important to note that typically, there is only a correlation between the ground truth Y and the mediator Z . For instance, the presence of a cow does not imply the presence of grass. Yet, in the view of models, this correlation can be seen as an indirect causal effect without human knowledge. With this view, all confounders between Y and X are captured by Z , eliminating any unblocked backdoor pathways [34] between them. Our data generation process description under correlation shifts and the DAG align with recent studies such as [47].

An example. Consider a binary classification problem of cows and camels (see Fig. 3). We assume that the animal category Y and background Z are the two attributes that contribute to the generation of an image X . Our goal is to predict the animal category Y from image X . The image X is the result of the total effect of the two attributes. We assume that the value of Y is changed from “camel” to “cow” during the data generation process. So the animal in the image X is changed to “cow” (the red arrow in Fig. 3). Meanwhile, the cow is more likely to be on the grass, while the camel is more likely to be in the desert. Therefore, Z may change from “desert” to “grass” as the animal category Y and the label L changes, changing the background in the image X . This process is represented by the $Y \rightarrow L \rightarrow Z$ path in Fig. 3, where the class label L is just the ground truth Y that has been affected by the label noise N_L .

2.2 Problem Setting

We consider a standard domain generalization setting, where the data come from different environments $e \in \mathcal{E}_{all}$. Assume that we have the training data collected from a finite subset of training environments \mathcal{E}_{train} , where $\mathcal{E}_{train} \subset \mathcal{E}_{all}$. For every environment $e \in \mathcal{E}_{train}$, the training dataset $\{(x_i^e, l_i^e, e)\}_{i=1}^{N_e}$ is sampled

from the distribution \mathcal{D}^e , where N_e is the number of training data in environment e . The PDF of the distribution is $\mathbb{P}^e(X^e, L^e) = \mathbb{P}(X, L \mid E = e)$, where X is the instance (e.g., an image), L is the class label, and E is the domain label. The goal of domain generalization is to train a model with data from training environments \mathcal{E}_{train} that generalizes well to all environments $e \in \mathcal{E}_{all}$. Our goal is to find a predictor $h^* : \mathcal{X} \rightarrow \mathcal{Y}$ in the hypothesis class \mathcal{H} such that the worst-case risk is minimized:

$$h^* = \operatorname{argmin}_{h \in \mathcal{H}} \max_{e \in \mathcal{E}_{all}} R^e(h), \quad (1)$$

where $R^e(h)$ is the risk of predictor h in environment e .

Many existing works, such as [10], focus on theoretically analyzing the relationship between causal classifiers and worst-case risk, which indicate that the model learning the stable direct effects is robust when the environment changes, i.e. a minimax solution of Eq. 1. Built upon these theoretical analyses, our work focuses on how to enable the model to learn the direct effects in the data. It is desirable to cut off the pathway between Z and L so that they are independent. To this end, we designed a novel framework with improved training process and model selection.

3 Method

3.1 Recovering Indirect Effects

Since the variable Z on the indirect-effect pathway is often not observable, we design an extractor to recover the representation \hat{Z} of the indirect effect from X by learning a discriminator head in the first stage. From Fig. 3, we observe that the indirect-effect representation Z and the class label L form a Markov blanket for the domain label E , which means that E is independent of other variables given L and Z . Hence the discriminator head needs Z and L to predict E . If we take the output of the extractor and L as the input of the discriminator head, the discriminator head will force the extractor to recover Z from X . Specifically, assume that the dataset is sampled from N_S training domains. We set up an extractor $G(\cdot; \Theta_G) : \mathcal{X} \rightarrow \mathcal{Z}$ and a discriminator head $D(\cdot, \cdot; \Theta_D) : \mathcal{Z} \times \mathcal{L} \rightarrow [0, 1]^{N_S}$ that outputs the probability that a sample belongs to each training domain, and update the parameters of both models by minimizing the prediction error of domain label e :

$$\Theta_G^*, \Theta_D^* := \operatorname{argmin}_{\Theta_G, \Theta_D} \mathbb{E}_{x, y, e} \operatorname{CE}(D(G(x; \Theta_G), l; \Theta_D), e), \quad (2)$$

where CE is the Cross Entropy loss, Θ_G and Θ_D stand for the parameters of the extractor G and the discriminator head D , respectively, and (x, l, e) is a training sample. We use the learned extractor to obtain the representation $\hat{z}_i^e = G(x_i^e; \Theta_G^*)$ for every instance (x_i^e, l_i^e, e) .

Many methods learned domain discriminators by a minimax problem [1, 14, 22]. These methods extracted features that could maximize the domain discriminator error. In our approach, on the other hand, the representation vector \hat{Z} is obtained by minimizing the domain discrimination error. This makes

Table 1. The changes of joint distribution of Z and Y after balancing in the “80%” domain of CMNIST.

Y	Z	Before Balancing	After Balancing
1	1	0.4	$\frac{0.4+0.1\alpha}{1+\alpha}$
1	0	0.1	$\frac{0.1+0.4\alpha}{1+\alpha}$
0	1	0.1	$\frac{0.1+0.4\alpha}{1+\alpha}$
0	0	0.4	$\frac{0.4+0.1\alpha}{1+\alpha}$

our model easier to optimize and more stable than a minimax game. Moreover, many other methods predict the domain label directly from the image, while our method uses a carefully designed order of inputting images x and class labels l , leveraging the Markov blanket property. This design enables us to extract the \hat{Z} that truly represents Z and successfully reduce the correlation between Z and Y , as shown in Fig. 5 and Fig. 8, thereby leading to performance improvements.

3.2 Eliminating Indirect Effects in Training (TB)

In the model training stage, we remove the indirect effects from the data by creating balanced batches based on the representation \hat{Z} , which is referred to as **TB** in the following. We start by defining the balanced batch.

Definition 2 (Balanced Batch). *For any sample in a balanced batch, denoted by $(x_i^e, l_i^e, e, \hat{z}_i^e)$, there exists a corresponding sample $(x_j^e, l_j^e, e, \hat{z}_j^e)$ with probability P , such that $\hat{z}_i^e = \hat{z}_j^e$, $l_i^e \neq l_j^e$, and $\mathbb{P}_{Batch}(L) = \mathbb{P}_D(L)$, where $\mathbb{P}_{Batch}(L)$ and $\mathbb{P}_D(L)$ are marginal probability density functions of L in the batch and the training set, respectively.*

Ideally, for each sample x_i , we can find a corresponding sample x_j with the same indirect-effect representation $\hat{z}_i^e = \hat{z}_j^e$. However, we cannot always find exactly equal \hat{z} as in the ideal case. To resolve this problem, for each sample $(x_i^e, l_i^e, e, \hat{z}_i^e)$, we search for another sample $(x_j^e, l_j^e, e, \hat{z}_j^e)$ such that \hat{z}_j^e is the nearest neighbor of \hat{z}_i^e . To ensure that the marginal distribution of label L does not change, we include the matched sample into the batch with a probability that depends on the proportion of each class of samples in the training set.

Taking the cow and camel classification problem in Fig. 3 as an example, for each cow image in the batch, we search for a camel image with the same background, e.g., a camel standing on grass for a cow standing on grass, as well as a cow image with the same background for each camel image. Thus our training batches consist of pairs of images. We train the model on balanced batches constructed as described above.

A Toy Example. “80%” domain in CMNIST can be a simple example to intuitively explain why the balancing can reduce spurious correlations. We denote by $Y \in \{0, 1\}$ the label and by $Z \in \{0, 1\}$ the color. Assuming that a proportion α of the samples are successfully matched with their corresponding samples,

Algorithm 1. Direct-Effect Risk Minimization (DRM)

Input: Training set D ; validation set V ; initial predictor f_{θ_0} ; training steps T ; checkpoint frequency C ; learning rate ϵ ;

Output: Predictor f_{θ_T} ;

1: Update Θ_G, Θ_D and get Θ_G^*, Θ_D^* by the following equation:

$$\Theta_G^*, \Theta_D^* := \underset{\Theta_G, \Theta_D}{\operatorname{argmin}} \mathbb{E}_{x,y,e} \operatorname{CE}(D(G(x; \Theta_G), l; \Theta_D), e);$$

2: $V_b \leftarrow \{\}$; # balanced validation set
3: **for** (x^e, l^e) in V **do**
4: $\hat{z}^e \leftarrow G(x^e; \Theta_G^*)$;
5: Search V for (x_b^e, l_b^e) with the closest \hat{z}_b^e to \hat{z}^e , and $l^e \neq l_b^e$;
6: Add (x^e, l^e) , (x_b^e, l_b^e) to V_b ;
7: **end for**
8: $t \leftarrow 0$;
9: **while** $t \leq T$ **do**
10: Sample a batch $B = \{(x_i^e, l_i^e)\}_{i=1}^{\text{batchsize}}$ from D ;
11: **for** (x^e, l^e) in batch **do**
12: $\hat{z}^e \leftarrow G(x^e; \Theta_G^*)$;
13: Search D for (x_b^e, l_b^e) with the closest \hat{z}_b^e to \hat{z}^e , and $l^e \neq l_b^e$;
14: Add (x_b^e, l_b^e) to B ;
15: **end for**
16: Run ERM or other algorithms on B and update f_{θ_t} ;
17: **if** $C|t$ **then**
18: Evaluate model on balanced validation set V_b ;
19: **end if**
20: $t \leftarrow t + 1$;
21: **end while**

i.e., those with the same Z but a different Y . The joint distribution of Z and Y changes as shown in the following Table 1. Before balancing, it can be easily calculated that the Pearson correlation coefficient (PCC) $\rho_{Z,Y} = 0.6$, while the post-balancing PCC becomes:

$$\rho_{Z,Y} = \frac{\operatorname{Cov}(Z, Y)}{\sigma_Z \sigma_Y} = \frac{\frac{0.4+0.1\alpha}{1+\alpha} - 0.25}{0.25} = 0.6 \times \frac{1-\alpha}{1+\alpha}.$$

The value of the correlation coefficient monotonically decreases when $\alpha \in [0, 1]$. Therefore, given that our algorithm has successfully extracted the representation of color \hat{Z} in the first stage, the spurious correlation decreases as the proportion of successful matches increases. We demonstrate in Fig. 5 that in real-world datasets, balancing can actually reduce spurious correlations.

3.3 Model Selection (VB)

DomainBed [15] highlights that using random hyperparameter search affects method performance, particularly for correlation-shifted datasets. Researchers

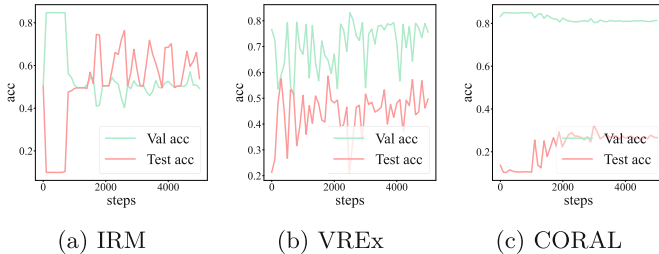


Fig. 4. The inconsistency of validation set accuracy and test set accuracy during the training process. (a) is for IRM, (b) is for VREx, and (c) is for CORAL.

should disclaim oracle-selection results and restrict access to the test domain. In domain generalization, training and testing domain distributions differ notably, and there is a large performance gap between selecting a model on the test and the training domain distribution. Figure 4 demonstrates that training-domain validation often leads to inconsistent validation and test accuracies. High-validation-accuracy checkpoints might underperform on the test set. The random hyperparameter search implemented by DomainBed avoids test domain access, so its reported results for some methods might be lower. A key reason for this performance drop, we postulate, is the misleading spurious correlation in the validation set, since models relying on this cannot generalize to test domains with opposite correlations.

Given the inconsistency, our DRM framework incorporates a novel model selection method using the mentioned balancing approach, termed **VB** in subsequent sections. We generate balanced validation sets as previously described. The validation data, sourced from training domains, aligns with the training-domain validation protocol of DomainBed, ensuring no model access to the test set.

In our primary results (Table 2), we run ERM on balanced batches and evaluate the models on balanced validation sets (ERM+VB+TB). Notably, our VB and TB methods can be integrated with numerous existing methods, enhancing their performance. We show these results in Table 3. Our experiments adhered to the DomainBed protocol, and for fairness, we compared DRM only with methods adhering to the same protocol without test domain access.

The pseudo-code description of the whole DRM framework is shown in Algorithm 1.

4 Experiments

We compare DRM with 16 baseline methods: ERM [44], IRM [2], GroupDRO [40], Mixup [49, 51], MLDG [20], CORAL [43], MMD [22], DANN [14], CDANN [23], MTL [6], SagNet [32], ARM [53], VREx [18], RSC [17], CAD & CondCAD [39], CausIRL with CORAL or MMD [9], EQRM [12], SWAD [8], and EOA [3],

sourced from the DomainBed benchmark [15]. We assess DRM on correlation-shifted datasets categorized by [48, 50], with spurious correlations between class labels and features, e.g., image background. Generalizing to the test domain can be challenging due to possible spurious-correlation flips between training and testing. While i.i.d. algorithms like ERM significantly drop in such cases, DRM demonstrates enhanced performance. Adhering to DomainBed’s protocol in all stages, we conducted random hyperparameter searches and utilized DomainBed’s codebase for method evaluations. Further experiment details are in the Appendix.

4.1 Datasets

We evaluate our approach on *CMNIST* dataset, *3DShapes* dataset, *DSprites* dataset, and *CelebA* dataset, which are common correlation shift datasets used to evaluate domain generalization methods [40, 48, 50].

Colored MNIST [2, 19] introduces a spurious correlation between colors and digits, using red or green coloring. The color-label correlations in three environments are +90%, +80%, and −90%. For instance, in the +90% environment, 90% of images labeled 1 are red, while 90% labeled 0 are green. The dataset also flips 25% of class labels, leading to a 75% shape-label correlation, which is less than the color-label correlation. Thus, an i.i.d. learning method like ERM is biased towards learning color-label correlations. For a broader and realistic evaluation, we introduce another four correlation shift datasets: *3DShapes* [7], *DSprites* [30], *CelebA-HB*, and *CelebA-NS* [26]. Their stable and spurious features are “floor hue” and “orientation”, “Position X” and “Position Y”, “No Beard” and “Wearing Hat”, and “Smiling” and “Wearing Necktie”, respectively. Dataset specifics are in the Appendix.

While focusing on the correlation shift, we also experiment on the diversity-shifted datasets in DomainBed, such as *PACS* [21] and *Office-Home (O-H)* [46], to confirm that DRM does not hurt model performance on such datasets. We regard diversity shift as a distinct domain generalization problem from correlation shift, aligning with [50]. For instance, a diversity shift example is when training images are art paintings and cartoons, but test images are photos. Our results indicate that ERM faces greater performance drops on correlation-shifted datasets than on diversity-shifted ones in the o.o.d. scenario, suggesting different remedy approaches. While foundation models trained with vast data perform well in the latter case, our paper offers a solution for the former.

Notably, the domain shift type is not tied to whether a dataset is real-world. Both *CelebA-NS* and *CelebA-HB* in our paper are real-world datasets with correlation shifts. Such shifts often stem from selection bias in real-world settings, as simulated by the *CelebA* datasets. However, due to shuffling of datasets in research, the presence of this selection bias is often overlooked, despite its significance.

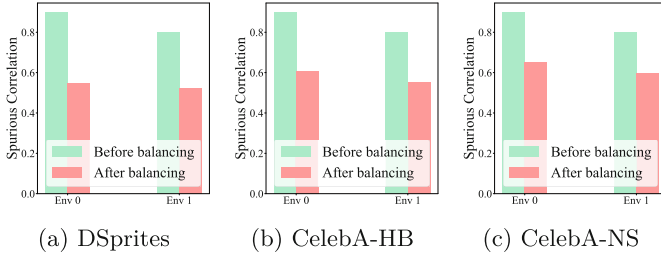


Fig. 5. The spurious correlation before & after balancing.

4.2 Results

Table 2 shows the performance of our approach under correlation shift. Following the DomainBed protocol, ERM and domain generalization algorithms reported by DomainBed exhibit a notable performance decline during tests with reversed correlation compared to training. Their accuracies align closely, mirroring *CMNIST* results in DomainBed [15]. For *CMNIST*, where the information-theoretic optimal accuracy is 75% given the 25% noise, ERM and other algorithms achieve no more than 10.5% in the challenging “−90%” setting—significantly below random guessing. Conversely, our DRM method reaches 69.7% accuracy, surpassing others by almost 60%. Additionally, our method maintains its effectiveness in the “+90%” and “+80%” domains. On average, DRM surpasses ERM by 20% and the leading prior method by 15%. This performance remains consistent across other datasets. Our method leads by roughly 50% in the toughest domain and offers over 15% improvement for correlation shift problems.

Results Interpretation. We attribute the enhanced performance of DRM to its capacity to substantially reduce spurious correlations in both training and validation sets, even on more realistic datasets, as illustrated in Fig. 5. Figure 6 presents eight examples of balanced *CMNIST* image pairs. In these pairs, a label-1 image with a red background pairs with a label-0 image with a red background, and a label-0 image with a green background pairs with a label-1 image with a green background. This design diminishes the correlation between color and label. Consequently, models trained with balanced data exhibit improved o.o.d. generalization capabilities.

Ablation Study. Our approach, focused on enhancing the sampling phase, allows easy integration of training set balancing (TB) and validation set balancing (VB) with other algorithms. We examine the contributions of VB and TB in our approach, with results presented in Table 3. Both TB and VB significantly boost the o.o.d. performance of the original methods, underscoring their effectiveness and importance within our general framework for addressing

Table 2. Experimental results on the correlation-shifted datasets, where the experiments are run by following the DomainBed setting. **Min** and **Avg** are the minimum value and the average accuracy for all test environments, respectively. The benchmark results on *CMNIST* are copied from DomainBed [15], while other benchmark results are obtained by running the DomainBed code.

Algorithm	Copied from [15]		Results Obtained Using the DomainBed Code [15]							
	CMNIST		3DShapes		DSprites		CelebA-HB		CelebA-NS	
	Min	Avg	Min	Avg	Min	Avg	Min	Avg	Min	Avg
ERM	10.0 ± 0.1	51.5 ± 0.1	10.1 ± 0.1	53.3 ± 0.1	13.8 ± 0.5	54.0 ± 0.1	16.8 ± 1.2	52.0 ± 0.5	21.1 ± 0.4	52.7 ± 0.5
IRM	10.2 ± 0.3	52.0 ± 0.1	10.0 ± 0.0	53.2 ± 0.1	14.5 ± 0.3	54.0 ± 0.1	20.4 ± 2.1	52.1 ± 0.7	21.5 ± 0.9	53.2 ± 0.4
GroupDRO	10.0 ± 0.2	52.1 ± 0.0	10.5 ± 0.4	53.4 ± 0.1	15.0 ± 0.4	54.4 ± 0.2	18.3 ± 1.5	52.8 ± 0.9	21.2 ± 0.2	53.3 ± 0.2
Mixup	10.1 ± 0.1	52.1 ± 0.2	10.2 ± 0.1	53.4 ± 0.2	14.0 ± 0.3	53.9 ± 0.0	17.9 ± 3.4	52.4 ± 1.1	22.2 ± 1.5	53.7 ± 0.7
MLDG	9.8 ± 0.1	51.5 ± 0.1	10.1 ± 0.1	53.5 ± 0.1	14.3 ± 0.3	54.2 ± 0.1	20.0 ± 2.1	53.0 ± 0.7	22.7 ± 1.7	53.7 ± 0.6
CORAL	9.9 ± 0.1	51.5 ± 0.1	10.0 ± 0.0	53.3 ± 0.1	13.8 ± 0.2	53.9 ± 0.2	17.7 ± 1.6	52.4 ± 0.6	22.1 ± 1.1	53.4 ± 0.4
MMD	9.9 ± 0.3	51.5 ± 0.2	10.0 ± 0.1	53.2 ± 0.1	14.4 ± 0.0	51.4 ± 2.1	17.4 ± 1.8	50.7 ± 0.5	22.5 ± 0.6	53.3 ± 0.1
DANN	10.0 ± 0.0	51.5 ± 0.3	10.0 ± 0.0	53.3 ± 0.0	14.7 ± 0.3	54.1 ± 0.3	16.9 ± 1.7	51.7 ± 0.3	21.8 ± 1.5	53.7 ± 0.8
CDANN	10.2 ± 0.1	51.7 ± 0.1	10.0 ± 0.0	53.3 ± 0.1	14.4 ± 0.2	54.0 ± 0.1	18.6 ± 2.6	52.5 ± 0.6	22.5 ± 1.2	53.9 ± 0.4
MTL	10.5 ± 0.1	51.4 ± 0.1	10.1 ± 0.0	53.4 ± 0.1	14.8 ± 0.5	54.3 ± 0.1	23.5 ± 1.4	53.7 ± 0.6	27.6 ± 1.2	54.9 ± 0.3
SagNet	10.3 ± 0.1	51.7 ± 0.0	10.1 ± 0.1	53.4 ± 0.1	13.6 ± 0.1	54.0 ± 0.0	14.9 ± 0.9	50.4 ± 0.3	22.0 ± 0.6	53.1 ± 0.2
ARM	10.2 ± 0.0	56.2 ± 0.2	10.0 ± 0.0	55.2 ± 0.3	14.5 ± 0.6	59.7 ± 0.4	22.8 ± 2.3	54.1 ± 0.6	21.1 ± 1.4	53.0 ± 0.5
VREx	10.2 ± 0.0	51.8 ± 0.1	10.8 ± 0.3	53.5 ± 0.1	13.8 ± 0.3	53.9 ± 0.1	19.2 ± 1.9	52.5 ± 0.7	20.3 ± 0.4	53.2 ± 0.3
RSC	10.0 ± 0.2	51.7 ± 0.2	10.1 ± 0.1	53.2 ± 0.1	13.3 ± 0.2	53.8 ± 0.1	18.9 ± 1.1	52.5 ± 0.5	23.7 ± 0.8	54.3 ± 0.5
CAD	10.4 ± 0.1	51.9 ± 0.1	10.1 ± 0.0	53.3 ± 0.1	13.3 ± 0.2	54.1 ± 0.2	20.1 ± 2.3	52.1 ± 1.3	22.9 ± 0.8	53.7 ± 1.5
CondCAD	9.7 ± 0.1	51.9 ± 0.1	10.1 ± 0.1	53.3 ± 0.1	15.4 ± 0.2	54.3 ± 0.1	16.1 ± 0.4	51.7 ± 0.6	23.9 ± 0.5	54.1 ± 0.3
CausIRL _{CORAL}	10.0 ± 0.0	51.8 ± 0.1	10.1 ± 0.0	54.1 ± 0.1	15.8 ± 0.3	54.1 ± 0.1	17.8 ± 1.3	52.5 ± 0.6	22.3 ± 0.3	52.9 ± 0.2
CausIRL _{MMD}	10.3 ± 0.1	51.5 ± 0.1	10.1 ± 0.0	40.7 ± 1.8	44.4 ± 3.1	53.7 ± 2.0	18.4 ± 0.7	52.3 ± 0.3	23.5 ± 0.6	53.8 ± 0.5
EQRM	10.3 ± 0.1	52.3 ± 0.1	10.3 ± 0.0	53.1 ± 0.1	16.0 ± 0.6	54.2 ± 0.2	17.9 ± 1.3	52.2 ± 0.7	20.6 ± 0.2	53.1 ± 0.2
SWAD	13.1 ± 1.9	52.7 ± 0.6	10.1 ± 0.0	53.2 ± 0.1	15.0 ± 0.3	53.9 ± 0.1	18.7 ± 0.1	51.2 ± 0.8	22.9 ± 0.2	54.0 ± 0.2
EOA	10.1	51.0	10.0	53.1	16.8	54.3	14.4	52.5	21.5	53.9
DRM (ours)	69.7 ± 1.5	71.2 ± 0.6	74.5 ± 0.2	74.8 ± 0.1	73.3 ± 0.5	73.8 ± 0.2	61.0 ± 4.9	66.1 ± 0.6	59.9 ± 2.6	65.4 ± 1.2

correlation shift issues. VB, in particular, reveals potential in leveraging existing methods such as IRM, enabling them to substantially outperform ERM when VB is added. Employing both VB and TB further enhances the performance of existing methods. ERM performs well in this case because VB and TB have largely eliminated spurious correlations.

Accuracy Curves Analysis. Many domain generalization methods design new loss functions by incorporating invariant constraints, such as IRM and VREx, which leads to a very unstable training process. In contrast, as shown in Fig. 7, the fluctuation of the accuracy curve of our DRM in the training phase is relatively small. The validation accuracy of our method is basically consistent with the test accuracy, showing the same trend in the figure.

4.3 Foundation Models and O.o.d. Generalization

The significance of data scale is evident in domain generalization. We demonstrate that models trained with vast data perform well on diversity-shifted datasets, but show little improvement on correlation-shifted ones. We evaluate using the *Colored MNIST* and *CelebA-HB* for correlation shift, and *PACS* and *Office-Home* for diversity shift. Experiments with the large-scale pre-trained models CLIP [37] and EVA [13] were conducted through linear probing. Table 4

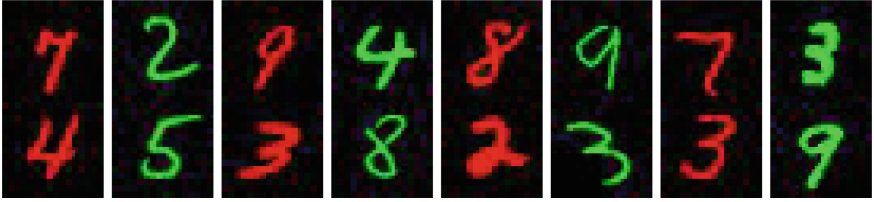


Fig. 6. Eight examples of balanced *CMNIST* image pairs.

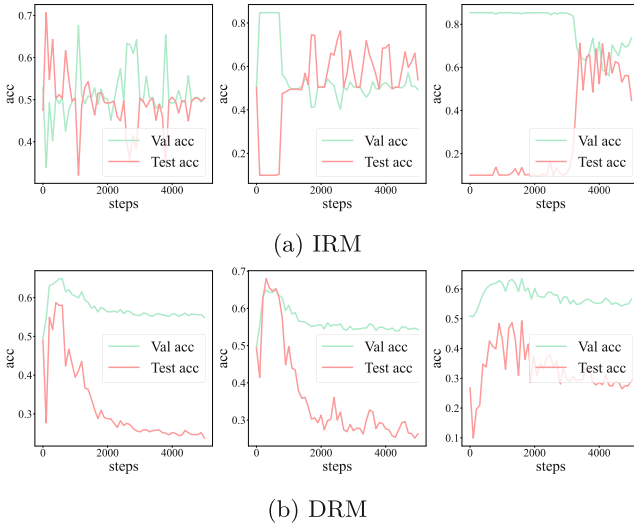


Fig. 7. Accuracy curves of IRM and our DRM with different random seed for hyper-parameter search.

reveals CLIP (ViT-L [11]) and EVA (ViT, 1B parameters) in the o.o.d. scenario surpass ResNet-50 in the i.i.d. scenario for *PACS* and *Office-Home*. This indicates increasing data scale, parameters, and refining model architecture can mitigate the diversity shift problem. Yet, they cannot generalize well in the correlation shift scenario. Therefore, we believe that our approach closes the gap between foundation models and domain generalization to some extent.

4.4 Visual Explanation

We visualize results using the *CelebA-HB* and *CelebA-NS* datasets. For *CelebA-HB*, the indirect effect is the pathway between the spurious feature and label, which is “Wearing Hat” and “No Beard”, respectively. For *CelebA-NS*, the indirect effect is the pathway between “Wearing Necktie” and “Smiling”.

Table 3. Ablation study for *CMNIST*, *DSprites*, and *CelebA-HB*. VB stands for balancing during validation, while TB stands for balancing during training.

Algorithm	CMNIST		DSprites		CelebA-HB	
	Min	Avg	Min	Avg	Min	Avg
ERM	10.0 ± 0.1	51.5 ± 0.1	13.8 ± 0.5	54.0 ± 0.1	16.8 ± 1.2	52.0 ± 0.5
ERM+VB	30.7 ± 1.1	58.2 ± 0.7	35.1 ± 2.2	61.1 ± 1.5	37.4 ± 1.6	59.1 ± 0.4
ERM+VB+TB	69.7 ± 1.5	71.2 ± 0.6	73.3 ± 0.5	73.8 ± 0.2	61.0 ± 4.9	66.1 ± 0.6
IRM	10.2 ± 0.3	52.0 ± 0.1	14.5 ± 0.3	54.0 ± 0.1	20.4 ± 2.1	52.1 ± 0.7
IRM+VB	40.1 ± 5.2	61.4 ± 1.7	50.1 ± 7.7	66.1 ± 3.2	47.4 ± 4.1	61.5 ± 1.6
IRM+VB+TB	67.6 ± 1.5	69.5 ± 0.4	71.3 ± 1.0	73.0 ± 0.6	65.7 ± 1.6	66.9 ± 0.3
VREx	10.2 ± 0.0	51.8 ± 0.1	13.8 ± 0.3	53.9 ± 0.1	19.2 ± 1.9	52.5 ± 0.7
VREx+VB	49.9 ± 5.0	65.1 ± 3.2	52.3 ± 7.9	66.8 ± 5.1	46.8 ± 2.3	61.4 ± 1.4
VREx+VB+TB	68.3 ± 0.3	70.6 ± 0.2	72.9 ± 0.4	73.6 ± 0.2	61.8 ± 3.0	65.8 ± 1.0
CORAL	9.9 ± 0.1	51.5 ± 0.1	13.8 ± 0.2	53.9 ± 0.2	17.7 ± 1.6	52.4 ± 0.6
CORAL+VB	33.2 ± 0.7	59.2 ± 0.3	28.8 ± 2.2	59.0 ± 1.3	47.3 ± 1.9	61.6 ± 0.9
CORAL+VB+TB	68.0 ± 0.8	70.7 ± 0.3	71.9 ± 1.0	73.3 ± 0.7	60.2 ± 4.6	64.8 ± 1.9
CDANN	10.2 ± 0.1	51.7 ± 0.1	14.4 ± 0.2	54.0 ± 0.1	18.6 ± 2.6	52.5 ± 0.6
CDANN+VB	45.9 ± 10.5	63.8 ± 4.6	48.7 ± 6.8	65.7 ± 3.2	48.4 ± 2.7	62.1 ± 1.2
CDANN+VB+TB	66.4 ± 0.7	69.2 ± 0.1	73.0 ± 0.1	73.7 ± 0.3	59.4 ± 3.1	65.6 ± 0.9

Analysis of Indirect Effect Representation. Using t-SNE [28], we reduce dimension of \hat{Z} to 2, as depicted in Fig. 8. Data points with identical spurious features cluster, indicating that \hat{Z} aptly represents the spurious feature, thus our methods can match samples correctly during training and validation.

Attention Map. In Fig. 9, we present the attention maps of the last convolution layer for ERM (the first row) and DRM (the second row). The model trained by ERM focuses on the spurious feature “Wearing Hat” and “Wearing Necktie”, while the model trained by DRM focuses on the stable feature “No Beard” and “Smiling”.

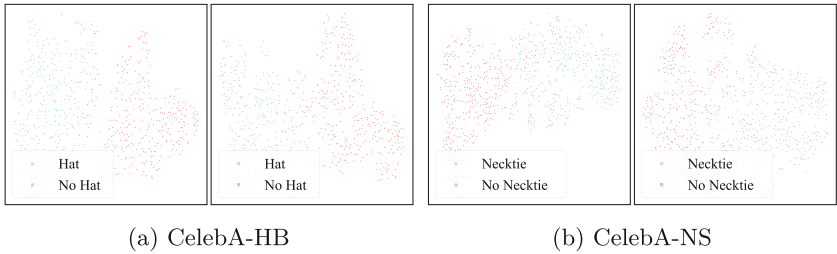
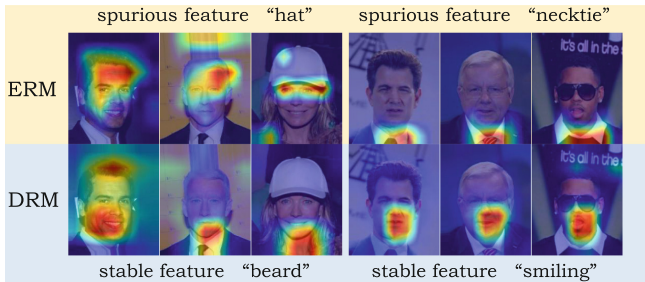
5 Related Works

Domain Generalization with Causality. Numerous studies have integrated causality inference tools into domain generalization. Causality has been shown to be robust across domains [35]. Some studies identify extractable causal factors [41, 42] and link causality to generalization [10]. [33] defined natural direct effects. Though this concept appeared in earlier works [16, 36, 52], no work analyzed domain generalization using this framework.

Matching Based Methods. Matching eliminates selection bias in causal inference by pairing similar instances [38]. [29, 47] presented unsupervised matching

Table 4. Results of different methods on typical correlation and diversity-shifted datasets. O-H represents *Office-Home*.

OOD	Method	Correlation Shift		Diversity Shift	
		CMNIST	CelebA-HB	PACS	O-H
✗	ERM (RN50)	86.6	83.6	96.3	80.4
✓	ERM (RN50)	51.5 (-35.1)	52.0 (-31.6)	85.5	66.5
✓	CLIP (RN50)	48.9 (-37.7)	55.8 (-27.8)	64.1	48.3
✓	CLIP (ViT-B)	52.2 (-34.4)	54.8 (-28.8)	95.7	82.3
✓	CLIP (ViT-L)	52.5 (-34.1)	53.8 (-29.8)	98.4	88.3
✓	EVA (ViT-1B)	51.3 (-35.3)	53.6 (-30.0)	98.7	90.7
✓	DRM (RN50)	71.2 (-15.4)	66.1 (-17.5)	84.8	65.7

**Fig. 8.** Visualization of 2D t-SNE result of Z . Different colors represent different spurious features, “Wearing Hat” vs. “No Hat”, or “Wearing Neckline” vs. “No Neckline”. Each subfigure represents the results of two training domains when “-90%” is the test domain, which is the most difficult to generalize.**Fig. 9.** Attention maps of ERM and our method, respectively. **The left half** is on the *CelebA-HB* dataset. **The right half** is on the *CelebA-NS* dataset.

and propensity score matching methods. Our method also involves mini-batch balancing (matching) in its second stage. However, in the first stage, we derive an indirect-effect representation through a domain discriminator and introduce

a balancing-based model selection method. These unique features enhance the efficacy of our method compared to previous approaches.

6 Conclusion

In this paper, we introduce direct and indirect effects from causal inference to domain generalization. We propose a method to extract the indirect-effect representation and remove the indirect effects during training. We also introduce a novel approach for model selection in the o.o.d. setting. Our approaches can be integrated with existing methods to enhance their performance substantially. Experimental results confirm that our approach attains SOTA performance.

Acknowledgements. Yuhui Li and Chao Zhang are supported by the National Nature Science Foundation of China under Grant 62071013 and National Key R&D Program of China under Grant 2018AAA0100300. Hongyang Zhang is supported by the NSERC Discovery Grant RGPIN-2022-03215, DGEGR-2022-00357.

References

1. Albuquerque, I., Monteiro, J., Darvishi, M., Falk, T.H., Mitliagkas, I.: Generalizing to unseen domains via distribution matching. arXiv preprint [arXiv:1911.00804](https://arxiv.org/abs/1911.00804) (2019)
2. Arjovsky, M., Bottou, L., Gulrajani, I., Lopez-Paz, D.: Invariant risk minimization. arXiv preprint [arXiv:1907.02893](https://arxiv.org/abs/1907.02893) (2019)
3. Arpit, D., Wang, H., Zhou, Y., Xiong, C.: Ensemble of averages: improving model selection and boosting performance in domain generalization. *Adv. Neural. Inf. Process. Syst.* **35**, 8265–8277 (2022)
4. Beery, S., Van Horn, G., Perona, P.: Recognition in terra incognita. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) *Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XVI*, pp. 472–489. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-030-01270-0_28
5. Bellot, A., van der Schaar, M.: Accounting for unobserved confounding in domain generalization. arXiv preprint [arXiv:2007.10653](https://arxiv.org/abs/2007.10653) (2020)
6. Blanchard, G., Lee, G., Scott, C.: Generalizing from several related classification tasks to a new unlabeled sample. In: *Advances in Neural Information Processing Systems*, vol. 24 (2011)
7. Burgess, C., Kim, H.: 3d shapes dataset. <https://github.com/deepmind/3dshapes-dataset/> (2018)
8. Cha, J., et al.: SWAD: domain generalization by seeking flat minima. *Adv. Neural. Inf. Process. Syst.* **34**, 22405–22418 (2021)
9. Chevalley, M., Bunne, C., Krause, A., Bauer, S.: Invariant causal mechanisms through distribution matching. arXiv preprint [arXiv:2206.11646](https://arxiv.org/abs/2206.11646) (2022)
10. Christiansen, R., Pfister, N., Jakobsen, M.E., Gnecco, N., Peters, J.: A causal framework for distribution generalization (2020). <https://doi.org/10.48550/ARXIV.2006.07433>, <https://arxiv.org/abs/2006.07433>
11. Dosovitskiy, A., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint [arXiv:2010.11929](https://arxiv.org/abs/2010.11929) (2020)

12. Eastwood, C., et al.: Probable domain generalization via quantile risk minimization. *Adv. Neural. Inf. Process. Syst.* **35**, 17340–17358 (2022)
13. Fang, Y., et al.: Eva: Exploring the limits of masked visual representation learning at scale. arXiv preprint [arXiv:2211.07636](https://arxiv.org/abs/2211.07636) (2022)
14. Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., Lempitsky, V.: Domain-adversarial training of neural networks. *J. Mach. Learn. Res.* **17**(1), 2030–2096 (2016)
15. Gulrajani, I., Lopez-Paz, D.: In search of lost domain generalization. In: *International Conference on Learning Representations* (2020)
16. Heskes, T., Sijben, E., Bucur, I.G., Claassen, T.: Causal shapley values: exploiting causal knowledge to explain individual predictions of complex models. *Adv. Neural. Inf. Process. Syst.* **33**, 4778–4789 (2020)
17. Huang, Z., Wang, H., Xing, E.P., Huang, D.: Self-challenging improves cross-domain generalization. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) *ECCV 2020. LNCS*, vol. 12347, pp. 124–140. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58536-5_8
18. Krueger, et al.: Out-of-distribution generalization via risk extrapolation (rex). In: *International Conference on Machine Learning*, pp. 5815–5826. PMLR (2021)
19. LeCun, Y.: The mnist database of handwritten digits. <http://yann.lecun.com/exdb/mnist/> (1998)
20. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.: Learning to generalize: meta-learning for domain generalization. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32 (2018)
21. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.M.: Deeper, broader and artier domain generalization. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5542–5550 (2017)
22. Li, H., Pan, S.J., Wang, S., Kot, A.C.: Domain generalization with adversarial feature learning. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5400–5409 (2018)
23. Li, Y., et al.: Deep domain generalization via conditional invariant adversarial networks. In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 624–639 (2018)
24. Lin, Y., Dong, H., Wang, H., Zhang, T.: Bayesian invariant risk minimization. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16021–16030 (2022)
25. Liu, C., et al.: Learning causal semantic representation for out-of-distribution prediction. *Adv. Neural. Inf. Process. Syst.* **34**, 6155–6170 (2021)
26. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: *Proceedings of International Conference on Computer Vision (ICCV)* (December 2015)
27. Lu, C., Wu, Y., Hernández-Lobato, J.M., Schölkopf, B.: Invariant causal representation learning for out-of-distribution generalization. In: *International Conference on Learning Representations* (2021)
28. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. *J. Mach. Learn. Res.* **9**(11) (2008)
29. Mahajan, D., Tople, S., Sharma, A.: Domain generalization using causal matching. In: *International Conference on Machine Learning*, pp. 7313–7324. PMLR (2021)
30. Matthey, L., Higgins, I., Hassabis, D., Lerchner, A.: dsprites: Disentanglement testing sprites dataset. <https://github.com/deepmind/dsprites-dataset/> (2017)

31. Nagarajan, V., Andreassen, A., Neyshabur, B.: Understanding the failure modes of out-of-distribution generalization. In: International Conference on Learning Representations (2020)
32. Nam, H., Lee, H., Park, J., Yoon, W., Yoo, D.: Reducing domain gap via style-agnostic networks. arXiv preprint [arXiv:1910.11645](https://arxiv.org/abs/1910.11645) **2**(7), 8 (2019)
33. Pearl, J.: Direct and indirect effects. Probabilistic and Causal Inference: The Works of Judea Pearl, p. 373 (2001)
34. Pearl, J.: Causality. Cambridge university press (2009)
35. Peters, J., Bühlmann, P., Meinshausen, N.: Causal inference by using invariant prediction: identification and confidence intervals. *J. Royal Stat. Society: Series B (Statistical Methodology)* **78**(5), 947–1012 (2016)
36. Qi, J., Niu, Y., Huang, J., Zhang, H.: Two causal principles for improving visual dialog. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10860–10869 (2020)
37. Radford, A., et al.: Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning, pp. 8748–8763. PMLR (2021)
38. Rosenbaum, P.R., Rubin, D.B.: The central role of the propensity score in observational studies for causal effects. *Biometrika* **70**(1), 41–55 (1983)
39. Ruan, Y., Dubois, Y., Maddison, C.J.: Optimal representations for covariate shift. In: International Conference on Learning Representations (2021)
40. Sagawa, S., Koh, P.W., Hashimoto, T.B., Liang, P.: Distributionally robust neural networks for group shifts: on the importance of regularization for worst-case generalization. arXiv preprint [arXiv:1911.08731](https://arxiv.org/abs/1911.08731) (2019)
41. Schölkopf, B., Janzing, D., Peters, J., Sgouritsa, E., Zhang, K., Mooij, J.: On causal and anticausal learning. arXiv preprint [arXiv:1206.6471](https://arxiv.org/abs/1206.6471) (2012)
42. Schölkopf, B., et al.: Toward causal representation learning. *Proc. IEEE* **109**(5), 612–634 (2021)
43. Sun, B., Saenko, K.: Deep CORAL: correlation alignment for deep domain adaptation. In: Hua, G., Jégou, H. (eds.) *Computer Vision – ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8–10 and 15–16, 2016, Proceedings, Part III*, pp. 443–450. Springer International Publishing, Cham (2016). https://doi.org/10.1007/978-3-319-49409-8_35
44. Vapnik, V.: *Statistical learning theory* wiley. New York **1**(624), 2 (1998)
45. Vedantam, R., Lopez-Paz, D., Schwab, D.J.: An empirical investigation of domain generalization with empirical risk minimizers. *Adv. Neural. Inf. Process. Syst.* **34**, 28131–28143 (2021)
46. Venkateswara, H., Eusebio, J., Chakraborty, S., Panchanathan, S.: Deep hashing network for unsupervised domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5018–5027 (2017)
47. Wang, X., Saxon, M., Li, J., Zhang, H., Zhang, K., Wang, W.Y.: Causal balancing for domain generalization. arXiv preprint [arXiv:2206.05263](https://arxiv.org/abs/2206.05263) (2022)
48. Wiles, O., Goyal, S., Stimberg, F., Alvisè-Rebuffi, S., Ktena, I., Cemgil, T., et al.: A fine-grained analysis on distribution shift. arXiv preprint [arXiv:2110.11328](https://arxiv.org/abs/2110.11328) (2021)
49. Xu, M., Zhang, J., Ni, B., Li, T., Wang, C., Tian, Q., Zhang, W.: Adversarial domain adaptation with domain mixup. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 6502–6509 (2020)
50. Ye, N., et al.: Ood-bench: Quantifying and understanding two dimensions of out-of-distribution generalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition pp. 7947–7958 (2022)

51. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: beyond empirical risk minimization. In: International Conference on Learning Representations (2018)
52. Zhang, J., Bareinboim, E.: Fairness in decision-making-the causal explanation formula. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
53. Zhang, M., Marklund, H., Gupta, A., Levine, S., Finn, C.: Adaptive risk minimization: A meta-learning approach for tackling group shift. **8** 9 (2020) [arXiv:2007.02931](https://arxiv.org/abs/2007.02931)



Federated Frank-Wolfe Algorithm

Ali Dadras^(✉), Sourasekhar Banerjee, Karthik Prakhya, and Alp Yurtsever

Umeå University, Umeå, Sweden

{ali.dadras,sourasekhar.banerjee,karthik.prakhya,alp.yurtsever}@umu.se

Abstract. Federated learning (FL) has gained a lot of attention in recent years for building privacy-preserving collaborative learning systems. However, FL algorithms for constrained machine learning problems are still limited, particularly when the projection step is costly. To this end, we propose a Federated Frank-Wolfe Algorithm (FEDFW). FEDFW features data privacy, low per-iteration cost, and communication of sparse signals. In the deterministic setting, FEDFW achieves an ε -suboptimal solution within $\mathcal{O}(\varepsilon^{-2})$ iterations for smooth and convex objectives, and $\mathcal{O}(\varepsilon^{-3})$ iterations for smooth but non-convex objectives. Furthermore, we present a stochastic variant of FEDFW and show that it finds a solution within $\mathcal{O}(\varepsilon^{-3})$ iterations in the convex setting. We demonstrate the empirical performance of FEDFW on several machine learning tasks.

Keywords: Federated learning · Frank-Wolfe · Conditional gradient method · Projection-free · Distributed optimization

1 Introduction

We present a new variant of the Frank-Wolfe (FW) algorithm, FEDFW, designed for the increasingly popular Federated Learning (FL) paradigm in machine learning. Consider the following constrained empirical risk minimization template:

$$\min_{\mathbf{x} \in \mathcal{D}} F(\mathbf{x}) := \frac{1}{n} \sum_{i=1}^n f_i(\mathbf{x}), \quad (1)$$

where $\mathcal{D} \subseteq \mathbb{R}^p$ is a convex and compact set. We define the diameter of \mathcal{D} as $D := \max_{\mathbf{x}, \mathbf{y} \in \mathcal{D}} \|\mathbf{x} - \mathbf{y}\|$. The function $F : \mathbb{R}^p \rightarrow \mathbb{R}$ represents the objective function, and $f_i : \mathbb{R}^p \rightarrow \mathbb{R}$ (for $i = 1, \dots, n$) represent the loss functions of the clients, where n is the number of clients. Throughout, we assume f_i is L -smooth, meaning that it has Lipschitz continuous gradients with parameter L .

FL holds great promise for solving optimization problems over a large network, where clients collaborate under the coordination of a server to find a

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70352-2_4.

common good model. Privacy is an explicit goal in FL; clients work together towards a common goal by utilizing their own data without sharing it. As a result, FL exhibits remarkable potential for data science applications involving privacy-sensitive information. Its applications range from learning tasks (such as training neural networks) on mobile devices without sharing personal data [1] to medical applications of machine learning, where hospitals collaborate without sharing sensitive patient information [2].

Most FL algorithms focus on unconstrained optimization problems, and extending these algorithms to handle constrained problems typically requires projection steps. However, in many machine learning applications, the projection cost can create a computational bottleneck, preventing us from solving these problems at a large scale. The FW algorithm [3] has emerged as a preferred method for addressing these problems in machine learning. The main workhorse of the FW algorithm is the linear minimization oracle (LMO),

$$\text{lmo}(\mathbf{y}) := \underset{\mathbf{x} \in \mathcal{D}}{\operatorname{argmin}} \langle \mathbf{y}, \mathbf{x} \rangle. \quad (2)$$

Evaluating linear minimization is generally less computationally expensive than performing the projection step. A famous example illustrating this is the nuclear-norm constraint: projecting onto a nuclear-norm ball often requires computing a full-spectrum singular value decomposition. In contrast, linear minimization involves finding the top singular vector, a task that can be efficiently approximated using methods such as the power method or Lanczos iterations.

To our knowledge, FW has not yet been explored in the context of FL. This paper aims to close this gap. Our primary contribution lies in adapting the FW method for FL with convergence guarantees.

The paper is organized as follows: Sect. 2 provides a brief review of the literature on FL and the FW method. In Sect. 3, we introduce FEDFW. Unlike traditional FL methods, FEDFW does not overwrite clients' local models with the global model sent by the server. Instead, it penalizes clients' loss functions by using the global model. We present the convergence guarantees of FEDFW in Sect. 3.1. Specifically, our method provably finds a ε -suboptimal solution after $\mathcal{O}(\varepsilon^{-2})$ iterations for smooth and convex objective functions (refer to Theorem 1). In the case of non-convex objectives, the complexity increases to $\mathcal{O}(\varepsilon^{-3})$ (refer to Theorem 2). Section 4 introduces several design variations of FEDFW, including a stochastic variant. Section 5 presents numerical experiments on various machine learning tasks with both convex and non-convex objective functions. Finally, Sect. 6 provides concluding remarks along with a discussion on the limitations of the proposed method. Detailed proofs and technical aspects are deferred to the supplementary material.

2 Related Work

Federated Learning. FL is a distributed learning paradigm that, unlike most traditional distributed settings, focuses on a scenario where only a subset of

clients participate in each training round, data is often heterogeneous, and clients can perform different numbers of iterations in each round [4, 5]. FEDAVG [4] has been a cornerstone in the FL literature, demonstrating practical capabilities in addressing key concerns such as privacy and security, data heterogeneity, and computational costs. Although it is shown that fixed points of some FEDAVG variants do not necessarily converge to the minimizer of the objective function, even in the least squares problem [6], and can even diverge [7], the convergence guarantees of FEDAVG have been studied under different assumptions (see [8–15] and the references therein). However, all these works on the convergence guarantees of FEDAVG are restricted to unconstrained problems.

Constrained or composite optimization problems are ubiquitous in machine learning, often used to impose structural priors such as sparsity or low-rankness. To our knowledge, FEDDR [16] and FEDDA [17] are the first FL algorithms with convergence guarantees for constrained problems. The former employs Douglas-Rachford splitting, while the latter is based on the dual averaging method [18], to solve composite optimization problems, including constrained problems via indicator functions, within the FL setting. [19] introduced a ‘fast’ variant of FEDDA, achieving rapid convergence rates with linear speedup and reduced communication rounds for composite strongly convex problems. FEDADMM [20] was proposed for federated composite optimization problems involving a non-convex smooth term and a convex non-smooth term in the objective. Moreover, [21] proposed a FL algorithm based on a proximal augmented Lagrangian approach to address problems with convex functional constraints. None of these works address our problem template, where the constraints are challenging to project onto but allow for an efficient solution to the linear minimization problem.

Frank-Wolfe Algorithm. The FW algorithm, also known as the conditional gradient method or CGM, was initially introduced in [3] to minimize a convex quadratic objective over a polytope, and was extended to general convex objectives and arbitrary convex and compact sets in [22]. Following the seminal works in [23, 24], the method gained popularity in machine learning.

The increasing interest in FW methods for data science applications has led to the development of new results and variants. For example, [25] established convergence guarantees for FW with non-convex objective functions. Additionally, online, stochastic, and variance-reduced variants of FW have been proposed; see [26–31] and the references therein. FW has also been combined with smoothing strategies for non-smooth and composite objectives [32–35], and with augmented Lagrangian methods for problems with affine equality constraints [36, 37]. Furthermore, various design variants of FW, such as the away-step and pairwise step strategies, can offer computational advantages. For a comprehensive overview of FW-type methods and their applications, we refer to [38, 39].

The most closely related methods to our work are the distributed FW variants. However, the variants in [40–42] are fundamentally different from FEDFW as they require sharing gradient information of the clients with the server or with the neighboring nodes. In FEDFW, clients do not share gradients, which

is critical for data privacy [43, 44]. Other distributed FW variants are proposed in [45–47]. However, the method proposed by [46] is limited to the convex low-rank matrix optimization problem, and the methods in [45, 47] assume that the problem domain is block separable.

3 Federated Frank-Wolfe Algorithm

In essence, any first-order optimization algorithm can be adapted for a simplified federated setting by transmitting local gradients to the server at each iteration. These local gradients can be aggregated to compute the full gradient and distributed back to the clients. Although it is possible to implement the standard FW algorithm in FL this way, this baseline has two major problems. First, it relies on communication at each iteration, which raises scalability concerns, as extending this approach to multiple local steps is not feasible. Secondly, sharing raw gradients raises privacy concerns, as sensitive information and data points can be inferred with high precision from transmitted gradients [43]. Consequently, most FL algorithms are designed to exchange local models or step-directions rather than gradients. Unfortunately, a simple combination of the FW algorithm with a model aggregation step fails to find a solution to (1), as we demonstrate with a simple counterexample in the supplementary material. Therefore, developing FEDFW requires a special algorithmic approach, which we elaborate on below.

We start by rewriting problem (1) in terms of the matrix decision variable $\mathbf{X} := [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]$, as follows:

$$\min_{\mathbf{X} \in \mathcal{D}^n} \frac{1}{n} \sum_{i=1}^n f_i(\mathbf{X}\mathbf{e}_i) + \delta_{\mathcal{C}}(\mathbf{X}). \quad (3)$$

Here, \mathbf{e}_i denotes the i th standard unit vector, and $\delta_{\mathcal{C}}$ is the indicator function for the consensus set:

$$\mathcal{C} := \{[\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathbb{R}^{p \times n} : \mathbf{x}_1 = \mathbf{x}_2 = \dots = \mathbf{x}_n\}. \quad (4)$$

It is evident that problems (1) and (3) are equivalent. However, the latter formulation represents the local models of the clients as the columns of the matrix \mathbf{X} , offering a more explicit representation for FL.

The original FW algorithm is ill-suited for solving problem (3) due to the non-smooth nature of the objective function because of the indicator function. Drawing inspiration from techniques proposed in [33], we adopt a quadratic penalty strategy to address this challenge. The main idea is to perform FW updates on a surrogate objective which replaces the hard constraint $\delta_{\mathcal{C}}$ with a smooth function that penalizes the distance between \mathbf{X} and the consensus set \mathcal{C} :

$$\hat{F}_t(\mathbf{X}) = \frac{1}{n} \sum_{i=1}^n f_i(\mathbf{X}\mathbf{e}_i) + \frac{\lambda_t}{2} \text{dist}^2(\mathbf{X}, \mathcal{C}), \quad (5)$$

where $\lambda_t \geq 0$ is the penalty parameter. Note that the surrogate function is parameterized by the iteration counter t , as it is crucial to amplify the impact of the penalty function by gradually increasing λ_t at a specific rate through the iterations. This adjustment will ensure that the generated sequence converges to a solution of the original problem in (3).

To perform an FW update with respect to the surrogate function, first, we need to compute the gradient of \hat{F}_t , given by

$$\begin{aligned} \nabla \hat{F}_t(\mathbf{X}) &= \frac{1}{n} \sum_{i=1}^n \nabla f_i(\mathbf{X}\mathbf{e}_i) \mathbf{e}_i^\top + \lambda_t (\mathbf{X} - \text{proj}_{\mathcal{C}}(\mathbf{X})) \\ &= \frac{1}{n} \sum_{i=1}^n \nabla f_i(\mathbf{x}_i) \mathbf{e}_i^\top + \lambda_t \sum_{i=1}^n (\mathbf{x}_i - \bar{\mathbf{x}}) \mathbf{e}_i^\top \end{aligned} \quad (6)$$

where $\bar{\mathbf{x}} := \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i$. Then, we call the linear minimization oracle:

$$\mathbf{S}^t \in \underset{\mathbf{X} \in \mathcal{D}^n}{\text{argmin}} \langle \nabla \hat{F}_t(\mathbf{X}^t), \mathbf{X} \rangle. \quad (7)$$

Since \mathcal{D}^n is separable for the columns of \mathbf{X} , we can evaluate (7) in parallel for $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$. Define \mathbf{s}_i^t as

$$\mathbf{s}_i^t \in \underset{\mathbf{x} \in \mathcal{D}}{\text{argmin}} \left\langle \frac{1}{n} \nabla f_i(\mathbf{x}_i^t) + \lambda_t (\mathbf{x}_i^t - \bar{\mathbf{x}}^t), \mathbf{x} \right\rangle, \quad (8)$$

where $\bar{\mathbf{x}}^t := \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i^t$. Then, $\mathbf{S}^t = \sum_{i=1}^n \mathbf{s}_i^t \mathbf{e}_i^\top$.

Finally, we update the decision variable by $\mathbf{X}^{t+1} = (1 - \eta_t) \mathbf{X}^t + \eta_t \mathbf{S}^t$, which can be computed column-wise in parallel:

$$\mathbf{x}_i^{t+1} = (1 - \eta_t) \mathbf{x}_i^t + \eta_t \mathbf{s}_i^t, \quad (9)$$

where $\eta_t \in [0, 1]$ is the step-size.

This establishes the fundamental update rule for our proposed algorithm, FEDFW. Note that communication is required only during the computation of $\bar{\mathbf{x}}^t$, which constitutes our aggregation step. All other computations can be performed locally by the clients. Algorithm 1 presents FEDFW and several design variants, which are further detailed in Sect. 4.

3.1 Convergence Guarantees

This section presents the convergence guarantees of FEDFW. We begin with the guarantees for problems with a smooth and convex objective function.

Theorem 1. *Consider problem (1) with L -smooth and convex loss functions f_i . Then, estimation $\bar{\mathbf{x}}^t$ generated by FEDFW with step-size $\eta_t = \frac{2}{t+1}$ and penalty parameter $\lambda_t = \lambda_0 \sqrt{t+1}$ for any $\lambda_0 > 0$ satisfies*

$$F(\bar{\mathbf{x}}^t) - F(\bar{\mathbf{x}}^*) \leq \mathcal{O}(t^{-1/2}). \quad (10)$$

Algorithm 1. FEDFW: Federated Frank-Wolfe Algorithm (+variants)

input $\mathbf{x}_i^1 \in \mathbb{R}^p$, $\forall i \in [n]$, λ_t , η_t , ρ_t , $\bar{\mathbf{x}}^1 = \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i^1$, $\mathbf{y}_i^1 = \mathbf{0}$, $\mathbf{d}_i^1 = \mathbf{0}$
for round $t = 1, 2, \dots, T$ **do**
 — **Client-level local training** —————
 for client $i = 1, 2, \dots, n$ **do**
 — **FEDFW:** $\mathbf{g}_i^t = \frac{1}{n} \nabla f_i(\mathbf{x}_i^t) + \lambda_t(\mathbf{x}_i^t - \bar{\mathbf{x}}^t)$
 — **FEDFW+:** $\mathbf{y}_i^{t+1} = \mathbf{y}_i^t + \lambda_0(\mathbf{x}_i^t - \bar{\mathbf{x}}^t)$
 $\mathbf{g}_i^t = \frac{1}{n} \nabla f_i(\mathbf{x}_i^t) + \lambda_t(\mathbf{x}_i^t - \bar{\mathbf{x}}^t) + \mathbf{y}_i^{t+1}$
 — **FEDFW-STO:** $\mathbf{d}_i^{t+1} = (1 - \rho_t)\mathbf{d}_i^t + \rho_t \frac{1}{n} \nabla f_i(\mathbf{x}_i^t, \omega_i^t)$
 $\mathbf{g}_i^t = \mathbf{d}_i^{t+1} + \lambda_t(\mathbf{x}_i^t - \bar{\mathbf{x}}^t)$
 $\mathbf{s}_i^t = \arg \min\{\langle \mathbf{g}_i^t, \mathbf{x} \rangle : \mathbf{x} \in \mathcal{D}\}$
 $\mathbf{x}_i^{t+1} = (1 - \eta_t)\mathbf{x}_i^t + \eta_t \mathbf{s}_i^t$
 Client communicates \mathbf{s}_i^t to the server.
 end for
 — **Server-level aggregation** —————
 $\bar{\mathbf{x}}^{t+1} = (1 - \eta_t)\bar{\mathbf{x}}^t + \eta_t (\frac{1}{n} \sum_{i=1}^n \mathbf{s}_i^t)$
 Server communicates $\bar{\mathbf{x}}^{t+1}$ to the clients.
end for

Remark 1. Our proof is inspired by the analysis in [33]. However, a distinction lies in how the guarantees are expressed. In [33], the authors demonstrate the convergence of \mathbf{x}_i^t towards a solution by proving that both the objective residual and the distance to the feasible set converge to zero. In contrast, we establish the convergence of $\bar{\mathbf{x}}^t$, representing a feasible point, focusing only on the objective residual. We present detailed proof in the supplementary material.

It is worth noting that the convergence guarantees of FEDFW are slower compared to those of existing unconstrained or projection-based FL algorithms. For instance, in the smooth convex setting with full gradients, FEDAVG [4] achieves a rate of $\mathcal{O}(t^{-1})$ in the objective residual. In a convex composite problem setting, FEDDA [17] converges at a rate of $\mathcal{O}(t^{-2/3})$. While FEDFW guarantees a slower rate of $\mathcal{O}(t^{-1/2})$, it is important to highlight that FEDFW employs cheap linear minimization oracles.

Next, we present the convergence guarantees of FEDFW for non-convex problems. For unconstrained non-convex problems, the gradient norm is commonly used as a metric to demonstrate convergence to a stationary point. However, this metric is not suitable for constrained problems, as the gradient may not approach zero if the solution resides on the boundary of the feasible set. To address this, we use the following gap function, standard in FW analysis [25]:

$$\text{gap}(\mathbf{x}) := \max_{\mathbf{u} \in \mathcal{D}} \langle \nabla F(\mathbf{x}), \mathbf{x} - \mathbf{u} \rangle. \quad (11)$$

It is straightforward to show that $\text{gap}(\mathbf{x})$ is non-negative for all $\mathbf{x} \in \mathcal{D}$, and it attains zero if and only if \mathbf{x} is a first-order stationary point of Problem (1).

Theorem 2. Consider problem (1) with L -smooth loss functions f_i . Suppose the sequence $\bar{\mathbf{x}}^t$ is generated by FEDFW with the fixed step-size $\eta_t = T^{-2/3}$, and penalty parameter $\lambda_t = \lambda_0 T^{1/3}$ for an arbitrary $\lambda_0 > 0$. Then,

$$\min_{1 \leq t \leq T} \text{gap}(\bar{\mathbf{x}}^t) \leq \mathcal{O}(T^{-1/3}). \quad (12)$$

Remark 2. We present the proof in the supplementary material. Our analysis introduces a novel approach, as [33] does not explore non-convex problems. While our focus is primarily on problems (1) and (3), our methodology can be used to derive guarantees for a broader setting of minimization of a smooth non-convex function subject to affine constraints over a convex and compact set.

As with our previous results, the convergence rate in the non-convex setting is slower compared to FEDAVG, which achieves an $\mathcal{O}(t^{-1/2})$ rate in the gradient norm (note the distinction between the gradient norm and squared gradient norm metrics). For composite FL problems with a non-convex smooth loss and a convex non-smooth regularizer, FEDDR [16] achieves an $\mathcal{O}(t^{-1/2})$ rate in the norm of a proximal gradient mapping. In contrast, our guarantees are in terms of the Frank-Wolfe (FW) gap. To our knowledge, FEDDA does not offer guarantees in the non-convex setting.

3.2 Privacy and Communication Benefits

FEDFW offers low communication overhead since the communicated signals are the extreme points of \mathcal{D} , which typically have low dimensional representation. For example, if \mathcal{D} is ℓ_1 (resp., nuclear) norm-ball, then the signals \mathbf{s}_i are 1-sparse (resp., rank-one). Additionally, linear minimization is a nonlinear oracle, the reverse operator of which is highly ill-conditioned. Retrieving the gradient from its linear minimization output is generally unfeasible. For example, if \mathcal{D} is the ℓ_1 norm-ball, then \mathbf{s}_i merely reveals the sign of the maximum entry of the gradient. In the case of a box constraint, \mathbf{s}_i only reveals the gradient signs. For the nuclear norm-ball, \mathbf{s}_i unveils only the top eigenvectors of the gradient. Furthermore, FW is robust against additive and multiplicative errors in the linear minimization step [24]; consequently, we can introduce noise to augment data privacy without compromising the convergence guarantees.

In a simple numerical experiment, we demonstrate the privacy benefits of communicating linear minimization outputs instead of gradients. This experiment is based on the Deep Leakage algorithm [43] using the CIFAR100 dataset. Our experiment compares reconstructed images (*i.e.*, leaked data points) obtained from shared gradients versus shared linear minimization outputs, under ℓ_1 and ℓ_2 norm constraints. Figure 1 displays the final reconstructed images alongside the Peak Signal-to-Noise Ratio (PSNR) across iterations. It is evident that reconstruction via linear minimization oracles, particularly under the ℓ_1 ball constraint, is significantly more challenging than raw gradients.

4 Design Variants of FEDFW

This section discusses several design variants and extensions of FEDFW.

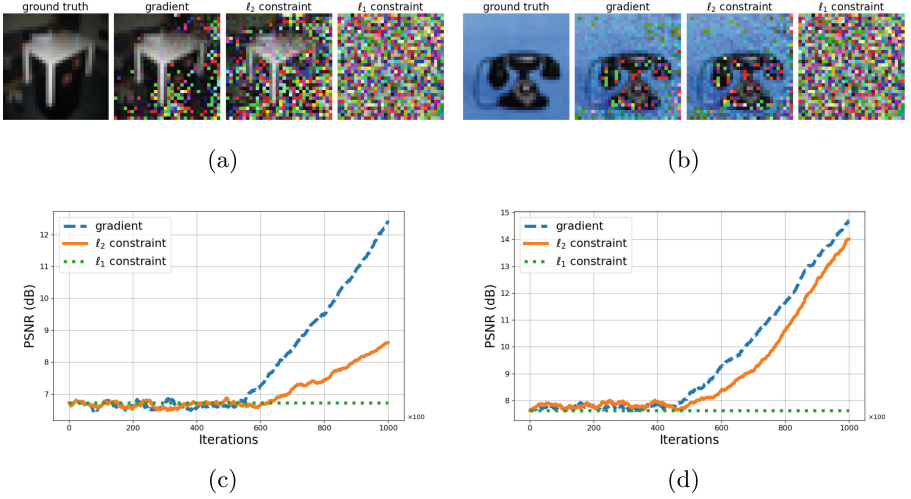


Fig. 1. Privacy benefits of sharing linear minimization outputs vs gradients. The Deep Leakage Algorithm can recover CIFAR-100 data points from shared gradients. Sharing linear minimization outputs enhances privacy. (a) and (b) compares reconstructions from gradients and LMO outputs with ℓ_2 and ℓ_1 -norm ball constraints after 10^5 iterations for two different data points. (c) and (d) present the reconstruction PSNR as a function of iterations for the corresponding images.

4.1 FEDFW with stochastic gradients

Consider the following stochastic problem template:

$$\min_{\mathbf{x} \in \mathcal{D}} F(\mathbf{x}) := \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{\omega_i} [f_i(\mathbf{x}, \omega_i)]. \quad (13)$$

Here, ω_i is a random variable with an unknown distribution \mathcal{P}_i . The client loss function $f_i(\mathbf{x}) := \mathbb{E}_{\omega_i} [f_i(\mathbf{x}, \omega_i)]$ is defined as the expectation over this unknown distribution; hence we cannot compute its gradient. We design FEDFW-STO for solving this problem.

We assume that at each iteration, every participating client can independently draw a sample ω_i^t from their distribution \mathcal{P}_i . $\nabla f_i(\mathbf{x}, \omega_i^t)$ serves as an unbiased estimator of $\nabla f_i(\mathbf{x})$. Additionally, we adopt the standard assumption that the estimator has bounded variance.

Assumption 1 (Bounded variance). Let $\nabla f_i(\mathbf{x}, \omega_i)$ denote the stochastic gradients. We assume that it satisfies the following condition for some $\sigma < \infty$:

$$\mathbb{E}_{\omega_i} \left[\|\nabla f_i(\mathbf{x}, \omega_i) - \nabla f_i(\mathbf{x})\|^2 \right] \leq \sigma^2. \quad (14)$$

Unfortunately, FW does not readily extend to stochastic settings by replacing the gradient with an unbiased estimator of bounded variance. Instead, adapting

FW for stochastic settings, in general, requires a variance reduction strategy. Inspired by [30, 34], we employ the following averaged gradient estimator to tackle this challenge. We start by $\mathbf{d}_i^0 = \mathbf{0}$, and iteratively update

$$\mathbf{d}_i^{t+1} = (1 - \rho_t)\mathbf{d}_i^t + \rho_t \frac{1}{n} \nabla f_i(\mathbf{x}_i^t, \omega_i^t), \quad (15)$$

for some $\rho_t \in (0, 1]$. FEDFW-STO uses \mathbf{d}_i^{t+1} in place of the gradient in the linear minimization step; pseudocode is shown in Algorithm 1. Although \mathbf{d}_i^{t+1} is not an unbiased estimator, it offers the advantage of reduced variance. The balance between bias and variance can be adjusted by modifying ρ_t , and the analysis relies on finding the right balance, reducing variance sufficiently while maintaining the bias within tolerable limits.

Theorem 3. *Consider problem (13) with L -smooth and convex loss functions f_i . Suppose Assumption 1 holds. Then, the sequence $\bar{\mathbf{x}}^t$ generated by FEDFW-STO in Algorithm 1, with step-size $\eta_t = \frac{9}{t+8}$, penalty parameter $\lambda_t = \lambda_0 \sqrt{t+8}$ for an arbitrary $\lambda_0 > 0$, and $\rho_t = \frac{4}{(t+7)^{2/3}}$ satisfies*

$$\mathbb{E}[F(\bar{\mathbf{x}}^t)] - F(\mathbf{x}^*) \leq \mathcal{O}(t^{-1/3}). \quad (16)$$

Remark 3. Our analysis in this setting is inspired by [34]; however, we establish the convergence of the feasible point $\bar{\mathbf{x}}^t$. This differs from the guarantees in [34], which demonstrate the convergence of \mathbf{x}_i^t towards a solution by proving that both the expected objective residual and the expected distance to the feasible set converge to zero. We present the detailed proof in the supplementary material.

In the smooth convex stochastic setting, FEDAVG achieves a convergence rate of $\mathcal{O}(t^{-1/2})$. This rate also applies to FEDDA when addressing composite convex problems. Additionally, under the assumption of strong convexity, FAST-FEDDA [19] achieves an accelerated rate of $\mathcal{O}(t^{-1})$. In comparison, FEDFW-STO converges with $\mathcal{O}(t^{-1/3})$ rate; however, it benefits from the use of inexpensive linear minimization oracles.

4.2 FEDFW with Partial Client Participation

A key challenge in FL is to tackle random device participation schedules. Unlike a classical distributed optimization scheme, in most FL applications, clients have some autonomy and are not entirely controlled by the server. Due to various factors, such as network congestion or resource constraints, clients may intermittently participate in the training process. This obstacle can be tackled in FEDFW by employing a block-coordinate Frank-Wolfe approach [48]. Given that the domain of problem (3) is block-separable, we can extend our FEDFW analysis to block-coordinate updates.

Suppose that in every round t , the client i participates in the training procedure with a fixed probability of $\mathbf{p}_i \in (0, 1]$. For simplicity, we assume the participation rate is the same among all clients, *i.e.*, $\mathbf{p}_1 = \dots = \mathbf{p}_n := \mathbf{p}$, but

non-uniform participation can be addressed similarly. Instate the convex optimization problem described in Theorem 1 but with the random client participation scheme. At round t , the training procedure follows the same as in Algorithm 1 for the participating clients, and $\mathbf{x}_i^{t+1} = \mathbf{x}_i^t$ for the non-participants. Then, the estimation $\bar{\mathbf{x}}^t$ generated with the step-size $\eta_t = \frac{2}{\mathbf{p}(t-1)+2}$ and penalty parameter $\lambda_t = \lambda_0 \sqrt{\mathbf{p}(t-1)+2}$ converges to a solution with rate

$$\mathbb{E}[F(\bar{\mathbf{x}}^t) - F(\mathbf{x}^*)] \leq \mathcal{O}((\mathbf{p}t)^{-1/2}). \quad (17)$$

Similarly, if we consider the non-convex setting of Theorem 2 with randomized client participation, and use the block-coordinate FEDFW with step-size $\eta_t = (\mathbf{p}T + 1)^{-\frac{2}{3}}$, and penalty parameter $\lambda_t = \lambda_0(\mathbf{p}T + 1)^{\frac{1}{3}}$, we get

$$\min_{1 \leq t \leq T} \mathbb{E}[\text{gap}(\bar{\mathbf{x}}^t)] \leq \mathcal{O}((\mathbf{p}T)^{-1/3}). \quad (18)$$

The proofs are provided in the supplementary material.

4.3 FEDFW with Split Constraints for Stragglers

FL systems are frequently implemented across heterogeneous pools of client hardware, leading to the ‘straggler effect’—delays in execution resulting from clients with less computation or communication speeds. In FEDFW, we can mitigate this issue by assigning tasks to straggling clients more compatible with their computational capabilities. Theoretically, this adjustment can be achieved through certain special reformulations of the problem defined in (3). Specifically, the constraint $\mathbf{X} \in \mathcal{D}^n$ can be refined to $\mathbf{X} \in \bigcap_{i=1}^n \mathcal{D}_i$, where $\bigcap_{i=1}^n \mathcal{D}_i = \mathcal{D}$. This modification does not affect the solution set, due to the consensus constraint.

In general, in FEDFW, most of the computation occurs during the linear minimization step. Suppose that the resources of the client i are limited, particularly for arithmetic computations. In this case, we can select \mathcal{D}_i as a superset of \mathcal{D} where linear minimization computations are more straightforward. For instance, a Euclidean (or Frobenius) norm-ball encompassing \mathcal{D} could be an excellent choice. Then, \mathbf{s}_i^t becomes proportional to the negative of \mathbf{g}_i^t with appropriate normalization based on the radius of \mathcal{D}_i , facilitating computation with minimal effort. On the other hand, if the primary bottleneck is communication, we might opt for \mathcal{D}_i characterized by sparse extreme points, such as an ℓ_1 -norm ball containing \mathcal{D} or by low-rank extreme points like those in a nuclear-norm ball. This strategy results in sparse (or low-rank) \mathbf{s}_i^t , thereby streamlining communication.

4.4 FEDFW with Augmented Lagrangian

FEDFW employs a quadratic penalty strategy to handle the consensus constraint. We also propose an alternative variant, FEDFW+, which is modeled after the augmented Lagrangian strategy in [37]. The pseudocode for FEDFW+ is presented in Algorithm 1. We compare the empirical performance of FEDFW and FEDFW+ in Sect. 5. The theoretical analysis of FEDFW+ is omitted here; for further details, we refer readers to [37].

5 Numerical Experiments

In this section, we evaluate and compare the empirical performance of our methods against FEDDR, which serves as the baseline algorithm, on the convex multiclass logistic regression (MCLR) problem and the non-convex tasks of training convolutional neural networks (CNN) and deep neural networks (DNN). For each problem, we consider two different choices for the domain \mathcal{D} : namely the ℓ_1 and ℓ_2 ball constraints, each with a radius of 10. We assess the models' performance based on validation accuracy, validation loss, and the Frank-Wolfe gap (11). To evaluate the effect of data heterogeneity, we conducted experiments using both IID and non-IID data distributions across clients. The code for the numerical experiments can be accessed via <https://github.com/sourasb05/Federated-Frank-Wolfe.git>.

Datasets. We use several datasets in our experiments: MNIST [49], CIFAR-10 [50], EMNIST [51], and a synthetic dataset generated as described in [52]. Specifically, the synthetic data is drawn from a multivariate normal distribution, and the labels are computed using softmax functions. We create data points of 60 features and from 10 different labels. For all datasets, we consider both IID and non-IID data distributions across the clients. In the non-IID scenario, each user has data from only 3 labels. We followed this rule for the synthetic data, as well as MNIST, CIFAR10, and EMNIST-10. For EMNIST-62, each user has data from 20 classes, with unequal distribution among users.

5.1 Comparison of Algorithms in the Convex Setting

We tested the performance of the algorithms on the strongly convex MCLR problem using the MNIST and CIFAR-10 datasets as well as the synthetic dataset. Table 1 presents the test accuracy results for the algorithms with IID and non-IID data distributions, and for two different choices of \mathcal{D} . In these experiments, we simulated FL with 10 clients, all participating fully ($p = 1$). We ran the algorithms for 100 communication rounds, with one local iteration per round.

5.2 Comparison of Algorithms in the Non-convex Setting

For the experiments in the non-convex setting we trained CNNs using the MNIST dataset and a DNN with two hidden layers using the synthetic dataset. We considered an FL system with 10 clients and full participation ($p = 1$). Similar to the previous case, we evaluated IID and non-IID data distributions as well as different choices of \mathcal{D} , and ran the methods for 100 communication rounds with a single local training step. Table 2 summarizes the resulting test accuracies.

Table 1. Comparison of algorithms on the convex MCLR problem with different datasets and choices of \mathcal{D} . We consider both IID and non-IID data distributions. The numbers represent test accuracy.

	IID			non-IID		
	MNIST	Synthetic	CIFAR10	MNIST	Synthetic	CIFAR10
	ℓ_2 constraint					
FEDDR	89.59 (± 0.0003)	78.24(± 0.007)	39.95 (± 0.001)	83.72(± 0.001)	92.97(± 0.005)	37.79(± 0.004)
FEDFW	86.96(± 0.009)	80.20 (± 0.01)	36.30(± 0.001)	86.95(± 0.001)	94.81 (± 0.001)	38.13 (± 0.003)
FEDFW+	86.50(± 0.001)	79.96(± 0.001)	36.30(± 0.001)	86.98 (± 0.001)	94.56(± 0.009)	37.20(± 0.004)
	ℓ_1 constraint					
FEDDR	72.18(± 0.0004)	79.00(± 0.004)	23.25 (± 0.00)	74.29(± 0.0)	93.81 (± 0.009)	24.77(± 0.0)
FEDFW	78.07 (± 0.005)	81.63(± 0.01)	21.86(± 0.003)	80.54 (± 0.0)	90.84(± 0.003)	25.08(± 0.004)
FEDFW+	69.17(± 0.004)	81.92 (± 0.008)	21.99(0.006)	71.32(± 0.002)	91.20(± 0.006)	25.16 (± 0.008)

Table 2. Comparison of algorithms on the non-convex tasks. We train a CNN using MNIST, and a DNN with synthetic data. We consider IID and non-IID data distributions, and different choices of \mathcal{D} . The numbers show test accuracy.

	IID		non-IID	
	MNIST	Synthetic	MNIST	Synthetic
	ℓ_2 constraint			
FEDDR	96.89 (± 0.0009)	75.96(± 0.03)	88.93(± 0.013)	93.85(± 0.009)
FEDFW	95.87(± 0.01)	81.70(± 0.008)	92.70 (± 0.002)	96.13 (± 0.007)
FEDFW+	95.05(± 0.005)	81.96 (± 0.006)	91.79(± 0.005)	96.08(± 0.004)
	ℓ_1 constraint			
FEDDR	11.72(± 0.0)	78.59 (± 0.006)	16.75(± 0.01)	93.51 (± 0.006)
FEDFW	23.88 (± 0.005)	75.52(± 0.008)	37.62 (± 0.008)	91.53(± 0.01)
FEDFW+	20.40(± 0.003)	76.44(± 0.004)	36.27(± 0.006)	91.96(± 0.003)

5.3 Comparison of Algorithms in the Stochastic Setting

Finally, we compared the performance of FEDFW-STO against FEDDR in the stochastic setting, where only stochastic gradients are accessible. For this experiment, we consider an FL network with 100 clients with full participation ($p = 1$). Over this network, we trained the MCLR model using EMNIST-10, EMNIST-62, CIFAR10, and the synthetic dataset. We used a mini-batch size of 64, one local iteration per communication round, and ran the algorithms for 300 communication rounds. Table 3 summarizes the test accuracies obtained in this experiment. FEDFW-STO outperformed FEDDR in our experiments in the stochastic setting.

5.4 Impact of Hyperparameters

We conclude our experiments with an ablation study to investigate how varying hyperparameters impact the performance of FEDFW.

Table 3. Comparison of algorithms in the stochastic setting on the convex MCLR problem with different datasets and ℓ_2 ball constraint. We consider both IID and non-IID data distributions. The numbers represent test accuracy.

	IID		non-IID	
	EMNIST-10	EMNIST-62	EMNIST-10	EMNIST-62
FEDDR	92.18(± 0.01)	39.58(± 0.00)	85.70(± 0.002)	41.09(± 0.002)
FEDFW-STO	93.79(± 0.00)	41.22(± 0.00)	91.88(± 0.01)	60.51(± 0.002)
	Synthetic	CIFAR10	Synthetic	CIFAR10
FEDDR	67.68(± 0.003)	36.39(± 0.004)	84.70(± 0.01)	34.91(± 0.008)
FEDFW-STO	72.01(± 0.004)	38.52(± 0.01)	87.32(± 0.003)	37.83(± 0.01)

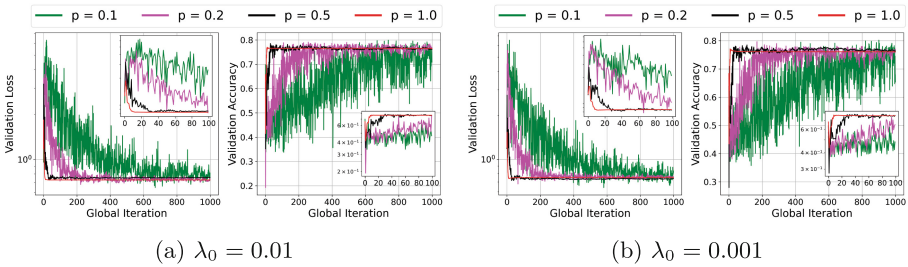


Fig. 2. Effect of participation p on FEDFW. The experiment was conducted with MCLR using synthetic data, an ℓ_1 constraint, and two different choices of λ_0 .

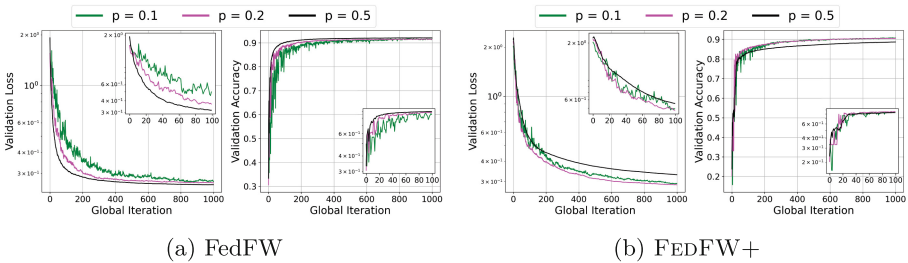
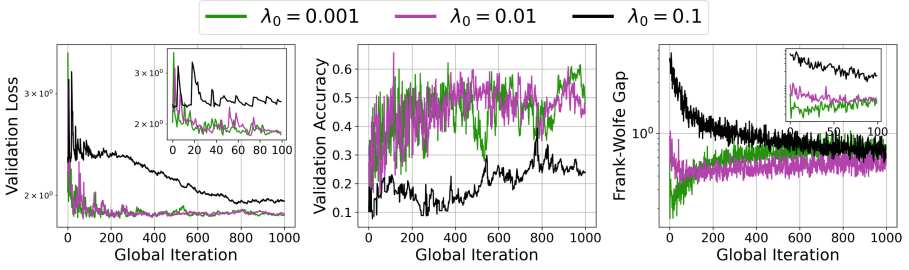
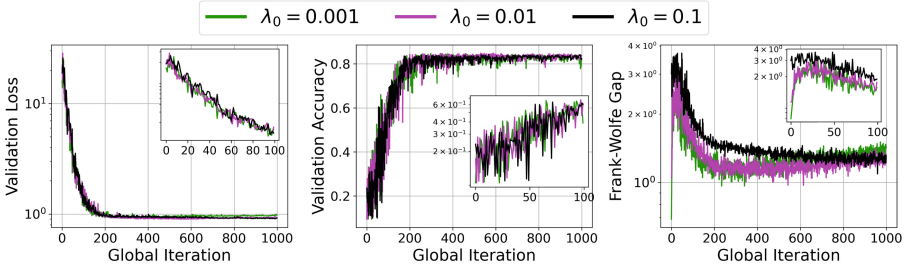


Fig. 3. Effect of participation p on FEDFW and FEDFW+. We trained a DNN model using synthetic data, an ℓ_2 constraint, and a fixed $\lambda_t = 10^{-3}$.

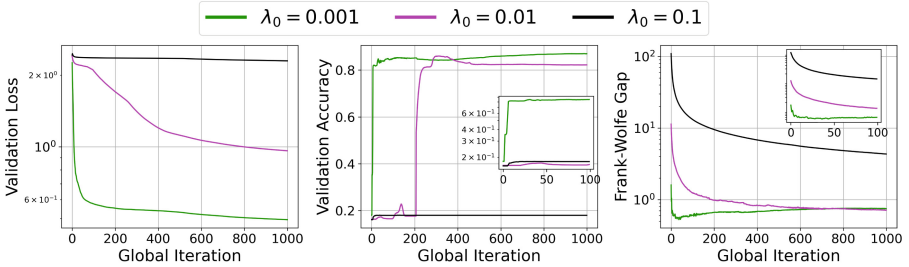
Impact of Partial Participation (p). Figure 2 shows the validation accuracy and loss of FEDFW algorithm for synthetic data and MCLR model. Figure 3 depicts the validation accuracy and loss of FEDFW and FEDFW+ algorithms for synthetic data and DNN model. Both convex and non-convex experiments show faster convergence for higher participation probability. It is worth mentioning that variations in λ_0 do not alter the influence of p . These observations are in accordance with the theoretical guarantees presented in Sect. 4.2.



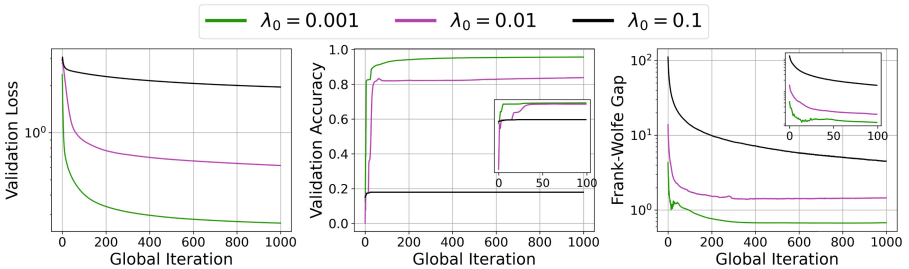
(a) Convex MCLR with MNIST data, participation ratio $p = 0.5$, and ℓ_1 constraint.



(b) Convex MCLR with MNIST data, participation ratio $p = 0.5$, and ℓ_2 constraint.



(c) Non-convex DNN with synthetic data, full participation $p = 1$, and ℓ_1 constraint.



(d) Non-convex DNN with synthetic data, full participation $p = 1$, and ℓ_2 constraint.

Fig. 4. Effect of the initial penalty (λ_0) on FEDFW. (a) and (b) show the results for the convex setting, (c) and (d) demonstrates the non-convex setting.

Impact of Initial Penalty Parameter (λ_0). Figure 4 illustrates the effect of hyperparameters on the convergence of loss, Frank-Wolfe gap, and validation accuracy of the algorithms. A higher λ_0 leads to a larger gap in the initial iterations of the algorithm due to its regularization effect. In other words, increasing λ_0 enforces the update direction towards the consensus set, which in turn increases the gap value in the first iteration. The exact expressions for the constants in the convergence guarantees, which are detailed in the supplementary material, can guide the optimal choice of λ_0 .

6 Conclusions

We introduced a FW-type method for FL and established its theoretical convergence rates. The proposed method, FEDFW, guarantees $\mathcal{O}(t^{-1/2})$ convergence rates when the objective function smooth and convex. If we remove the convexity assumption, the rate reduces to $\mathcal{O}(t^{-1/3})$. With access to only stochastic gradients, FEDFW achieves an $\mathcal{O}(t^{-1/3})$ convergence rate in the convex setting. Additionally, we proposed an empirically faster version of FEDFW by incorporating an augmented Lagrangian dual update.

We conclude with a brief discussion on the limitations of our work. The primary limitation of FEDFW is its slower convergence rates compared to state-of-the-art FL methods. Developing a tighter bound for FEDFW, with multiple local steps, is an area for future research. Additionally, the analysis of FEDFW+ is left to future work. Another important piece of future work is the convergence analysis of FEDFW-STO for non-convex objectives. Finally, the development and analysis of an extension for asynchronous updates also remain as future work.

Acknowledgements. This work was supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. We also acknowledge support from the Swedish Research Council under the grant registration number 2023-05476. The computations were enabled by the Berzelius resource provided by the Knut and Alice Wallenberg Foundation at the National Supercomputer Centre. Additionally, computations in an earlier version of this work were enabled by resources provided by the Swedish National Infrastructure for Computing (SNIC) at Chalmers Centre for Computational Science and Engineering (C3SE) partially funded by the Swedish Research Council through grant agreement no. 2018-05973. We appreciate the discussions with Yikun Hou on the numerical experiments and implementation. We acknowledge the use of OpenAI’s ChatGPT for editorial assistance in preparing this manuscript.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Lim, W.Y.B., et al.: Federated learning in mobile edge networks: a comprehensive survey. *IEEE Commun. Surv. Tutor.* **22**(3), 2031–2063 (2020)
2. Wang, J., et al.: A field guide to federated optimization. [arXiv:2107.06917](https://arxiv.org/abs/2107.06917) (2021)
3. Frank, M., Wolfe, P.: An algorithm for quadratic programming. *Naval Res. Logist. Q.* **3**(1–2), 95–110 (1956)
4. McMahan, B., Moore, E., Ramage, D., Hampson, S., Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: *Artificial Intelligence and Statistics*, pp. 1273–1282. PMLR (2017)
5. Konečný, J., McMahan, H.B., Ramage, D., Richtárik, P.: Federated optimization: distributed machine learning for on-device intelligence. [arXiv:1610.02527](https://arxiv.org/abs/1610.02527) (2016)
6. Pathak, R., Wainwright, M.J.: Fedsplit: an algorithmic framework for fast federated optimization. In: *Advances in Neural Information Processing Systems*, vol. 33, pp. 7057–7066 (2020)
7. Zhang, X., Hong, M., Dhople, S., Yin, W., Liu, Y.: Fedpd: a federated learning framework with adaptivity to non-IID data. *IEEE Trans. Sig. Process.* **69**, 6055–6070 (2021)
8. Stich, S.U.: Local SGD converges fast and communicates little. [arXiv:1805.09767](https://arxiv.org/abs/1805.09767) (2018)
9. Li, X., Huang, K., Yang, W., Wang, S., Zhang, Z.: On the convergence of FedAVG on non-IID data. In: *International Conference on Learning Representations* (2019)
10. Haddadpour, F., Kamani, M.M., Mahdavi, M., Cadambe, V.: Local SGD with periodic averaging: tighter analysis and adaptive synchronization. In: *Advances in Neural Information Processing Systems*, vol. 32 (2019)
11. Yu, H., Yang, S., Zhu, S.: Parallel restarted SGD with faster convergence and less communication: demystifying why model averaging works for deep learning. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 1, pp. 5693–5700 (2019)
12. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. *Proc. Mach. Learn. Syst.* **2**, 429–450 (2020)
13. Woodworth, B., et al.: Is local SGD better than minibatch SGD? In: *International Conference on Machine Learning*, pp. 10334–10343. PMLR (2020)
14. Woodworth, B.E., Patel, K.K., Srebro, N.: Minibatch vs local SGD for heterogeneous distributed learning. In: *Advances in Neural Information Processing Systems*, vol. 33, pp. 6281–6292 (2020)
15. Al-Shedivat, M., Gillenwater, J., Xing, E., Rostamizadeh, A.: Federated learning via posterior averaging: a new perspective and practical algorithms. [arXiv:2010.05273](https://arxiv.org/abs/2010.05273) (2020)
16. Tran Dinh, Q., Pham, N.H., Phan, D., Nguyen, L.: FedDR-randomized Douglas-Rachford splitting algorithms for nonconvex federated composite optimization. In: *Advances in Neural Information Processing Systems*, vol. 34, pp. 30326–30338 (2021)
17. Yuan, H., Zaheer, M., Reddi, S.: Federated composite optimization. In: *International Conference on Machine Learning*, pp. 12253–12266. PMLR (2021)
18. Nesterov, Y.: Primal-dual subgradient methods for convex problems. *Math. Program.* **120**(1), 221–259 (2009)
19. Bao, Y., Crawshaw, M., Luo, S., Liu, M.: Fast composite optimization and statistical recovery in federated learning. In: *International Conference on Machine Learning*, pp. 1508–1536. PMLR (2022)

20. Wang, H., Marella, S., Anderson, J.: Fedadmm: a federated primal-dual algorithm allowing partial participation. In: 2022 IEEE 61st Conference on Decision and Control (CDC), pp. 287–294. IEEE (2022)
21. He, C., Peng, L., Sun, J.: Federated learning with convex global and local constraints. In: OPT 2023: Optimization for Machine Learning (2023)
22. Levitin, E.S., Polyak, B.T.: Constrained minimization methods. *USSR Comput. Math. Math. Phys.* **6**(5), 1–50 (1966)
23. Hazan, E.: Sparse approximate solutions to semidefinite programs. In: Laber, E.S., Bornstein, C., Nogueira, L.T., Faria, L. (eds.) *LATIN 2008*. LNCS, vol. 4957, pp. 306–316. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78773-0_27
24. Jaggi, M.: Revisiting frank-wolfe: projection-free sparse convex optimization. In: *International Conference on Machine Learning*, pp. 427–435. PMLR (2013)
25. Lacoste-Julien, S.: Convergence rate of Frank-Wolfe for non-convex objectives. [arXiv:1607.00345](https://arxiv.org/abs/1607.00345) (2016)
26. Hazan, E., Kale, S.: Projection-free online learning. In: *International Conference on Machine Learning*. PMLR (2012)
27. Hazan, E., Luo, H.: Variance-reduced and projection-free stochastic optimization. In: *International Conference on Machine Learning*, pp. 1263–1271. PMLR (2016)
28. Reddi, S.J., Sra, S., Póczos, B., Smola, A.: Stochastic Frank-Wolfe methods for nonconvex optimization. In: 2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton), pp. 1244–1251. IEEE (2016)
29. Yurtsever, A., Sra, S., Cevher, V.: Conditional gradient methods via stochastic path-integrated differential estimator. In: *International Conference on Machine Learning*. pp. 7282–7291. PMLR (2019)
30. Mokhtari, A., Hassani, H., Karbasi, A.: Stochastic conditional gradient methods: from convex minimization to submodular maximization. *J. Mach. Learn. Res.* **21**(105), 1–49 (2020)
31. Néglar, G., et al.: Stochastic Frank-Wolfe for constrained finite-sum minimization. In: *international Conference on Machine Learning*, pp. 7253–7262. PMLR (2020)
32. Lan, G.: An optimal method for stochastic composite optimization. *Math. Program.* **133**(1), 365–397 (2012)
33. Yurtsever, A., Fercoq, O., Locatello, F., Cevher, V.: A conditional gradient framework for composite convex minimization with applications to semidefinite programming. In: *International Conference on Machine Learning*, pp. 5727–5736. PMLR (2018)
34. Locatello, F., Yurtsever, A., Fercoq, O., Cevher, V.: Stochastic Frank-Wolfe for composite convex minimization. In: *Advances in Neural Information Processing Systems*, vol. 32 (2019)
35. Dresdner, G., Vladarean, M.L., Rätsch, G., Locatello, F., Cevher, V., Yurtsever, A.: Faster one-sample stochastic conditional gradient method for composite convex minimization. In: *International Conference on Artificial Intelligence and Statistics*, pp. 8439–8457. PMLR (2022)
36. Gidel, G., Pedregosa, F., Lacoste-Julien, S.: Frank-Wolfe splitting via augmented Lagrangian method. In: *International Conference on Artificial Intelligence and Statistics*, pp. 1456–1465. PMLR (2018)
37. Yurtsever, A., Fercoq, O., Cevher, V.: A conditional-gradient-based augmented Lagrangian framework. In: *International Conference on Machine Learning*, pp. 7272–7281. PMLR (2019)
38. Kerdreux, T.: Accelerating conditional gradient methods. Ph.D. thesis, Université Paris sciences et lettres (2020)

39. Bomze, I.M., Rinaldi, F., Zeffiro, D.: Frank–Wolfe and friends: a journey into projection-free first-order optimization methods. *4OR* **19**, 313–345 (2021)
40. Wai, H.T., Lafond, J., Scaglione, A., Moulines, E.: Decentralized Frank-Wolfe algorithm for convex and nonconvex problems. *IEEE Trans. Autom. Control* **62**(11), 5522–5537 (2017)
41. Mokhtari, A., Hassani, H., Karbasi, A.: Decentralized submodular maximization: bridging discrete and continuous settings. In: *International Conference on Machine Learning*, pp. 3616–3625. PMLR (2018)
42. Gao, H., Xu, H., Vucetic, S.: Sample efficient decentralized stochastic Frank-Wolfe methods for continuous DR-submodular maximization. In: *Thirtieth International Joint Conference on Artificial Intelligence* (2021)
43. Zhu, L., Liu, Z., Han, S.: Deep leakage from gradients. In: *Advances in Neural Information Processing Systems*, vol. 32 (2019)
44. Li, Z., Zhang, J., Liu, L., Liu, J.: Auditing privacy defenses in federated learning via generative gradient leakage. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10132–10142 (2022)
45. Wang, Y.X., Sadhanala, V., Dai, W., Neiswanger, W., Sra, S., Xing, E.: Parallel and distributed block-coordinate Frank-Wolfe algorithms. In: *International Conference on Machine Learning*, pp. 1548–1557. PMLR (2016)
46. Zheng, W., Bellet, A., Gallinari, P.: A distributed Frank-Wolfe framework for learning low-rank matrices with the trace norm. *Mach. Learn.* **107**(8), 1457–1475 (2018)
47. Zhang, M., Zhou, Y., Ge, Q., Zheng, R., Wu, Q.: Decentralized randomized block-coordinate Frank-Wolfe algorithms for submodular maximization over networks. *IEEE Trans. Syst. Man Cybern. Syst.* (2021)
48. Lacoste-Julien, S., Jaggi, M., Schmidt, M., Pletscher, P.: Block-coordinate Frank-Wolfe optimization for structural SVMs. In: *International Conference on Machine Learning*, pp. 53–61. PMLR (2013)
49. LeCun, Y., et al.: Gradient-based learning applied to document recognition. *Proc. IEEE* **86**(11), 2278–2324 (1998)
50. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images, Toronto, ON, Canada (2009)
51. Cohen, G., et al.: EMNIST: extending MNIST to handwritten letters. In: *IJCNN*, pp. 2921–2926. IEEE (2017)
52. Dinh, T., Tran, C., Nguyen, N.: Personalized federated learning with Moreau envelopes. *J. Adv. Neural Inf. Process. Syst.* **33**, 21394–21405 (2020)



Bootstrap Latents of Nodes and Neighbors for Graph Self-supervised Learning

Yunhui Liu^{1,2}, Huaisong Zhang³, Tieke He^{1,2(✉)}, Tao Zheng^{1,2},
and Jianhua Zhao^{1,2}

¹ State Key Laboratory for Novel Software Technology,
Nanjing University, Nanjing, China

hetieke@gmail.com

² Software Institute, Nanjing University, Nanjing, China

³ Tsinghua Shenzhen International Graduate School,
Tsinghua University, Shenzhen, China

Abstract. Contrastive learning is a significant paradigm in graph self-supervised learning. However, it requires negative samples to prevent model collapse and learn discriminative representations. These negative samples inevitably lead to heavy computation, memory overhead and class collision, compromising the representation learning. Recent studies present that methods obviating negative samples can attain competitive performance and scalability enhancements, exemplified by bootstrapped graph latents (BGRL). However, BGRL neglects the inherent graph homophily, which provides valuable insights into underlying positive pairs. Our motivation arises from the observation that subtly introducing a few ground-truth positive pairs significantly improves BGRL. Although we can't obtain ground-truth positive pairs without labels under the self-supervised setting, edges in the graph can reflect noisy positive pairs, i.e., neighboring nodes often share the same label. Therefore, we propose to expand the positive pair set with node-neighbor pairs. Subsequently, we introduce a cross-attention module to predict the supportiveness score of a neighbor with respect to the anchor node. This score quantifies the positive support from each neighboring node, and is encoded into the training objective. Consequently, our method mitigates class collision from negative and noisy positive samples, concurrently enhancing intra-class compactness. Extensive experiments are conducted on five benchmark datasets and three downstream task node classification, node clustering, and node similarity search. The results demonstrate that our method generates node representations with enhanced intra-class compactness and achieves state-of-the-art performance. Our implementation code is available at <https://github.com/Cloudy1225/BLNN>.

Keywords: Self-Supervised Learning · Graph Representation Learning · Graph Neural Networks

1 Introduction

Graph self-supervised learning (GSSL) is a promising paradigm for learning more informative representations without human annotations. Typically, GSSL models

are pre-trained using well-designed pretext objectives, which serve as effective initializations for diverse downstream tasks [16]. Consequently, GSSL has made substantial advancements in graph representation learning. It offers performance, generalizability, and robustness metrics comparable to or even surpassing those of supervised methods [2, 25, 27].

A major branch of GSSL is graph contrastive learning (GCL) methods [38, 39], which aim to learn representations by maximizing the agreement between two augmented samples (positive pair) while minimizing the similarities with other samples (negative pairs). The constructed negative pairs is crucial for preventing model collapse and generating discriminative representations [29]. Consequently, current GCL methods inherently rely on increasing the quantity and quality of negative samples. This reliance not only introduces additional computational and memory costs but also leads to the class collision issue, where different samples from the same class are erroneously considered negative pairs, thereby impeding representation learning for classification [22]. To address these issues, recent non-contrastive methods have explored the prospect of learning without negative samples [1, 12, 14, 24, 25, 34]. Among these methods, Bootstrapped Graph Latents (BGRL) [25], derived from BYOL [5], has achieved competitive performance and heightened scalability. BGRL learns node representations by using representations of one augmented view to predict another view, i.e., maximizing the similarity between the prediction and its paired target. Simultaneously, BGRL strategically leverages the asymmetry between the online branch (with gradient) and the target branch (without gradient) to alleviate model collapse.

However, BGRL fails to account for inherent graph homophily, which indicates the phenomenon that neighboring nodes tend to share the same semantic label and thus offers valuable insights into underlying positive pairs. *Why does exploiting the homophily pattern make sense?* In practice, some supervised metric learning methods [10, 31, 33], which employ architectures and objectives akin to self-supervised learning, have illustrated that introducing more ground-truth positive pairs (i.e., samples with the same label) significantly enhances representation learning for classification. Such success inspires us that mining potential positive pairs could empower the model to learn highly intra-class-compacted representations, which are more conducive to classification. Our hypothesis is validated through empirical studies in Sect. 4.1. Unfortunately, unlike the supervised setting, obtaining ground-truth positive pairs is unfeasible due to the absence of labels under the self-supervised setting. But fortunately, the homophily pattern is evident in various real-world graphs [18], where neighboring nodes can be seen as noisy positive pairs. Consequently, exploiting such neighbor information holds promise for graph self-supervised learning.

Based on the above analysis, we propose Bootstrap Latents of Nodes and Neighbors (BLNN) to enhance Bootstrapped Graph Latents by incorporating neighbor information. Specifically, we first expand the positive pair set with node-neighbor pairs based on the graph homophily pattern. However, although connected nodes tend to share the same label in the homophily scenario, there

also exist inter-class edges, especially near the decision boundary between two classes. Treating these inter-class connected nodes as positive (i.e., false positive) pairs would inevitably compromise overall performance. To alleviate this class collision caused by false positive pairs, we further introduce an attention module to compute a supportiveness score of each neighbor representation with respect to the current view anchor node. This score serves as a soft measure of the supportiveness associated with each neighbor contributing to the current anchor node during loss computations. Basically, a higher supportiveness often stands for a higher weight to intra-class node-neighbor pairs. To this end, our BLNN incorporates soft positive node-neighbor pairs to support the anchor node for loss computations, resulting in more intra-class-compacted and discriminative node representations. The contributions of our work can be summarized as follows:

- We empirically demonstrate the efficacy of introducing more ground-truth positive pairs in boosting the negative-sample-free method BGRL. And we propose exploiting the graph homophily to mining positive pairs in the absence of labels.
- We expand the positive pair set with node-neighbor pairs and propose a cross-attention module to weight the contribution of each neighbor to loss computations. This approach mitigates class collision resulting from false positive node-neighbor pairs.
- Extensive experiments are conducted on five benchmark datasets and three downstream task node classification, node clustering, and node similarity search. The results demonstrate that our method generates node representations with enhanced intra-class compactness and achieves state-of-the-art performance.

2 Related Work

2.1 Graph Self-supervised Learning

Recently, numerous research efforts have been devoted to graph self-supervised learning, and a branch based on multi-view learning has garnered attention owing to its superior performance. The basic idea involves ensuring consensus among multiple views derived from the same sample under different graph transformations to optimize model parameters [16]. A crucial aspect of these methods is the prevention of trivial solutions, where all representations converge either to a constant point (i.e., complete collapse) or to a subspace (i.e., dimensional collapse). The existing methods can be broadly classified into two groups: contrastive and non-contrastive approaches, each delineated by its strategy for mitigating model collapse.

Contrastive-Based Methods typically follow the criterion of mutual information maximization [7], whose objective functions involve contrasting positive pairs with negative ones. Pioneering works, such as DGI [27] and GMI [21], focus on unsupervised representation learning by maximizing mutual information between node-level representations and a graph summary vector, employing

the Jensen-Shannon estimator [20]. MVGRL [6] proposes to learn both node-level and graph-level representations by performing node diffusion and contrasting node representations to augmented graph representation. GRACE [38] and its variants GCA [39], gCooL [13], CSGCL [2] learn node representations by pulling together the representations of the same node in two augmented views while pushing away the representations of the other nodes in two views [29]. Despite the success of contrastive learning on graphs, they require a large number of negative samples with carefully crafted encoders and augmentation techniques to learn discriminative representations, making them suffer seriously from heavy computation, memory overhead and class collision [22].

Non-contrastive Methods discard negative samples, necessitating specialized strategies to avoid collapsed solutions. CCA-SSG [34], G-BT [1] and iGCL [14] learn augmentation invariant information while introducing feature decorrelation to capture orthogonal features and prevent dimensional collapse. BGRL [25], derived from BYOL [5], introduces an online network along with a target network, where the target network is updated with a moving average of the online network to avoid collapse. AFGRL [12] identifies nodes as positive samples by considering both local structural information and global graph semantics, sidestepping the need for an augmented graph view and negative sampling. SGCL [24] uncovers the hidden factors contributing to BGRL’s success and simplifies the architecture design. In this paper, we propose mining potential positive pairs from neighboring nodes to enhance BGRL.

2.2 Generation of Positive and Negative Pairs

There are two common approaches to generating positive and negative pairs, depending on the availability of label information. In the supervised setting, where label information is available, positive pairs consist of samples within the same class, while negative pairs comprise samples from different classes [10,31,33]. In the self-supervised setting without label information, a typical strategy is to generate different views of the original sample via augmentation [9]. Here, two views of the same sample serve as positive pairs for each other, while those of different samples serve as negative pairs. However, such instance discrimination based methods inevitably a class collision issue, which means even for very similar samples, they still need to be pushed apart.

To mitigate the class collision issue, some studies focus on mining positive pairs from nearest neighbors [3,4,12,37] while some propose methods without negative pairs [5,12,25,34]. In the domain of graph, AF-GCL [28] regards multi-hop neighboring nodes as potential positive pairs, utilizing well-designed similarity metrics to identify the most similar nodes as positive pairs; nevertheless, this method still necessitates a considerable number of negative pairs. AFGRL [12] and HomoGCL [15] identify positive pairs by considering the local structural information and the global semantics of graphs, but they require performing time-consuming K-means clustering on the entire set of node representations to capture global semantic information. Our BLNN differs from previous work

in the following three highlights: 1) BLNN, derived from BGRL [25], is a non-contrastive method, eliminating the introduction of class collision arising from false negative pairs. 2) BLNN treats all one-hop node-neighbor pairs as candidate positive pairs, simplifying the selection of candidate neighbors from the K-NN search. 3) BLNN employs a cross-attention module, instead of the time-consuming K-means, to mitigate class collision caused by noisy positive node-neighbor pairs.

3 Preliminary

3.1 Problem Statement

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ represent an attributed graph, where $\mathcal{V} = \{v_1, v_2, \dots, v_n\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denote the node set and the edge set, respectively. The graph \mathcal{G} is associated with a feature matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$, where $\mathbf{x}_i \in \mathbb{R}^p$ represents the feature of v_i , and an adjacency matrix $\mathbf{A} \in \{0, 1\}^{n \times n}$, where $\mathbf{A}_{i,j} = 1$ if and only if $(v_i, v_j) \in \mathcal{E}$. During training in the self-supervised setting, no task-specific labels are provided for \mathcal{G} . The primary objective is to learn an embedding function $f_\theta(\mathbf{A}, \mathbf{X})$ that transforms \mathbf{X} to \mathbf{H} , where $\mathbf{H} \in \mathbb{R}^{n \times d}$ and $d \ll p$. The pre-trained representations are intended to encapsulate both attribute and structure information inherent in \mathcal{G} and can be easily transferable to various downstream tasks such as node classification, node clustering, and node similarity search.

3.2 Graph Homophily

Graph homophily suggests that neighboring nodes often belong to the same class, offering valuable prior knowledge in real-world graphs such as citation networks, co-purchase networks, or friendship networks [18]. A well-used metric for quantifying graph homophily is edge homophily, which is defined as the fraction of intra-class edges:

$$\mathcal{H} = \frac{1}{|\mathcal{E}|} \sum_{(v_i, v_j) \in \mathcal{E}} \mathbb{I}(y_i = y_j), \quad (1)$$

where y_i denotes the class of v_i and \mathbb{I} represents the indicator function. In Table 1, edge homophily values for five benchmark datasets are presented. The table illustrates that the majority of edges are intra-class, indicating the potential to mine positive pairs from node-neighbor pairs.

3.3 Bootstrapped Graph Latents

We first introduce the pioneer work Bootstrapped Graph Latents (BGRL) [25], which aims to maximize the similarity between representations of the same node generated from two different augmented graph views and employs asymmetric architectures to avoid collapsed representations. BGRL consists of three major components: 1) a random graph augmentation generator \mathcal{T} ; 2) two asymmetric

graph encoders, i.e., the online encoder f_θ and the target encoder f_ϕ ; 3) an objective function to maximize the similarity between the positive pair.

Graph View Augmentation. Given the adjacency matrix \mathbf{A} and feature matrix \mathbf{X} of a graph \mathcal{G} , BGRL employs feature masking and edge dropping to enhance both graph attributes and topological information (see Appendix A.3). The augmentation function \mathcal{T} comprises all possible graph transformation operations, and each $t \sim \mathcal{T}$ corresponds to a specific transformation applied to graph \mathcal{G} . At each training epoch, BGRL first samples two random augmentation functions $t^1 \sim \mathcal{T}$ and $t^2 \sim \mathcal{T}$, and then generates two views $\mathcal{G}^1 = (\mathbf{A}^1, \mathbf{X}^1)$ and $\mathcal{G}^2 = (\mathbf{A}^2, \mathbf{X}^2)$ based on the chosen functions.

Node Representations Generation. Different from the classical contrastive learning frameworks with a shared graph encoder, BGRL employs two asymmetric graph encoders to avoid representation collapse. The online encoder f_θ generates an online representations from the first augmented graph, $\mathbf{H}^1 = f_\theta(\mathbf{A}^1, \mathbf{X}^1)$. Similarly, the target encoder f_ϕ produces a target representation of the second augmented graph, $\mathbf{H}^2 = f_\phi(\mathbf{A}^2, \mathbf{X}^2)$. The online representation is then input into a node-level predictor, p_θ (implemented as a MLP), which produces a prediction of the target representation, $\mathbf{Z}^1 = p_\theta(\mathbf{H}^1)$.

Positive Pair Similarity Maximization. The learning process of BGRL centers around maximizing the cosine similarity between the predicted target representations \mathbf{Z}^1 and the true target representations \mathbf{H}^2 , i.e., positive pairs. The objective function is defined as

$$\mathcal{L}_{BGRL} = -\frac{1}{n} \sum_{i=1}^n \frac{\mathbf{z}_i^1 \cdot \mathbf{h}_i^2}{\|\mathbf{z}_i^1\| \|\mathbf{h}_i^2\|}, \quad (2)$$

where (\cdot) denotes the dot production, and $\|\cdot\|$ represents the ℓ_2 normalization. Notably, only the online encoder parameters θ are updated with respected to the gradients from the objective function while the target encoder parameters ϕ are updated as an exponential moving average (EMA) of θ with a decay rate t , i.e., $\phi = t\phi + (1-t)\theta$. Therefore, BGRL utilizes the outputs from the ensemble-optimized parameters as targets, progressively enhancing the model in a step-by-step fashion, an approach commonly known as bootstrapping.

4 Methodology

In this section, we present an overview of the proposed BLNN, as depicted in Fig. 1. In Sect. 4.1, we empirically analyze our motivation to introduce more ground-truth positive pairs from node-neighbor pairs for graph self-supervised learning. Then, we describe how to mine high-confidence positive information from node-neighbor pairs in Sect. 4.2.

4.1 Motivation

As discussed in the introduction, some supervised metric learning methods [10, 31, 33], which employ architectures and objectives similar to self-supervised

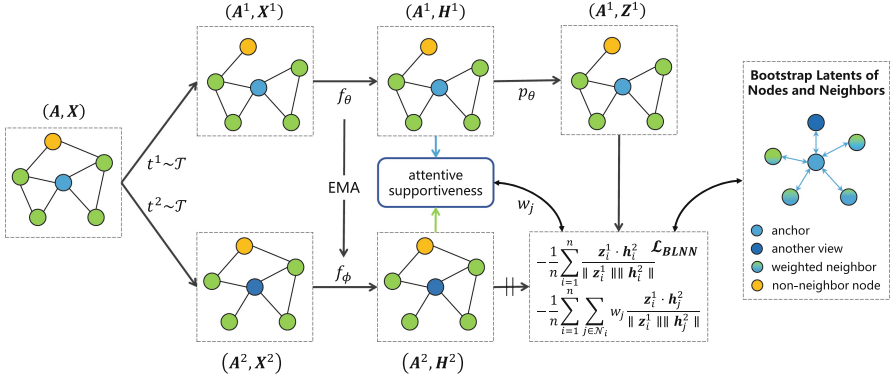


Fig. 1. Overview of our proposed BLNN method. Given a graph, we first generate two different views using augmentations t^1, t^2 . From these, we use encoders f_θ, f_ϕ to form online and target node representations $\mathbf{H}^1, \mathbf{H}^2$. They are then fed into the attention module to compute the supportiveness w_j of the neighbor v_j w.r.t. the anchor node v_i . The predictor p_θ uses \mathbf{H}^1 to form a prediction \mathbf{Z}^1 of the target \mathbf{H}^2 . The final objective is computed as a combination of the alignment of node-itself pairs and the supportiveness-weighted alignment of node-neighbor pairs. Note that the alignment is achieved by maximizing the cosine similarity between corresponding rows of \mathbf{Z}^1 and \mathbf{H}^2 , flowing gradients only through \mathbf{Z}^1 . The target parameters ϕ are updated as an exponentially moving average of θ .

learning, have shown that introducing more ground-truth positive pairs significantly enhances representation learning for classification. Such success inspires us that mining potential positive pairs could empower BGRL to learn highly intra-class-compacted representations, which are more conducive to classification.

Empirical Analysis. To verify our hypothesis, we conduct experiments by incorporating a small subset of the whole ground-truth positive pair set from an oracle perspective and assessing its influence on classification. According to the graph homophily, neighboring nodes often share the same class. Therefore, we first treat all node-neighbor pairs as noisy candidate positive pairs. Subsequently, we manually filter out inter-class pairs, retaining only the intra-class pairs as the clean positive pairs. We then extend the objective function Eq. (2) with an additional alignment of above intra-class node-neighbor pairs to train BGRL. Figure 2 illustrates the results of node classification across three datasets, revealing two key observations: 1) The incorporation of clean positive node-neighbor pairs consistently and significantly improves classification performance. 2) However, simply treating raw node-neighbor pairs as ground-truth positive pairs yields only marginal improvement or even performance degradation, as raw node-neighbor pairs include inter-class pairs, which would cause class collision.

Based on the above observations, we propose to enhance BGRL using two key strategies: 1) expanding the positive pair set with node-neighbor pairs; 2) mitigating class collision caused by false positive node-neighbor pairs via a cross-attention weighting module.

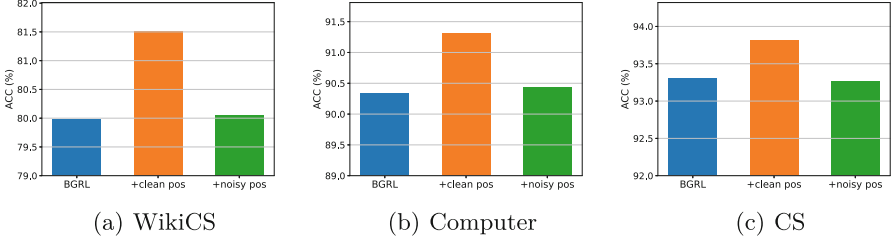


Fig. 2. Empirical studies on WikiCS, Computer and CS. “noisy pos” indicates raw node-neighbor pairs in the input graph, while “clean pos” indicates clean node-neighbor pairs that all are intra-class pairs.

4.2 Bootstrap Latents of Nodes and Neighbors

Motivated by the observations presented in Sect. 4.1, we introduce Bootstrap Latents of Nodes and Neighbors (BLNN) to enhance Bootstrapped Graph Latents (BGRL). We follow the BGRL framework illustrated in Sect. 3.3.

Objective Function. Our BLNN first treats node-neighbor pairs as candidate positive pairs, leveraging the neighbor set \mathcal{N}_i to support the anchor node v_i . Subsequently, it introduces an adaptive measurement of supportiveness through a cross-attention module to mitigate class collision resulting from false positive node-neighbor pairs. Specifically, for each neighbor $v_j \in \mathcal{N}_i$, we input its target representation \mathbf{h}_j^2 and the anchor’s online representation \mathbf{h}_i^1 into the attention module for cross-attention computations. This attention module predicts a supportiveness value w_j , which we use to adjust the contribution of \mathbf{h}_j^2 to the anchor’s prediction \mathbf{z}_i^1 during training. The loss function of our BLNN can be written as:

$$\begin{aligned}
 \mathcal{L}_{BLNN} = & - \underbrace{\frac{1}{n} \sum_{i=1}^n \frac{\mathbf{z}_i^1 \cdot \mathbf{h}_i^2}{\|\mathbf{z}_i^1\| \|\mathbf{h}_i^2\|}}_{\text{Bootstrap Latents of Nodes}} \\
 & - \underbrace{\frac{1}{n} \sum_{i=1}^n \sum_{j \in \mathcal{N}_i} w_j \frac{\mathbf{z}_i^1 \cdot \mathbf{h}_j^2}{\|\mathbf{z}_i^1\| \|\mathbf{h}_j^2\|}}_{\text{Bootstrap Latents of Neighbors}}.
 \end{aligned} \tag{3}$$

Attention Weighting. The attention module, which softly measure the positiveness of node-neighbor pairs, simply consists of a cross-attention operator, and a softmax activation. Formally, given the anchor’s online representation \mathbf{h}_i^1 and its neighboring node’s target representation \mathbf{h}_j^2 , the supportiveness score can be computed as:

$$w_j = \text{softmax}_j(e_{ij}) = \frac{\exp(e_{ij}/\tau)}{\sum_{k \in \mathcal{N}_i} \exp(e_{ik}/\tau)}, \tag{4}$$

where $e_{ij} = \mathbf{h}_i^1 \cdot \mathbf{h}_j^2 / \|\mathbf{h}_i^1\| \|\mathbf{h}_j^2\|$ is the cosine similarity between \mathbf{h}_i^1 and \mathbf{h}_j^2 and τ is a temperature parameter. This attention module assigns higher weights to ground-truth positive node-neighbor pairs than false positive node-neighbor pairs, thus mitigating class collision caused by aligning false node-neighbor pairs.

Comparison with BGRL. Our BLNN enhances BGRL by introducing potential positive node-neighbor pairs in the absence of ground-truth labels. It inherits BGRL’s advantages, such as the negative-free property, which naturally address class collision caused by false negative pairs. Different from the original BGRL framework, which aligns only augmented views with the anchor node, the cross-attention design in BLNN enriches the diversity of positive nodes to support the anchor node in a soft and adaptive manner. This design empowers us to leverage more positive pairs, enhancing intra-class compactness. Additionally, the computations for supportiveness scores and node-neighbor alignment loss exhibit a time complexity linear with the number of edges $\mathcal{O}(|\mathcal{E}|)$. Given the sparsity of real-world graphs, i.e., $\mathcal{O}(|\mathcal{E}|) \ll \mathcal{O}(|\mathcal{V}|^2)$, such complexity increase compared to BGRL is acceptable and our model maintains lower time complexity than contrastive learning baselines [13, 36, 38, 39].

Algorithm 1. Bootstrap Latents of Nodes and Neighbors

Input: $\mathcal{G} = (\mathbf{A}, \mathbf{X})$

Parameter: Temperature τ , BGRL-related hyperparameters

Output: The graph encoder f_θ

- 1: Initialize model parameters;
 - 2: **while** not converge **do**
 - 3: Sample two augmentation functions $t^1, t^2 \sim \mathcal{T}$;
 - 4: Generate augmented views $(\mathbf{A}^1, \mathbf{X}^1), (\mathbf{A}^2, \mathbf{X}^2)$;
 - 5: Obtain online representations $\mathbf{H}^1 = f_\theta(\mathbf{A}^1, \mathbf{X}^1)$;
 - 6: Obtain target representations $\mathbf{H}^2 = f_\phi(\mathbf{A}^2, \mathbf{X}^2)$;
 - 7: Compute positiveness scores of node-neighbor pairs via Eq. (4);
 - 8: Predict the target representations $\mathbf{Z}^1 = p_\theta(\mathbf{H}^1)$;
 - 9: Calculate the objective function via Eq. (3);
 - 10: Update the parameters of f_θ, p_θ via SGD;
 - 11: Update the parameters of f_ϕ via an EMA of f_θ ;
 - 12: **end while**
 - 13: **return** f_θ .
-

5 Experiments

In this section, we design the experiments to evaluate our proposed BLNN and answer the following research questions. **RQ1:** Does BLNN outperform existing baseline methods on node classification, node clustering, and node similarity search? **RQ2:** How does each component of BLNN benefit the performance? **RQ3:** Can the supportiveness score measure the positiveness of node-neighbor

pairs? **RQ4:** Is BLNN sensitive to the hyperparameter τ ? **RQ5:** How to intuitively understand BLNN can enhance intra-class compactness of learned representations?

5.1 Experiment Setup

Datasets. We adopt five publicly available real-world benchmark datasets, including one reference network WikiCS [19], two co-purchase networks Photo, Computer [23], and two co-authorship networks CS, Physics [23] to conduct the experiments throughout the paper. The statistics of the datasets are provided in Table 1. More details can be found in Appendix A.1.

Table 1. Dataset statistics. \mathcal{H} is the fraction of intra-class node-neighbor pairs.

Dataset	#Nodes	#Edges	#Feats	#Classes	\mathcal{H} (%)
WikiCS	11,701	431,726	300	10	65.47
Photo	7,650	238,163	745	8	82.72
Computer	13,752	491,722	767	10	77.72
CS	18,333	163,788	6,805	15	80.81
Physics	34,493	495,924	8,415	5	93.14

Baselines. We compare BLNN with a variety of baselines, including supervised methods MLP, GCN [11], and GAT [26]; contrastive methods DGI [27], MVGRL [6], GRACE [38], GCA [39], AF-GCL [28], COSTA [36], FastGCL [30], gCooL [13], ProGCL [32], and CGKS [35]; non-contrastive methods CCA-SSG [34], GBT [1], AFGRL [12], GraphMAE [8], and BGRL [25]. All the baseline results are taken from previously published papers. And brief introductions of the baselines can be found in Appendix A.2.

Evaluation Protocol. We evaluate BLNN on three tasks, i.e., node classification, node clustering and node similarity search. We first train the model in an unsupervised manner. For node classification, we use the learned representations to train and test a simple logistic regression classifier with twenty 1:1:8 train/validation/test random splits (twenty public splits for WikiCS) [25]. We apply K-means to the learned representations, initializing the cluster numbers with fixed values. For node similarity search, we use pairwise cosine similarity to identify nearest node neighbors [12]. Evaluations are conducted at every 250 epochs, and we report the best results [12, 25].

Metrics. Following AFGRL [12], we use accuracy for node classification, normalized mutual information (NMI) and homogeneity (Hom.) for node clustering. For node similarity search, we introduce $S@k$, which is average ratio among the k nearest neighbors sharing the same label as the query node. Formulas of these metrics can be found in Appendix A.4.

Implementation Details. Since our BLNN is derived from BGRL, we implement BLNN based on the official code¹ of BGRL. To ensure a fair comparison, all BGRL-related hyperparameters are the same as those specified in the original BGRL paper. We perform a grid-search on the introduced temperature hyperparameter τ . All experiments are conducted on a 32 GB V100 GPU. Our implementation code is available at <https://github.com/Cloudy1225/BLNN>. More details can be found in Appendix A.5.

Table 2. Node classification results measured by accuracy along with standard deviations. The baseline results are taken from previously published papers. ‘-’ denotes the absence of the result in the original paper. The *Input* column illustrates the data used in the training stage, and \mathbf{Y} denotes labels.

Method	Input	WikiCS	Photo	Computer	CS	Physics
MLP	\mathbf{X}, \mathbf{Y}	71.98 ± 0.00	78.53 ± 0.00	73.81 ± 0.00	90.37 ± 0.00	93.58 ± 0.00
GCN	$\mathbf{A}, \mathbf{X}, \mathbf{Y}$	77.19 ± 0.12	92.42 ± 0.22	86.51 ± 0.54	93.03 ± 0.31	95.65 ± 0.16
GAT	$\mathbf{A}, \mathbf{X}, \mathbf{Y}$	77.65 ± 0.11	92.56 ± 0.35	86.93 ± 0.29	92.31 ± 0.24	95.47 ± 0.15
DGI	\mathbf{A}, \mathbf{X}	78.25 ± 0.56	91.69 ± 1.07	87.98 ± 0.81	92.15 ± 0.63	94.51 ± 0.52
MVGRL	\mathbf{A}, \mathbf{X}	77.57 ± 0.46	92.04 ± 0.98	87.39 ± 0.92	92.11 ± 0.12	95.33 ± 0.03
GRACE	\mathbf{A}, \mathbf{X}	78.64 ± 0.33	92.46 ± 0.18	88.29 ± 0.11	92.17 ± 0.04	95.26 ± 0.22
GCA	\mathbf{A}, \mathbf{X}	78.35 ± 0.05	92.53 ± 0.16	87.85 ± 0.31	93.10 ± 0.01	95.68 ± 0.05
AF-GCL	\mathbf{A}, \mathbf{X}	79.01 ± 0.51	92.49 ± 0.31	89.68 ± 0.19	91.92 ± 0.10	95.12 ± 0.15
COSTA	\mathbf{A}, \mathbf{X}	79.12 ± 0.02	92.56 ± 0.45	88.32 ± 0.03	92.94 ± 0.10	95.60 ± 0.02
FastGCL	\mathbf{A}, \mathbf{X}	79.20 ± 0.07	92.91 ± 0.07	89.35 ± 0.09	92.71 ± 0.07	95.53 ± 0.02
gCool	\mathbf{A}, \mathbf{X}	78.74 ± 0.04	93.18 ± 0.12	88.85 ± 0.14	93.32 ± 0.02	-
ProGCL	\mathbf{A}, \mathbf{X}	78.68 ± 0.12	93.30 ± 0.09	89.28 ± 0.15	93.51 ± 0.06	-
CGKS	\mathbf{A}, \mathbf{X}	79.20 ± 0.10	92.40 ± 0.10	88.50 ± 0.20	93.00 ± 0.20	-
CCA-SSG	\mathbf{A}, \mathbf{X}	79.08 ± 0.53	93.14 ± 0.14	88.74 ± 0.28	93.32 ± 0.22	95.38 ± 0.06
G-BT	\mathbf{A}, \mathbf{X}	76.83 ± 0.73	92.46 ± 0.35	87.93 ± 0.36	92.91 ± 0.25	95.25 ± 0.13
AFGRL	\mathbf{A}, \mathbf{X}	77.62 ± 0.49	93.22 ± 0.28	89.88 ± 0.33	93.27 ± 0.17	95.69 ± 0.10
GraphMAE	\mathbf{A}, \mathbf{X}	79.54 ± 0.58	92.98 ± 0.35	89.88 ± 0.10	93.08 ± 0.17	95.40 ± 0.06
BGRL	\mathbf{A}, \mathbf{X}	79.98 ± 0.10	93.17 ± 0.30	90.34 ± 0.19	93.31 ± 0.13	95.73 ± 0.05
BLNN	\mathbf{A}, \mathbf{X}	80.48 ± 0.52	93.54 ± 0.23	91.02 ± 0.23	93.61 ± 0.15	95.86 ± 0.10

5.2 Experiment Results

Performance Analysis (RQ1). The experimental results of node classification are presented in Table 2, revealing that our BLNN outperforms both self-supervised and even supervised baselines. This superiority can be attributed to two primary factors: 1) The pioneering BGRL of BLNN can effectively learn discriminative node representations, achieving competitive performance. 2) BLNN introduces additional potential positive pairs, enhancing the intra-class compactness of representations learned by BGRL. Node clustering results are detailed

¹ <https://github.com/nerdslab/bgml>.

in Table 3, demonstrating BLNN’s superior performance across four datasets, except Physics. Notably, BLNN exhibits significant improvement over BGRL, especially on WikiCS, Computer and Physics, with an increase ranging from 5% to 8%. These enhancements underscore the effectiveness of incorporating positive node-neighbor pairs to generate more intra-class compact representations. Table 4 illustrates the node similarity search results, with BLNN demonstrating the best performance. This outcome aligns with expectations, as BLNN is designed to softly pull together representations of nodes and their neighbors, where neighboring nodes often share the same label in graphs.

Table 3. Performance on node clustering. The baseline results are taken from the published AFGRL paper.

Dataset	WikiCS		Photo		Computer		CS		Physics	
	NMI	Hom.	NMI	Hom.	NMI	Hom.	NMI	Hom.	NMI	Hom.
GRACE	42.82	44.23	65.13	66.57	47.93	52.22	75.62	79.09	-	-
GCA	33.73	35.25	64.43	65.75	52.78	58.16	76.20	79.65	-	-
AFGRL	41.32	43.07	65.63	67.43	55.20	60.40	78.59	81.61	72.89	73.54
BGRL	39.69	41.56	68.41	70.04	53.64	58.69	77.32	80.41	55.68	60.18
BLNN	47.17	49.11	71.05	72.18	58.79	64.33	78.97	82.08	62.41	67.39

Table 4. Performance on node similarity search. The baseline results are taken from the published AFGRL paper.

Dataset	WikiCS		Photo		Computer		CS		Physics	
	S@5	S@10	S@5	S@10	S@5	S@10	S@5	S@10	S@5	S@10
GRACE	77.54	76.45	91.55	91.06	87.38	86.43	91.04	90.59	-	-
GCA	77.86	76.73	91.12	90.52	88.26	87.42	91.26	91.00	-	-
AFGRL	78.11	76.60	92.36	91.73	89.66	88.90	91.80	91.42	95.25	94.86
BGRL	77.39	76.17	92.45	91.95	89.47	88.55	91.12	90.86	95.04	94.64
BLNN	80.27	79.04	92.61	91.96	89.91	89.12	91.90	91.59	95.39	95.01

Ablation Studies (RQ2). To verify the benefit of each component of BLNN, we conduct ablation studies with different variants of BGRL: BGRL with raw noisy node-neighbor pairs (BGRL_{noisy}), BGRL with clean node-neighbor pairs (BGRL_{clean}), and our proposed BLNN (BGRL with supportiveness-weighted node-neighbor pairs). Results are reported in Table 5. We can find that simply treating raw node-neighbor pairs as ground-truth positive pairs results in only marginal improvement or even performance degradation, as raw node-neighbor pairs include inter-class pairs, which would cause class collision. Our supportiveness weighting strategy, implemented through an attention module, effectively

mitigates this class collision, yielding superior performance. However, there is still a gap between our BLNN and the ideal solution $\text{BGRL}_{\text{clean}}$, which necessitates the availability of all labels. These results further confirm our motivation described in Sect. 4.1.

Table 5. Ablation study on node classification.

Variant	WikiCS	Photo	Computer	CS	Physics
BGRL	79.98	93.17	90.34	93.31	95.73
BLNN	80.48	93.54	91.02	93.61	95.86
$\text{BGRL}_{\text{noisy}}$	80.05	93.33	90.44	93.27	95.59
$\text{BGRL}_{\text{clean}}$	81.51	93.66	91.31	93.92	95.98

Case Study (RQ3). Our attention module is implemented based on cosine similarities of node-neighbor pairs and is expected to assign higher weights to true positive node-neighbor pairs than false positive pairs. Here, we conduct a twofold case study on Computer to verify that: 1) node-neighbor pairs with higher cosine similarity tend to share the same label; 2) our attention module indeed assigns higher weights to true positive node-neighbor pairs. We first sort all node-neighbor pairs based on the learned cosine similarity and then divide them into intervals of size 10,000 to compute the homophily in each interval. As shown in Fig. 3(a), the cosine similarity effectively estimates the probability of neighbor nodes being positive, with more similar node-neighbor pairs exhibiting larger homophily, which validates the efficacy of leveraging cosine similarity in our attention module. Moreover, we select an anchor node with 949 neighbors, sorting all anchor-neighbor pairs according to the supportiveness weights predicted by the attention module. We also partition them into intervals of size 50 to calculate homophily within each interval. As shown in Fig. 3(b), our attention module generally assigns higher weights to true positive node-neighbor pairs compared to false positive pairs.

Hyperparameter Analysis (RQ4). We investigate the impact of the temperature τ in Eq. (4) on node classification by varying τ from 0.1 to 2.0 in increments of 0.1. Figure 4 presents the ACC scores on Photo, Computer and CS. It is observed that, our BLNN almost always achieves better performance than BGRL with respect to different τ . In general, BLNN exhibits robustness to the temperature τ . Analysis for BGRL-related hyperparameters can be found in the original BGRL paper [25].

Visualization and Compactness of Representations (RQ5). To gain a more intuitive insight into node representations, we provide the t-SNE [17] visualizations of the raw features and representations learned by BGRL and BLNN, along with intra-class compactness score on Computer. The intra-class compactness score is defined as the mean cosine similarity among all intra-class node

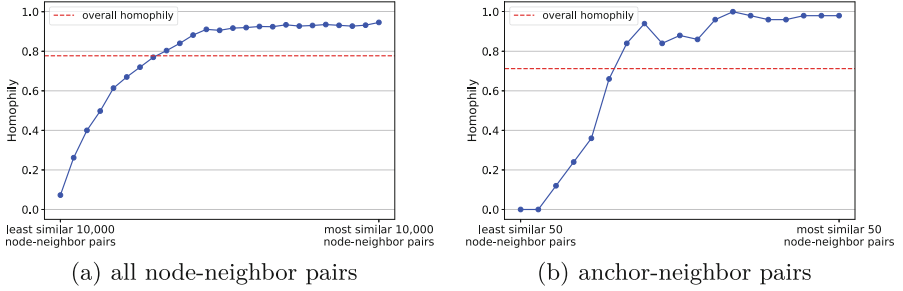


Fig. 3. Case study to verify the efficacy of our attention module.

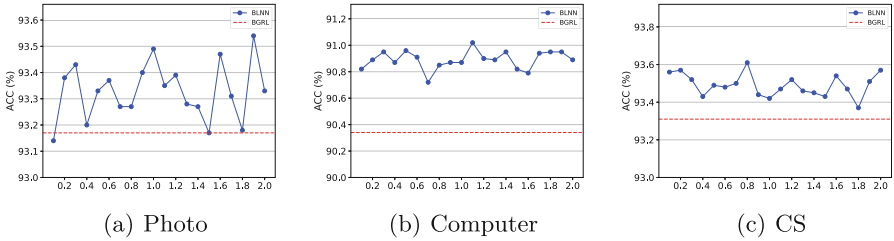


Fig. 4. Visualization of the impact of τ on node classification.

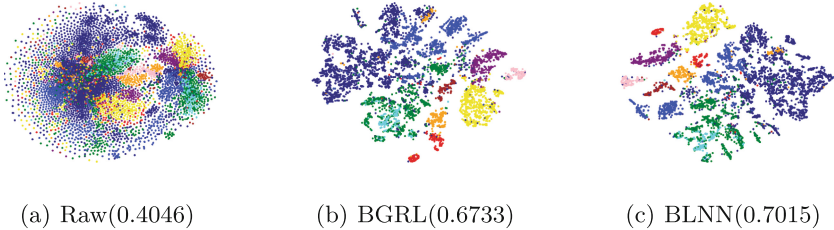


Fig. 5. t-SNE visualization and intra-class compactness of node representations on Computer. ‘(*)’ indicates the mean intra-class pair-wise cosine similarity.

pairs (the formula can be found in Appendix A.4). As shown in Fig. 5, the representations learned by BLNN exhibit higher intra-class compactness, thus underscoring the effectiveness of mining positive node-neighbor pairs.

6 Conclusion

In this paper, we introduce Bootstrap Latents of Nodes and Neighbors (BLNN). Our proposal is motivated by the empirical observation that introducing ground-truth positive node-neighbor pairs can yield significant improvements for BGRL. We thus expand the positive pair set with node-neighbor pairs and propose a

cross-attention module to weight the contribution of each neighbor to loss computations. This module prioritizes higher weights for ground-truth positive node-neighbor pairs compared to false positive node-neighbor pairs, thereby alleviating class collision resulting from the alignment of false node-neighbor pairs. Extensive experiments demonstrate that our BLNN effectively improves the intra-class compactness of learned representations, establishing its state-of-the-art performance in three downstream tasks across five benchmark datasets.

Acknowledgments. This work is partially supported by the National Key Research and Development Program of China (2021YFB1715600), and the National Natural Science Foundation of China (62306137).

References

1. Bielak, P., Kajdanowicz, T., Chawla, N.V.: Graph Barlow twins: a self-supervised representation learning framework for graphs. *Knowl.-Based Syst.* **256**, 109631 (2022)
2. Chen, H., Zhao, Z., Li, Y., Zou, Y., Li, R., Zhang, R.: CSGCL: community-strength-enhanced graph contrastive learning. In: *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pp. 2059–2067 (2023)
3. Dwibedi, D., Aytar, Y., Tompson, J., Sermanet, P., Zisserman, A.: With a little help from my friends: Nearest-neighbor contrastive learning of visual representations. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9588–9597 (2021)
4. GE, C., Wang, J., Tong, Z., Chen, S., Song, Y., Luo, P.: Soft neighbors are positive supporters in contrastive visual representation learning. In: *The Eleventh International Conference on Learning Representations* (2023)
5. Grill, J.B., et al.: Bootstrap your own latent a new approach to self-supervised learning. In: *Proceedings of the 34th International Conference on Neural Information Processing Systems. NIPS’20* (2020)
6. Hassani, K., Khasahmadi, A.H.: Contrastive multi-view representation learning on graphs. In: *Proceedings of the 37th International Conference on Machine Learning. ICML’20* (2020)
7. Hjelm, R.D., et al.: Learning deep representations by mutual information estimation and maximization. In: *International Conference on Learning Representations* (2019)
8. Hou, Z., et al.: GraphMAE: self-supervised masked graph autoencoders. In: *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 594–604 (2022)
9. Ji, W., Deng, Z., Nakada, R., Zou, J., Zhang, L.: The power of contrast for feature learning: a theoretical analysis. *J. Mach. Learn. Res.* **24**(330), 1–78 (2023)
10. Khosla, P., et al.: Supervised contrastive learning. In: *Advances in Neural Information Processing Systems*, vol. 33, pp. 18661–18673 (2020)
11. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: *International Conference on Learning Representations* (2017)
12. Lee, N., Lee, J., Park, C.: Augmentation-free self-supervised learning on graphs. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 7372–7380 (2022)

13. Li, B., Jing, B., Tong, H.: Graph communal contrastive learning. In: Proceedings of the ACM Web Conference 2022. WWW'22, pp. 1203–1213 (2022)
14. Li, H., Cao, J., Zhu, J., Luo, Q., He, S., Wang, X.: Augmentation-free graph contrastive learning of invariant-discriminative representations. *IEEE Trans. Neural Netw. Learn. Syst.* (2023)
15. Li, W.Z., Wang, C.D., Xiong, H., Lai, J.H.: HomoGCL: rethinking homophily in graph contrastive learning. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. KDD '23, pp. 1341–1352. Association for Computing Machinery, New York, NY, USA (2023)
16. Liu, Y., et al.: Graph self-supervised learning: a survey. *IEEE Trans. Knowl. Data Eng.* **35**(6), 5879–5900 (2022)
17. van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**(86), 2579–2605 (2008)
18. McPherson, M., Smith-Lovin, L., Cook, J.M.: Birds of a feather: homophily in social networks. *Rev. Sociol.* **27**, 415–444 (2001)
19. Mernyei, P., Cangea, C.: Wiki-cs: a Wikipedia-based benchmark for graph neural networks. *arXiv abs/2007.02901* (2020)
20. Nowozin, S., Cseke, B., Tomioka, R.: f-GAN: training generative neural samplers using variational divergence minimization. IN: *Advances in Neural Information Processing Systems*, 29 (2016)
21. Peng, Z., et al.: Graph representation learning via graphical mutual information maximization. In: *Proceedings of the Web Conference 2020*, pp. 259–270 (2020)
22. Saunshi, N., Plevrakis, O., Arora, S., Khodak, M., Khandeparkar, H.: A theoretical analysis of contrastive unsupervised representation learning. In: *International Conference on Machine Learning*, pp. 5628–5637. PMLR (2019)
23. Shchur, O., Mumme, M., Bojchevski, A., Günnemann, S.: Pitfalls of graph neural network evaluation. IN: *Relational Representation Learning Workshop. NeurIPS 2018* (2018)
24. Sun, W., Li, J., Chen, L., Wu, B., Bian, Y., Zheng, Z.: Rethinking and simplifying bootstrapped graph latents. In: *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pp. 665–673 (2024)
25. Thakoor, S., et al.: Large-scale representation learning on graphs via bootstrapping. In: *International Conference on Learning Representations* (2022)
26. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: *International Conference on Learning Representations* (2018)
27. Veličković, P., Fedus, W., Hamilton, W.L., Liò, P., Bengio, Y., Hjelm, R.D.: Deep graph infomax. In: *International Conference on Learning Representations* (2019)
28. Wang, H., Zhang, J., Zhu, Q., Huang, W.: Augmentation-free graph contrastive learning with performance guarantee. *arXiv preprint arXiv:2204.04874* (2022)
29. Wang, T., Isola, P.: Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In: *Proceedings of the 37th International Conference on Machine Learning*, vol. 119, pp. 9929–9939 (2020)
30. Wang, Y., Sun, W., Xu, K., Zhu, Z., Chen, L., Zheng, Z.: FastGCL: fast self-supervised learning on graphs via contrastive neighborhood aggregation (2022)
31. Wen, Y., et al.: Pairwise similarity learning is simple. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5308–5318 (2023)
32. Xia, J., Wu, L., Wang, G., Chen, J., Li, S.Z.: ProGCL: rethinking hard negative mining in graph contrastive learning. In: *International Conference on Machine Learning* (2021)

33. Yi, L., Liu, S., She, Q., McLeod, A.I., Wang, B.: On learning contrastive representations for learning with noisy labels. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 16682–16691 (2022)
34. Zhang, H., Wu, Q., Yan, J., Wipf, D., Yu, P.S.: From canonical correlation analysis to self-supervised graph neural networks. In: Advances in Neural Information Processing Systems, vol. 34, pp. 76–89 (2021)
35. Zhang, Y., Chen, Y., Song, Z., King, I.: Contrastive cross-scale graph knowledge synergy. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 3422–3433 (2023)
36. Zhang, Y., Zhu, H., Song, Z., Koniusz, P., King, I.: Costa: covariance-preserving feature augmentation for graph contrastive learning. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (2022)
37. Zheng, M., et al.: Weakly supervised contrastive learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 10042–10051 (2021)
38. Zhu, Y., Xu, Y., Yu, F., Liu, Q., Wu, S., Wang, L.: Deep graph contrastive representation learning. arXiv abs/2006.04131 (2020)
39. Zhu, Y., Xu, Y., Yu, F., Liu, Q., Wu, S., Wang, L.: Graph contrastive learning with adaptive augmentation. In: Proceedings of the Web Conference 2021. WWW’21, pp. 2069–2080 (2021)



Deep Sketched Output Kernel Regression for Structured Prediction

Tamim El Ahmad^{1(✉)}, Junjie Yang¹, Pierre Laforgue²,
and Florence d'Alché-Buc¹

¹ LTCI, Télécom Paris, IP Paris, Palaiseau, France

tamim.elahmad@telecom-paris.fr

² Department of Computer Science, University of Milan, Milan, Italy

pierre.laforgue@unimi.it

Abstract. By leveraging the kernel trick in the output space, kernel-induced losses provide a principled way to define structured output prediction tasks for a wide variety of output modalities. In particular, they have been successfully used in the context of surrogate non-parametric regression, where the kernel trick is typically exploited in the input space as well. However, when inputs are images or texts, more expressive models such as deep neural networks seem more suited than non-parametric methods. In this work, we tackle the question of how to train neural networks to solve structured output prediction tasks, while still benefiting from the versatility and relevance of kernel-induced losses. We design a novel family of deep neural architectures, whose last layer predicts in a data-dependent finite-dimensional subspace of the infinite-dimensional output feature space deriving from the kernel-induced loss. This subspace is chosen as the span of the eigenfunctions of a randomly-approximated version of the empirical kernel covariance operator. Interestingly, this approach unlocks the use of gradient descent algorithms (and consequently of any neural architecture) for structured prediction. Experiments on synthetic tasks as well as real-world supervised graph prediction problems show the relevance of our method.

Keywords: Structured prediction · Deep learning · Kernel methods

1 Introduction

Learning to predict complex outputs, such as graphs or any other composite object, raises many challenges in machine learning [3, 19, 51]. The most important of them is undoubtedly the difficulty of leveraging the geometry of the

T. El Ahmad and J. Yang—Equal contribution.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70352-2_6.

output space. In supervised graph prediction, for instance, it is often required to use node permutation-invariant and node size-insensitive distances, such as the Fused Gromov-Wasserstein distance [69]. In that regard, surrogate methods such as Output Kernel Regression [25, 33, 71] offer a powerful and flexible framework by using the kernel trick in the output space. By appropriately choosing the output kernel, it is possible to incorporate various kinds of information, both in the model and in the loss function [13, 15, 49]. One important limitation of this approach, however, is that the induced output features may be infinite-dimensional.

If leveraging the kernel trick in the input space may be a solution [12, 16], such non-parametric methods are usually outperformed by more expressive models such as neural networks when input data consist of images or texts. In the context of structured prediction, deep learning has led to impressive results for specific tasks, such as semantic segmentation [37] or the protein 3D structure prediction [32]. To create versatile deep models, the main approach explored in the literature is the energy-based approach, which consists of converting structured prediction into learning a scalar score function [4, 27, 42, 43]. However, these methods usually fail to go beyond structured prediction problems which can be reformulated as high-dimensional multi-label classification problems, as pointed out by [26]. Besides, this approach requires a two-step strategy, since the energy function is first learned thanks to the training data, and then maximized at inference time. To obtain an end-to-end model, [5] uses direct risk minimization techniques, and [67] introduces inference networks, a neural architecture that approximates the inference problem. In this work, we choose to benefit from the versatility of kernel-induced losses, and deploy it to neural networks. To this end, we address the infinite-dimensionality of the output features by computing a finite-dimensional basis within the output feature space, defined as the eigenbasis of a sketched version of the output empirical covariance operator.

Sketching [45, 73] is a dimension-reduction technique based on random linear projections. In the context of kernel methods, it has mainly been explored through the so-called Nyström approximation [56, 72], or via specific distributions such as Gaussian or Randomized Orthogonal Systems [40, 75]. Previous works tackle sketched scalar kernel regression by providing a low-rank approximation of the Gram matrix [2, 20], reducing the number of parameters to learn at the optimization stage [40, 75], providing data-dependent random features [39, 72, 74], or leveraging an orthogonal projection operator in the feature space [56]. This last interpretation has been used to learn large-scale dynamical systems [47], and structured prediction [22].

In our proposition to solve structured prediction from complex input data, we make the following contributions:

- We introduce Deep Sketched Output Kernel Regression, a novel family of deep neural architectures whose last layer predicts a data-dependent finite-dimensional representation of the outputs, that lies in the infinite-dimensional feature space deriving from the kernel-induced loss.

- This last layer is computed beforehand, and is the eigenbasis of the sketched empirical covariance operator, unlocking the use of gradient-based techniques to learn the weights of the previous layers for any neural architecture.
- We empirically show the relevance of our approach on a synthetic least squares regression problem, and provide a strategy to select the sketching size.
- We show that DSOKR performs well on two text-to-molecule datasets.

2 Deep Sketched Output Kernel Regression

In this section, we set up the problem of structured prediction. Specifically, we consider surrogate regression approaches for kernel-induced losses. By introducing a last layer able to make predictions in a Reproducing Kernel Hilbert Space (RKHS), we unlock the use of deep neural networks as hypothesis space.

Consider the general regression task from an input domain \mathcal{X} to a structured output domain \mathcal{Y} (e.g., the set of labeled graphs of arbitrary size). Learning a mapping from \mathcal{X} to \mathcal{Y} naturally requires taking into account the structure of the output space. One way to do so is the *Output Kernel Regression* (OKR) framework [10, 12, 16, 25, 71], which is part of the family of surrogate regression methods [14, 15].

Output Kernel Regression. A positive definite (p.d.) kernel $k : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$ is a symmetric function such that for all $n \geq 1$, and any $(y_i)_{i=1}^n \in \mathcal{Y}^n$, $(\alpha_i)_{i=1}^n \in \mathbb{R}^n$, we have $\sum_{i,j=1}^n \alpha_i k(y_i, y_j) \alpha_j \geq 0$. Such a kernel is associated with a canonical feature map $\psi : y \in \mathcal{Y} \mapsto k(\cdot, y)$, which is uniquely associated with a Hilbert space of functions $\mathcal{H} \subset \mathbb{R}^{\mathcal{Y}}$, the RKHS, such that $\psi(y) \in \mathcal{H}$ for all $y \in \mathcal{Y}$, and $h(y) = \langle h, \psi(y) \rangle_{\mathcal{H}}$ for any $(h, y) \in \mathcal{H} \times \mathcal{Y}$. Given a p.d. kernel k , ψ its canonical feature map and \mathcal{H} its RKHS, the OKR approach that we consider in this work exploits the kernel-induced squared loss:

$$\Delta(y, y') := \|\psi(y) - \psi(y')\|_{\mathcal{H}}^2 = k(y, y) - 2k(y, y') + k(y', y'). \quad (1)$$

The versatility of loss (1) stems from the large variety of kernels that have been designed to compare structured objects [7, 24, 38]. In multi-label classification, for instance, choosing the linear kernel or the Tanimoto kernel induces respectively the Hamming and the F1-loss [65]. In label ranking, Kemeny and Hamming embeddings define respectively Kendall’s τ distance and the Hamming loss [38, 50]. For sequence prediction tasks, n-gram kernels have been proven useful [17, 33, 50], while an abundant collection of kernels has been designed for graphs, based either on bags of structures or information propagation, see Appendix B and [7] for examples.

If kernel-induced losses can be computed easily thanks to the kernel trick, note that most of them are however non-differentiable. In particular, this largely compromises their use within deep neural architectures, that are however key to achieve state-of-the-art performances in many applications. In this work, we close this gap and propose an approach that benefits from both the expressivity of neural networks for input image/textual data, as well as the relevance of

kernel-induced losses for structured outputs. Formally, let ρ be a joint probability distribution on $\mathcal{X} \times \mathcal{Y}$. Our goal is to design a family $(f_\theta)_{\theta \in \Theta} \subset \mathcal{Y}^{\mathcal{X}}$ of neural networks with outputs in \mathcal{Y} that can minimize the kernel-induced loss, i.e., that can solve

$$\min_{\theta \in \Theta} \mathbb{E}_{(x,y) \sim \rho} \left[\|\psi(y) - \psi(f_\theta(x))\|_{\mathcal{H}}^2 \right]. \tag{2}$$

To do so, we assume that we can access a training sample $\{(x_1, y_1), \dots, (x_n, y_n)\}$

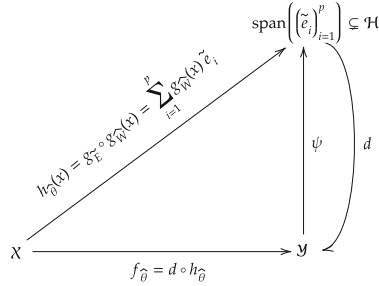


Fig. 1. Illustration of DSOKR model.

drawn i.i.d. from ρ . Since learning f_θ through ψ is difficult, we employ a two-step method. First, we solve the surrogate empirical problem

$$\hat{\theta} \in \arg \min_{\theta \in \Theta} L(\theta) = \arg \min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^n \|h_\theta(x_i) - \psi(y_i)\|_{\mathcal{H}}^2, \tag{3}$$

where $(h_\theta)_{\theta \in \Theta} \subset \mathcal{H}^{\mathcal{X}}$ is a family of neural networks with outputs in \mathcal{H} . We then retrieve the solution by solving for any prediction the pre-image problem

$$f_{\hat{\theta}}(x) = \arg \min_{y \in \mathcal{Y}} \|h_{\hat{\theta}}(x) - \psi(y)\|_{\mathcal{H}}^2. \tag{4}$$

This approach nonetheless raises a major challenge. Indeed, the dimension of the canonical feature space \mathcal{H} may be infinite, making the training very difficult. The question we have to answer now is: *how can we design a neural architecture that is able to learn infinite-dimensional output kernel features?*

Neural Networks with Infinite-Dimensional Outputs. We propose a novel architecture of neural networks to compute the function h_θ with values in \mathcal{H} , as illustrated in Fig. 1. Let $p \geq 1$, our architecture is the composition of two networks: an input neural network, denoted $g_W: \mathcal{X} \rightarrow \mathbb{R}^p$, with generic weights $W \in \mathcal{W}$, and a last layer composed of a unique *functional* neuron, denoted $g_E: \mathbb{R}^p \rightarrow \mathcal{H}$, that predicts in \mathcal{H} . The latter depends on the kernel k used in the loss definition, and on a finite basis $E = ((e_j)_{j=1}^p) \in \mathcal{H}^p$ of elements in \mathcal{H} . We let $\theta = (W, E)$, and for any $x \in \mathcal{X}$, we have

$$h_\theta(x) := g_E \circ g_W(x), \quad (5)$$

where g_W typically implements a $L - 1$ neural architecture encompassing, multilayered perceptrons, convolutional neural networks, or transformers. Instead, g_E computes a linear combination of some basis functions $E = (e_j)_{j=1}^p \in \mathcal{H}^p$

$$g_E : z \in \mathbb{R}^p \mapsto \sum_{j=1}^p z_j e_j \in \mathcal{H}. \quad (6)$$

With this architecture, computations remain finite, and the input neural network outputs the coefficients of the basis expansion, generating predictions in \mathcal{H} .

Remark 1 (Input Neural net's last layers). Since the neural network g_W learns the coordinates of the surrogate estimator in the basis, its last layers are always mere fully connected ones, regardless of the nature of the output data at hand.

2.1 Learning Neural Networks with Infinite-Dimensional Outputs

Learning the surrogate regression model h_θ now boils down to computing $\theta = (W, E)$. We propose to solve this problem in two steps. First, we learn a suitable E using only the output training data $(\psi(y_i))_{i=1}^n$ in an unsupervised fashion. Then, we use standard gradient-based algorithms to learn W through the frozen last layer, minimizing the loss on the whole supervised training sample $(x_i, \psi(y_i))_{i=1}^n$.

Estimating the Functional Last Unit g_E . A very first idea is to choose E as the non-orthogonal dictionary $\psi(y_j)_{j=1}^n$. But this choice induces a very large output dimension (namely, $p = n$) for large training datasets.

An alternative consists in using Kernel Principal Component Analysis (KPCA) [58]. Given a marginal probability distribution over \mathcal{Y} , let $C = \mathbb{E}_y[\psi(y) \otimes \psi(y)]$ be the covariance operator associated with k , and $\hat{C} = (1/n) \sum_{i=1}^n \psi(y_i) \otimes \psi(y_i)$ its empirical counterpart. Let S be the sampling operator that transforms a function $f \in \mathcal{H}$ into the vector $(1/\sqrt{n})(f(x_1), \dots, f(x_n))^T$ in \mathbb{R}^n , and denote by $S^\#$ its adjoint. We have $S^\# : \alpha \in \mathbb{R}^n \mapsto (1/\sqrt{n}) \sum_{i=1}^n \alpha_i \psi(y_i) \in \mathcal{H}$, and $\hat{C} = S^\# S$. KPCA provides the eigenbasis of \hat{C} by computing the SVD of the output Gram matrix, for a prohibitive computational cost of $\mathcal{O}(n^3)$. In practice, though, it is often the case that the so-called *capacity condition* holds [15, 22], i.e., that the spectrum of the empirical covariance operator enjoys a large eigendecay. It is then possible to efficiently approximate the eigenbasis of \hat{C} using random projections techniques [45], also known as sketching, solving this way the computational and memory issues.

Sketching for Kernel Methods. Sketching [73] is a dimension reduction technique based on random linear projections. Since the goal is to reduce the dependency on the number of training samples n in kernel methods, such linear projections can be encoded by a randomly drawn matrix $R \in \mathbb{R}^{m \times n}$, where $m \ll n$.

Standard examples include Nyström approximation [46], where each row of R is randomly drawn from the rows of the identity matrix I_n , also called sub-sampling sketches, and Gaussian sketches [75], where all entries of R are i.i.d. Gaussian random variables. As they act as a random training data sub-sampler and then largely reduce both the time and space complexities induced by kernel methods, sub-sampling sketches are the most popular sketching type applied to kernels, while Gaussian sketches are less computationally efficient but offer better statistical properties. Hence, given a sketching matrix $R \in \mathbb{R}^{m \times n}$, one can define $\tilde{\mathcal{H}}_{\mathcal{Y}} = \text{span}((\sum_{j=1}^n R_{ij} \psi(y_j))_{i=1}^m)$ which is a low-dimensional linear subspace of \mathcal{H} of dimension at most m . One can even compute the basis \tilde{E} of $\tilde{\mathcal{H}}_{\mathcal{Y}}$, providing the last layer $g_{\tilde{E}}$.

Sketching to Estimate g_E . We here show how to compute the basis \tilde{E} of $\tilde{\mathcal{H}}_{\mathcal{Y}}$. Let $m < n$, and $R \in \mathbb{R}^{m \times n}$ be a sketching matrix. Let $\tilde{K} = RKR^T \in \mathbb{R}^{m \times m}$ be the sketched Gram matrix, and $\{(\sigma_i(\tilde{K}), \tilde{v}_i), i \in [m]\}$ its eigenpairs, in descending order. We set $p = \text{rank}(\tilde{K})$. Note that $p \leq m$, and that $p = m$ for classical examples, e.g. full-rank K and sub-sample without replacement or Gaussian R . The following proposition provides the eigenfunctions of the sketched empirical covariance operator.

Proposition 1. [22, Proposition 2] *The eigenfunctions of the sketched empirical covariance operator $\tilde{C} = S^\# R^T R S$ are the $\tilde{e}_j = \sqrt{\frac{n}{\sigma_j(\tilde{K})}} S^\# R^T \tilde{v}_j \in \mathcal{H}$, for $j \leq p$.*

Hence, computing the eigenfunctions of \tilde{C} provides a basis of \mathcal{H} of dimension p . Note that in sketched KPCA, which has been explored via Nyström approximation in [63,64], one solves for $i = 1, \dots, m$

$$f_i = \arg \max_{f \in \mathcal{H}} \left\{ \langle f, \hat{C} f \rangle_{\mathcal{H}} : f \in \tilde{\mathcal{H}}_{\mathcal{Y}}, \|f\|_{\mathcal{H}} = 1, f \perp \{f_1, \dots, f_{i-1}\} \right\} \quad (7)$$

where $\tilde{\mathcal{H}}_{\mathcal{Y}} = \text{span}((\sum_{j=1}^n R_{ij} \psi(y_j))_{i=1}^m)$. Let \tilde{P} be the orthogonal projector onto the basis $(\tilde{e}_1, \dots, \tilde{e}_p)$, solving Equation (7) is equivalent to compute the eigenfunctions of the projected empirical covariance operator $\tilde{P} \hat{C} \tilde{P}$, i.e., to compute the KPCA of the projected kernel $\langle \tilde{P} \psi(\cdot), \tilde{P} \psi(\cdot) \rangle_{\mathcal{H}}$. Besides, as for the SVD of \tilde{C} , sketched KPCA needs the SVD of \tilde{K} to obtain its square root, but also requires the additional $\tilde{K}^{1/2} R K^2 R^T \tilde{K}^{1/2}$ SVD computation.

Remark 2 (Random Fourier Features). Another popular kernel approximation is the Random Fourier Features [44,52,57]. They approximate a kernel function as the inner product of small random features using Monte-Carlo sampling when the kernel writes as the Fourier transform of a probability distribution. Such an approach, however, defines a new randomly approximated kernel, then a new randomly approximated loss, which can induce learning difficulties due to the bias and variance inherent to the approximation. Unlike RFF, sketching is not limited to kernels writing as the Fourier transform of a probability distribution

and to defining an approximated loss, it allows the building of a low-dimensional basis within the original feature space of interest.

Learning the Input Neural Network g_W . Equipped with the basis $\tilde{E} = (\tilde{e}_j)_{j \leq p}$, we can compute a novel expression of the loss $L(\theta) = L(\tilde{E}, W)$, see Appendix A for the proof.

Algorithm 1. Deep Sketched Output Kernel Regression (DSOKR)

input: training $\{(x_i, y_i)\}_{i=1}^n$, validation $\{(x_i^{\text{val}}, y_i^{\text{val}})\}_{i=1}^{n_{\text{val}}}$ pairs, test inputs $\{x_i^{\text{te}}\}_{i=1}^{n_{\text{te}}}$, candidate outputs test inputs $\{y_i^c\}_{i=1}^{n_c}$, normalized output kernel k , sketching matrix $R \in \mathbb{R}^{m \times n}$, neural network g_W

init : $\tilde{K} = RKR^\top \in \mathbb{R}^{m \times m}$ where $K = (k(y_i, y_j))_{1 \leq i, j \leq n} \in \mathbb{R}^{n \times n}$

// 1. a. Training of g_E : computations for the basis \tilde{E}

- Construct $\tilde{D}_p \in \mathbb{R}^{p \times p}$, $\tilde{V}_p \in \mathbb{R}^{m \times p}$ such that $\tilde{V}_p \tilde{D}_p \tilde{V}_p^\top = \tilde{K}$ (SVD of \tilde{K})
- $\tilde{\Omega} = \tilde{D}_p^{-1/2} \tilde{V}_p^\top \in \mathbb{R}^{p \times m}$

// 1. b. Training of g_W : solving the surrogate problem

- $\tilde{\psi}(y_i) = \tilde{\Omega} R k^{y_i} \in \mathbb{R}^p, \forall 1 \leq i \leq n$, $\tilde{\psi}(y_i^{\text{val}}) = \tilde{\Omega} R k^{y_i^{\text{val}}} \in \mathbb{R}^p, \forall 1 \leq i \leq n_{\text{val}}$
- $\hat{W} = \arg \min_{W \in \mathcal{W}} \frac{1}{n} \sum_{i=1}^n \|g_W(x_i) - \tilde{\psi}(y_i)\|_2^2$ (training of g_W with training $\{(x_i, \tilde{\psi}(y_i))\}_{i=1}^n$ and validation $\{(x_i^{\text{val}}, \tilde{\psi}(y_i^{\text{val}}))\}_{i=1}^{n_{\text{val}}}$ pairs and Mean Squared Error loss)

// 2. Inference

- $\tilde{\psi}(y_i^c) = \tilde{\Omega} R k^{y_i^c} \in \mathbb{R}^p, \forall 1 \leq i \leq n_c$
- $f_{\hat{\theta}}(x_i^{\text{te}}) = y_j^c$ where $j = \arg \max_{1 \leq j \leq n_c} g_{\hat{W}}(x_i^{\text{te}})^\top \tilde{\psi}(y_j^c), \forall 1 \leq i \leq n_{\text{te}}$

return $f_{\hat{\theta}}(x_i^{\text{te}}), \forall 1 \leq i \leq n_{\text{te}}$

Proposition 2. Given the pre-trained basis $\tilde{E} = (\tilde{e}_j)_{j \leq p}$, $L(\tilde{E}, W)$ expresses as

$$L(\tilde{E}, W) = \frac{1}{n} \sum_{i=1}^n \|g_W(x_i) - \tilde{\psi}(y_i)\|_2^2, \quad (8)$$

where $\tilde{\psi}(y) = (\tilde{e}_1(y), \dots, \tilde{e}_p(y))^\top = \tilde{D}_p^{-1/2} \tilde{V}_p^\top R k^y \in \mathbb{R}^p$, $\tilde{V}_p = (\tilde{v}_1, \dots, \tilde{v}_p)$, $\tilde{D}_p = \text{diag}(\sigma_1(\tilde{K}), \dots, \sigma_p(\tilde{K}))$, and $k^y = (k(y, y_1), \dots, k(y, y_n))$.

Finally, given \tilde{E} and Proposition 2, learning the full network h_θ boils down to learning the input neural network g_W and thus finding a solution \hat{W} to

$$\min_{W \in \mathcal{W}} \frac{1}{n} \sum_{i=1}^n \|g_W(x_i) - \tilde{\psi}(y_i)\|_2^2. \quad (9)$$

A classical stochastic gradient descent algorithm can then be applied to learn W . Compared to the initial loss (3), the relevance of (9) is governed by the quality of

the approximation of \widehat{C} by \widetilde{C} . If our approach regularises the solution (the range of the surrogate estimator h_θ is restricted from \mathcal{H} to E), this restriction may not be limiting if we set $m \geq p$ high enough to capture all the information contained in \widetilde{C} . We discuss strategies to correctly set m at the beginning of Sect. 3.

Remark 3 (Beyond the square loss). Equipped with such an architecture $g_W \circ g_E$, one can easily consider any loss that writes $\Delta(y, y') = c(\|\psi(y) - \psi(y')\|_{\mathcal{H}}^2)$, where $c: \mathbb{R}_+ \rightarrow \mathbb{R}_+$ is a non-decreasing sub-differentiable function. For instance, in the presence of output outliers, one could typically consider robust losses such as the Huber or ϵ -insensitive losses, that correspond to different choices of function c [30, 41, 62].

2.2 The Pre-image Problem at Inference Time

We focus now on the decoding part, i.e., on computing

$$d \circ h_{\hat{\theta}}(x) = \arg \min_{y \in \mathcal{Y}} k(y, x) - 2g_{\widehat{W}}(x)^\top \widetilde{\psi}(y) = \arg \max_{y \in \mathcal{Y}} g_{\widehat{W}}(x)^\top \widetilde{\psi}(y)$$

if we assume k to be normalized, i.e. $k(y, y') = 1, \forall y, y' \in \mathcal{Y}$. For a test set $X^{\text{te}} = (x_1^{\text{te}}, \dots, x_{n_{\text{te}}}^{\text{te}}) \in \mathcal{X}^{n_{\text{te}}}$ and a candidate set $Y^c = (y_1^c, \dots, y_{n_c}^c) \in \mathcal{Y}^{n_c}$, for all $1 \leq i \leq n_{\text{te}}$, the prediction is given by

$$f_{\hat{\theta}}(x_i^{\text{te}}) = y_j^c \quad \text{where} \quad j = \arg \max_{1 \leq j \leq n_c} g_{\widehat{W}}(x_i^{\text{te}})^\top \widetilde{\psi}(y_j^c). \quad (10)$$

Hence, the decoding is particularly suited to problems for which we have some knowledge of the possible outcomes, such as molecular identification problems [11]. When the output kernel is differentiable, it may also be solved using standard gradient-based methods. Finally, some ad-hoc ways to solve the pre-image problem exist for specific kernels, see e.g., [17] for the sequence prediction via n-gram kernels, or [38] for label ranking via Kemeny, Hamming, or Lehmer embeddings. The DSOKR framework is summarized in Algorithm 1.

3 Experiments

In this section, we first present a range of strategies to select the sketching size and an analysis of our proposed DSOKR on a synthetic dataset. Besides, we show the effectiveness of DSOKR through its application to two real-world Supervised Graph Prediction (SGP) tasks: SMILES to Molecule and Text to Molecule. The code to reproduce our results is available at: <https://github.com/tamim-el/dsokr>.

Sketching Size Selection Strategy. A critical hyper-parameter of DSOKR is the sketching size m . Indeed, the optimal choice is the dimension of the subspace containing the output features. However, to estimate this dimension, one has to compute the eigenvalues of K , which has the prohibitive complexity of $\mathcal{O}(n^3)$. Hence, a first solution is to compute the Approximate Leverage Scores (ALS)

as described in [1]. This is an approximation of the eigenvalues of K that relies on sub-sampling $n_S < n$ entries within the whole training set. Moreover, we use another technique that we call *Perfect h*. Considering any pair (x, y) in a validation set, we replace $g_W(x)$ by the “perfect” coefficients of the expansion, i.e., for each $j = 1, \dots, p$, $\langle \tilde{e}_j, \psi(y) \rangle_{\mathcal{H}}$ and define “perfect” surrogate estimator h_ψ as follows

$$h_\psi(x) = \sum_{j=1}^p \langle \tilde{e}_j, \psi(y) \rangle_{\mathcal{H}} \tilde{e}_j = \sum_{j=1}^p \tilde{\psi}(y)_j \tilde{e}_j. \quad (11)$$

Then, we evaluate the performance of this “perfect” surrogate estimator h_ψ on a validation set to select m . Hence, *Perfect h* allows to select the minimal m in the range given by ALS such that the performance of h_ψ reaches an optimal value.

3.1 Analysis of DSOKR on Synthetic Least Squares Regression

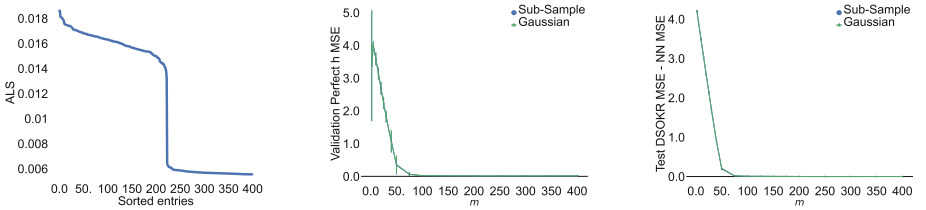


Fig. 2. Sorted 400 highest ALS (left), validation MSE of *Perfect h* w.r.t. m (center) and the difference between test MSE of DSOKR and NN w.r.t. m (right).

Dataset. We generate a synthetic dataset of least-squares regression, using then a linear output kernel, with $n = 50,000$ training data points, $\mathcal{X} = \mathbb{R}^{2,000}$, $\mathcal{Y} = \mathbb{R}^{1,000}$, and $\mathcal{H} = \mathcal{Y} = \mathbb{R}^{1,000}$. The goal is to build this dataset such that the outputs lie in a subspace of \mathcal{Y} of dimension $d = 50 < 1,000$. Hence, given d randomly drawn orthonormal vectors $(u_j)_{j=1}^d$, for all $1 \leq i \leq n$, the outputs are such that $y_i = \sum_{j=1}^d \alpha(x_i)_j u_j + \varepsilon_i$, where α is a function of the inputs and $\varepsilon_i \sim \mathcal{N}(0, \sigma^2 I_{1,000})$ are i.i.d. with $\sigma^2 = 0.01$. We generate i.i.d. normal distributed inputs $x_i \sim \mathcal{N}(0, C)$, where $(\sigma_j(C) = j^{-1/2})_{j=1}^{2,000}$ and its eigenvectors are randomly drawn. Finally, we draw $H \in \mathbb{R}^{d \times 2,000}$ with i.i.d. coefficients from the standard normal distribution, and the outputs are given for $1 \leq i \leq n$ by

$$y_i = U H x_i + \varepsilon_i, \quad (12)$$

where $U = (u_1, \dots, u_d) \in \mathbb{R}^{1,000 \times d}$. We generate validation and test sets of $n_{\text{val}} = 5,000$ and $n_{\text{te}} = 10,000$ points in the same way.

Experimental Settings. We first compute the ALS as described above. We take as regularisation penalty $\lambda = 10^{-4}$, sampling parameter $n_S = \sqrt{n}$ and probability vector $(p_i = 1/n)_{i=1}^n$ (uniform sampling). Then, we perform the sketching size selection strategy *Perfect h*. Note that using a linear output kernel, $\psi : y \in \mathbb{R}^{1,000} \mapsto y$, then $\tilde{e}_i = (1/\sqrt{\sigma_i(\tilde{K})})\tilde{v}_i^\top RY$, where $Y = (y_1, \dots, y_n)^\top \in \mathbb{R}^{n \times 1,000}$, and

$$h_{\hat{\theta}}(x) = Y^\top R^\top \tilde{V}_p \tilde{D}_p^{-1/2} g_{\tilde{W}}(x). \quad (13)$$

Finally, we perform our DSOKR model whose neural network g_W is a Single-Layer Perceptron, i.e. with no hidden layer, and compare it with an SLP whose output size is 1,000, and trained with a Mean Squared Error loss, that we call "NN". We select the optimal number of epochs thanks to the validation set and evaluate the performance via the MSE. We use the ADAM [36] optimizer. For the *Perfect h* and DSOKR models and any sketching size $m \in [2, 400]$, we average the results over five replicates of the models. We use uniform sub-sampling without replacement and Gaussian sketching distributions.

Experimental Results. Figure 2 (left) presents the sorted 400 highest leverage scores. This gives a rough estimate of the optimal sketching size since the leverage scores converge to a minimal value starting from 200 approximately, which is an upper bound of the true basis dimension $d = 50$. Figure 2 (center) shows that *Perfect h* is a relevant strategy to fine-tune m since the obtained optimal value is $m = 75$, which is very close to $d = 50$. This small difference comes from the added noise ε_i . Moreover, this value corresponds to the optimal value based on the DSOKR test MSE. In fact, Fig. 2 (right) presents the performance DSOKR for many m values compared with NN. DSOKR performance converges to the NN’s performance for $m = 75$ as well. Hence, we show that DSOKR attains optimal performance if its sketching size is set as the dimension of the output marginal distribution’s range, which can be estimated thanks to the ALS and the *Perfect h* strategies. There is no difference between sub-sample and Gaussian sketching since the dataset is rather simple. Moreover, note that the neural network of the DSOKR model for $m = 75$ contains 150,075 parameters, whereas the NN model contains 2,001,000 parameters. Then, our sketched basis strategy, even in the context of multi-output regression, allows to reduce the size of the last layer, simplifying the regression problem and reducing the number of weights to learn.

3.2 SMILES to Molecule: SMI2Mol

Dataset. We use the QM9 molecule dataset [54,55], containing around 130,000 small organic molecules. These molecules have been processed using RDKit¹, with aromatic rings converted to their Kekule form and hydrogen atoms removed. We also remove molecules containing only one atom. Each molecule contains up to 9 atoms of Carbon, Nitrogen, Oxygen, or Fluorine, along with three types

¹ RDKit: Open-source cheminformatics. <https://www.rdkit.org>.

of bonds: single, double, and triple. As input features, we use the Simplified Molecular Input Line-Entry System (SMILES), which are strings describing their chemical structure. We refer to the resulting dataset as **SMI2Mol**.

Experimental Set-Up. Using all SMILES-Molecule pairs, we build five splits using different seeds. Each split has 131,382 training samples, 500 validation samples, and 2,000 test samples. In DSOKR, g_W is a Transformer [68]. The SMILES strings are tokenized into character sequences as inputs for the Transformer encoder. To define the loss on output molecules, we cross-validate several graph kernels, including the Weisfeiler-Lehman subtree kernel (WL-VH) [60], the neighborhood subgraph pairwise distance kernel (NSPD) [18], and the core Weisfeiler-Lehman subtree kernel (CORE-WL) [48]. We use the implementation of the graph kernels provided by the Python library GraKel [61]. We employ SubSample sketching for the output kernel. The sketching size m is fixed using our proposed *Perfect h* strategy. Our method is benchmarked against SISOKR [22], NNBary-FGW [9], and ILE-FGW [9]. For ILE-FGW and SISOKR, we additionally use SubSample sketching [56] for input kernel approximation. To ensure a fair comparison, both SISOKR and ILE-FGW adopt the 3-gram kernel for the input strings, whereas NNBary-FGW and DSOKR use a Transformer encoder. The performance is evaluated using Graph Edit Distance (GED), implemented by the NetworkX package [28].

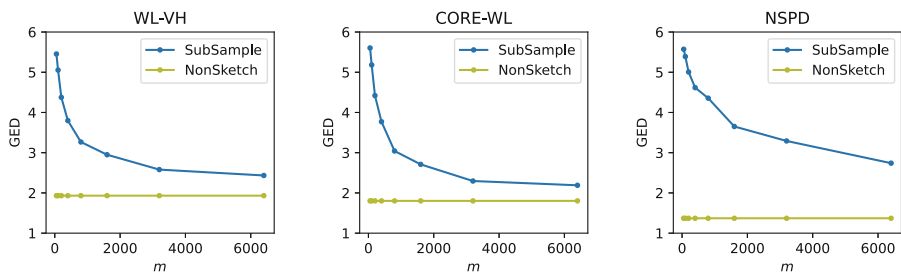


Fig. 3. The GED w/ edge feature w.r.t. the sketching size m for *Perfect h* for three graph kernels on SMI2Mol ($m > 6400$ is too costly computationally).

Table 1. Edit distance of different methods on SMI2Mol test set

	GED w/o edge feature ↓	GED w/ edge feature ↓
SISOKR	3.330 ± 0.080	4.192 ± 0.109
NNBary-FGW	5.115 ± 0.129	-
Sketched ILE-FGW	2.998 ± 0.253	-
DSOKR	1.951 ± 0.074	2.960 ± 0.079

Experimental Results. Figure 3 displays the GED obtained by *Perfect h* concerning various graph kernels. Based on this visualization, we have set the sketching sizes of WL-VH, CORE-WL, and NPSD to 3200, 3200, and 6400 respectively. Table 1 showcases the performance of various methods of SGP. Notably, DSOKR outperforms all baseline methods. It is evident that while graph kernels and the fused Gromov-Wasserstein (FGW) distance induce a meaningful feature space, the capabilities of SISOKR and ILE-FGW are constrained by the input kernels, thus highlighting the relevance of our proposed method. For further insight, a comparison of some prediction examples is provided in Fig. 4 and Appendix C.1.

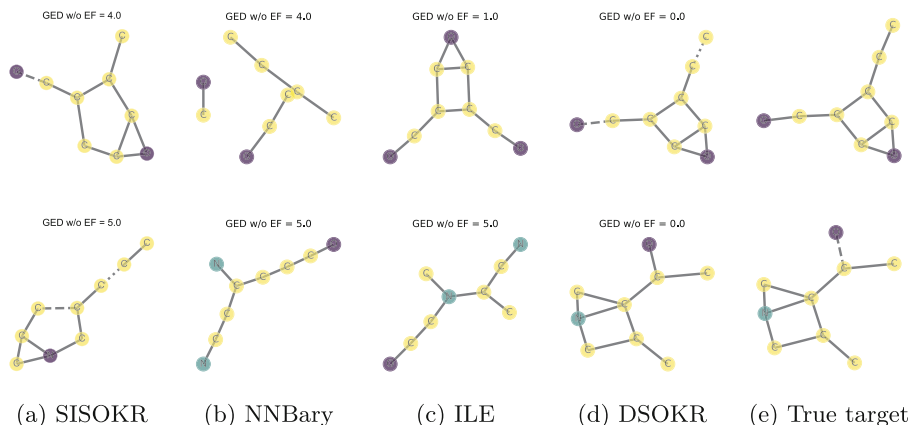


Fig. 4. Predicted molecules on the SMI2Mol dataset.

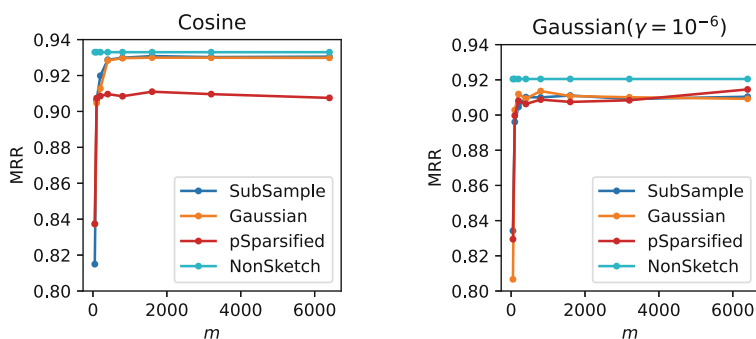


Fig. 5. The MRR scores on ChEBI-20 validation set w.r.t. m for *Perfect h* when the output kernel is Cosine or Gaussian on the ChEBI-20 dataset.

Table 2. Performance of different methods on ChEBI-20 test set. All the methods based on NNs use SciBERT as input text encoder for fair comparison. The number in the ensemble setting indicates the number of single models used.

	Hits@1 \uparrow	Hits@10 \uparrow	MRR \uparrow
SISOKR	0.4%	2.8%	0.015
SciBERT Regression	16.8%	56.9%	0.298
CMAM - MLP	34.9%	84.2%	0.513
CMAM - GCN	33.2%	82.5%	0.495
CMAM - Ensemble (MLP \times 3)	39.8%	87.6%	0.562
CMAM - Ensemble (GCN \times 3)	39.0%	87.0%	0.551
CMAM - Ensemble (MLP \times 3 + GCN \times 3)	44.2%	88.7%	0.597
DSOKR - SubSample Sketch	48.2%	87.4%	0.624
DSOKR - Gaussian Sketch	49.0%	87.5%	0.630
DSOKR - Ensemble (SubSample \times 3)	51.0%	88.2%	0.642
DSOKR - Ensemble (Gaussian \times 3)	50.5%	87.9%	0.642
DSOKR - Ensemble (SubSample \times 3 + Gaussian \times 3)	50.0%	88.3%	0.640

3.3 Text to Molecule: ChEBI-20

Dataset. The ChEBI-20 [21] dataset contains 33,010 pairs of compounds and descriptions. The compounds come from PubChem [34,35], and their descriptions (more than 20 words) from the Chemical Entities of Biological Interest (ChEBI) database [29]. The dataset is divided as follows: 80% for training, 10% for validation, and 10% for testing. The candidate set contains all compounds. The mean and median number of atoms per molecule is 32 and 25 respectively, and the mean and median number of words per description is 55 and 51 respectively.

Experimental Set-Up. For our method DSOKR, we use SciBERT [6] with an additional linear layer to parameterize g_W . The maximum length of the input tokens is set to 256. Mol2vec [31] is used as the output molecule representation, which is a vector of dimension 300. Based on the Mol2vec representation, we conduct cross-validation using the following kernels: Cosine kernel and Gaussian kernel with gamma chosen from $\{10^{-9}, 10^{-6}, 10^{-3}, 1\}$, along with the following three sketches: sub-sampling [56], Gaussian [75], and p -sparsified [23]. The sketching size for all combinations of the output kernels and sketches is determined using the *Perfect h* strategy. As for the baselines, we consider SciBERT Regression, Cross-Modal Attention Model (CMAM) [21], and SISOKR. In the case of SciBERT Regression, we address the regression problem using Mean Squared Error loss, where the output space is the embedding space of Mol2vec, within a function space parameterized by SciBERT. CMAM aims to enhance the cosine similarity between the text embedding and the corresponding molecule in true pairs by employing a contrastive loss function. Specifically, the former is

derived from SciBERT, while the latter is generated using either a multi-layer perceptron (MLP) or a graph convolutional network (GCN) atop the Mol2vec representation. We reproduce the results of CMAM with the codes² released by [21]. In SISOKR, we use SciBERT embeddings as input features, leveraging the cosine kernel atop them. We maintain the identical output kernel sketching setup as in DSOKR. For all methods, we train the model using the best hyperparameters with three random seeds and report the one with the best validation performance. The performance is evaluated with mean reciprocal rank (MRR), Hits@1 and Hits@10. We could not benchmark AMAN [76], as no implementation is publicly available.

Ensemble. In [21], the authors propose an ensemble strategy to enhance the results by aggregating the ranks obtained by different training of their models. If for each $1 \leq t \leq T$, R_t denotes the ranking returned by the model t , the new score is computed as follows

$$s(y_i) = \sum_{t=1}^T \omega_t R_t(y_i) \quad s.t. \quad \sum_{i=1}^T \omega_t = 1 \quad (14)$$

for each y_i in the candidate set. In our case, the computation of DSOKR’s last layer g_E depends on a draw of the sketching matrix R , which means that DSOKR is particularly well-suited to the aggregation via multiple draws of the sketching matrix R_t and the training of the corresponding neural networks g_{W_t} . Hence, we explore two more ways of aggregating multiple DSOKR models, by averaging or maximizing these models’ scores, i.e. for any input x and candidate y ,

$$s(x, y) = \sum_{t=1}^T \omega_t g_{\hat{W}_t}(x)^\top \tilde{\psi}_t(y) \quad \text{or} \quad s(x, y) = \arg \max_{1 \leq t \leq T} g_{\hat{W}_t}(x)^\top \tilde{\psi}_t(y). \quad (15)$$

We explore all three ensemble methods for DSOKR models and subsequently select the optimal one based on its validation performance.

Experimental Results. Figure 5 illustrates the validation MRR scores with *Perfect h*, for many m values, and either Cosine or Gaussian output kernels. It is evident that for both the Cosine kernel and Gaussian kernel (with $\gamma = 10^{-6}$) employing various sketching methods, the MRR score stabilizes as the sketching size exceeds 100, and that Cosine outperforms Gaussian. This observation allows us to choose $m = 100$, smaller than the original Mol2vec dimension, which is 300. Table 2 presents a comprehensive comparison of DSOKR with various baseline models. Firstly, comparing DSOKR with SISOKR reveals the critical importance of employing deep neural networks when dealing with complex structured inputs and DSOKR makes it possible in the case of functional output space. Secondly, the notable improvement over SciBERT Regression underscores the value of employing kernel sketching to derive more compact and better output features, thereby facilitating regression problem-solving. Lastly, DSOKR outperforms the sota CMAP for both single and ensemble models. See Appendix C.2 for more details.

² <https://github.com/cnedwards/text2mol>.

4 Conclusion

We designed a new architecture of neural networks able to minimize kernel-induced losses for structured prediction and achieving sota performance on molecular identification. An interesting avenue for future work is to derive excess risk for this estimator by combining deep learning theory and surrogate regression bounds.

Acknowledgments. Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Commission. Neither the European Union nor the granting authority can be held responsible for them. This project has received funding from the European Union’s Horizon Europe research and innovation programme under grant agreement 101120237 (ELIAS), the Télécom Paris research chair on Data Science and Artificial Intelligence for Digitalized Industry and Services (DSALDIS) and the PEPR-IA through the project FOUNDRY.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Alaoui, A., Mahoney, M.W.: Fast randomized kernel ridge regression with statistical guarantees. In: *NeurIPS*, vol. 28 (2015)
2. Bach, F.: Sharp analysis of low-rank kernel matrix approximations. In: *COLT*, pp. 185–209 (2013)
3. Bakir, G., Hofmann, T., Smola, A.J., Schölkopf, B., Taskar, B.: *Predicting Structured Data*. The MIT Press, Cambridge (2007)
4. Belanger, D., McCallum, A.: Structured prediction energy networks. In: *ICML*, pp. 983–992 (2016)
5. Belanger, D., Yang, B., McCallum, A.: End-to-end learning for structured prediction energy networks. In: *ICML*, vol. 70, pp. 429–439 (2017)
6. Beltagy, I., Lo, K., Cohan, A.: SciBERT: a pretrained language model for scientific text. In: *EMNLP-IJCNLP*, pp. 3615–3620. Hong Kong, China (2019)
7. Borgwardt, K., Ghisu, E., Llinares-López, F., O’Bray, L., Rieck, B.: Graph kernels: state-of-the-art and future challenges. *Found. Trends Mach. Learn.* **13**(5–6), 531–712 (2020)
8. Borgwardt, K., Kriegel, H.: Shortest-path kernels on graphs. In: *Fifth IEEE International Conference on Data Mining (ICDM’05)*, pp. 8 pp.– (2005)
9. Brogat-Motte, L., Flamary, R., Brouard, C., Rousu, J., D’Alché-Buc, F.: Learning to predict graphs with fused Gromov-Wasserstein barycenters. In: *ICML*, vol. 162, pp. 2321–2335 (2022)
10. Brouard, C., d’Alché-Buc, F., Szafranski, M.: Semi-supervised penalized output kernel regression for link prediction. In: *ICML*, pp. 593–600 (2011)
11. Brouard, C., Shen, H., Dührkop, K., d’Alché-Buc, F., Böcker, S., Rousu, J.: Fast metabolite identification with input output kernel regression. *Bioinformatics* **32**(12), 28–36 (2016)

12. Brouard, C., Szafranski, M., d'Alché Buc, F.: Input output kernel regression: supervised and semi-supervised structured output prediction with operator-valued kernels. *JMLR* **17**(1), 6105–6152 (2016)
13. Cabannes, V.A., Bach, F., Rudi, A.: Fast rates for structured prediction. In: *COLT*, pp. 823–865 (2021)
14. Ciliberto, C., Rosasco, L., Rudi, A.: A consistent regularization approach for structured prediction. In: *NeurIPS*, pp. 4412–4420 (2016)
15. Ciliberto, C., Rosasco, L., Rudi, A.: A general framework for consistent structured prediction with implicit loss embeddings. *JMLR* **21**(98), 1–67 (2020)
16. Cortes, C., Mohri, M., Weston, J.: A general regression technique for learning transductions. In: *ICML*, pp. 153–160 (2005)
17. Cortes, C., Mohri, M., Weston, J.: A general regression framework for learning string-to-string mappings. In: *Predicting Structured Data* (2007)
18. Costa, F., Grave, K.D.: Fast neighborhood subgraph pairwise distance kernel. In: *ICML*, pp. 255–262 (2010)
19. Deshwal, A., Doppa, J.R., Roth, D.: Learning and inference for structured prediction: a unifying perspective. In: *IJCAI* (2019)
20. Drineas, P., Mahoney, M.W., Cristianini, N.: On the nyström method for approximating a gram matrix for improved kernel-based learning. *JMLR* **6**(12) (2005)
21. Edwards, C., Zhai, C., Ji, H.: Text2Mol: cross-modal molecule retrieval with natural language queries. In: *EMNLP*, pp. 595–607 (2021)
22. El Ahmad, T., Brogat-Motte, L., Laforgue, P., d'Alché-Buc, F.: Sketch in, sketch out: accelerating both learning and inference for structured prediction with kernels (2023)
23. El Ahmad, T., Laforgue, P., d'Alché Buc, F.: Fast kernel methods for generic lipschitz losses via p -sparsified sketches. *TMLR* (2023)
24. Gärtner, T.: *Kernels for Structured Data*, Series in Machine Perception and Artificial Intelligence, vol. 72. WorldScientific (2008)
25. Geurts, P., Wehenkel, L., d'Alché Buc, F.: Kernelizing the output of tree-based methods. In: *ICML*, pp. 345–352 (2006)
26. Graber, C., Meshi, O., Schwing, A.: Deep structured prediction with nonlinear output transformations. In: *NeurIPS*, vol. 31 (2018)
27. Gygli, M., Norouzi, M., Angelova, A.: Deep value networks learn to evaluate and iteratively refine structured outputs. In: *ICML*, p. 1341–1351 (2017)
28. Hagberg, A.A., Schult, D.A., Swart, P.J.: Exploring network structure, dynamics, and function using NetworkX. In: *Proceedings of the 7th Python in Science Conference*, pp. 11–15 (2008)
29. Hastings, J., et al.: ChEBI in 2016: improved services and an expanding collection of metabolites. *Nucleic Acids Res.* **44**(D1), D1214–D1219 (2016)
30. Huber, P.J.: Robust estimation of a location parameter. *Ann. Math. Stat.* 73–101 (1964)
31. Jaeger, S., Fulle, S., Turk, S.: Mol2Vec: unsupervised machine learning approach with chemical intuition. *J. Chem. Inf. Model.* **58**(1), 27–35 (2018)
32. Jumper, J., et al.: Highly accurate protein structure prediction with alphafold. *Nature* **596**(7873), 583–589 (2021)
33. Kadri, H., Ghavamzadeh, M., Preux, P.: A generalized kernel approach to structured output learning. In: *ICML*, pp. 471–479 (2013)
34. Kim, S., et al.: PubChem 2019 update: improved access to chemical data. *Nucleic Acids Res.* **47**(D1), D1102–D1109 (2019)
35. Kim, S., et al.: PubChem substance and compound databases. *Nucleic Acids Res.* **44**(D1), D1202–D1213 (2016)

36. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: ICLR (2015)
37. Kirillov, A., et al.: Segment anything. In: ICCV, pp. 4015–4026 (2023)
38. Korba, A., Garcia, A., d'Alché-Buc, F.: A structured prediction approach for label ranking. In: NeurIPS, vol. 31 (2018)
39. Kpotufe, S., Sriperumbudur, B.K.: Gaussian sketching yields a J-L lemma in RKHS. In: Chiappa, S., Calandra, R. (eds.) AISTATS (2020)
40. Lacotte, J., Pilanci, M.: Adaptive and oblivious randomized subspace methods for high-dimensional optimization: sharp analysis and lower bounds. *IEEE Trans. Inf. Theory* **68**(5), 3281–3303 (2022)
41. Laforgue, P., Lambert, A., Brogat-Motte, L., d'Alché Buc, F.: Duality in RKHS with infinite dimensional outputs: application to robust losses. In: ICML, pp. 5598–5607 (2020)
42. LeCun, Y., Chopra, S., Hadsell, R., Ranzato, A., Huang, F.J.: A tutorial on energy-based learning. In: Predicting Structured Data (2006)
43. Lee, J.Y., Patel, D., Goyal, P., Zhao, W., Xu, Z., McCallum, A.: Structured energy network as a loss. In: NeurIPS (2022)
44. Li, Z., Ton, J.F., Oglic, D., Sejdinovic, D.: Towards a unified analysis of random Fourier features. *JMLR* **22**(108), 1–51 (2021)
45. Mahoney, M.W., et al.: Randomized algorithms for matrices and data. *Found. Trends Mach. Learn.* **3**(2), 123–224 (2011)
46. Meanti, G., Carratino, L., Rosasco, L., Rudi, A.: Kernel methods through the roof: handling billions of points efficiently. In: Advances in Neural Information Processing Systems (NeurIPS), vol. 33 (2020)
47. Meanti, G., Chatalic, A., Kostic, V., Novelli, P., Pontil, M., Rosasco, L.: Estimating Koopman operators with sketching to provably learn large scale dynamical systems. In: NeurIPS (2023)
48. Nikolentzos, G., Meladianos, P., Limnios, S., Vazirgiannis, M.: A degeneracy framework for graph similarity. In: IJCAI (2018)
49. Nowak, A., Bach, F., Rudi, A.: Sharp analysis of learning with discrete losses. In: AISTAT (2019)
50. Nowak, A., Bach, F., Rudi, A.: Consistent structured prediction with max-min margin Markov networks. In: ICML (2020)
51. Nowozin, S., Lampert, C.H.: Structured learning and prediction in computer vision. *Found. Trends Comput. Graph. Vision* **6** (2011)
52. Rahimi, A., Recht, B.: Random features for large scale kernel machines. In: NeurIPS, vol. 20, pp. 1177–1184 (2007)
53. Ralaivola, L., Swamidass, S.J., Saigo, H., Baldi, P.: Graph kernels for chemical informatics. *Neural Netw.* **18**(8), 1093–1110 (2005)
54. Ramakrishnan, R., Dral, P.O., Rupp, M., von Lilienfeld, O.A.: Quantum chemistry structures and properties of 134 kilo molecules. *Sci. Data* **1** (2014)
55. Ruddigkeit, L., van Deursen, R., Blum, L.C., Reymond, J.L.: Enumeration of 166 billion organic small molecules in the chemical universe database GDB-17. *J. Chem. Inf. Model.* **52**(11) (2012)
56. Rudi, A., Camoriano, R., Rosasco, L.: Less is more: Nyström computational regularization. In: NeurIPS, vol. 28 (2015)
57. Rudi, A., Rosasco, L.: Generalization properties of learning with random features. In: NeurIPS, pp. 3215–3225 (2017)
58. Schölkopf, B., Smola, A., Müller, K.R.: Kernel principal component analysis. In: ICANN (1997)
59. Seidman, S.B.: Network structure and minimum degree. *Soc. Netw.* **5**(3), 269–287 (1983)

60. Shervashidze, N., Schweitzer, P., van Leeuwen, E.J., Mehlhorn, K., Borgwardt, K.M.: Weisfeiler-Lehman graph kernels. *JMLR* (2011)
61. Siglidis, G., Nikolentzos, G., Limnios, S., Giatsidis, C., Skianis, K., Vazirgiannis, M.: Grakel: a graph kernel library in python. *JMLR* (2020)
62. Steinwart, I., Christmann, A.: Sparsity of SVMs that use the epsilon-insensitive loss. In: *NeurIPS* (2008)
63. Sterge, N., Sriperumbudur, B., Rosasco, L., Rudi, A.: Gain with no pain: efficiency of kernel-PCA by Nyström sampling. In: *AISTATS* (2020)
64. Sterge, N., Sriperumbudur, B.K.: Statistical optimality and computational efficiency of Nystrom kernel PCA. *JMLR* **23**(337), 1–32 (2022)
65. Tanimoto, T.: *An Elementary Mathematical Theory of Classification and Prediction*. International Business Machines Corporation (1958)
66. Tripp, A., Bacallado, S., Singh, S., Hernández-Lobato, J.M.: Tanimoto random features for scalable molecular machine learning. In: *NeurIPS* (2023)
67. Tu, L., Gimpel, K.: Learning approximate inference networks for structured prediction. In: *ICLR* (2018)
68. Vaswani, A., et al.: Attention is all you need. In: *NeurIPS* (2017)
69. Vayer, T., Courty, N., Tavenard, R., Laetitia, C., Flamary, R.: Optimal transport for structured data with application on graphs. In: *ICML* (2019)
70. Weisfeiler, B., Leman, A.: The reduction of a graph to canonical form and the algebra which appears therein. *NTI, Ser.* **2**(9), 12–16 (1968)
71. Weston, J., Chapelle, O., Vapnik, V., Elisseeff, A., Schölkopf, B.: Kernel dependency estimation. In: *NeurIPS*, pp. 897–904. MIT Press (2003)
72. Williams, C., Seeger, M.: Using the Nyström method to speed up kernel machines. In: *NeurIPS*, vol. 13, pp. 682–688 (2001)
73. Woodruff, D.P.: Sketching as a tool for numerical linear algebra. *Found. Trends Theor. Comput. Sci.* **10**(1–2), 1–157 (2014)
74. Yang, T., Li, Y.F., Mahdavi, M., Jin, R., Zhou, Z.H.: Nyström method vs random Fourier features: a theoretical and empirical comparison. In: *NeurIPS*, vol. 25 (2012)
75. Yang, Y., Pilanci, M., Wainwright, M.J., et al.: Randomized sketches for kernels: fast and optimal nonparametric regression. *Ann. Stat.* **45**(3), 991–1023 (2017)
76. Zhao, W., Zhou, D., Cao, B., Zhang, K., Chen, J.: Adversarial modality alignment network for cross-modal molecule retrieval. *IEEE Trans. Artif. Intell.* **5**(1) (2024)



Hyperbolic Delaunay Geometric Alignment

Aniss Aiman Medbouhi^(✉), Giovanni Luca Marchetti, Vladislav Polianskii, Alexander Kravberg, Petra Poklukar, Anastasia Varava, and Danica Kragic

Division of Robotics, Perception and Learning, Department of Intelligent Systems,
School of Electrical Engineering and Computer Science, KTH Royal Institute of
Technology, Stockholm, Sweden
medbouhi@kth.se

Abstract. Hyperbolic machine learning is an emerging field aimed at representing data with a hierarchical structure. However, there is a lack of tools for evaluation and analysis of the resulting hyperbolic data representations. To this end, we propose Hyperbolic Delaunay Geometric Alignment (HyperDGA) – a similarity score for comparing datasets in a hyperbolic space. The core idea is counting the edges of the hyperbolic Delaunay graph connecting datapoints across the given sets. We provide an empirical investigation on synthetic and real-life biological data and demonstrate that HyperDGA outperforms the hyperbolic version of classical distances between sets. Furthermore, we showcase the potential of HyperDGA for evaluating latent representations inferred by a Hyperbolic Variational Auto-Encoder.

Keywords: Hyperbolic Geometry · Hierarchical Data · Evaluation

1 Introduction

Hyperbolic geometry is a non-Euclidean geometry characterized by constant negative curvature [1], which is particularly suitable for low-dimensional tree embedding. In contrast to the Euclidean case, a hyperbolic space requires only two dimensions to embed any tree with arbitrary low distortion due to the exponential volume growth away from the origin [26]. This has motivated the use of hyperbolic geometry in machine learning for representing data that exhibit a *hierarchical* structure. Examples of such data include geographic communication networks [12], internet networks [2], words in a natural language [17, 28], or single-cell data in biology [13]. Recent work in dimensionality reduction [5, 9, 25] and generative modeling [15, 16] has shown that performance in downstream tasks is improved when hierarchical data is represented in a hyperbolic space.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70352-2_7.

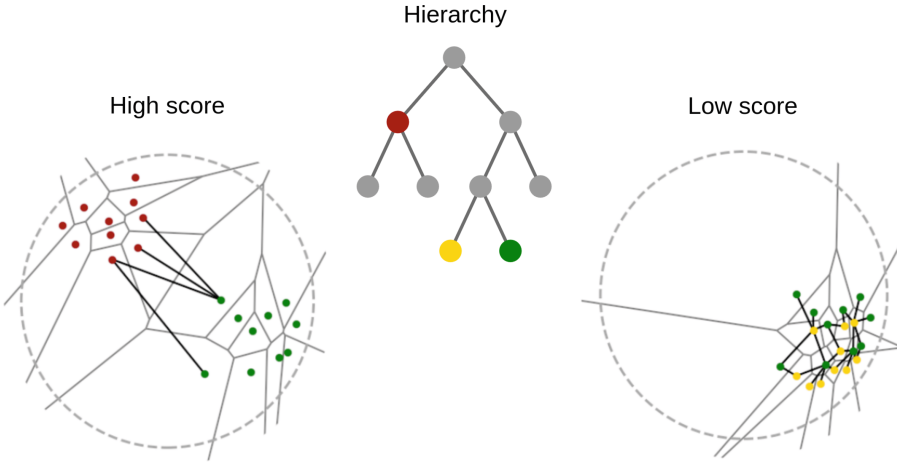


Fig. 1. Illustration of HyperDGA for a hyperbolic representation of hierarchical data. Our score counts heterogeneous edges (black) of the Delaunay graph, which is dual to the hyperbolic Voronoi diagram (gray). Data closer in the hierarchy (green & yellow) have lower scores than data further apart (red & green). (Color figure online)

Despite the abundance of hyperbolic machine learning models, there is a lack of methods for analyzing hyperbolic embeddings or for evaluating hyperbolic data representations independently of any downstream task. This raises the need to design evaluation metrics suitable for hyperbolic geometry. To this end, we propose Hyperbolic Delaunay Geometric Alignment (HyperDGA) – a similarity score between two sets embedded in a hyperbolic space. Our method is based on a hyperbolic Delaunay triangulation that takes into account the topology and geometry of the data representation. Inspired by a recently-introduced score deemed Delaunay Component Analysis (DCA) [22, 23], HyperDGA relies on counting the edges of the Delaunay graph that are *heterogeneous* i.e., connecting points across the given sets – see Fig. 1 for an illustration.

We validate HyperDGA empirically on synthetic and real-life datasets represented in a hyperbolic space, and compare it to hyperbolic versions of standard distances for sets – the Chamfer and the Wasserstein distances. Namely, we investigate correlation of the metrics with noise in a synthetic dataset simulating protein evolution, and evaluate the latent representation of a Hyperbolic Variational Auto-Encoder trained on the same dataset. Moreover, we compare the measured distances on real-life biological data representing cells. We demonstrate that HyperDGA outperforms the Chamfer distance, and performs comparably or better than the Wasserstein distance.

In summary, our contributions include a geometric similarity score between two sets in a hyperbolic space, and an experimental investigation showcasing how HyperDGA can be used for hyperbolic data analysis and evaluation of hyperbolic representations. The code for HyperDGA and the experiments is accessible at the public repository: <https://github.com/anissmedbouhi/HyperDGA>.

2 Related Work

Since HyperDGA is designed for data analysis of hyperbolic embeddings, we start by reviewing hyperbolic dimensionality reduction techniques, as well as hyperbolic Variational Auto-Encoders (VAEs). Moreover, we review recent methods for evaluating data representations in the Euclidean space, since this has not been investigated yet in the hyperbolic case.

Hyperbolic Dimensionality Reduction. Classical dimensionality reduction methods have been extended to the hyperbolic case, for example Multidimensional Scaling [25], t -Distributed Stochastic Neighbor Embeddings [9], Principal Component Analysis [5, 8], and contrastive learning [4, 17]. All these methods embed data in a lower-dimensional hyperbolic space in order to exploit the inherent hierarchical structure. HyperDGA is applicable on top of these dimensionality reduction techniques to analyze the resulting hyperbolic data representations.

Hyperbolic VAEs. Hyperbolic VAEs were first proposed by [15, 16] as hyperbolic analogues of the VAE [11, 24]. These models differ in two aspects: the model of hyperbolic geometry for the latent space, and the type of hyperbolic Gaussian distribution deployed. Specifically, the hyperbolic models involved are the Poincaré ball model and the hyperboloid model respectively, while the distributions are the Riemannian normal and the wrapped normal, respectively. Mixed-Curvature VAEs [27] can be seen as a generalization of the above-cited hyperbolic VAEs. The main difference is that the latent space consists of a product of spaces of constant curvature selected among the Euclidean, spherical, or hyperbolic one. Thus, the latent space has a non-constant curvature, resulting in more flexibility to represent the data.

Evaluation of Euclidean Data Representations. Several methods have been proposed to compare data representations in terms of their geometric and topological structure. For example, Geometry Score [10] computes a topological approximation of the given datasets. This results in a the score that is robust to noise, but is computationally expensive and requires tuning several hyperparameters. Similarly, Improved Precision and Recall Metric (IPR) [14] approximates the underlying data manifold by a union of hyperspheres, but only contains a single hyperparameter controlling their sizes. IPR consists of two scores *precision* and *recall*, counting the number of points of a dataset in the other dataset’s approximated manifold. Geometric Component Analysis (GeomCA) [23] is inspired by IPR and similarly provides local and global geometric scores. GeomCA approximates the manifold via an ϵ -proximity graph connecting neighboring datapoints, where ϵ is a hyperparameter controlling the neighbor size. The latter is difficult to tune because clusters of different scales may appear in the data. DCA [22] amends to this shortcoming by deploying the same evaluation scores while approximating the data manifold via the 1-skeleton of the Delaunay triangulation. In summary, the

above methods are designed specifically for Euclidean data representations and cannot be applied directly to different geometries. On the contrary, HyperDGA extends DCA to the hyperbolic space, and therefore is suitable for analyzing and evaluating hyperbolic data representations.

3 Background

HyperDGA is based on the computation of a hyperbolic Delaunay graph. The latter captures geometric and topological information in data, which motivates our similarity score. However, a core challenge is computing the Delaunay graph in a hyperbolic space. To this end, we deploy the *Klein-Beltrami model* of hyperbolic geometry, where the geodesics are straight lines. As a consequence, the computation of the Delaunay graph can be reduced to the Euclidean case, which in turn can be addressed via standard algorithms from computational geometry. In what follows, we introduce the necessary geometric background on Delaunay triangulations and the Klein-Beltrami model.

3.1 Voronoi Cells and Delaunay Graph

A *Delaunay triangulation* is a collection of simplices in a metric space that is *dual* to the *Voronoi diagram* [3]. The latter partitions the space into regions called *Voronoi cells*. Intuitively, given a set of points, each one defines a Voronoi cell with the property that points in \mathbb{R}^n belong to the cell of their nearest neighbor. More formally, let \mathcal{X} be a metric space with distance function $d: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}_{\geq 0}$ and let $P \subseteq \mathcal{X}$ be a finite subset.

Definition 1. *The Voronoi cell of $p \in P$ is:*

$$V(p) = \{x \in \mathcal{X} \mid \forall q \in P \ d(x, q) \geq d(x, p)\}. \quad (1)$$

The Delaunay triangulation is the collection of simplices with vertices $\sigma \subseteq P$ such that $\bigcap_{p \in \sigma} V(p) \neq \emptyset$.

We refer to the 1-skeleton of the Delaunay triangulation as *Delaunay graph*. In the case of the Euclidean space $\mathcal{X} = \mathbb{R}^n$, the Voronoi cells are convex n -dimensional polytopes that intersect at the boundary and cover the ambient space – see Fig. 2, left. Moreover, for generic P , the Delaunay triangulation is embedded as a collection of non-overlapping simplices that cover the convex hull of P .

3.2 The Klein-Beltrami Model

The n -dimensional hyperbolic space is, by definition, the unique simply-connected Riemannian manifold with constant curvature equal to -1 . It admits several isometric models – see the supplementary material for an overview with

other models¹. As anticipated, we deploy the Klein-Beltrami model since its geodesics are straight lines, which enables to reduce the derivations and computations to the Euclidean case, in particular for computing the hyperbolic Delaunay triangulation.

The n -dimensional Klein-Beltrami model is defined as the Riemannian manifold with ambient space the Euclidean unit ball $\mathbb{K}^n = \{z \in \mathbb{R}^n \mid \|z\| < 1\}$, equipped with the metric tensor:

$$g(z) = \left(\frac{1}{\|z\|^2 - 1} I_n + \frac{1}{(\|z\|^2 - 1)^2} z \otimes z \right), \quad (2)$$

where \otimes denotes the tensor product, I_n the identity matrix, and $\langle \cdot, \cdot \rangle$ the Euclidean scalar product. As for any Riemannian manifold, \mathbb{K}^n can be seen as a metric space when equipped with the geodesic distance $d(x, y) = \inf_{\gamma} \int_{[0,1]} \sqrt{\gamma'(t)^\dagger g(\gamma(t)) \gamma'(t)} dt$, where $\gamma: [0, 1] \rightarrow \mathbb{K}^n$ is a smooth curve with $\gamma(0) = x, \gamma(1) = y$. Here, γ' denotes the first derivative and \dagger the transpose. Explicitly, the distance is given by the expression:

$$d(x, y) = \operatorname{arccosh} \left(\frac{1 - \langle x, y \rangle}{\sqrt{(1 - \|x\|^2)(1 - \|y\|^2)}} \right). \quad (3)$$

4 Method

In this section, we describe HyperDGA – our proposed method for comparing data representations in a hyperbolic space. An overview of the steps for computing HyperDGA is provided in Algorithm 1.

Algorithm 1: HyperDGA Overview

- Data:** Two datasets A, B represented in a hyperbolic space.
 - Result:** Similarity score $\text{HyperDGA}(A, B)$.
 - Step 1:** Convert A and B to \mathbb{K}^n via an isometry.
 - Step 2:** Compute the hyperbolic Delaunay triangulation of $A \cup B$.
 - Step 2.1:** Compute the Euclidean power diagram of $A \cup B$.
 - Step 2.2:** Remove the non-Delaunay simplices.
 - Step 3:** Compute $\text{HyperDGA}(A, B)$ via Equation 6.
-

4.1 Conversion to Klein-Beltrami

Step 1 of Algorithm 1 consists in converting the datasets A, B from the given model of hyperbolic geometry to the Klein-Beltrami ball model \mathbb{K}^n . This is possible since there exist isometric diffeomorphisms between all the hyperbolic models. For example, two popular models in hyperbolic machine learning are the Poincaré ball model \mathbb{P}^n and the Lorentz hyperboloid model \mathbb{L}^n – see the supplementary material for details (see Footnote 1). The following maps convert these models to the Klein-Beltrami model:

¹ <https://github.com/anissmedbouhi/HyperDGA/blob/main/supp.pdf>.

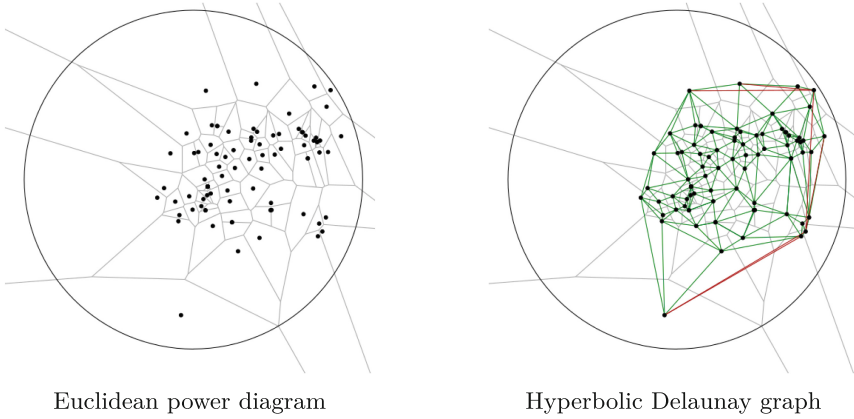


Fig. 2. Depiction of the Euclidean power diagram (gray) equivalent to the hyperbolic Voronoi diagram, the hyperbolic Delaunay graph (green), and the pruned edges (red). Data is obtained from real-life cells (neoblasts 4 & 7, [21]) via Poincaré embedding [13] and converted to the Klein-Beltrami model. (Color figure online)

$$\begin{aligned}
 - z &= (z_0, \dots, z_n) \in \mathbb{L}^n \mapsto \left(\frac{z_1}{z_0}, \dots, \frac{z_n}{z_0} \right) \in \mathbb{K}^n \\
 - z &\in \mathbb{P}^n \mapsto \frac{2z}{1+\|z\|^2} \in \mathbb{K}^n.
 \end{aligned}$$

4.2 Hyperbolic Voronoi Diagram in \mathbb{K}^n

The goal of **Step 2** in Algorithm 1 is to compute the hyperbolic Delaunay triangulation of the union of A and B in the Klein-Beltrami model. As shown by [18], the hyperbolic Voronoi cells in the Klein-Beltrami model correspond to weighted Euclidean Voronoi cells, referred to as *power cells*. We introduce the latter below in a general metric space \mathcal{X} .

Definition 2. The power cell of $p \in P$ with weights $\{r_p\}_{p \in P} \subseteq \mathbb{R}_{\geq 0}$ is:

$$R(p) = \{x \in \mathcal{X} \mid \forall q \in P \ d_q(x) \geq d_p(x)\}, \tag{4}$$

where $d_p(x) = d(x, p)^2 - r_p^2$.

If all the weights r_p are set to 0, power cells specialize to Voronoi cells. By leveraging on the non-isometric embedding of the Klein-Beltrami model into the Euclidean space, it is possible to reduce the computation of hyperbolic Voronoi cells to Euclidean power cells with appropriate points and weights. This is performed in **Step 2.1**, where we compute the Euclidean power diagram of the union of A and B – see Fig. 2 (left).

Theorem 1 ([18]). *Given $P \subseteq \mathbb{K}^n$, there exists an explicit set $S \subseteq \mathbb{R}^n$ and weights $\{r_s\}_{s \in S}$ such that the hyperbolic Voronoi cells of P correspond to restrictions to \mathbb{K}^n of power cells of S .*

Explicitly, the set S and weights $\{r_s\}_{s \in S}$ from Theorem 1 are obtained from points $p \in P$ via:

$$s = \frac{p}{2\sqrt{1 - \|p\|^2}}, \quad r_s^2 = \frac{\|p\|^2}{4(1 - \|p\|^2)} - \frac{1}{\sqrt{1 - \|p\|^2}}. \quad (5)$$

Power cells, together with their intersections, can be computed in the Euclidean space via standard algorithms from computational geometry, for example, by lifting the points to a hyperboloid and constructing a convex hull [7]. This enables us to compute hyperbolic Voronoi cells, together with hyperbolic Delaunay triangulations, via Theorem 1.

However, one subtlety is that power cells might intersect outside the unit ball in \mathbb{R}^n . Since these intersections do not define hyperbolic Delaunay simplices, they need to be removed – see Fig. 2 (right). By duality, the simplices correspond to (potentially unbounded) convex polytopes in \mathbb{R}^n . Therefore, it is possible to determine whether the latter intersect \mathbb{K}^n by finding their point of minimum norm, which defines a quadratic programming problem [6]. When $n = 2$, this reduces to determine whether segments and half-lines intersect the unit circle in the Euclidean plane, which can be addressed by elementary algebraic methods. This is achieved in **Step 2.2** of Algorithm 1.

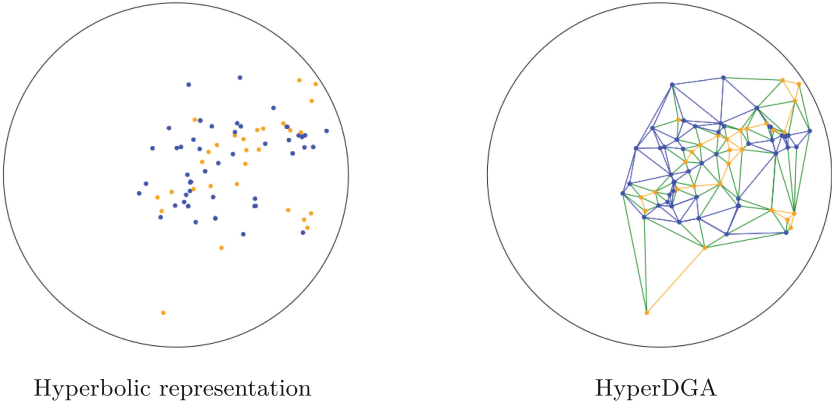


Fig. 3. Left: Klein-Beltrami hyperbolic representation (via Poincaré embedding [13]) of neoblasts 4 & 7 [21] (blue and orange). Right: Depiction of their corresponding homogeneous edges (blue and orange) and heterogeneous edges (green). (Color figure online)

4.3 HyperDGA

We now introduce HyperDGA, a similarity score between two sets of datapoints embedded in a hyperbolic space. This is computed in the final **Step 3** of Algorithm 1. Inspired by DCA [22], we leverage the Delaunay graph constructed on

the union of both datasets. This graph naturally captures the geometric and topological structure of the data by connecting neighboring points via edges whose length adapts to local density changes. The core idea is to measure the geometric alignment of two finite subsets $A, B \subset \mathbb{K}^n$ by counting the proportion of *heterogeneous* edges of the Delaunay graph i.e., the edges connecting points of A and B – see Fig. 3 for an illustration. As opposed to heterogeneous edges, the *homogeneous* ones connect pairs of points that both belong to either A or B . Intuitively, low HyperDGA scores indicate that the sets A and B are geometrically aligned. This leads to the following definition.

Definition 3. Let \mathcal{E} be the edges of the Delaunay graph of $A \cup B$ and consider the heterogeneous edges $\tilde{\mathcal{E}} \subseteq \mathcal{E}$ connecting a point in A to one in B . We define:

$$\text{HyperDGA}(A, B) = 1 - \frac{|\tilde{\mathcal{E}}|}{|\mathcal{E}|} \in [0, 1]. \quad (6)$$

5 Experiments

We provide an empirical investigation of HyperDGA via two experiments on synthetic data and one on real-life data. The goal of the first experiment is to demonstrate the effectiveness of HyperDGA in measuring hyperbolic set distances, and to showcase the capabilities of inferring the noise injected in the original data by analyzing its hyperbolic embedding. The second experiment is aimed at showcasing the relevance of HyperDGA as a tool to evaluate latent representations of a Hyperbolic VAE. Finally, in the last experiment, we validate HyperDGA on real-life biological datasets of varying complexity by showing that the measured hyperbolic distances are coherent with the semantics or domain knowledge.

We compare HyperDGA to the two baselines given by hyperbolic versions of classical metrics between finite sets. Specifically, we consider the (symmetrized) *Chamfer* and *Wasserstein* distances, which are defined for $A, B \subset \mathbb{K}^n$ of the same cardinality as:

$$\begin{array}{cc} \text{Wasserstein} & \text{Chamfer} \\ \min_{\pi: A \rightarrow B} \sum_{p \in A} d(p, \pi(p)) & \sum_{p \in A} \min_{q \in B} d^2(p, q) + \sum_{q \in B} \min_{p \in A} d^2(p, q) \end{array}$$

Here, d is the Klein-Beltrami distance (Eq. 3) and π denotes a bijection between A and B representing a discrete optimal transport.

We implement all of the experiments the Python programming language. Our code is available at a public repository². All the experiments are performed on a machine with CPU Intel Core i7 and with 16GB of RAM.

² <https://github.com/anissmedbouhi/HyperDGA>.

5.1 Synthetic Data with Hyperbolic VAE

For our two experiments on synthetic data, following [16], we consider a binary tree A_0 of depth $d = 11$ whose nodes consist of binary sequences representing a simple simulation of protein evolution. By construction, the graph geodesic distance between any pair of nodes in the tree is equal to the Hamming distance between their associated sequences. Furthermore, we consider a perturbed version of A_0 denoted A_ϵ where we randomly flip the value of each coordinate of each node with probability ϵ . The resulting points in A_ϵ are encoded in a Euclidean space of $2^d - 1$ dimensions (one dimension for each binary coordinate). We then train a Hyperbolic VAE [16] on a dataset T sampled with $\epsilon = 0.1$, in order to represent data in a hyperbolic space. The architecture of the model consists of a Multi Layer Perceptron of depth 3 and 100 hidden variables at each layer for both encoder and decoder, with hyperbolic tangent as the activation function and a latent space of dimension 2. We train the Hyperbolic VAE for 100000 epochs. In what follows, we denote the encoder and the decoder by E and D , respectively.

Experiment 1: Distance Behavior and Noise Inference. Following the procedure explained above, we construct 10 synthetic datasets in 2047 dimensions A_ϵ , $\epsilon \in \{0.1, 0.2, \dots, 0.9, 0.99\}$. As ϵ increases, a larger amount of noise is injected in the binary tree data, resulting in more mutations. This can be visualized in Fig. 4 (top), where the hyperbolic encoding of A_ϵ shifts visibly. Our goal is comparing the encoding $E(T)$ of the training set with the encoding $E(A_\epsilon)$ of the perturbed data. To this end, we measure their similarity in terms of HyperDGA and of the hyperbolic versions of Chamfer and Wasserstein distances.

As evident from Fig. 5, $\text{HyperDGA}(E(T), E(A_\epsilon))$ is strictly monotonic with respect to ϵ , which is coherent with the increase of the amount of noise in the data. The decrease of the number of heterogeneous edges of the hyperbolic Delaunay graph is visualized in Fig. 4 (bottom). Moreover, as reported in Table 1, all the considered hyperbolic distances are strongly correlated with the perturbation parameter ϵ . HyperDGA with hyperbolic Wasserstein exhibit the largest correlation. From Fig. 5, it is clear that the noise injected in the original data can be inferred from the latent representation. In other words, in a context of *speciation*, the amount of mutations between two populations can be inferred from their latent representation by measuring distances between sets via HyperDGA or the baselines.

Table 1. Comparison of the hyperbolic distances in terms of their Pearson correlation with the perturbation ϵ .

Hyperbolic distance	HyperDGA	Chamfer	Wasserstein
Correl. w/ ϵ	0.97	0.81	0.98

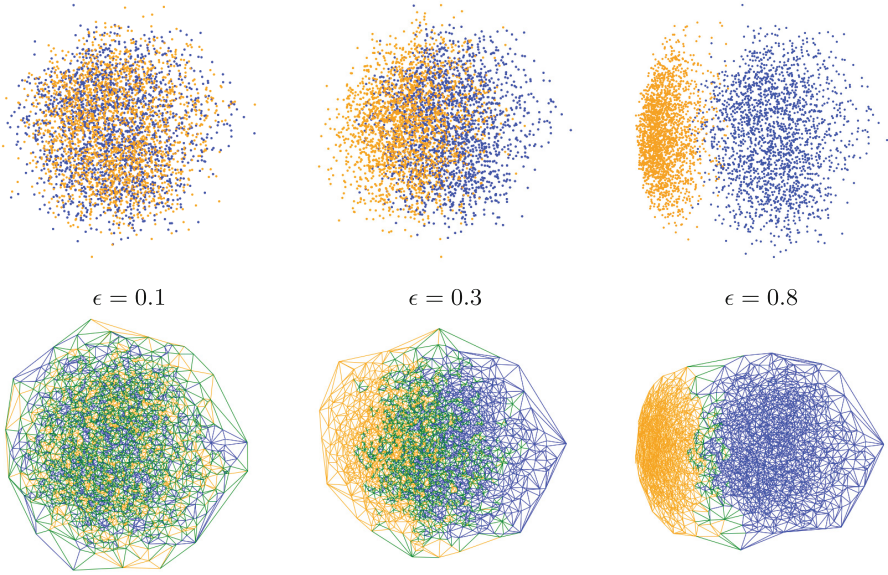


Fig. 4. Top: Poincaré hyperbolic encodings of the training set (blue) and A_ϵ (orange). Bottom: Corresponding Klein-Beltrami visualizations of HyperDGA, with homogeneous edges (blue & orange) and heterogeneous ones (green). (Color figure online)

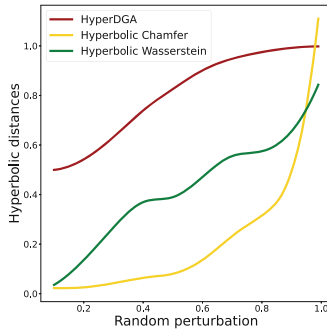


Fig. 5. Hyperbolic distances between the encodings of the training set and A_ϵ in function of the random perturbation ϵ .

Experiment 2: Evaluation of Latent Representation. In this experiment, we compare HyperDGA to the baselines as a method to evaluate the representation inferred by a Hyperbolic VAE. To this end, during training, we measure the given distance between the encoding of the training set $E(T)$ and the encoding of the decoding of its encoding $E(D(E(T)))$, as illustrated in Fig. 6. We then investigate the correlation between this quantity and the loss of the VAE, as

well as with a supervised performance score. Intuitively, as the training of the Hyperbolic VAE progresses, its latent representation and reconstruction improve, resulting in alignment between the latent distributions we consider. Following [16], the supervised performance score is given by the absolute difference between the geodesic distances in the binary tree data A_0 – computed as the Hamming distance in the input data space – and the hyperbolic latent distances of the encoding of A_0 . In other words, the latter score evaluates whether the encoder is isometric with respect to the binary tree data. We compute the correlation every 100 epochs of training, up to 1700 epochs.

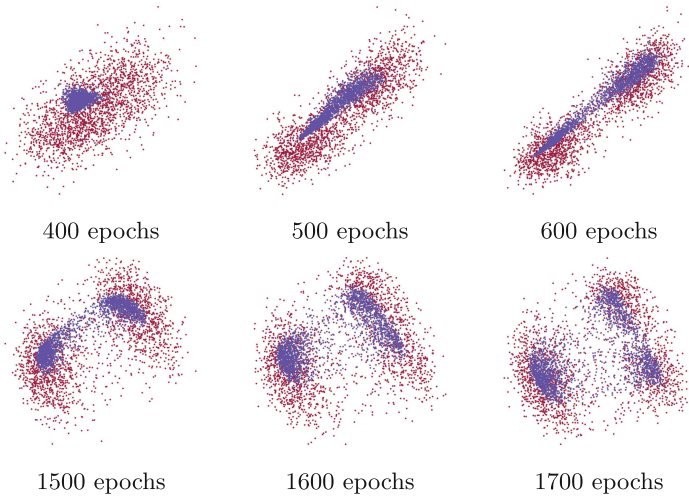


Fig. 6. Visualization of cluster creation in the encoding of the training set (red) and the encoding of its decoding (purple) while training a Hyperbolic VAE. (Color figure online)

Table 2 reports the means and standard deviations over three random initializations of the VAE. Moreover, Fig. 7 displays plots of the various quantities involved as training progresses, while Fig. 6 shows visualizations of the hyperbolic latent space. All three methods exhibit a strong Pearson correlation (>0.5) with the loss and with the performance score. Even further, HyperDGA outperforms the baselines in that it exhibits a stronger correlation. Jumps in HyperDGA

Table 2. Comparison of the hyperbolic distances in terms of their Pearson correlation with the loss and the performance score.

Hyperbolic distance	HyperDGA	Chamfer	Wasserstein
Correl. w/ Loss	0.76 ± 0.07	0.52 ± 0.11	0.60 ± 0.06
Correl. w/ Performance	-0.74 ± 0.07	-0.58 ± 0.17	-0.65 ± 0.07

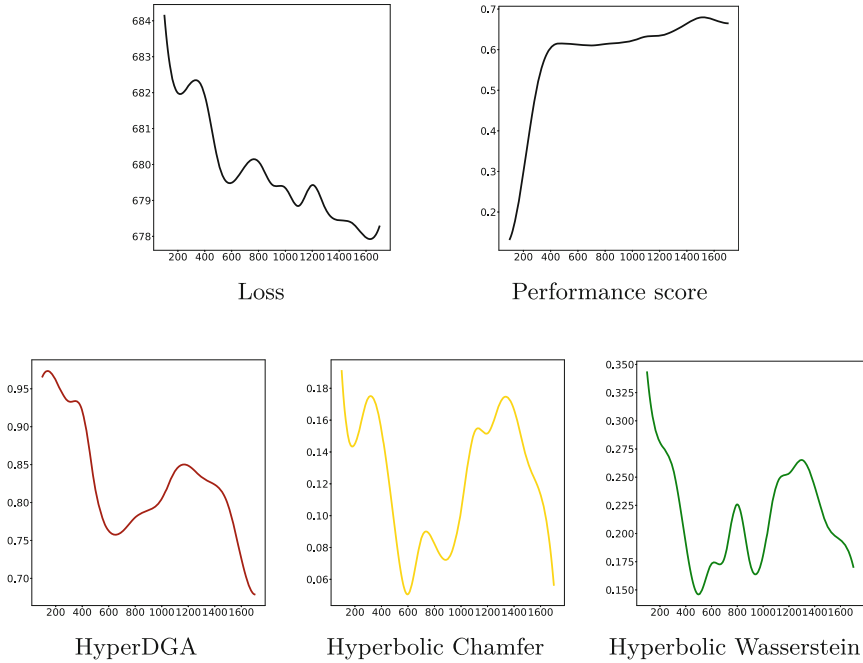


Fig. 7. Plots of different metrics in function of the training iteration step, for one seed of the neural network.

are associated with the creation of new clusters, as illustrated in Fig. 6 around epochs 500 (from one to two clusters) and 1600 (from two to three clusters). This means that in addition to the geometric information, HyperDGA takes into account topological changes in the latent representation.

5.2 Real-Life Biological Data With Poincaré Embedding

In our last experiment, we validate HyperDGA on hyperbolic embeddings of real-life biological data. In order to embed the data, we rely on the dimensionality reduction technique from [13]. The latter is based on the Poincaré embedding method [17], but is specifically designed to visualize single cell data in biology and to uncover hierarchies. We consider three RNA single-cell sequencing datasets to account for different scenarios and complexity in terms of geometric and topological structure in data:

- *Olsson*: the mouse myelopoiesis data from [19] containing 382 cells³.
- *Paul*: the mouse myeloid progenitors MARS-seq data from [20] containing 2730 cells⁴.

³ Accession code at <https://tinyurl.com/olssondata>.

⁴ Accession code at <https://tinyurl.com/pauldata>.

- *Planaria*: the planaria droplet-based single cell transcriptome profiling from [21] containing 21612 cells⁵.

Each dataset consists of several groups of cells and exhibits a hierarchical structure due to the cellular differentiation process where stem cells specialize into, for example, muscle or neuron cells. We expect developmentally similar groups to yield a low score, while dissimilar groups to yield a high score. The pre-processed data are downloaded from <https://tinyurl.com/3vw32jad>.

Experiment 3: Validation on Real-Life Biological Data. For each dataset, we measure HyperDGA and the baselines between each pair of three groups of cells and verify that the resulting values are coherent with the domain knowledge. The hyperbolic embeddings of the three groups for each dataset are displayed in Fig. 8. For *Olsson* and *Paul*, we consider two similar classes that are close in the hierarchical tree data: *Mono* and *Gran* for *Olsson*, *13Baso* and *14Mo* for *Paul*. We compare these classes to a third class that lies further in the hierarchy: *HSPC-1* for *Olsson*, *7MEP* for *Paul*. For *Planaria*, we divide the whole dataset in three different groups of cells according to [21]: *Neoblasts*, *Progenitors* and *Differentiated* cells. Given the semantics from biology [21], we expect the distance between the neoblasts and the differentiated cells to be larger than both the distance between the neoblasts and the progenitors, and the distance between the progenitors and the differentiated cells.

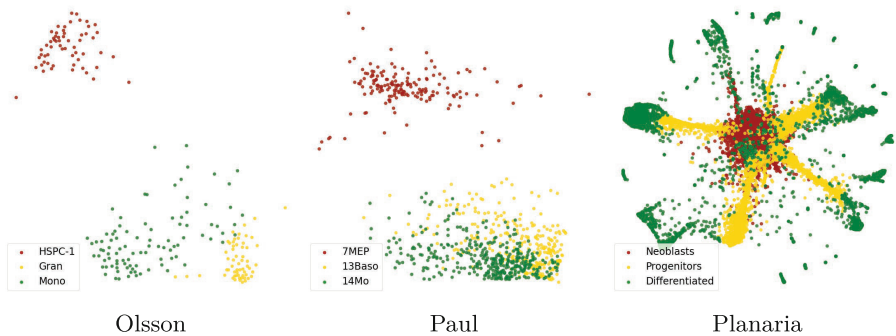


Fig. 8. Poincaré embeddings of three classes from three different datasets.

All the results are reported in Table 3 – see the supplementary material for HyperDGA visualizations⁶. As evident from the results, HyperDGA and the baselines are consistent with the domain knowledge for all the scenarios among the three datasets. Furthermore, for the embeddings of the neoblasts versus the progenitors from the *Planaria* dataset (see Fig. 8), we argue that the values of

⁵ Accession code at <https://tinyurl.com/plassdata>.

⁶ <https://github.com/anissmedbouhi/HyperDGA/blob/main/supp.pdf>.

Table 3. Hyperbolic distances (HyperDGA, Chamfer, Wasserstein) of three classes for three datasets: *Olsson* (top), *Paul* (middle) and *Planaria* (bottom).

	<i>Gran</i>	<i>HSPC-1</i>
<i>Mono</i>	(0.943, 2.6, 0.9)	(0.988, 5.6, 1.3)
<i>Gran</i>	/	(0.987, 16.7, 2.7)
	<i>14Mo</i>	<i>7MEP</i>
<i>13Baso</i>	(0.713, 0.08, 0.25)	(0.991, 2.70, 0.71)
<i>14Mo</i>	/	(0.995, 3.18, 0.70)
	<i>Progenitors</i>	<i>Differentiated</i>
<i>Neoblasts</i>	(0.93, 0.07, 0.4)	(0.97, 2.08, 1.2)
<i>Progenitors</i>	/	(0.96, 1.42, 0.9)

HyperDGA are topologically and geometrically more consistent with the true semantics. Specifically, these two embeddings are visibly different and with significant non-overlapping regions, whereas both Chamfer and Wasserstein give remarkably low distances between them.

6 Conclusions, Limitations and Future Work

We have introduced HyperDGA – a tool for hyperbolic data analysis to complement hyperbolic dimensionality reduction techniques and to evaluate hyperbolic representation learning models. We have demonstrated on synthetic and real-life biological data the relevance of HyperDGA as a method to measure the geometric alignment between two sets represented in a hyperbolic space.

One limitation of HyperDGA is that, by construction, it takes values in rational numbers i.e., $\text{HyperDGA}(A, B) \in \mathbb{Q}$. Since \mathbb{Q} is totally disconnected, HyperDGA is discontinuous in A and B . However, continuity and, especially, differentiability are desirable properties in the context of optimization since they enable gradient-based methods, such as gradient descent. The latter is potentially useful, since similarity scores between sets can be exploited as optimization objectives for generative modelling and regularization. Therefore, designing continuous and differentiable versions of HyperDGA represents a promising line for future research.

Another future direction is to investigate practical applications of HyperDGA in hyperbolic machine learning together with domain experts. For example, in the context of biology, similarity scores such as HyperDGA can be deployed to measure alignment between different types of cells, which are commonly represented in a hyperbolic space [13, 29]. In particular, the geometric alignment

between tumoral versus healthy cells might provide insights into the evolution of a cancer, leading to a promising biomedical application of HyperDGA.

Acknowledgements. We wish to thank Yoshihiro Nagano for the technical help, and Mohammad Al-Jaff and Michael Welle for their feedback. We are also grateful to the anonymous reviewers for their comments. This work has been supported by the Swedish Research Council, Knut and Alice Wallenberg Foundation, and the European Research Council (ERC AdG BIRD).

References

1. Beltrami, E.: Saggio di interpretazione della geometria Non-Euclidea. s.n. (1868)
2. Boguñá, M., Papadopoulos, F., Krioukov, D.: Sustaining the internet with hyperbolic mapping. *Nat. Commun.* **1**, 62 (09 2010)
3. Boissonnat, J.D., Yvinec, M.: *Algorithmic Geometry*. Cambridge University Press, Cambridge (1998)
4. Chamberlain, B.P., Clough, J., Deisenroth, M.P.: Neural embeddings of graphs in hyperbolic space. arXiv preprint [arXiv:1705.10359](https://arxiv.org/abs/1705.10359) (2017)
5. Chami, I., Gu, A., Nguyen, D.P., Re, C.: Horopca: hyperbolic dimensionality reduction via horospherical projections. In: Meila, M., Zhang, T. (eds.) *Proceedings of the 38th International Conference on Machine Learning*. *Proceedings of Machine Learning Research*, vol. 139, pp. 1419–1429. PMLR (2021)
6. De Loera, J.A., Haddock, J., Rademacher, L.: The minimum Euclidean-norm point in a convex polytope: Wolfe’s combinatorial algorithm is exponential. In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 545–553 (2018)
7. Edelsbrunner, H., Seidel, R.: Voronoi diagrams and arrangements. In: *Proceedings of the First Annual Symposium on Computational Geometry*, pp. 251–262 (1985)
8. Fletcher, P., Lu, C., Pizer, S., Joshi, S.: Principal geodesic analysis for the study of nonlinear statistics of shape. *IEEE Trans. Med. Imaging* **23**, 995–1005 (2004)
9. Guo, Y., Guo, H., Yu, S.X.: Co-SNE: dimensionality reduction and visualization for hyperbolic data. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21–30 (2022)
10. Khrulkov, V., Oseledets, I.: Geometry score: a method for comparing generative adversarial networks. In: Dy, J., Krause, A. (eds.) *Proceedings of the 35th International Conference on Machine Learning*. *Proceedings of Machine Learning Research*, vol. 80, pp. 2621–2629. PMLR (2018)
11. Kingma, D.P., Welling, M.: Auto-encoding variational Bayes. In: Bengio, Y., LeCun, Y. (eds.) *ICLR* (2014)
12. Kleinberg, R.: Geographic routing using hyperbolic space. In: *IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*, pp. 1902–1909 (2007)
13. Klimovskaia, A., Lopez-Paz, D., Bottou, L., Nickel, M.: Poincaré maps for analyzing complex hierarchies in single-cell data. *Nat. Commun.* (2020)
14. Kynkäänniemi, T., Karras, T., Laine, S., Lehtinen, J., Aila, T.: Improved precision and recall metric for assessing generative models. In: Wallach, H., Larochelle, H., Beygelzimer, A., d’Alché-Buc, F., Fox, E., Garnett, R. (eds.) *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc. (2019)

15. Mathieu, E., Le Lan, C., Maddison, C.J., Tomioka, R., Whye Teh, Y.: Continuous hierarchical representations with poincaré variational auto-encoders. In: *Advances in Neural Information Processing Systems* (2019)
16. Nagano, Y., Yamaguchi, S., Fujita, Y., Koyama, M.: A wrapped normal distribution on hyperbolic space for gradient-based learning. In: *International Conference on Machine Learning* (2019)
17. Nickel, M., Kiela, D.: Poincaré embeddings for learning hierarchical representations. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) *Advances in Neural Information Processing Systems*. vol. 30. Curran Associates, Inc. (2017)
18. Nielsen, F., Nock, R.: Hyperbolic voronoi diagrams made easy. In: *2010 International Conference on Computational Science and Its Applications*, pp. 74–80 (2010)
19. Olsson, A., et al.: Single-cell analysis of mixed-lineage states leading to a binary cell fate choice. *Nature* **537** (08 2016)
20. Paul, F., et al.: Transcriptional heterogeneity and lineage commitment in myeloid progenitors. *Cell* **163** (11 2015)
21. Plass, M., et al.: Cell type atlas and lineage tree of a whole complex animal by single-cell transcriptomics. *Science* **360**, eaaq1723 (2018)
22. Poklukar, P., Polianskii, V., Varava, A., Pokorný, F.T., Jensfelt, D.K.: Delaunay component analysis for evaluation of data representations. In: *International Conference on Learning Representations* (2022)
23. Poklukar, P., Varava, A., Kragic, D.: Geomca: geometric evaluation of data representations. In: Meila, M., Zhang, T. (eds.) *Proceedings of the 38th International Conference on Machine Learning*. *Proceedings of Machine Learning Research*, vol. 139, pp. 8588–8598. PMLR (2021)
24. Rezende, D.J., Mohamed, S., Wierstra, D.: Stochastic backpropagation and approximate inference in deep generative models. In: Xing, E.P., Jebara, T. (eds.) *Proceedings of the 31st International Conference on Machine Learning*. *Proceedings of Machine Learning Research*, vol. 32, pp. 1278–1286. PMLR, Beijing, China (2014)
25. Sala, F., De Sa, C., Gu, A., Re, C.: Representation tradeoffs for hyperbolic embeddings. In: Dy, J., Krause, A. (eds.) *Proceedings of the 35th International Conference on Machine Learning*. *Proceedings of Machine Learning Research*, vol. 80, pp. 4460–4469. PMLR (2018)
26. Sarkar, R.: Low distortion delaunay embedding of trees in hyperbolic plane. In: van Kreveld, M., Speckmann, B. (eds.) *GD 2011*. LNCS, vol. 7034, pp. 355–366. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-25878-7_34
27. Skopek, O., Ganea, O.E., Bécigneul, G.: Mixed-curvature variational autoencoders. In: *International Conference on Learning Representations* (2020)
28. Tifrea, A., Bécigneul, G., Ganea, O.E.: Poincaré glove: hyperbolic word embeddings. In: *7th International Conference on Learning Representations (ICLR)* (2019)
29. Zhou, Y., Sharpee, T.O.: Hyperbolic geometry of gene expression. *iScience* **24**(3), 102225 (2021)



ApmNet: Toward Generalizable Visual Continuous Control with Pre-trained Image Models

Haitao Wang and Hejun Wu^(✉)

School of Computer Science and Engineering, Sun Yat-sen University,
Guangzhou, China

wuhejun@mail.sysu.edu.cn

Abstract. In this paper, we propose ApmNet, an effective asymmetric two-stream framework for generalizable visual continuous control. Since pre-trained image models (PIM) have achieved remarkable success in many fields, researchers have attempted to extend PIM to visual continuous control tasks. However, directly applying the PIM to the visual control tasks is difficult due to the absence of “task-specific” information in pre-training. Fine-tuning the PIM for a control task can be an effective solution. However, it is prone to overfitting, potentially causing the PIM to lose the generalization ability gained through pre-training. To address these issues, ApmNet adopts an asymmetric dual-stream network design. ApmNet uses a frozen pre-trained Masked Autoencoder (MAE) as a visual backbone for policy learning. The reconstructed distorted view as data augmentation introduces a very powerful generalization ability to ApmNet. Then ApmNet uses a separate pre-trained network with an adapter module to fuse different pre-trained representations and generate actions. The adapter module can ensure that ApmNet bridges the distribution shift between pre-training data and vision states. After training, ApmNet abandoned MAE and only relied on the separate pre-trained network with an adapter as the final vision backbone. Through this asymmetric strategy, ApmNet can achieve good generalization ability by only fine-tuning a small number of parameters. Extensive experiments on DMControl, DMControl-GB, and MetaWorld benchmarks verify the effectiveness of our ApmNet. Empirical evidence suggests that ApmNet outperforms previous state-of-the-art methods in terms of sample efficiency and generalization.

Keywords: Pre-trained image models · Visual reinforcement learning · Generalization · Masked autoencoder

1 Introduction

Large-scale pre-training [4, 5] have led to a series of breakthroughs for Computer Vision (CV) and Natural Language Processing (NLP). In recent years, some

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70352-2_8.

researchers have extended the large-scale pre-training paradigm to the field of visual continuous control [27, 42]. These works first use out-of-domain datasets to pre-train a visual encoder. Then they apply the pre-trained image models without fine-tuning to continuous control tasks by processing the vision states frame-by-frame. Compared to the simple Learning-from-Scratch (LfS) methods, Learning-from-Pretraining (LfP) can effectively improve the training efficiency and generalization ability of visual continuous control.

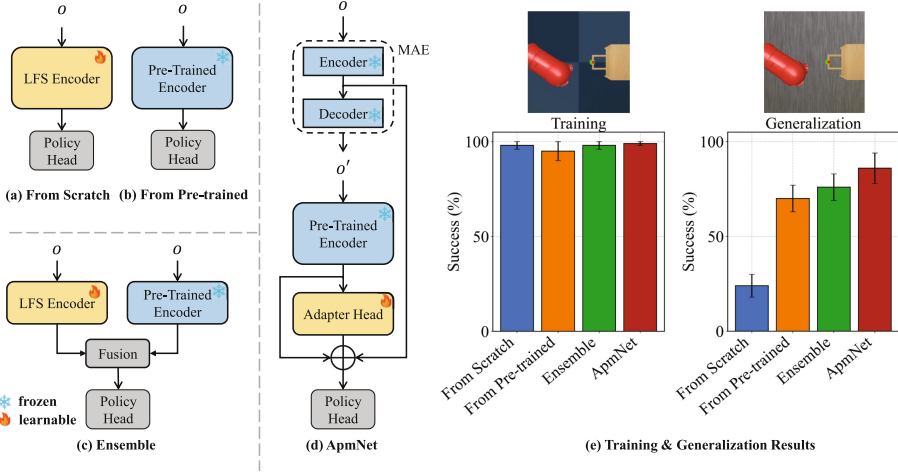


Fig. 1. Prior methods for learning policy. (a) from scratch, (b) from a pre-trained visual encoder with a frozen backbone, (c) an ensemble model that contains both a pre-trained frozen encoder and learnable encoder, and (d) the proposed asymmetric architecture. The right Figure (e) shows their performances on Metaworld generalization task, evaluated on both seen (training) and unseen (generalization) instances.

Indeed, the experiences from the fields of CV/NLP [17, 37] indicate that directly applying pre-trained image models to continuous control tasks may not be optimal. This is due to the distributional shift that exists between the vision states of continuous control and large-scale pre-trained datasets [21, 27]. An intuitive solution would be to fine-tune the pre-trained image model. However, recent research [42] has shown that when fine-tuned the pre-trained model for control tasks, the pre-trained model will lose most of their pre-trained generalization capabilities.

To better understand the effect of the pre-trained image model in visual continuous control tasks, we compare the LfS, LfP, and ensemble model, in Metaworld generalization tasks [34]. Compared to LfS (Fig. 1 (a)), LfP (Fig. 1 (b)) achieves a higher success rate in generalization task but poor in training. This result demonstrates that there is a data distribution shift under visual state and pre-trained datasets. Compared to LfS and LfP, the ensemble model (Fig. 1 (c))

achieves a higher success rate in both the training and generalization environment. This observation indicates that the ensemble models exhibit significant improvements across all tasks. However, the naive approach of ensembling models results in notable additional computational costs due to inferring two models simultaneously. This limitation hinders practical applications in real-world scenarios. Therefore, it is impressive to explore methods for consolidating the ensemble model’s knowledge into a single model, mitigating the computational burden.

To this end, we propose ApmNet (**A**dapting **p**re-trained **m**odels), a simple yet effective asymmetric two-stream framework for generalizable visual continuous control tasks. As shown in Fig. 1 (d), ApmNet uses a frozen pre-trained Masked Autoencoder (MAE) as a visual backbone for both data augmentation and policy learning. The reconstructed distorted view as data augmentation introduces the very powerful “**generalization ability**” to continuous control tasks. Then we introduce a separate pre-trained network with an adapter module to fuse different pre-trained representations and generate actions. The learnable adapter incorporates “**domain-specific**” features from the raw observations to help handle distribution shifts. It is worth noting that we only use MAE in the model training phase, while in the model testing and deployment stages, we only retain the separated pre-trained network and adapter. Through this asymmetric training strategy, ApmNet not only possesses high sampling efficiency and generalization ability but also reduces computational consumption during the application phase.

To validate the effectiveness of our framework, we conduct a series of experiments on 3 challenging benchmarks: DMControl Suite [31], DMControl Generalization Benchmark (DMControl-GB) [14], and MetaWorld manipulation tasks [41]. Empirical studies have demonstrated that ApmNet outperforms previous state-of-the-art visual control methods in terms of sample efficiency and generalization. Below, we highlight the contribution of this paper:

1. We propose ApmNet, a simple yet effective asymmetric two-stream framework. Our framework can adapt any pre-trained image model to a reinforcement learning agent on visual continuous control tasks. The asymmetric architecture enables the model to retain both generalized and domain-specific features with low computational cost.
2. ApmNet outperforms state-of-the-art methods in 3 visual control benchmarks, showing significant improvement of our method in generalization and sample efficiency.

2 Related Work

2.1 Pre-trained Models for Policy Learning

Using pre-trained models from other domains for visual continuous control has achieved promising results [9, 18, 28, 35, 40]. For example, RRL [27] and PIE-G [42] use pre-trained ResNet for state representation learning that can generalize

to unseen visual scenarios in a zero-shot manner. VRL3 [33] proposes a multi-stage pre-trained framework for solving visual control tasks. APV [26] introduces a framework that learns representations useful for understanding the dynamics via generative pre-training on videos. Recently, some studies [2, 8] investigate the use of pre-trained language models such as BERT [6] for understanding and interpreting visual scenarios. However, most of the above methods require additional computational costs during fine-tuning or directly use the pre-trained encoder without fine-tuning. Our proposed ApmNet leverages existing pre-trained visual models, tunes only a small number of model parameters (much more efficient than full fine-tuning), and achieves performance that is comparable to, or even better than, that of previous state-of-the-art methods.

2.2 Data Augmentation for Policy Learning

Data augmentation has become a widely used technique in visual continuous control to improve sample efficiency and generalization [14, 24, 39]. The advantages of data augmentation for visual control can be viewed from two aspects: (1) It can significantly increase the amount of original data, thus improving the sample efficiency. Typical work includes methods like RAD [20], DrQ [19], DrQ-v2 [38] and DrAC [22], which use classical manual augmentation techniques such as flipping and cropping for data augmentation, significantly improving the sample efficiency of visual RL. (2) It introduces additional diversity to the original training data, thereby enhancing the agent’s robustness to variations. For example, CURL [29] and ATC [30] use data augmentation techniques to construct positive and negative sample pairs for contrastive representation learning, aiming to learn robust representations of states. Our ApmNet differs from theirs in that we use a generative model for data augmentation, regularize both actor and critic, and focus on both the problems of generalization and sample efficiency.

3 Preliminaries

3.1 Continuous Control from Image

In this work, we formalize the problem of continuous control tasks as an MDP (Markov Decision Process) problem. We consider a standard reinforcement learning (RL) setup where an agent interacts with an environment. At each timestep t , the agent takes an action a_t in a state s_t , and receives a scalar reward r_t while the environment transitions to a new state s_{t+1} . We formulate the standard RL task as a Markov Decision Process (MDP) [3], which is defined as a tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$. Here, \mathcal{S} is the state space, \mathcal{A} is the action space, $P : \mathcal{S} \times \mathcal{A} \rightarrow \Delta(\mathcal{S})$ is the transition function of system that defines a probability distribution over the next state given the current state and action, where $\Delta(\mathcal{S})$ is the distribution over the state space \mathcal{S} . The reward function $R : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ assigns a scalar reward to each state-action pair (s_t, a_t) , and $\gamma \in [0, 1)$ is the discount factor. The

objective of RL is to learn a policy π that maximizes the expected discounted sum of rewards:

$$J = \mathbb{E}_{s_0, a_0, s_1, a_1, \dots} \left[\sum_{t=0}^{\infty} \gamma^t r_t \right] \quad (1)$$

where s_0 is the initial state and $a_t = \pi(s_t)$ is the action selected by the policy at timestep t .

To model image-based decision-making task as an RL task, we formulate this task as an infinite-horizon MDP. Typically, a single video frame does not capture the full latent state of the system. Therefore, we approximate the current state of the system by stacking three consecutive prior video frames. The resulting MDP can be described as a tuple $(\mathcal{X}, \mathcal{A}, P, R, \gamma)$, where \mathcal{X} is the collection of state $x_t = (s_t, s_{t+1}, s_{t+2})$, represented as a three-stack of video frames. In this work, we use neural networks to process the states. Let f_θ denote the encoder parameterized by θ , which maps x_t into a d -dimensional feature. Then, the extracted feature will be fed into a policy network π_ω to get the action a_t at timestep t , i.e., $a_t = \pi_\omega(f_\theta(x_t))$.

3.2 Masked Autoencoders

The Masked Autoencoder (MAE) [15] is an effective and scalable visual learning method that operates by randomly masking patches of input visual data and reconstructing the missing patches in the latent space. In particular, the observation o_t is first processed by a patchify stem, as introduced in the Vision Transformer (ViT) [7], which reshapes it into a sequence of 2D patches h_t . A subset of patches is then randomly masked with a ratio of m to obtain h_t^m . The remaining patches are embedded into low-dimensional vectors using a ViT encoder. Finally, a simple ViT decoder reconstructs the observation by processing tokens from the encoder through Transformer layers followed by a linear output head.

$$\text{Patchify: } h_t \sim f^{\text{patch}}(o_t) \quad (2)$$

$$\text{Masking: } h_t^m \sim p^{\text{mask}}(h_t^m | h_t, m) \quad (3)$$

$$\text{ViT encoder: } z_t^m \sim E_\phi(z_t^m | h_t^m) \quad (4)$$

$$\text{ViT decoder: } \hat{o}_t \sim D_\theta(\hat{o}_t | z_t^m) \quad (5)$$

The entire model is jointly optimized to minimize the mean squared error loss (MSE) between the reconstructed and original patches.

4 Method

In this section, we introduce our ApmNet, a simple yet effective framework for visual continuous control that benefits from pre-trained image models to enhance the generalization and sample efficiency. We first introduce the current way of using pre-trained image models for continuous control. Then, we present our asymmetric architecture. Finally, we explain our policy learning framework for continuous control.

4.1 ApmNet Architecture

Because fine-tuning pre-trained image models requires significant computational resources, prior methods (as shown in Fig. 1(b)) [27, 42] use the frozen pre-trained image models as the feature extractor for continuous control tasks. However, these approaches lack flexibility and lead to suboptimal control performance, mainly due to the fact that the datasets used in pre-trained image models have a different distribution from visual continuous control tasks. In addition, although ensemble models (as shown in Fig. 1(c)) can avoid the aforementioned issues, they also come with challenges such as difficult deployment and high computational cost.

Vision Backbone and Mask-reconstruct Augmentation: To enable the model to learn flexible task-specific features while preserving the generalization advantages of the pre-trained network, ApmNet adopts the design of an asymmetric two stream architecture. As shown in Fig. 1(d), ApmNet explicitly leverages the self-supervised pre-trained MAE as the vision backbone. Inspired by a recent study [36] that has shown pre-trained MAE is not only a powerful feature extractor but also a powerful data enhancer, ApmNet uses the reconstructed distorted view as data augmentation for policy learning. Specifically, given the random binary mask M and the t -th step state x_t , ApmNet first divide the masked state $M \odot x_t$ into non-overlapped patches and discard the masked patches. Then the remaining unmasked patches are fed into the pre-trained vit encoder E_ϕ and decoder D_θ to generate the reconstructed state $\hat{x}_t = D_\theta(E_\phi(M \odot x_t))$. The reconstructed state \hat{x}_t is seen as the augmented version of x_t and then input into another pre-trained image encoder (ResNet-18 [16]) for secondary feature extraction.

Although MAE can produce new data, it might not fully simulate the noise and variability present in real data, leading to a discrepancy between the generated data and the actual data. To tackle this issue, inspired by recent progress on offline RL [1], ApmNet employ a simple method, symmetric sampling, to introduce masked augmented states into policy learning. Specifically, for each batch, ApmNet sample 50% of the data for masked data augmentation, and the remaining 50% undergo linear transformations. As we will see in later sections, this simple sampling strategy is surprisingly effective across a variety of tasks.

Adapter Tuning for Control: The pre-trained MAE and ResNet provide strong generalization capabilities for visual control. Subsequently, ApmNet incorporates a very simple yet effective structure: the Adapter module, to introduce “task-specific” information to the overall model. As shown in Fig. 2(a), the adapter is a bottleneck architecture that consists of two convolutional layers and an activation layer in the middle. The first convolutional layer projects the input to a low dimension and the second convolutional layer projects it back to the original dimension. ApmNet add this simple adapter after each residual block of ResNet. During training, all the other layers of the ResNet are frozen

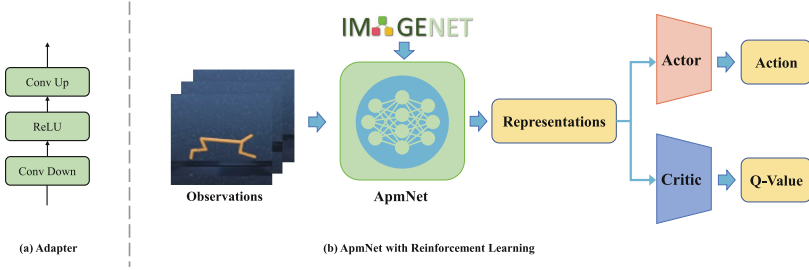


Fig. 2. (a) The architecture of the Adapter. (b) ApmNet with Reinforcement Learning.

while only the adapters are updated. In this way, ApmNet is able to introduce “task-specific” information to the overall model with minimal computational cost. It is worth noting that the primary emphasis of this paper is on resolving the issue of pre-trained models lacking “task-specific” information, rather than designing a new adapter module. The flexibility of ApmNet allows for any type of adapter module to be seamlessly integrated into its structure.

4.2 Asymmetric Policy Learning

The overview of ApmNet with reinforcement learning algorithm for visual control is shown in Fig. 2(b). The asymmetry of ApmNet is reflected in the use of different visual backbones during the training and testing phases. During the training phase, we adopt the entire ApmNet as the vision backbone that helps the agent to learn a useful low-dimensional state representation. We implement DrQ-v2 [38] as the basic visual reinforcement learning algorithm for downstream control tasks. DrQ-v2 is an off-policy actor-critic approach that uses data augmentation to learn directly from visual data. Our agent is trained with two Q_{θ_k} value functions and the corresponding target $Q_{\bar{\theta}_k}$. The critic loss is

$$\mathcal{L}(\theta_k) = \mathbb{E}_{\tau \sim \mathcal{D}} [(Q_{\theta_k}(s_t, a_t) - y)^2] \quad \forall k \in \{1, 2\} \quad (6)$$

where $\tau = (s_t, a_t, r_{t:t+n-1}, s_{t+n})$ is the mini-batch of transitions. y is the n -step TD target:

$$y = \sum_i^{n-1} \tau^i r_{t+i} + \tau^n \min_{k=1,2} Q_{\bar{\theta}_k}(s_{t+n}, a_{t+n}) \quad (7)$$

The actor π_ω is trained with the following loss:

$$\mathcal{L}(\mathcal{D}) = -\mathbb{E}_{s_t \sim \mathcal{D}} \left[\min_{k=1,2} Q_{\theta_k}(s_t, a_t) \right] \quad (8)$$

where s_t is augmented using our symmetric sampling strategy: a 50% probability for non-linear data augmentation using MAE and a 50% probability for weak data augmentation (e.g., random shift). In the setting of generalization, we follow

the way of using the strong augmentation method (e.g., mixup) instead of weak data augmentation. During the testing phase, we discard the MAE component of ApmNet and only use ResNet with the adapter module as the vision backbone. Through this asymmetric approach, the capabilities of different pre-trained models are retained in the adapter module via the backpropagation algorithm during the training phase. By retaining only one branch of the visual encoder during the testing phase, we can significantly preserve the feature extraction capabilities of the model while reducing computational costs.

5 Experiments

In this section, we explore how ApmNet can affect the agent’s generalization performance and sample efficiency. We compare our method with other state-of-the-art baselines on a wide spectrum of tasks. These tasks include DeepMind control suite (DMControl) [31], DeepMind control generalization Benchmark (DMControl-GB) [14], and MetaWorld manipulation tasks [41]. We also conduct a series of ablation studies to take a closer look at the proposed method. It is worth noting that since the best baseline is different in each of the three benchmarks, we compare ApmNet with the best-performing algorithm in each benchmark. All pre-trained model weights used in ApmNet come from the open source community and can be downloaded free of charge from the github of the paper’s author¹.

5.1 Environments Setup

DMControl. To quantify the sample efficiency of ApmNet, we evaluate ApmNet on various continuous control tasks from DMControl [31]. Following the common setup in this benchmark [38], the episode length is 1000 steps with the action repeat of 2, and the reward ranges from 0 to 1. We select several environments from the DMControl’s simple, medium, and difficult tasks for experimental evaluation.

DMcontrol-GB. To quantify the generalization ability, we evaluate ApmNet and baselines on DMControl-GB [31]. RL agents are trained in an original DMControl environment, and we measure generalization to three distinct test distributions: (1) environments with random colors (color hard); (2) environments with simple natural videos as background (video easy); and (3) environments with difficult natural videos as background (video hard).

MetaWorld. To better emulate real-world problems with visual RL, we evaluate ApmNet on various vision-based manipulation tasks from MetaWorld [41]. In all manipulation tasks, the episode length is 500 steps with the action repeat of 2. Detailed parameter settings for all experimental environments are shown in the **Appendix**

¹ ResNet: <https://github.com/pytorch/vision>; MAE: <https://github.com/facebookresearch/mae>.

Table 1. Generalization on unseen video backgrounds. Episode return of methods trained in a fixed environment and evaluated on DMControl-GB with two types of unseen dynamic video background, i.e., *video easy* and *video hard*. ApmNet outperforms the baselines in most of the tasks.



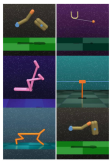
Setting	DMControl-GB	DrQ	DrQ-v2	SpawnNet	PIE-G	ApmNet
	Cartpole, Swingup	138±9	130±3	352±63	401±21	574±33
	Walker, Stand	289±49	151±13	885±36	852±56	916±21
	Walker, Walk	104±22	34±11	623±73	600±28	680±30
	Cup, Catch	92±23	97±27	799±121	786±47	830±13
	Cheetah, Run	32±13	23±5	161±27	154±17	203±5
	Finger, Spin	71±45	21±4	820±18	762±59	834±33
	Cartpole, Swingup	485±105	267±41	752±42	587±61	785±14
	Walker, Stand	873±83	560±48	960±25	957±12	942±45
	Walker, Walk	682±89	175±117	829±58	871±22	929±6
	Cup, Catch	318±157	454±60	889±95	892±68	931±18
	Cheetah, Run	102±30	64±22	301±20	287±20	325±55
	Finger, Spin	533±119	456±15	842±23	837±107	896±62

Table 2. Generalization on random color environments. Episode return of methods trained in a fixed environment and evaluated on DMControl-GB with random colors. ApmNet outperforms the baselines in most of the tasks.

Setting	DMControl-GB	SAC	DrQ	DrQ-v2	SpawnNet	PIE-G	ApmNet
	Cartpole, Swingup	248±24	586±52	277±80	837±23	749±46	692±33
	Walker, Stand	365±79	770±71	413±61	942±26	960±15	983±8
	Walker, Walk	144±19	520±91	168±90	760±145	884±20	923±22
	Cup, Catch	151±36	365±210	468±99	961±10	964±7	986±6
	Cheetah, Run	133±26	100±27	109±45	273±23	369±53	421±67
	Finger, Spin	164±33	277±43	326±64	771±104	801±95	911±65

5.2 Evaluation on Generalization Ability

We compare the generalization ability of ApmNet with strong baselines: **SAC** [11]: a widely used off-policy RL algorithm; **DrQ** [19]: a visual RL algorithm with augmented states based on SAC; **DrQ-v2** [38]: the state-of-the-art visual RL algorithm in terms of continuous control; **SpawnNet** [21]: the state-of-the-art method in terms of generalization through ensemble learning; **PIE-G** [42]: another state-of-the-art method generalization by using pre-trained encoder. For a fair comparison, all baselines have employed the same data augmentation method.

Generalization on Unseen and/or Moving Backgrounds. The generalization abilities of ApmNet and the baselines are evaluated on the DMControl-

Table 3. Generalization on MetaWorld generalization tasks. Evaluation on distracting textures. ApmNet is robust to the texture changing.

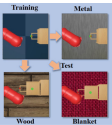
	Setting	SAC	DrQ-v2	SpawnNet	PIE-G	ApmNet
	Training	100%	98%	98%	98%	99%
	Metal	0%	24%	76%	70%	86%
	Wood	0%	46%	92%	92%	95%
	Blanket	0%	8%	76%	73%	78%

Table 4. Parameter comparison between ApmNet and Baselines.

Method	Encoder	Fine-tuning	Tunable Params
DrQ-v2	CNN	Fine-tune CNN	30.36k
SpawnNet	ViT+CNN	Fine-tune CNN	30.36k
PIE-G	ResNet	Frozen	—
ApmNet	MAE+ResNet	Adapter	8.300K

GB benchmark. We evaluate ApmNet on the challenging generalization setting: *video easy* and *video hard*, and compare to several recent state-of-the-art baselines. Results are shown in Table 1. We find that ApmNet outperforms previous methods in **11** out of **12** instances. In addition, Table 4 shows the number of model parameters and the number of tunable parameters for ApmNet and baselines. As can be seen, ApmNet surpasses DrQ-v2, PIE-G, and SpawnNet comprehensively in the generalization task tests with the introduction of only a few tunable parameters. These results suggest that (1) non-linear data augmentation and efficient adapter tuning can handle generalization tasks of RL better. (2) ApmNet achieves a good balance between generalization performance and speed.

Generalization on Color Hard Observations. We then evaluate ApmNet on DMControl-GB with the randomly jittered color setting. As shown in Table 2, ApmNet obtains better or competitive performance in **5** out of **6** instances. These results further demonstrate the superiority of ApmNet.

Generalization on Manipulation Tasks. Furthermore, we conduct experiments on the MetaWorld generalization benchmark to test the ApmNet’s generalization ability in manipulation tasks with different background textures. The visualized observations are shown in Table 3. We use the success rate as the evaluation metric for its goal-conditioned nature. From Table 3, we can find that ApmNet can achieve better comparable generalization performance in all settings.

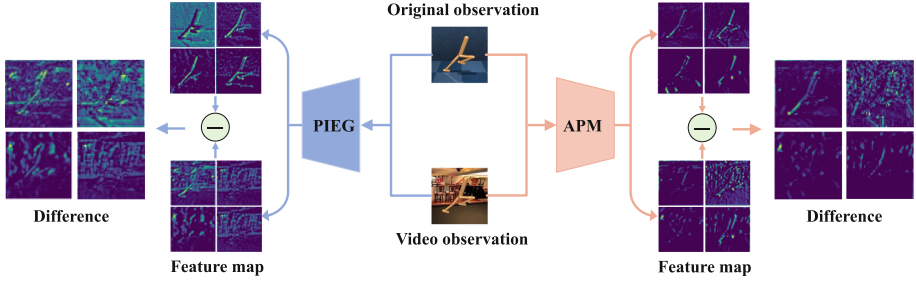


Fig. 3. Visualized feature map. Since both PIE-G and our ApmNet utilize the ImageNet pre-trained ResNet-18 as the backbone, we chose to visualize the feature maps from 4 identical pre-trained convolutional kernels. The difference of the feature maps with ApmNet as the encoder is cleaner and more noticeable than that with PIE-G, indicating that ApmNet is more robust to background noise.

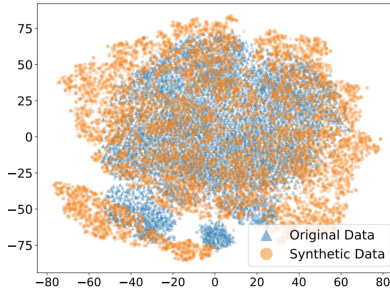


Fig. 4. Visualization of sampled synthetic data and original data. The data is sampled from the ‘Cheetah, Run’ task in DMControl.

Visualization Analysis. Attempting to explain why our method is better, we visualize the difference between the feature maps extracted from our ApmNet and the encoder used in PIE-G. We assume that in visual control generalization tasks, the encoder treats observations of the same state with different backgrounds as similar. Therefore, the difference in their feature maps should be as minimal as possible. As shown in Fig. 3, the encoder of ApmNet produces “cleaner” feature map differences than PIE-G, with more distinct contours and less susceptibility to background interference. Numerically, we calculate the average pixel intensity in the difference of feature maps. The intensity is decreased by 34% with ApmNet than that with PIE-G.

We also visualize the distribution of sampled synthetic data and original data using T-SNE [32]. The distribution visualization is shown in Fig. 4. We find that the data augmented by our MAE is of high fidelity, covering or even broadening the original data distribution, which makes ApmNet perform better in generalization tasks. Further, we also visualize the synthetic data of MAE. A detailed description can be found in the **Appendix**.

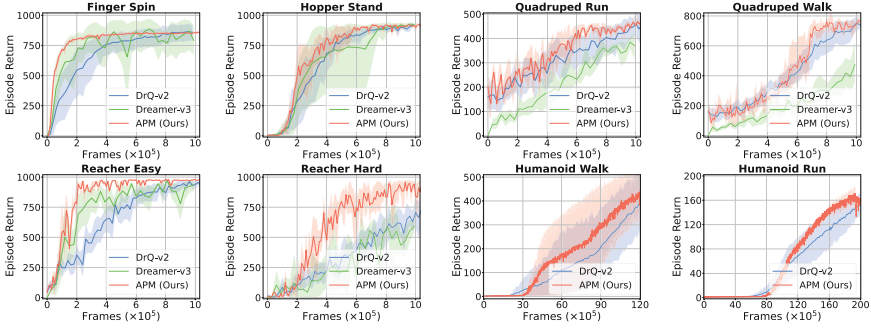


Fig. 5. Sample efficiency (Evaluation). Learning curves on 8 continuous control tasks from DMControl suite as measured on the episode return. The solid line and shaded regions represent the mean and standard deviations calculated over 5 seeds. ApmNet achieves better sample efficiency in all instances.

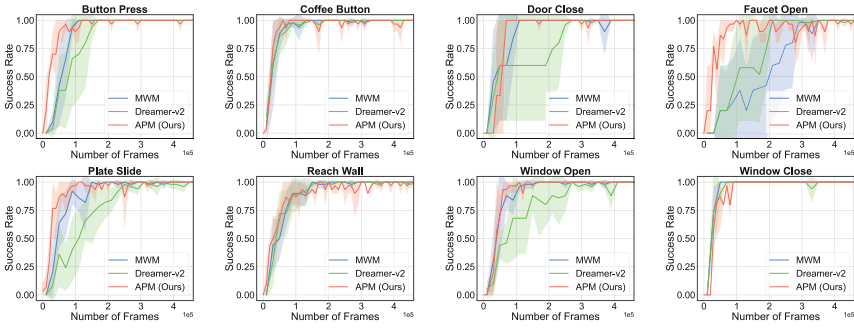


Fig. 6. Sample efficiency (Evaluation). Learning curves on 8 visual robotic manipulation tasks from MetaWorld as measured on the success rate. The solid line and shaded regions represent the mean and standard deviations calculated over 5 seeds. ApmNet achieves better or comparable performance in 7 out of 8 instances.

5.3 Evaluation on Sample Efficiency

We evaluate the sample efficiency of ApmNet on 8 challenging continuous control tasks on the DeepMind Control Suite and 8 manipulation tasks on MetaWorld environment, respectively. We compare the sample efficiency of ApmNet with the state-of-the-art sample-efficiency baselines: DrQ-v2 [38], Dreamer-v2 [12], MWM [25], and Dreamer-v3 [13]. It is worth noting that PIE-G and SpawnNet are algorithms tailored for generalization tasks, and their sample efficiency is not state-of-the-art.

Sample Efficiency on DMControl. Figure 5 demonstrates that ApmNet achieves better sample efficiency and asymptotic performance than DrQ-v2 and Dreamer-v3 in 8 out of 8 tasks. In particular, we also observe that ApmNet gains significant improvement in humanoid control problems, i.e., Humanoid

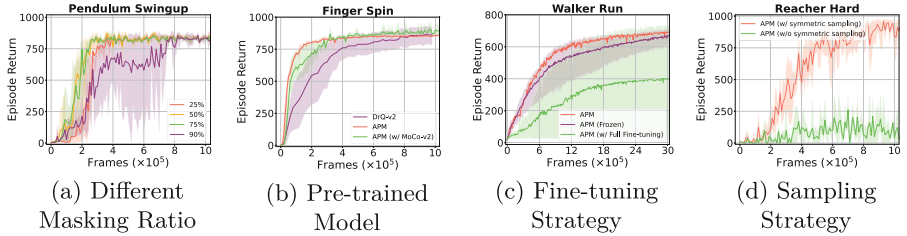


Fig. 7. Ablation study. Learning curves on 4 continuous tasks from DMControl Suite that investigate the effect of (a) Masking Ratio, (b) Pre-trained Model, (c) Fine-tuning Strategy, (d) Sampling Strategy. The solid line and shaded regions represent the mean and standard deviations calculated over 5 seeds.

Walk and Humanoid Run. This indicates our approach’s superiority in complex control problems.

Sample Efficiency on MetaWorld. Figure 6 demonstrates that in many cases, ApmNet, despite being a model-free method, can rival the sample efficiency of state-of-the-art model-based MWM. We note, however, that on some manipulation tasks (for example *Window Close*) MWM and Dreamer-v2 outperform ApmNet. We will investigate this discrepancy in future work.

5.4 Ablation Study

In this section, we conduct several ablation studies to dissect the effect of each component.

Masking Ratio. We report the performance with varying masking ratio $m \in \{0.25, 0.5, 0.75, 0.9\}$ in Fig. 7(a). We find that $m \in \{0.5, 0.75\}$ achieves better performance than $m = 0.25$ because strong regularization can prevent the model from finding a shortcut from input pixels. However, too strong regularization, i.e., $m = 0.9$, degrades the performance.

Pre-trained Model. We also investigate the effectiveness of other visual representations, MoCo-v2 [5]. Figure 7(b) shows that ApmNet with the pre-trained representation of MoCo-v2 can also obtain competitive sample efficiency.

Fine-tuning Strategy. We conduct research to fully fine-tuning and freeze the encoder’s parameters instead of using our adapter tuning. Figure 7(c) suggests that compared with the adapter tuning, the full fine-tuning representations lead to a drop in sample efficiency. The reason is that during the fine-tuning process, the encoder has to adapt to the new data distribution and cannot inherit the useful representations learned from ImageNet. When comparing ApmNet with frozen parameters, we find that by introducing the adapter, ApmNet can not

only inherit the useful representations learned from ImageNet but also extract more valuable information from the specific task, thereby improving the sample efficiency in RL.

Sampling Strategy. Additionally, we evaluate the effect of symmetric sampling. As we see in Fig. 7(d), with 100% MAE augmented data, the agent cannot learn effective strategies. The reason is that the data generated by MAE has a distribution shift compared to the original data, which prevents the agent from observing real samples and thus hinders it from learning effective strategies. After incorporating the symmetric sampling strategy, the agent can not only observe the real environment but also perceive out-of-sample data from the generative model. Our experiments demonstrate that ApmNet enhances both the sample efficiency and generalization capability of visual continuous control.

6 Conclusion and Future Work

In this work, we propose ApmNet, a simple yet effective method that empowers RL agents to benefit from both the readily available visual priors and domain-specific information while minimizing computational costs. We show that ApmNet’s use of MAE and asymmetric learning not only improves performance on training instances but more importantly, offers better generalization ability to unseen environment tasks. ApmNet achieves better performance than prior state-of-the-art methods on 3 challenging benchmarks.

Future Work. ApmNet is a highly flexible framework, and due to space limitations, we have not fully explored its potential. In future work, we will delve into the applications of various pre-trained models within the ApmNet framework. For instance, apart from MAE, it would be intriguing to investigate whether other generative models such as diffusion model [23] and GAN [10], can achieve better results. Furthermore, examining how the quality of images generated by these models affects the performance of control tasks and exploring ways to construct suitable Adapter modules are also exciting research directions.

Acknowledgments. This work was mainly supported by the National Key Research and Development Program of China under Grant 2023YFB4301900. This paper was also supported by the National Natural Science Foundation of China (NSFC) (Grant No. 62272497).

References

1. Ball, P.J., Smith, L., Kostrikov, I., Levine, S.: Efficient online reinforcement learning with offline data. arXiv preprint [arXiv:2302.02948](https://arxiv.org/abs/2302.02948) (2023)
2. Banino, A., Badia, A.P., Walker, J., Scholtes, T., Mitrovic, J., Blundell, C.: Coberl: contrastive bert for reinforcement learning. arXiv preprint [arXiv:2107.05431](https://arxiv.org/abs/2107.05431) (2021)

3. Bellman, R.: A Markovian decision process. *J. Math. Mech.* 679–684 (1957)
4. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.E.: A simple framework for contrastive learning of visual representations. *ArXiv* [arxiv:2002.05709](https://arxiv.org/abs/2002.05709) (2020)
5. Chen, X., Fan, H., Girshick, R., He, K.: Improved baselines with momentum contrastive learning. *arXiv preprint* [arXiv:2003.04297](https://arxiv.org/abs/2003.04297) (2020)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. *arXiv preprint* [arXiv:1810.04805](https://arxiv.org/abs/1810.04805) (2018)
7. Dosovitskiy, A., et al.: An image is worth 16x16 words: transformers for image recognition at scale. *arXiv preprint* [arXiv:2010.11929](https://arxiv.org/abs/2010.11929) (2020)
8. Du, Y., et al.: Guiding pretraining in reinforcement learning with large language models. *arXiv preprint* [arXiv:2302.06692](https://arxiv.org/abs/2302.06692) (2023)
9. Ebert, F., et al.: Bridge data: boosting generalization of robotic skills with cross-domain datasets. *arXiv preprint* [arXiv:2109.13396](https://arxiv.org/abs/2109.13396) (2021)
10. Goodfellow, I.J., et al.: Generative adversarial networks. *CoRR* [arxiv:1406.2661](https://arxiv.org/abs/1406.2661) (2014)
11. Haarnoja, T., Zhou, A., Abbeel, P., Levine, S.: Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor. *ArXiv* [arxiv:1801.01290](https://arxiv.org/abs/1801.01290) (2018)
12. Hafner, D., Lillicrap, T., Norouzi, M., Ba, J.: Mastering atari with discrete world models. *arXiv preprint* [arXiv:2010.02193](https://arxiv.org/abs/2010.02193) (2020)
13. Hafner, D., Pasukonis, J., Ba, J., Lillicrap, T.: Mastering diverse domains through world models. *arXiv preprint* [arXiv:2301.04104](https://arxiv.org/abs/2301.04104) (2023)
14. Hansen, N., Wang, X.: Generalization in reinforcement learning by soft data augmentation. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 13611–13617 (2020)
15. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009 (2022)
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778 (2015)
17. Huang, X., Zhou, H., Yao, K., Han, K.: Froster: frozen clip is a strong teacher for open-vocabulary action recognition. *ArXiv* [arxiv:2402.03241](https://arxiv.org/abs/2402.03241) (2024)
18. Khandelwal, A., Weihs, L., Mottaghi, R., Kembhavi, A.: Simple but effective: clip embeddings for embodied AI. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14829–14838 (2022)
19. Kostrikov, I., Yarats, D., Fergus, R.: Image augmentation is all you need: regularizing deep reinforcement learning from pixels. *ArXiv* [arxiv:2004.13649](https://arxiv.org/abs/2004.13649) (2020)
20. Laskin, M., Lee, K., Stooke, A., Pinto, L., Abbeel, P., Srinivas, A.: Reinforcement learning with augmented data. *ArXiv* [arxiv:2004.14990](https://arxiv.org/abs/2004.14990) (2020)
21. Lin, X., So, J., Mahalingam, S., Liu, F., Abbeel, P.: Spawnnnet: learning generalizable visuomotor skills from pre-trained networks. *ArXiv* [arxiv:2307.03567](https://arxiv.org/abs/2307.03567) (2023)
22. Raileanu, R., Goldstein, M., Yarats, D., Kostrikov, I., Fergus, R.: Automatic data augmentation for generalization in reinforcement learning. In: *Neural Information Processing Systems* (2021)
23. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: *Conference on Computer Vision and Pattern Recognition 2022*, pp. 10674–10685 (2022)

24. Schwarzer, M., Anand, A., Goel, R., Hjelm, R.D., Courville, A.C., Bachman, P.: Data-efficient reinforcement learning with momentum predictive representations. ArXiv [arxiv:2007.05929](https://arxiv.org/abs/2007.05929) (2020)
25. Seo, Y., et al.: Masked world models for visual control. In: Conference on Robot Learning, pp. 1332–1344 (2023)
26. Seo, Y., Lee, K., James, S.L., Abbeel, P.: Reinforcement learning with action-free pre-training from videos. In: International Conference on Machine Learning, pp. 19561–19579 (2022)
27. Shah, R., Kumar, V.: Rrl: resnet as representation for reinforcement learning. ArXiv [arxiv:2107.03380](https://arxiv.org/abs/2107.03380) (2021)
28. Shridhar, M., Manuelli, L., Fox, D.: Cliport: what and where pathways for robotic manipulation. In: Conference on Robot Learning, pp. 894–906. PMLR (2022)
29. Srinivas, A., Laskin, M., Abbeel, P.: Curl: contrastive unsupervised representations for reinforcement learning. In: International Conference on Machine Learning (2020)
30. Stooke, A., Lee, K., Abbeel, P., Laskin, M.: Decoupling representation learning from reinforcement learning. ArXiv [arxiv:2009.08319](https://arxiv.org/abs/2009.08319) (2020)
31. Tassa, Y., et al.: Deepmind control suite. ArXiv [arxiv:1801.00690](https://arxiv.org/abs/1801.00690) (2018)
32. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. *J. Mach. Learn. Res.* **9**(11) (2008)
33. Wang, C., Luo, X., Ross, K.W., Li, D.: Vrl3: a data-driven framework for visual deep reinforcement learning. ArXiv [arxiv:2202.10324](https://arxiv.org/abs/2202.10324) (2022)
34. Wang, X., Lian, L., Yu, S.X.: Unsupervised visual attention and invariance for reinforcement learning. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6673–6683 (2021)
35. Xiao, T., Radosavovic, I., Darrell, T., Malik, J.: Masked visual pre-training for motor control. arXiv preprint [arXiv:2203.06173](https://arxiv.org/abs/2203.06173) (2022)
36. Xu, H., Ding, S., Zhang, X., Xiong, H., Tian, Q.: Masked autoencoders are robust data augmentors. ArXiv [arxiv:2206.04846](https://arxiv.org/abs/2206.04846) (2022)
37. Yang, T., Zhu, Y., Xie, Y., Zhang, A., Chen, C., Li, M.: Aim: adapting image models for efficient video action recognition. ArXiv [arxiv:2302.03024](https://arxiv.org/abs/2302.03024) (2023)
38. Yarats, D., Fergus, R., Lazaric, A., Pinto, L.: Mastering visual continuous control: improved data-augmented reinforcement learning. ArXiv [arxiv:2107.09645](https://arxiv.org/abs/2107.09645) (2021)
39. Yarats, D., Fergus, R., Lazaric, A., Pinto, L.: Reinforcement learning with prototypical representations. In: International Conference on Machine Learning (2021)
40. Yen-Chen, L., Zeng, A., Song, S., Isola, P., Lin, T.Y.: Learning to see before learning to act: Visual pre-training for manipulation. In: 2020 IEEE International Conference on Robotics and Automation (ICRA), pp. 7286–7293. IEEE (2020)
41. Yu, T., et al.: Meta-world: a benchmark and evaluation for multi-task and meta reinforcement learning. ArXiv [arxiv:1910.10897](https://arxiv.org/abs/1910.10897) (2019)
42. Yuan, Z., et al.: Pre-trained image encoder for generalizable visual reinforcement learning. *Adv. Neural. Inf. Process. Syst.* **35**, 13022–13037 (2022)



AdaHAT: Adaptive Hard Attention to the Task in Task-Incremental Learning

Pengxiang Wang¹(✉) , Hongbo Bo^{2,3}, Jun Hong⁴, Weiru Liu³,
and Kedian Mu¹

¹ School of Mathematical Sciences, Peking University, Beijing, China
wangpengxiang@stu.pku.edu.cn, mukedian@math.pku.edu.cn

² Population Health Sciences Institute, Newcastle University, Newcastle, UK
hongbo.bo@newcastle.ac.uk

³ School of Engineering Mathematics and Technology, University of Bristol,
Bristol, UK
weiru.liu@bristol.ac.uk

⁴ School of Computing and Creative Technologies, University of the West
of England, Bristol, UK
Jun.Hong@uwe.ac.uk

Abstract. Catastrophic forgetting is a major issue in task-incremental learning, where a neural network loses what it has learned in previous tasks after being trained on new tasks. A number of architecture-based approaches have been proposed to address this issue. However, the architecture-based approaches suffer from another issue on network capacity when the network learns long sequences of tasks. As the network is trained on an increasing number of new tasks in a long sequence of tasks, more parameters become static to prevent the network from forgetting what it has learned in previous tasks. In this paper, we propose an adaptive task-based hard attention mechanism which allows adaptive updates to static parameters by taking into account the information about previous tasks on both the importance of these parameters to previous tasks and the current network capacity. We develop a new neural network architecture incorporating our proposed Adaptive Hard Attention to the Task (AdaHAT) mechanism. AdaHAT extends an existing architecture-based approach, Hard Attention to the Task (HAT), to learn long sequences of tasks in an incremental manner. We conduct experiments on a number of datasets and compare AdaHAT with a number of baselines, including HAT. Our experimental results show that AdaHAT achieves better average performance over tasks than these baselines, especially on long sequences of tasks, demonstrating the benefits from balancing the trade-off between stability and plasticity of a network when learning such sequences of tasks. Our code is available at github.com/pengxiang-wang/continual-learning-arena.

Keywords: Continual learning · Task-incremental learning · Catastrophic forgetting

1 Introduction

One of the key features of human intelligence is the ability to learn continually and adapt over time. One of the major challenges for deep neural networks in continual learning is catastrophic forgetting [17,19], which results in drastic performance drops on previous tasks after the network is trained on new tasks. Continual learning addresses this issue by aiming to learn and accumulate knowledge over a sequence of tasks without catastrophic forgetting [4].

To overcome catastrophic forgetting in task-incremental learning, a scenario of continual learning in which an algorithm learns a sequence of distinct tasks in an incremental manner [4], various strategies have been proposed. Most of these strategies adopt the idea of exploring certain forms of information about previous tasks and taking into account the information when learning new tasks to prevent the network from forgetting the knowledge learned in previous tasks. For example, some replay-based approaches prevent forgetting by storing parts of the data from previous tasks, which replay algorithms use to consolidate previous knowledge [3,14,20]; some regularization-based approaches typically add regularization terms constructed using information about previous tasks to the loss function when training new tasks [11–13,18,27].

Several architecture-based approaches have been proposed to explore the inherent nature of parameter separability within the architecture of neural networks, focusing on reducing representational overlap between tasks in the network. The key idea in these approaches is to dedicate network parameters in different parts of the network to different tasks and to keep the parameters learned in previous tasks from being significantly changed when learning new tasks [2,15,16,22,26]. Therefore, they are also referred to as parameter isolation methods [4]. An important consideration in these approaches is the trade-off between stability and plasticity of the network, where stability emphasises performance over tasks while plasticity emphasizes saving network capacity by keeping parts of the network active for new tasks [25].

A recent architecture-based approach to reduce representational overlap is the task-based hard attention mechanism called Hard Attention to the Task (HAT) [22]. HAT learns layer-wise attention vectors (masks) for each task, concurrently to learning network parameters, to prevent updates to parameters important to previous tasks. When learning new tasks, HAT conditions updates to these parameters with all the masks learned in previous tasks, thus preventing forgetting what has been learned in these tasks. Specifically, HAT adjusts gradients during the training process to control the updates to the parameters directly. However, as the number of tasks increases, the network capacity is rapidly saturated, which greatly reduces the proportion of remaining active parameters for new tasks. This highlights the critical issue of balancing the trade-off between maintaining important parameters to previous tasks and preserving learning plasticity in the network for new tasks, i.e., the trade-off between stability and plasticity [25]. HAT emphasises stability over plasticity, unlike some other continual learning approaches which focus more on plasticity.

In this paper, we propose an extension to the task-based hard attention mechanism proposed in HAT to prevent the neural network from forgetting. Following the spirit of HAT, we propose a new mechanism, putting adaptive attention to the task, which allows adaptive updates to those static network parameters that have been attended to previous tasks when learning new tasks, taking into account the information about previous tasks on both the importance of these parameters to previous tasks and the current network capacity. These adaptive parameter updates help to reuse, in a measured way, parts of the network that have been made static for previous tasks. We call our proposed mechanism Adaptive Hard Attention to the Task (AdaHAT). AdaHAT has three distinctive characteristics: 1. It balances the trade-off between stability and plasticity of the network, i.e., between maintaining important parameters to previous tasks and preserving network capacity for new tasks; 2. It particularly suits for learning long sequences of tasks; 3. It is an adaptive process to update previously static parameters, based on both their effects on previous tasks and network capacity usage. For instance, when the network capacity is insufficient, larger updates are allowed to parameters that have been attended to fewer previous tasks, hence preserving more network capacity by compromising only a small amount of forgetting on previous tasks.

We implement AdaHAT, conduct experiments on a number of datasets and compare it with a number of baselines, particularly including HAT which AdaHAT extends. Our experimental results show that AdaHAT achieves better average performance over tasks than these baselines, especially on long sequences of tasks, demonstrating the benefits from balancing the stability-plasticity trade-off of the network when learning such sequences of tasks. The code for our implementation and experiments is available¹.

This paper makes the following contributions:

1. We propose a novel mechanism called Adaptive Hard Attention to the Task (AdaHAT), which allows adaptive updates to previously static parameters in a neural network, alleviating the problem of insufficient network capacity when learning long sequences of tasks;
2. We implement a neural network architecture based on the architecture developed in HAT [22], integrating our proposed task-based attention mechanism for adaptive parameter updating in the network while retaining the spirit of hard attention to the task proposed in HAT;
3. Our experimental results on a number of datasets show that while slightly causing forgetting, AdaHAT improves the average performance of the network after it has reached the network capacity limit, which effectively balances the trade-off between stability and plasticity in continual learning.

2 Related Work

Task-incremental learning is a continual learning scenario where the algorithm learns a sequence of distinct tasks in an incremental manner [24]. In this scenario,

¹ <https://github.com/pengxiang-wang/continual-learning-arena>.

it is crucial to balance the trade-off between stability and plasticity to ensure the optimal model performance over all tasks.

Catastrophic forgetting is a major issue in task-incremental learning, where a neural network loses what it has learned in previous tasks after being trained on new tasks. Major efforts have been focused on developing various mechanisms to prevent the network from catastrophic forgetting, often leveraging certain forms of information about previous tasks. In replay-based approaches, parts of the data from previous tasks are stored and replayed during the training on new tasks to prevent forgetting. Examples include iCaRL [20], GEM [14], and DER [3]. In regularization-based approaches, catastrophic forgetting is alleviated by incorporating regularization terms into the loss function, usually constructed from the information about previous tasks. Examples include LwF [13], EWC [11], SI [27], IMM [12], and VCL [18]. These approaches have an emphasis on stability in their forgetting prevention mechanisms, but generally still lean towards plasticity in the trade-off.

Architecture-based approaches adopt distinctly different strategies that overly prioritize stability, tilting the trade-off towards stability instead. They dedicate different parts of a neural network to different tasks, i.e., specifying task-specific parameters, which leverages the separability characteristic of the neural network architecture. One strategy develops incrementally parallel sub-networks to learn the sequence of tasks. A classical example is Progressive Neural Networks [21]. Other strategies allocate the parameter space within a fixed (sometimes dynamically expanded when needed) network architecture, with the allocation determined either by a set of rules, such as PackNet [16] and UCL [1], or by learnable masks over the architecture trained along with network parameters, such as Piggyback [15], HAT [22], CPG [8], and SupSup [26].

Architecture-based approaches generally suffer from the network capacity issue because they preserve and fix task-specific parameters for each previous task. In this case, the network can achieve the maximum stability and the best performance when there is sufficient capacity, but has to face drastic performance drops on new tasks when capacity runs out. In other words, they sacrifice learning plasticity for potential future tasks to maintain the stability for previous tasks [25].

To alleviate the network capacity issue, many architecture-based approaches introduce measures to carefully control their network capacity usage, with mechanisms such as sparsity regularization [22], which in turn limits their ability to learn previous tasks [25]. Others gain additional capacity resources by breaking the assumption of the fixed parameter space: some allow dynamically expanding the network when capacity runs out as more new tasks arrive. Some approaches like Progressive Neural Networks [21] even expand the network every time when a new task arrives, causing a progressively linear increase of the parameter space. In this case, the learning plasticity for new tasks leads to a significant storage problem with the ever-expanding network.

Additionally, most architecture-based approaches use many hyperparameters to control their network capacity usage, usually without directly leveraging

any information about previous tasks. These hyperparameters need to be tuned manually to determine how much capacity should be allocated to new tasks. For instance, PackNet uses a pruning ratio to explicitly allocate a certain percentage of network parameters to new tasks [16]. HAT requires manually setting the hyperparameter s_{\max} , where larger values provide stability for previous tasks, and smaller values enhance plasticity for new tasks [22]. However, in real-world continual learning scenarios, one usually does not know how many new tasks will arrive, or may even encounter an infinite sequence of tasks [4], making manual hyperparameter tuning infeasible. Even if the number of tasks is known, manual tuning requires repetitive retraining, and as the task sequence becomes longer, it may become necessary to make further adjustments to the hyperparameters, which undermines the key continuity characteristics of continual learning.

Network capacity is a critical factor affecting learning plasticity when incrementally learning long sequences of tasks [4]. To address this issue, we need to balance the stability-plasticity trade-off as well as prevent catastrophic forgetting at the same time. In this paper, we propose a novel adaptive task-based attention mechanism called Adaptive Hard Attention to the Task (AdaHAT) to balance the trade-off in architecture-based approaches, enabling the adaptive allocation of network capacity with taking into account the information about previous tasks. Our proposed mechanism, in particular, addresses the problem of lacking plasticity when incrementally learning long sequences of tasks.

3 Task-Incremental Learning with Adaptive Hard Attention to the Task

In this section, we present a new approach for task-incremental learning – our proposed adaptive attention mechanism called Adaptive Hard Attention to the Task (AdaHAT) to balance the trade-off between stability and plasticity. Our proposed mechanism extends the mechanism called Hard Attention to the Task (HAT) proposed in [22].

In the task-incremental learning scenario, a sequence of tasks $t = 1, \dots, N$, arrive at a neural network in an incremental manner, with each task associated with dataset $D^t = \{x^t, y^t\}$. The objective of task-incremental learning for the network is to learn the sequence of tasks, preventing performance drops on previous tasks when learning new tasks, and ultimately getting better performance over all tasks [4]. To achieve this objective, the stability-plasticity trade-off needs to be balanced properly.

We adopt the hard attention to the task mechanism proposed in HAT, in which layer-wise attention vectors \mathbf{m}_l^t with binary values are learned to pay hard attention on units in each layer $l = 1, \dots, L - 1$ to a new task t . The attention vectors are gated from layer-wise task embeddings \mathbf{e}_l^t with real values:

$$\mathbf{m}_l^t = \sigma (s\mathbf{e}_l^t) \quad (1)$$

where s is a positive scaling parameter and σ is the sigmoid gate function. The attention vectors are learned as the task embeddings are trained along with

network parameters. The binary attention vectors determine which part of the network is dedicated to the task. We use \mathbf{M}^t to denote all the attention vectors to the task t , Θ to denote the parameter space, therefore Θ^t , parameters in Θ dedicated to task t , are those masked by \mathbf{M}^t .

When training on a new task t , HAT conditions gradients in the backward pass according to cumulative attention vectors $\mathbf{m}_l^{\leq t}$ from all previous tasks. The cumulative attention vectors are recursively calculated by

$$\mathbf{m}_l^{\leq t} = \max\left(\mathbf{m}_l^t, \mathbf{m}_l^{\leq t-1}\right) \quad (2)$$

after learning task t , using element-wise maximum². This preserves the attention values for units in the network that are important to previous tasks, and allows these preserved values to condition network training on the new task.

To condition training on the new task t , HAT modifies the gradient $g_{l,ij}$ of the parameter $\theta_{l,ij}$ connecting the j -th unit in layer $l-1$ to the i -th unit in layer l , with a parameter-wise adjustment rate $a_{l,ij}$:

$$g'_{l,ij} = a_{l,ij} \cdot g_{l,ij}, \quad a_{l,ij} \in \{0, 1\} \quad (3)$$

where $g_{l,ij}$ is the gradient of parameter $\theta_{l,ij}$, and

$$a_{l,ij} = 1 - \min\left(m_{l,i}^{\leq t}, m_{l-1,j}^{\leq t}\right) \quad (4)$$

marks the hard clipping of the gradient, calculated as the reverse of the minimum of the two cumulative attention values, which results in binary values again. This means, those parameters become static without updates when the units connected at both ends are masked by the cumulative attention vectors, as their gradients are hard clipped.

As the task sequence becomes longer, more active parameters become static, gradually taking up more network capacity, hence reducing learning plasticity for new tasks. To address this issue, a regularization term controlling network sparsity is employed in HAT to promote low network capacity usage and high compactness of the masks:

$$\mathcal{L}'(f(x_t), y_t, \mathbf{M}^t, \mathbf{M}^{\leq t}) = \mathcal{L}(f(x_t), y_t) + cR(\mathbf{M}^t, \mathbf{M}^{\leq t}) \quad (5)$$

$$R(\mathbf{M}^t, \mathbf{M}^{\leq t}) = \frac{\sum_{l=1}^{L-1} \sum_{i=1}^{N_l} m_{l,i}^t (1 - m_{l,i}^{\leq t})}{\sum_{l=1}^{L-1} \sum_{i=1}^{N_l} (1 - m_{l,i}^{\leq t})} \quad (6)$$

where c denotes the regularization constant (a hyperparameter in HAT) and N_l denotes the number of units in layer l . This, to a certain extent, helps alleviate the issue on network capacity. However, the network capacity will eventually run out, as parameters will become static forever once they are learned to be dedicated to previous tasks. As shown in an experiment illustrated in Fig. 1,

² $\mathbf{m}_l^{\leq 0}$ starts with all zeros to calculate $\mathbf{m}_l^{\leq 1}$.

the cumulative attention vector of certain layer is rapidly occupied by value 1 as the network trains on new tasks, which means the network rapidly exhausts all its available parameter space, resulting in insufficient network capacity for the algorithm to be allocated for new tasks, hence significantly affecting the performance of the trained network on new tasks.

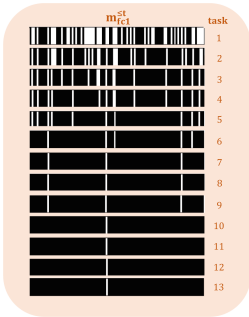


Fig. 1. The evolution of cumulative attention vector to previous tasks for layer "fc1" in a MLP architecture, represented by the cumulative attention vector $\mathbf{m}_{fc1}^{\leq t}$, with network parameters attended highlighted in black.

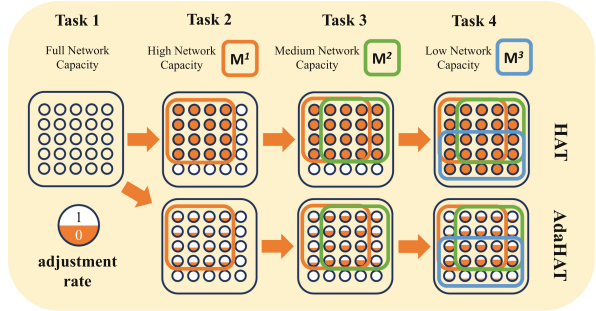


Fig. 2. Comparison between HAT and AdaHAT on the evolution of adjustment rates on gradients of network parameters (represented as circles). The hard clipping in HAT distinctly binarizes parameters into static (orange) and active (white) states, while the soft clipping in AdaHAT adaptively manages the adjustment rate between 0 and 1 based on both parameter importance (the number of bordered rounded rectangle) and network capacity. (Color figure online)

3.1 The Algorithm: Adaptive Updates to the Parameters in the Network with Summative Attention to Previous Tasks

Following the spirit of the hard attention to the task mechanism proposed in HAT [22], in order to address the network capacity issue in HAT on learning long sequences of tasks, we propose a new adaptive parameter updating approach extended from HAT called Adaptive Hard Attention to the Task (AdaHAT) based on the HAT architecture. We propose to replace the cumulative attention vector in HAT with our new summative attention vector, calculated as follows:

$$\mathbf{m}_i^{\leq t, \text{sum}} = \mathbf{m}_i^t + \mathbf{m}_i^{\leq t-1, \text{sum}} \quad (7)$$

The network capacity issue in HAT can be alleviated by allowing adaptive updates to parameters in the network that have become static for previous tasks. As we can see, the hard gradient clipping from Eq. (3) is the underlying cause of the network capacity issue in HAT because the parameters dedicated to previous tasks remain completely and permanently static when learning any new tasks.

In AdaHAT, we propose to softly clip the gradients as follows to release some network capacity from previously static parameters:

$$g'_{l,ij} = a_{l,ij}^* \cdot g_{l,ij}, \quad a_{l,ij}^* \in [0, 1] \quad (8)$$

where $a_{l,ij}^*$ represents the adjustment rate of AdaHAT, which ranges between 0 and 1, softly clipping the gradient $g_{l,ij}$. This allows updates to the parameter $\theta_{l,ij}$ that is adaptive to both the parameter importance to previous tasks and the current network capacity, as defined below. In the following parts, we first define two measures for parameter importance and network sparsity respectively and then integrate them into the adjustment rate.

Two pieces of information about previous tasks are considered to be crucial for playing the role in adjusting the gradient and determining how much parameter space for previous tasks can be released:

1. Parameter Importance

In HAT, cumulative attention vectors defined in Eq. (2) are used to preserve those important parameters to previous tasks. Higher cumulative attention values in these vectors indicate greater importance of the corresponding parameters, hence smaller updates are made to them in the backward pass when learning new tasks. However, this binary measure of importance is often indistinguishable. We propose to replace the cumulative attention vectors with our new measure of parameter importance in AdaHAT, called the Summative Attention Vectors, as defined in Eq. (7) above. For each parameter $\theta_{l,ij}$, we still calculate the minimum of the summative attention values of the connected units at both ends, $\min(m_{l,i}^{<t,\text{sum}}, m_{l-1,j}^{<t,\text{sum}})$, which represents the importance of the parameter to previous tasks. This value ranges from 0 up to $t - 1$, therefore encapsulates the information about previous tasks.

Thus, a higher summative attention value indicates that the parameter is dedicated to more previous tasks, hence smaller updates should be made, as it is important in terms of its contribution to previous tasks. For those parameters with lower importance, their space can be released more. Therefore, the adjustment rate should negatively correlate with this parameter importance value.

2. Network Sparsity

The sparsity regularization term in HAT reflects the network sparsity by measuring the compactness of hard attention values. Larger regularization values indicate that when learning new tasks, the algorithm is paying more attention to the active parameter space that is not dedicated to previous tasks, hence there is less need to make the adaptive updates to the parameters that have become static for previous tasks.

Furthermore, this regularization value is closely related to the current network capacity. Generally, when a smaller proportion of parameters in the network are static (i.e., sufficient network capacity), the regularization value tends to be larger, as there is a great possibility for the hard attention to be paid to active parameters. In such cases, since the network capacity is

sufficient, we want the algorithm not to hastily release the parameter space just learned and allocated for previous tasks, but to focus on learning those active parameters instead. Therefore, the adjustment rate should positively correlate with the network sparsity regularization value.

Considering this regularization term, AdaHAT tries its best to mimic HAT before the network reaches its capacity limit, then starts to make larger updates to the parameters for previous tasks afterwards. By retaining maximum stability before affecting the learning plasticity for new tasks, we follow the spirit of HAT in terms of high stability and leverage this benefit from HAT without losing our proposed adaptive features.

We now define our adjustment rate for AdaHAT, incorporating both parameter importance and network sparsity:

$$a_{l,ij}^* = \frac{r_l}{\min(m_{l,i}^{<t,\text{sum}}, m_{l-1,j}^{<t,\text{sum}}) + r_l}, \quad r_l = \frac{\alpha}{R(M^t, M^{<t}) + \epsilon} \quad (9)$$

where α is a hyperparameter that control the overall intensity of gradient adjustment. The constant value ϵ is set to 0.1 to help avoid division by zero. Note that for those active parameters, the adjustment rate is 1 as the parameter importance value is 0, so the soft clipping applies only to the gradient of parameters for previous tasks. As we can tell from Eq. (9), the adjustment rate correlates with the parameter importance negatively and network sparsity positively, where the two pieces of information collaboratively control the gradient adjustment thus manage the network capacity.

We present an illustration on how the adjustment rates evolve as new tasks arrive in Fig. 2 and the AdaHAT algorithm in Algorithm 1 summarizing the implementation of our proposed approach as described in this section.

4 Experiments

In this section, we first describe our experiment settings and metrics for performance, stability-plasticity trade-off and network capacity. We then present our experimental results on the performance of AdaHAT on learning long sequences of tasks and how AdaHAT achieves them by balancing stability-plasticity trade-off. We finally discuss the results of ablation study and hyperparameter selection. Our findings demonstrate that the proposed adaptive parameter updating approach, which takes into account both the parameter importance and the current network capacity, is effective. It can make adaptive gradient adjustments when learning new tasks, within a very small adjustment rate.

4.1 Setups

Task Sequences. AdaHAT aims to address the issue of insufficient network capacity in learning long sequences of tasks. Our experiments are conducted on long sequences of tasks comprising 20 tasks, which is longer than the sequences

Algorithm 1: Adaptive Hard Attention to the Task (AdaHAT)

Input: task sequence $D^t, t = 1, 2, 3, \dots$; learning rate η ; hyperparameters c, s_{\max} (from HAT) and α .

Output: network parameters Θ shared by all tasks; hard attention vectors (binary mask) M^t for each task $t = 1, 2, 3, \dots$

Initialize: task embeddings \mathbf{e}_t^l from $\mathcal{N}(0, 1)$; summative and cumulative attention vectors $\mathbf{m}_l^{\leq 0, \text{sum}}, \mathbf{m}_l^{\leq 0}$ both from all zeros.

for task $t = 1, 2, 3, \dots$ **do**

for epoch **do**

 Forward propagate through Θ^t under mask \mathbf{m}_l^t in Eq. (1);

 Calculate current network sparsity R in Eq. (6);

 Calculate loss \mathcal{L}^t in Eq. (5) and backpropagate to gradients $g_{l,ij}$ of network parameters $\theta_{l,ij}$ and gradients of task embeddings \mathbf{e}_t^l ;

 Calculate adjustment rate $a_{l,ij}^*$ in Eq. (9);

 Soft clip the gradients to $g'_{l,ij}$ in Eq. (8) with the adjustment rate;

 Update $\theta_{l,ij}$ using adjusted gradients $g'_{l,ij}$ and update \mathbf{e}_t^l ;

 Calculate the summative and cumulative attention vectors $\mathbf{m}_l^{\leq t, \text{sum}}, \mathbf{m}_l^{\leq t}$ of task t in Eq. (7) and Eq. (2).

commonly used in conventional continual learning experiment setups, where their task sequences usually comprise fewer than 10 tasks. This long task sequence setup allows us to better observe how architecture-based approaches behave once the network capacity has reached its limit and how it will evolve afterwards. We also experiment on much longer sequences of tasks comprising up to 50 tasks.

Data. We use the permuted version of MNIST [23] and the split version of CIFAR-100 [9] as our experiment datasets. We choose not to use permuted versions of CIFAR-10 and CIFAR-100 because they are overly challenging with long sequences of tasks for most baselines, leading their performance to a rather low level, which makes meaningful comparisons difficult. Besides, the split versions of MNIST and CIFAR-10 are unsuitable for our experiments as well, because their 10 classes cannot be divided into long sequences of tasks, for example, 20 tasks.

Baselines. To evaluate the performance of AdaHAT, we compare it with the following baselines:

- **Finetuning:** standard gradient descent [6];
- **LwF:** Learning without Forgetting [13];
- **EWC:** Elastic Weight Consolidation [11];
- **HAT:** Hard Attention to the Task [22].

We also compare the effectiveness of our proposed adaptive task-based hard attention mechanism with other gradient adjustment strategies. These strategies all adjust gradients in a naive or unmeaningful way, randomly or uniformly, without guidance by the information about previous tasks:

- **HAT-random:** HAT with a random value adjustment rate, where $a_{l,ij}$ in Eq. (3) is replaced by $\begin{cases} \text{rand}(0, 1), & \text{if } 1 - \min(m_{l,i}^{<t}, m_{l-1,j}^{<t}) = 0, \\ 1, & \text{otherwise.} \end{cases}$
- **HAT-const-alpha:** HAT with a fixed adjustment rate of constant value α (equals to α in Eq. (9)), where $a_{l,ij} = \begin{cases} \alpha, & \text{if } 1 - \min(m_{l,i}^{<t}, m_{l-1,j}^{<t}) = 0, \\ 1, & \text{otherwise.} \end{cases}$
- **HAT-const-1:** HAT with a fixed adjustment rate of constant value 1, which means there is no constraint on the backward pass imposed by any attention values to previous tasks.

Evaluation Metrics. We evaluate AdaHAT and the baselines using 4 metrics: performance, stability, plasticity and network capacity. We calculate Average Accuracy (AA) [25] and Forgetting Ratio (FR) [22] on each dataset over all tasks, which are the main performance metrics in task-incremental learning. Additionally, we evaluate the stability-plasticity trade-off to show which side each approach leans towards, or balanced, potentially. We evaluate it using backward and forward transfer metrics (BWT and FWT) [25].

Let $a_{t,N}$ denote the accuracy of the test model on dataset D^t after learning task N , $a_{t,N}^J$ denote the accuracy on dataset D^t of a randomly-initialized reference model jointly trained on $\cup_{t=1}^N D^t$, a_t^I and a_t^R denote the accuracy of a randomly-initialized reference model independently trained on D^t and a random stratified model, respectively. The metrics are defined as follows [25]:

$$AA_N = \frac{1}{N} \sum_{t=1}^N a_{t,N} \quad (10)$$

$$FR_N = \frac{1}{N} \sum_{t=1}^N \frac{a_{t,N} - a_t^R}{a_{t,N}^J - a_t^R} - 1 \quad (11)$$

$$BWT_N = \frac{1}{N-1} \sum_{t=1}^{N-1} (a_{t,N} - a_{t,t}) \quad (12)$$

$$FWT_N = \frac{1}{N-1} \sum_{t=2}^N (a_{t,t} - a_t^I) \quad (13)$$

Each experiment is repeated 5 times, and we report the mean and standard deviation for each metric.

To compare the network capacity usage of the models, we measure the network capacity using the average adjustment rate over all network parameters²:

$$NC = \frac{1}{\sum_l N_l} \sum_{l,i,j} a_{l,ij} \quad (14)$$

² AdaHAT denotes the adjustment rate as $a_{l,ij}^*$.

Note that the defined Network Capacity (NC) ranges from 0 to 1, with 0 indicating that all parameters can be adjusted freely (with no adjustment to their gradients), and 1 indicating that no parameter can be updated.

Networks. For experiments on Permuted MNIST, we use a fully-connected (MLP) network architecture including 3 hidden layers with dimensions 256, 100, and 64 as the feature extractor. For the experiments on Split CIFAR-100, we use ResNet-18 [7] as the feature extractor. Task embeddings are integrated into each layer to generate task-based attention vectors [22]. Rectified linear unit is used as the activation function. As this is a task-incremental learning scenario, each task has its own output head which is a single linear output layer, and the number of heads increases as new tasks arrive. All layers are randomly initialized with Xavier initialization [5], except for task embeddings \mathbf{e}_l^t , which are initialized with a standard normal distribution $\mathcal{N}(0, 1)$.

Training. All models are trained using the Adam optimizer [10] with a learning rate of 0.001. Since the experiments are intended to study the network after the network capacity has reached its limit, the hyperparameter s_{\max} in Eq. (1) is set to a large value of 400. The hyperparameter α in Eq. (9) is set to 10^{-6} . We train for 2 and 20 epochs for each task in Permuted MNIST and Split CIFAR-100, validating at each epoch to select the best model for testing. We randomly split 10% of the training data for each task as validation data. Batch sizes are set to 128 and 64. Permutation seeds are set to 1 through 20. All other random variables are set by a single seed from 1 for each experiment run.

We apply gradient clipping to network parameters with a specified value of 0.001 and weight decay with a factor of 0.00035 for all approaches except those using hard attention mechanism. Note that weight decay and gradient clipping are incompatible with the HAT mechanism, as additional gradient adjustments in these approaches can update those parameters meant to be static, potentially causing forgetting.

4.2 Results

We present our experimental results on sequences of 20 tasks in Table 1. AdaHAT demonstrates the best performance in terms of average accuracy and forgetting ratio, outperforming all baseline approaches, which shows the superiority of AdaHAT in incrementally learning long sequences of tasks.

We observe that Finetuning and HAT exhibit extreme BWT and FWT values. Finetuning leans towards plasticity with low BWT and high FWT, while HAT leans towards stability with high BWT and low FWT, both showing an imbalance on the stability-plasticity trade-off. AdaHAT, on the other hand, maintains relatively balanced BWT and FWT values, neither excessively high nor low, demonstrating a better trade-off between stability and plasticity. Notably, on the Split CIFAR-100 dataset, AdaHAT exhibits even higher stability and plasticity both. We can infer from these various approaches that achieving higher performance often hinges on effectively balancing BWT and FWT, under-

Table 1. Results on performance and stability-plasticity trade-off metrics (mean \pm std) of different approaches on the two datasets (20 tasks).

Dataset	Approach	AA(%)	FR (%)	BWT(%)	FWT (%)
Permuted MNIST	Finetuning	32.62 \pm 1.60	-73.78 \pm 1.84	-68.10 \pm 1.68	63.51 \pm 0.03
	LwF	26.95 \pm 1.80	-80.35 \pm 2.08	-72.59 \pm 1.91	62.04 \pm 0.09
	EWC	52.25 \pm 2.46	-51.38 \pm 2.83	-42.04 \pm 2.67	58.12 \pm 0.15
	HAT	67.64 \pm 1.27	-33.70 \pm 1.46	-0.11 \pm 0.18	32.49 \pm 1.12
	HAT-random	66.43 \pm 1.21	-35.10 \pm 1.39	-0.27 \pm 0.49	31.40 \pm 1.22
	HAT-const-alpha	68.08 \pm 1.18	-33.20 \pm 1.36	-1 * e ⁻³ \pm 0.00	32.92 \pm 1.23
	HAT-const-1	48.83 \pm 4.35	-55.14 \pm 5.02	-49.68 \pm 4.40	62.26 \pm 0.21
	AdaHAT	79.90 \pm 2.40	-19.43 \pm 2.76	-14.68 \pm 2.48	59.96 \pm 0.09
Split CIFAR-100	Finetuning	24.34 \pm 0.73	-91.66 \pm 1.32	-54.00 \pm 1.00	53.10 \pm 0.55
	LwF	34.56 \pm 0.94	-70.91 \pm 2.05	-48.03 \pm 1.01	57.61 \pm 0.40
	EWC	30.23 \pm 1.61	-79.84 \pm 3.13	-54.05 \pm 1.28	59.20 \pm 0.50
	HAT	32.44 \pm 1.58	-74.71 \pm 3.37	-45.59 \pm 1.49	53.11 \pm 0.34
	HAT-random	31.41 \pm 1.29	-76.98 \pm 2.45	-48.80 \pm 1.33	52.76 \pm 0.57
	HAT-const-alpha	32.16 \pm 2.48	-75.04 \pm 5.16	-44.49 \pm 2.57	51.86 \pm 0.82
	HAT-const-1	32.40 \pm 1.40	-75.58 \pm 3.08	-48.80 \pm 1.72	56.30 \pm 0.36
	AdaHAT	38.74 \pm 2.24	-62.37 \pm 4.64	-42.11 \pm 2.02	56.33 \pm 0.82

scoring the importance of maintaining a balanced stability-plasticity trade-off for optimal performance.

For baselines using HAT mechanism with other gradient adjustment strategies, we observe that almost none of them outperform HAT itself, even in the case with gradient adjustments allowed. This underscores the necessity for adjustments to be guided properly; without them, performance may even decline due to the gradient adjustment. For instance, in HAT-const-1, the allowance for full adjustment unbalances the trade-off heavily towards plasticity (low BWT and high FWT), which likely contributes to its poor performance.

Longer Task Sequences. To evaluate the performance of AdaHAT on longer task sequences, we conduct experiments on Permuted MNIST comprising 50 tasks. The evolution of performance of AdaHAT and baseline approaches as each new task arrives is shown in Fig. 3. HAT demonstrates slightly better performance before 8 tasks as it maximizes stability by making parameters for previous tasks completely static. However, the performance curve of HAT reaches a turning point at 8 tasks and then drastically drops, indicating that HAT exhausts network capacity at this point, exactly corresponding to Fig. 1 above, where HAT has almost run out of active parameter space. AdaHAT, on the other hand, maintains significant superiority over other baselines after this turning point, proving its capability and advantage in long task sequence settings.

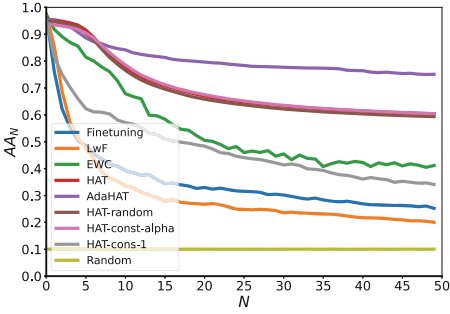


Fig. 3. Average Accuracy performance of different approaches on longer sequences of tasks (50 tasks, Permuted MNIST).

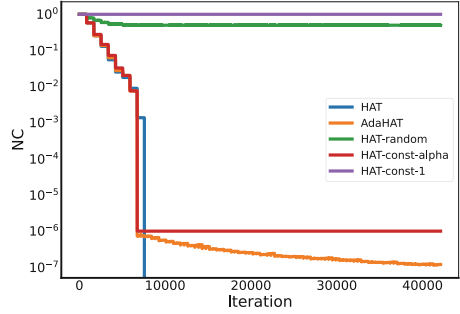


Fig. 4. Network Capacity usage in different gradient adjustment strategies (50 tasks, Permuted MNIST).

Note that we still find AdaHAT behaves close to HAT before 8 tasks, showing the effect of network sparsity information in the adjustment rate, which leverages the stability from HAT before affecting plasticity.

Network Capacity. We plot the evolution of Network Capacity (NC) usage against training iterations, in a specific experiment run (seed=1) on Permuted MNIST dataset comprising 50 tasks in Fig. 4. Over long-term training, the network capacity in HAT rapidly runs out. In HAT-random, HAT-const-alpha, and HAT-const-1, the capacity eventually stabilizes at values 0.5 (the mean value of $\text{rand}(0, 1)$), 10^{-6} (we set α to this value), and 1, respectively. AdaHAT behaves very similarly to HAT at first, but while HAT exhausts network capacity, AdaHAT manages it adaptively over time through an adaptive adjustment rate, making it converge to 0 but never reach 0. This is the key for AdaHAT to rebalance the trade-off when stability starts to affect plasticity, thus addressing the network capacity issue in HAT.

4.3 Ablation Study

To provide insights into the individual effects of the two pieces of information incorporated into the adjustment rate of AdaHAT, we design ablation experiments to analyze how each of them helps improve HAT individually by fixing the other information as a constant value:

- **AdaHAT-no-sum:** To study the effect of parameter importance, we fix all the summative attention $\min(m_{l,i}^{<t,\text{sum}}, m_{l-1,j}^{<t,\text{sum}})$ at a constant value t , so the adjustment rate solely depends on the sparsity regularization term. Note that we use t instead of 1 because we want to keep the same increasing magnitude as the summative attention. In other words, AdaHAT-no-sum always treats all previous tasks with the same high level of importance.

- **AdaHAT-no-reg:** To study the effect of network sparsity, we fix the regularization term $R(\mathbf{M}^t, \mathbf{M}^{<t})$ at a constant value 0, so the adjustment rate solely depends on the summative attention vectors. In other words, AdaHAT-no-reg always treats the network as if it is under low network sparsity.

We conduct the ablation experiments with the same settings as the previous experiments on longer task sequences. Results are shown in Fig. 5. Both ablated approaches fail to outperform AdaHAT, underscoring that the guidance from both pieces of information for the adjustment rate is crucial. However, they both surpass HAT in long sequences of tasks, underscoring that the adaptive adjustment mechanism itself is the key to addressing the network capacity issue in HAT.

We observe that AdaHAT-no-reg consistently performs worse than the AdaHAT. AdaHAT-no-reg adjusts more aggressively from the start as if the network capacity is insufficient, despite the network capacity being actually sufficient. It hastily breaks stability built from HAT before affecting plasticity, leading to significant initial performance drops, preventing it from catching up with AdaHAT in the longer run. This demonstrates the effect of network sparsity information leveraging the maximum benefits from HAT. Similarly, we observe that AdaHAT-no-sum underperforms AdaHAT in the same way as AdaHAT-no-reg, as it loses another piece of crucial information, leading to less efficient and less targeted gradient adjustments. This demonstrates the effect of parameter importance information leveraging information support from previous tasks. Note that an interesting observation is the performance of AdaHAT-no-sum slows dropping down and begins to approach that of AdaHAT in the long run. This is because, in the long run, the network has already lost sparsity and kept allocating its parts to new tasks for a long time, making all units tend to have an equal likelihood of being dedicated to a similar number of previous tasks, and making it increasingly difficult to distinguish parameter importance. The mechanism of AdaHAT becomes similar to AdaHAT-no-sum in the long run when the important scores lose their meaning, leading to their behavior being rather similar.

4.4 Hyperparameters

AdaHAT introduces only one additional hyperparameter, α , which acts as an additional regulation for the stability-plasticity trade-off by controlling the overall intensity of gradient adjustment. We evaluate α over a range of values: $10^{-7}, 2 \times 10^{-7}, \dots, 9 \times 10^{-7}, 10^{-6}, 2 \times 10^{-6}, \dots, 10^{-5}$. We conduct the hyperparameter experiments with the same settings as the previous experiment of AdaHAT on the Permuted MNIST dataset shown in Table 1.

We observe from Fig. 6 that AdaHAT is optimal when α is set to 10^{-6} in this 20-task experiment. Increasing α to 10^{-5} leads to significant performance drops as larger adjustments cause more forgetting. Conversely, smaller value of α is not ideal as well. For instance, while $\alpha = 10^{-7}$ performs well at first, it drops instead after 15 tasks. Thus, $\alpha = 10^{-6}$ achieves the optimal balance between stability and plasticity. Note that 10^{-6} is a very small value, indicating the updates of

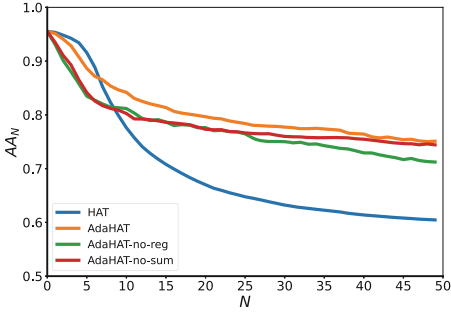


Fig. 5. Average Accuracy performance of different ablated approaches (50 tasks, Permuted MNIST).

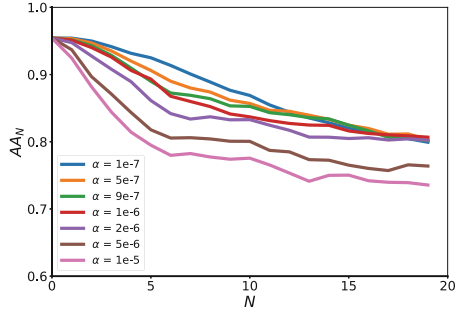


Fig. 6. Effect of hyperparameter α in AdaHAT on Average Accuracy performance (20 tasks, Permuted MNIST).

previously static parameters should be really subtle, which makes it even more important to design a proper and well-guided adjustment rate within this limit of small magnitude.

5 Conclusion

Catastrophic forgetting is one of the fundamental issues for deep neural networks, which has attracted a lot of research in continual learning. However, several existing architecture-based approaches that use hard attention mechanism to prevent the network from forgetting what it has learned in previous tasks tend to tilt the stability-plasticity trade-off towards stability, and suffer from the insufficient network capacity issue in long sequences of tasks. Consequently, these approaches perform well with sufficient network capacity but drastically become poor at the point the network capacity is exhausted when learning long sequences of tasks. In this paper, we propose a novel task-based attention mechanism, Adaptive Hard Attention to the Task (AdaHAT), which retains the stability benefits from HAT but aims to solve the aforementioned issues. Our experiment results show that AdaHAT outperforms several baselines on average performance, including HAT, especially on learning long sequence of tasks. The results on stability, plasticity and network capacity indicate that, the way that AdaHAT manages the network capacity adaptively over time in long sequences of tasks which balances the stability-plasticity trade-off, could be the underlying reason for its better performance.

Our proposed adaptive parameter updating approach also shows that in the architecture-based approaches, those masked static network parameters for previous tasks are acceptable to be updated in small magnitudes when the sequence of tasks becomes longer and reaches the network capacity limit, but must be a well-guided and adaptive way. The results show that two pieces of information about previous tasks, one is parameter importance to previous tasks constructed

from our proposed summative attention vectors, and the other is sparsity regularization term reflecting the current network capacity, both play a crucial role in the updates of those parameters, and also show the effectiveness and advantage of the adaptive way we incorporate them into a single adjustment rate. We believe more subtle information about previous tasks can be explored and exploited to help balance the stability-plasticity trade-off. We leave this for future work.

References

1. Ahn, H., Cha, S., Lee, D., Moon, T.: Uncertainty-based continual learning with adaptive regularization. In: *Advances in Neural Information Processing Systems*, vol. 32 (2019)
2. Aljundi, R., Chakravarty, P., Tuytelaars, T.: Expert gate: lifelong learning with a network of experts. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3366–3375 (2017)
3. Buzzega, P., Boschini, M., Porrello, A., Abati, D., Calderara, S.: Dark experience for general continual learning: a strong, simple baseline. *Adv. Neural. Inf. Process. Syst.* **33**, 15920–15930 (2020)
4. De Lange, M., et al.: A continual learning survey: defying forgetting in classification tasks. *IEEE Trans. Pattern Anal. Mach. Intell.* **44**(7), 3366–3385 (2021)
5. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–256. *JMLR Workshop and Conference Proceedings* (2010)
6. Goodfellow, I.J., Mirza, M., Xiao, D., Courville, A., Bengio, Y.: An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint [arXiv:1312.6211](https://arxiv.org/abs/1312.6211)* (2013)
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778 (2016)
8. Hung, C.Y., Tu, C.H., Wu, C.E., Chen, C.H., Chan, Y.M., Chen, C.S.: Compacting, picking and growing for unforgetting continual learning. In: *Advances in Neural Information Processing Systems*, vol. 32 (2019)
9. Kaushik, P., Gain, A., Kortylewski, A., Yuille, A.: Understanding catastrophic forgetting and remembering in continual learning with optimal relevance mapping. *arXiv preprint [arXiv:2102.11343](https://arxiv.org/abs/2102.11343)* (2021)
10. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. *arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980)* (2014)
11. Kirkpatrick, J., et al.: Overcoming catastrophic forgetting in neural networks. *Proc. Natl. Acad. Sci.* **114**(13), 3521–3526 (2017)
12. Lee, S.W., Kim, J.H., Jun, J., Ha, J.W., Zhang, B.T.: Overcoming catastrophic forgetting by incremental moment matching. In: *Advances in Neural Information Processing Systems*, vol. 30 (2017)
13. Li, Z., Hoiem, D.: Learning without forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.* **40**(12), 2935–2947 (2017)
14. Lopez-Paz, D., Ranzato, M.: Gradient episodic memory for continual learning. In: *Advances in Neural Information Processing Systems*, vol. 30 (2017)
15. Mallya, A., Davis, D., Lazebnik, S.: Piggyback: adapting a single network to multiple tasks by learning to mask weights. In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 67–82 (2018)

16. Mallya, A., Lazebnik, S.: PackNet: adding multiple tasks to a single network by iterative pruning. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 7765–7773 (2018)
17. McCloskey, M., Cohen, N.J.: Catastrophic interference in connectionist networks: the sequential learning problem. In: Psychology of Learning and Motivation, vol. 24, pp. 109–165. Elsevier (1989)
18. Nguyen, C.V., Li, Y., Bui, T.D., Turner, R.E.: Variational continual learning. arXiv preprint [arXiv:1710.10628](https://arxiv.org/abs/1710.10628) (2017)
19. Ratcliff, R.: Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychol. Rev.* **97**(2), 285 (1990)
20. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: iCaRL: incremental classifier and representation learning. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 2001–2010 (2017)
21. Rusu, A.A., et al.: Progressive neural networks. arXiv preprint [arXiv:1606.04671](https://arxiv.org/abs/1606.04671) (2016)
22. Serra, J., Suris, D., Miron, M., Karatzoglou, A.: Overcoming catastrophic forgetting with hard attention to the task. In: International Conference on Machine Learning, pp. 4548–4557. PMLR (2018)
23. Srivastava, R.K., Masci, J., Kazerounian, S., Gomez, F., Schmidhuber, J.: Compete to compute. In: Advances in Neural Information Processing Systems, vol. 26 (2013)
24. Van de Ven, G.M., Tolias, A.S.: Three scenarios for continual learning. arXiv preprint [arXiv:1904.07734](https://arxiv.org/abs/1904.07734) (2019)
25. Wang, L., Zhang, X., Su, H., Zhu, J.: A comprehensive survey of continual learning: theory, method and application. *IEEE Trans. Pattern Anal. Mach. Intell.* **46**, 5362–5383 (2024)
26. Wortsman, M., et al.: Supermasks in superposition. *Adv. Neural. Inf. Process. Syst.* **33**, 15173–15184 (2020)
27. Zenke, F., Poole, B., Ganguli, S.: Continual learning through synaptic intelligence. In: International Conference on Machine Learning, pp. 3987–3995. PMLR (2017)



Probabilistic Circuits with Constraints via Convex Optimization

Soroush Ghandi^{1(✉)}, Benjamin Quost^{1,2}, and Cassio de Campos¹

¹ Eindhoven University of Technology, Eindhoven, Netherlands
{s.ghandi,c.decampos}@tue.nl, benjamin.quost@utcf.fr

² University of Technology of Compiègne, Compiègne, France

Abstract. This work addresses integrating probabilistic propositional logic constraints into the distribution encoded by a probabilistic circuit (PC). PCs are a class of tractable models that allow efficient computations (such as conditional and marginal probabilities) while achieving state-of-the-art performance in some domains. The proposed approach takes both a PC and constraints as inputs, and outputs a new PC that satisfies the constraints. This is done efficiently via convex optimization without the need to retrain the entire model. Empirical evaluations indicate that the combination of constraints and PCs can have multiple use cases, including the improvement of model performance under scarce or incomplete data, as well as the enforcement of machine learning fairness measures into the model without compromising model fitness. We believe that these ideas will open possibilities for multiple other applications involving the combination of logics and deep probabilistic models.

Keywords: Probabilistic Circuits · Probabilistic Logic · Graphical Models

1 Introduction

Learning the underlying distribution of data has always been an integral part of many machine learning tasks, and generative probabilistic models are typically used to learn such distributions; Methods like VAEs [16] and GANs [10] can learn very complex, high dimensional distributions; however, they provide limited access to the learned distribution in the sense that many inferences (e.g. marginalization and conditioning) are practically intractable in the learned structures. On the other side of the spectrum, we have *tractable* probabilistic models, which provide better access to the learned distribution (and thus allow for a wider range of exact inferences) by trading off some fitting power.

Probabilistic Circuits (PCs) are tractable models that use a graph-based representation to encode high-dimensional distributions [17]. Besides being

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70352-2_10.

tractable, another advantage of PCs is that their graph-based representation allows them to process various inferences in polynomial time [34]. Learning a PC from data \mathcal{D} is defined as specifying a PC that represents the probability distribution underlying \mathcal{D} . This active line of research has seen several meaningful proposals in the past few years [1, 6–8, 13, 14, 19–21, 23, 25, 27, 30–33, 35, 36], but remains nevertheless open, given the difficulty of the task which involves both structure and parameter learning.

We address here the issue of *enhancing* a PC learned from data by using additional information and/or learning goals. To this end, we propose an approach for combining the PC with probabilistic propositional logic (PPL) [4, 11] constraints. More specifically, the approach takes a learned PC and updates (some of) its parameters in order to enforce the PPL constraints globally in the represented distribution. Our strategy can be seen as a “post-learning” method, which gives the advantages of versatility (existing models need not be retrained) as well as modularity: one may train a PC using any algorithm, as long as the resulting network keeps dependent variables (which may appear together in the same PPL constraint) together within the model; that is, they cannot appear factorized in the graph (further details are given in Sect. 3). This allows to build convex optimization problems (more precisely, constrained KL-divergence solvers) over parts of the distribution encoded in the PC so as to improve the corresponding model parameters via an efficient tractable method. On another note, being a post-learning method allows our approach to be considered as a form of belief revision for learned PCs. It is worth mentioning that a method to learn the structure and parameters of PCs with domain constraints is also proposed in [15], however with important differences. The learning method in [15] is data-driven (sample-based) and hence it is not applicable as a post-learning process; the constraints are limited to linear functions of conditional probabilities, and the proposed method is only tractable for deterministic PCs.

The benefits of having user-specified constraints are multi-fold. In order to illustrate them, we employ PPL constraints in a few (non-exhaustive) scenarios: (1) we improve the quality of models by enforcing that the yielded model matches the empirical marginal distributions under situations of (a) scarce data or (b) missing data; (2) we enforce fairness constraints into the model while at the same time avoiding a decrease in fitness. Overall, the experiments indicate that using PPL constraints often yields a better model (without compromising efficiency or accuracy), which is likely possible because of typical over-parameterizations that current large machine learning models impose. We emphasize that these applications of constraints are only a few examples of possible use, as we believe there are many other possibilities ahead to be tried.

2 Probabilistic Circuits

Probabilistic circuits (PCs) are a family of distribution representations facilitating many exact and efficient inference routines (see [2] for a nice introduction). A PC encodes a probabilistic model over a collection of variables \mathbf{X} ; it

is structured as a rooted directed acyclic graph \mathcal{G} , containing three types of nodes: (i) distribution nodes, (ii) sum nodes, and (iii) product nodes. Distribution nodes are the leaves of the graph \mathcal{G} , while sum and product nodes are the internal nodes. Each distribution node (leaf) v computes a probability distribution over some subset $\mathbf{X}' \subseteq \mathbf{X}$, i.e. an integrable function $p_v(\mathbf{x}') : \mathcal{X}' \rightarrow \mathbb{R}^+$ from the sample space of \mathbf{X}' to the non-negative real numbers. The *scope* of v is the set of variables $\text{sc}(v) := \mathbf{X}'$ over which the leaf computes a distribution. The scope of any internal node v (sum or product) is recursively defined as $\text{sc}(v) = \cup_{u \in \text{ch}(v)} \text{sc}(u)$, where $\text{ch}(v)$ is the set containing the children of v . Sum nodes compute convex combinations over their children, i.e. if v is a sum node, then v computes $v(\mathbf{x}) = \sum_{u \in \text{ch}(v)} w_{v,u} u(\mathbf{x})$, where $w_{v,u} \geq 0$. In a normalized PC, we have $\sum_{u \in \text{ch}(v)} w_{v,u} = 1$. Product nodes compute the product over their children, i.e. if v is a product node, then $v(\mathbf{x}) = \prod_{u \in \text{ch}(v)} u(\mathbf{x})$. The support of a node is the region where its associated function is strictly positive.

The main feature of PCs is that they facilitate a wide range of *tractable* inference routines, which go hand in hand with certain structural properties [2, 5]: (i) a sum node v is called *smooth* if its children have all the same scope: $\text{sc}(u) = \text{sc}(u')$, for any $u, u' \in \text{ch}(v)$; (ii) a product node v is called *decomposable* if its children have non-overlapping scopes: $\text{sc}(u) \cap \text{sc}(u') = \emptyset$, for any $u, u' \in \text{ch}(v)$, $u \neq u'$; (iii) a node is *consistent* if its support is non-empty. A PC is smooth (respectively decomposable) if all its sum (respectively product) nodes are smooth (respectively decomposable). A PC is consistent if all its nodes are consistent. The distribution $p(\text{sc}(v))$ represented by a node v in the PC is the function computed by the rules of the previous paragraph, and can be evaluated with a feed-forward pass. In order to ensure the tractability of queries, we can rely on smoothness and decomposability, but we also need leaf distribution nodes to compute inferences efficiently. This is a reason for many proposed PCs in the literature to assume that leaf nodes are univariate with some known distribution, such as Bernoulli, categorical, Gaussian, etc. Now, assume that we wish to compute a *marginal query*, that is, to evaluate the probability value over $\mathbf{X}_o \subset \mathbf{X}$ for evidence $\mathbf{X}_o = \mathbf{x}_o$, while marginalising $\mathbf{X}_{\neg o} = \mathbf{X} \setminus \mathbf{X}_o$. In smooth and decomposable PCs, this task reduces to performing marginalization at the leaves [28]: for each leaf v , one marginalizes $\text{sc}(v) \cap \mathbf{X}_{\neg o}$, and evaluates it for the values corresponding to $\text{sc}(v) \cap \mathbf{X}_o$. The desired marginal $p_{\mathbf{X}_o}(\mathbf{x}_o)$ results from evaluating internal nodes as in computing the complete distribution. Smoothness and consistency are sufficient to guarantee that the function of a PC represents a distribution. We also assume PCs are normalized (Fig. 1).

An important feature of normalized valid PCs is their interpretation as hierarchical, discrete mixture models [26, 38]:

$$p(\mathbf{x}) = \sum_z p(\mathbf{x}|z)p(z) = \sum_z p_z(\mathbf{x})p(z), \quad (1)$$

where Z is a discrete latent vector, which originates from the sum nodes of the structure. The number of states of Z , and thus of represented mixture components $p(\mathbf{x}|z)$, grows exponentially in the depth of the PC [24, 38]. While we

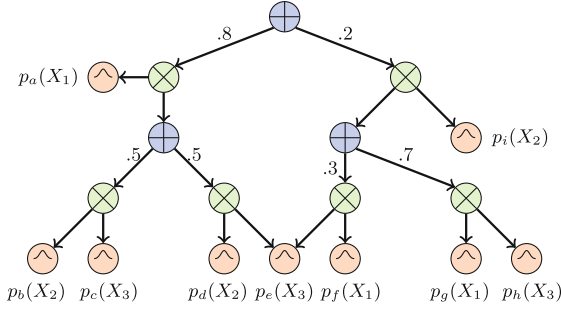


Fig. 1. Example of PC with variables X_1, \dots, X_3 . Sum nodes are in blue, product nodes in green, distribution leaf nodes in salmon. In this example, all leaf nodes are univariate. Subscriptions on each p in the figure are used to indicate that those are different leaf distributions (even if sometimes over the same variable). (Color figure online)

use this notation here and throughout the paper, we do not run computations directly in this formulation, but instead we make use of the graphical structure of the PC in order to perform efficient tractable inference, as usual for PCs.

3 Probabilistic Circuits with Constraints

We assume that a normalized valid PC has been produced (learned from data, designed by a human, etc.) over a domain with variables \mathbf{X} . Such a PC induces a joint distribution $p(\mathbf{X})$. The goal is to enforce some (linear) probabilistic propositional logic (PPL) constraints upon p . We work with constraints of the form:

$$\sum_{i_c} \tau_{i_c} \cdot p(F_{i_c}) \leq \alpha_c, \tag{2}$$

where each F_{i_c} is a propositional logic formula defined over Boolean variables $\mathbf{X}_c = \{X_{j_c}\}_{\forall j_c \subseteq \mathbf{X}}$, τ_{i_c}, α_c are real numbers, j_c (and i_c) are indexes of variables (terms) of the constraint c , and $c \in \mathcal{C}$ is an index over a set of PPL constraints. We assume that constraints are placed in buckets B (mathematically a bucket can be simply an index set indicating the constraints it contains) such that $\mathbf{X}_{B_1} \cap \mathbf{X}_{B_2} = \emptyset$ for all distinct buckets B_1, B_2 , where $\mathbf{X}_B = \cup_{c \in B} \mathbf{X}_c$ is the union of all variables appearing in a constraint inside bucket B . If any \mathbf{X}_{c_1} and \mathbf{X}_{c_2} of two constraints are not disjoint, then we put them together into the same bucket, so as to ensure that buckets have mutually exclusive sets of variables.

The constraints in each bucket B may obviously create dependencies among the variables \mathbf{X}_B . In order to avoid inconsistencies between such dependencies and those arising from the graph structure of the PC, we require that the variables in a bucket appear together in nodes of the model, that is, for any v, B , $X \in \mathbf{X}_B \cap \text{sc}(v) \Rightarrow \mathbf{X}_B \subseteq \text{sc}(v)$. Therefore, Eq. (1) can be recast as

$$p(\mathbf{X}) = \sum_z p(z) \prod_B p_z(\mathbf{X}_B) \prod_{X_i \in \mathbf{X} \setminus \cup \mathbf{X}_B} p_z(X_i), \tag{3}$$

where $p_z(\mathbf{X}_B)$ is a categorical distribution—note that notations $p_z(\mathbf{X}_B)$ and $p_z(X_i)$ employ a slight abuse, as the function itself is “aware” of the indexes of the variables in their arguments and may vary accordingly, for example, $p_z(X_i)$ is also a function of i and not only of X_i ; the same abuse holds elsewhere, for example in Expression (2). Equation (3) basically decomposes $p_z(\mathbf{x})$ of Eq. (1) into components that involve PPL variables (which remain together) and the other variables, which are assumed to be represented by univariate leaf-node distributions.

For ease of exposure, but also for the sake of compatibility with software that only deals with univariate leaf distributions, one can replace categorical distributions in leaf nodes with new sub-PCs. If one assumes independence among scope variables, then a product node with univariate leaf nodes suffices. If one wants to fully exploit the categorical distribution node, then a sum node with one child per parameter of the categorical distribution can be used. Figure 2 gives an example of dealing with a categorical “joint” distribution over two Boolean variables X_1, X_2 . Figure 2a shows the independent case, while Fig. 2b shows the joint approach to represent the distribution for X_1 and X_2 (note that Fig. 2b shows an example model of a fully parametrized distribution, and the structure can easily be altered with any other fully parametrized model, including for example lookup tables). The reader may have already noticed that large buckets of constraints will force the model to keep together many variables, which can be problematic as the number of parameters of the categorical joint distribution of all variables in a bucket B will grow exponentially in $|\mathbf{X}_B|$ (as in Fig. 2b, all possible configurations of \mathbf{X}_B would be listed). We will discuss this later, and ask the reader to assume that buckets (or equivalently scopes of leaf distribution nodes) are not large.

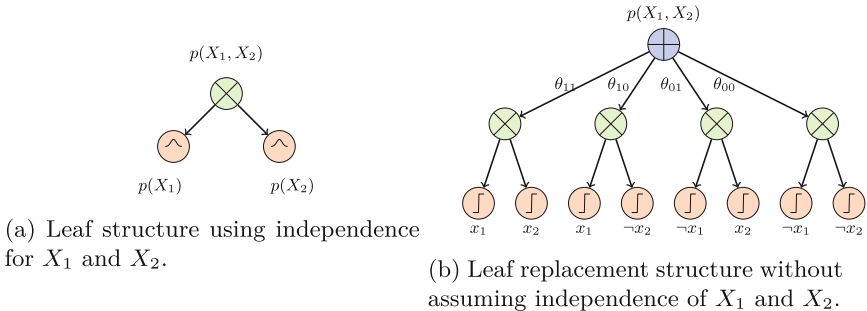


Fig. 2. Leaf distribution replacement structures that can be used to represent the parameters of a categorical variable for a bucket B with $\mathbf{X}_B = \{X_1, X_2\}$.

Given a PC representing $p(\mathbf{X})$ and PPL constraints, we aim to find an *efficient* approach to discover a new PC inducing a distribution $q^*(\mathbf{X})$ that is close to $p(\mathbf{X})$ while respecting the PPL constraints:

$$\begin{aligned}
q^*(\mathbf{X}) &= \underset{q(\mathbf{X})}{\operatorname{argmin}} \mathcal{L}(p(\mathbf{X}), q(\mathbf{X})) \\
\text{s.t. } \forall B, \forall c \in B &: \sum_{i_c} \tau_{i_c} \cdot q(F_{i_c}) \leq \alpha'_c, \quad q(\mathbf{X}) \in \mathcal{P}(\mathbf{X}),
\end{aligned} \tag{4}$$

where \mathcal{L} measures the discrepancy between two distributions, $\mathcal{P}(\mathbf{X})$ denotes a set of probability distributions over \mathbf{X} that can be represented by a PC, and B are buckets of PPL constraints. Optimization (4) is impractical, as it amounts to solving a complex optimization problem to search over $\mathcal{P}(\mathbf{X})$, even if \mathcal{L} is simple enough (a nonlinear optimization with nonlinear constraints). We aim to simplify this generic optimization, so that it can be done efficiently and in a tractable manner, while still attaining good results in practice. Therefore, we exploit the PC on which $p(\mathbf{X})$ was estimated, in order to constraint the search space of $q(\mathbf{X})$: we enforce $q(\mathbf{X})$ to have a shape similar to $p(\mathbf{X})$, i.e.

$$q(\mathbf{X}) = \sum_z p(z) \prod_B q_z(\mathbf{X}_B) \prod_{X_i \in \mathbf{X} \setminus \cup \mathbf{X}_B} p_z(X_i), \tag{5}$$

that is, only $\prod_B q_z(\mathbf{X}_B)$ will differ from the specification of $p(\mathbf{X})$. Plainly put, we only refine the distributions in the leaf nodes of the PC. Moreover, we use the Kullback-Leibler divergence $\mathcal{L}(p(\mathbf{X}), q(\mathbf{X})) = H(p(\mathbf{X}), q(\mathbf{X})) - H(p(\mathbf{X}))$ as discrepancy measure, where $H(\cdot)$ is the entropy and $H(\cdot, \cdot)$ the cross entropy. Clearly, we can focus on the cross entropy only, as the second term does not contain $q(\mathbf{X})$. Our first result is an upper bound on the cross entropy which allows us to run the optimization efficiently. The bound on the cross-entropy establishes an upper bound on the KL-divergence between $p(\mathbf{X})$ and $q(\mathbf{X})$.

Theorem 1. *Assume a PC representing a distribution $p(\mathbf{X})$ as in Eq. (3) and PPL constraints as in Eq. (2) (placed in disjoint buckets B) are given. Assume that $q(\mathbf{X})$ is a distribution induced by a PC with form as in Eq. (5). Then, $H(p(\mathbf{X}), q(\mathbf{X})) \leq \sum_B \mathbb{E}_z[H(p_z(\mathbf{X}_B), q_z(\mathbf{X}_B))] + H(p(\mathbf{X}', Z))$, where \mathbf{X}' are the variables not appearing in constraints.*

Proof. Note that we are particularly interested in terms with parameters in $q(\mathbf{X})$, as they will be optimized later. First, recall that

$$-H(p(\mathbf{X}), q(\mathbf{X})) = \sum_{\mathbf{x}} p(\mathbf{x}) \log q(\mathbf{x}), \tag{6}$$

and for any configuration \mathbf{x} of \mathbf{X} and for any arbitrary $z_0 \in Z$, we have:

$$q(\mathbf{x}) = \sum_z p(z) p'_z(\mathbf{x}') \prod_B q_z(\mathbf{x}_B) \geq p(z_0) p'_{z_0}(\mathbf{x}') \prod_B q_{z_0}(\mathbf{x}_B), \tag{7}$$

which holds because all terms are non-negative, where $\mathbf{X}' = \mathbf{X} \setminus \cup \mathbf{X}_B$ (variables not in any constraint), and $p'_z(\mathbf{X}') = \prod_{X_i \in \mathbf{X}'} p_z(X_i)$, for given z . By substituting (7) into (6), we can establish a lower bound on the negative cross-entropy term:

$$-H(p(\mathbf{X}), q(\mathbf{X})) \geq \sum_{\mathbf{x}} p(\mathbf{x}) \log[p(z_0)p'_{z_0}(\mathbf{x}') \prod_B q_{z_0}(\mathbf{x}_B)] \quad (8)$$

$$= \sum_{\mathbf{x}} \sum_z p(z)p'_z(\mathbf{x}') \prod_{\beta} p_z(\mathbf{x}_{\beta}) \log[p(z)p'_z(\mathbf{x}') \prod_B q_z(\mathbf{x}_B)], \quad (9)$$

with the arbitrary $z_0 \in Z$ in Expression (8) being chosen to be equal to z for each of the elements in the summation over z , thus resulting in Expression (9). Then, we can split Expression (9) into two parts (using the log of products as sum of logs), where only the second term depends on $q(\mathbf{X})$:

$$\begin{aligned} -H(p(\mathbf{X}), q(\mathbf{X})) &\geq \sum_{\mathbf{x}} \sum_z p(z)p'_z(\mathbf{x}') \prod_{\beta} p_z(\mathbf{x}_{\beta}) \log[p(z)p'_z(\mathbf{x}')] \\ &\quad + \sum_{\mathbf{x}} \sum_z p(z)p'_z(\mathbf{x}') \prod_{\beta} p_z(\mathbf{x}_{\beta}) \log[\prod_B q_z(\mathbf{x}_B)]. \end{aligned} \quad (10)$$

The first term in the RHS of Expression (10) can be reduced to $-H(p(\mathbf{X}', Z))$; it does not depend on $q(\mathbf{X})$, and will consequently not be analyzed further. The second term in the RHS can be manipulated as

$$\begin{aligned} &= \sum_B \sum_{\mathbf{x}} \sum_z p(z)p'_z(\mathbf{x}') \prod_{\beta} p_z(\mathbf{x}_{\beta}) \log q_z(\mathbf{x}_B) \\ &= \sum_B \sum_{\forall t} \sum_{\mathbf{x}_{B_t}} \sum_{\mathbf{x}'} \sum_z p(z)p'_z(\mathbf{x}') \prod_{\beta} p_z(\mathbf{x}_{\beta}) \log q_z(\mathbf{x}_B) \\ &= \sum_B \sum_z p(z) \sum_{\forall t} \prod_{\mathbf{x}_{B_t}} p_z(\mathbf{x}_{\beta}) \log q_z(\mathbf{x}_B) \sum_{\mathbf{x}'} p'_z(\mathbf{x}') \\ &= \sum_B \sum_z p(z) \sum_{\forall t} \prod_{\mathbf{x}_{B_t}} p_z(\mathbf{x}_{\beta}) \log q_z(\mathbf{x}_B) \\ &= \sum_B \sum_z p(z) \sum_{\mathbf{x}_B} p_z(\mathbf{x}_B) \log q_z(\mathbf{x}_B) \sum_{\forall t, B_t \neq B} \prod_{\beta \neq B} p_z(\mathbf{x}_{\beta}) \\ &= \sum_B \sum_z p(z) \sum_{\mathbf{x}_B} p_z(\mathbf{x}_B) \log q_z(\mathbf{x}_B) \prod_{\beta \neq B} \sum_{\mathbf{x}_{\beta}} p_z(\mathbf{x}_{\beta}) \\ &= \sum_B \sum_z p(z) \sum_{\mathbf{x}_B} p_z(\mathbf{x}_B) \log q_z(\mathbf{x}_B) \\ &= - \sum_B \mathbb{E}_z [H(p_z(\mathbf{X}_B), q_z(\mathbf{X}_B))]. \end{aligned} \quad (11)$$

Hence, $-H(p(\mathbf{X}), q(\mathbf{X})) \geq -H(p(\mathbf{X}', Z)) - \sum_B \mathbb{E}_z [H(p_z(\mathbf{X}_B), q_z(\mathbf{X}_B))]$, and the result follows. \square

Thus, we can adapt the PC at hand using the specified constraints by minimizing the upper bound on the desired discrepancy, leaving aside the term $H(p(\mathbf{X}', Z))$ which does not involve $q(\mathbf{X})$:

$$\begin{aligned}
 q^*(\mathbf{X}) &= \underset{q(\mathbf{X})}{\operatorname{argmin}} \sum_B \mathbb{E}_z [H(p_z(\mathbf{X}_B), q_z(\mathbf{X}_B))] \\
 \text{s.t. } \forall B, \forall c \in B &: \sum_{i_c} \tau_{i_c} \cdot q(F_{i_c}) \leq \alpha_c, \quad q(\mathbf{X}) \in \mathcal{P}(\mathbf{X}),
 \end{aligned} \tag{12}$$

Theorem 2 sheds light on the complexity of the procedure; it is based on considerably mild assumptions, as long as buckets do not involve too many variables.

Theorem 2. *Given the same inputs as Theorem 1, and assuming $|\mathbf{X}_B| \leq k$ for all buckets B , the solution q^* to the optimization in Optimization (12) can be found in polynomial time in the input size (while possibly exponential in k).*

Proof. The objective function is a sum over buckets containing (mutually) disjoint sets of variables, so we can solve Optimization (12) by solving separate optimizations for each bucket B :

$$\begin{aligned}
 \forall B: \quad q^*(\mathbf{X}_B) &= \underset{q(\mathbf{X}_B)}{\operatorname{argmin}} - \sum_z p(z) \sum_{\mathbf{x}_B} p_z(\mathbf{x}_B) \log q_z(\mathbf{x}_B) \\
 \text{s.t. } \forall c \in B &: \sum_{i_c} \tau_{i_c} \cdot q(F_{i_c}) \leq \alpha_c, \quad q(\mathbf{X}_B) \in \mathcal{P}(\mathbf{X}_B).
 \end{aligned} \tag{13}$$

(Note the abuse of notation here, as $q^*(\mathbf{X}_B)$ is used to indicate the parameters of model $q^*(\mathbf{X})$ that are associated with leaf nodes containing variables \mathbf{X}_B .) Optimization (13) can be solved for each B using convex optimization solvers, which run in polynomial time in the size of their inputs (and can be very efficient in practice). The values $p_z(\mathbf{x}_B)$ and $p(z)$ are fixed during the optimization and can be obtained directly from the PC model representing $p(\mathbf{X})$.

Assuming that $q_z(\mathbf{x}_B)$ is parameterized using values θ_{z,\mathbf{x}_B} representing a categorical distribution over \mathbf{X}_B conditional to $Z = z$ (same structure as in Fig. 2b), we obtain Optimization (14) for each bucket B . Note that in Optimization (14), each PPL formula F_{i_c} is written down as the sum of the worlds that satisfy the formula (we can query $F_{i_c}(\mathbf{x}_B)$ to see if each \mathbf{x}_B satisfies F_{i_c} , assuming $F_{i_c} = 1$ if so, and zero otherwise). Optimization (14) also connects the local parameters θ_{z,\mathbf{x}_B} with the marginal value of the candidate PC for \mathbf{x}_B , that is, $q(\mathbf{x}_B) = \sum_z p(z)\theta_{z,\mathbf{x}_B}$, which appears in the last expression of the optimization problem: thus, the imposed constraint is a global constraint in the joint model $q(\mathbf{X})$, and not simply a local constraint in the local parameters. Note also that $p(z) = q(z)$ (by assumption from Expression (5)).

$$\begin{aligned}
 \forall B: \quad q^*(\mathbf{X}_B) &= \underset{\theta_{z,\mathbf{x}_B}: \forall z, \mathbf{x}_B}{\operatorname{argmin}} - \sum_z p(z) \sum_{\mathbf{x}_B} p_z(\mathbf{x}_B) \log \theta_{z,\mathbf{x}_B}, \\
 \forall z, \mathbf{x}_B: \quad &\theta_{z,\mathbf{x}_B} \geq 0, \quad \forall z: 1 = \sum_{\mathbf{x}_B} \theta_{z,\mathbf{x}_B}, \\
 \text{s.t. } \forall c \in B &: \sum_{i_c} \tau_{i_c} \cdot \left(\sum_{\mathbf{x}_B} F_{i_c}(\mathbf{x}_B) \left(\sum_z p(z)\theta_{z,\mathbf{x}_B} \right) \right) \leq \alpha_c.
 \end{aligned} \tag{14}$$

Optimization (14), and hence Optimizations (4) and (13), will have a feasible solution so long as the set of PPL constraints has a feasible solution. This can be checked using linear programming using the constraints in Optimization (14). Therefore, it can be checked in polynomial time if the user provided an infeasible set of constraints. The number of buckets is bounded by the number of constraints C , which therefore also bounds the number of optimization calls. The optimization for bucket B has $O(|Z| \cdot |\mathbf{X}_B|)$ variables and $O(C \cdot |\mathbf{X}_B|)$ constraints (those are all very loose bounds), which is asymptotically bounded by the PC size plus constraints' size (that is, the input size), and convex optimization can be solved in polynomial time in the number of variables and constraints. \square

4 Experiments

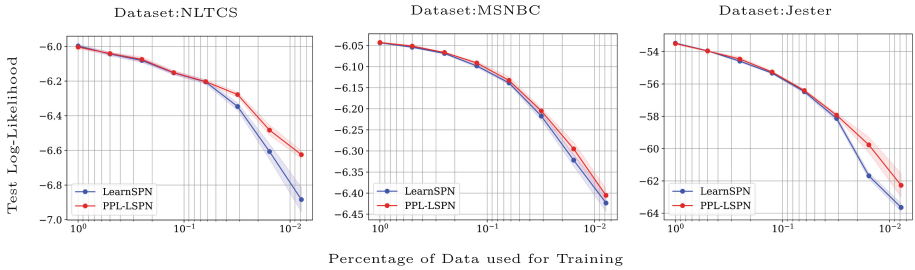


Fig. 3. LearnSPN vs. (constrained) PPL-LSPN trained on scarce datasets.

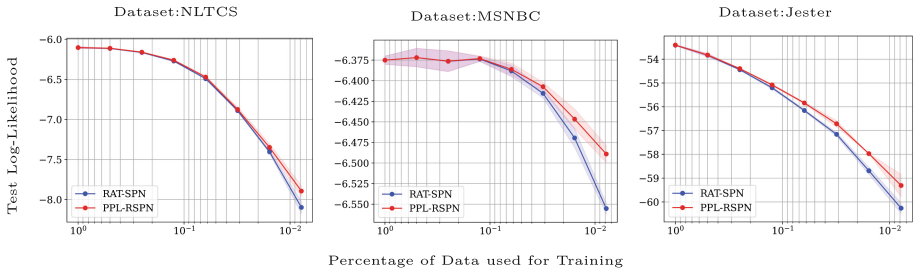


Fig. 4. RAT-SPN vs. (constrained) PPL-RSPN trained on scarce datasets.

We conduct a series of experiments to illustrate how constrained optimization can be utilized to shape a desirable performance or behavior in PCs. For the sake of this illustration, we focus on two use cases of constraints, namely (i) constraints over marginals of the distribution, and (ii) constraints for enforcing fairness in distributions.

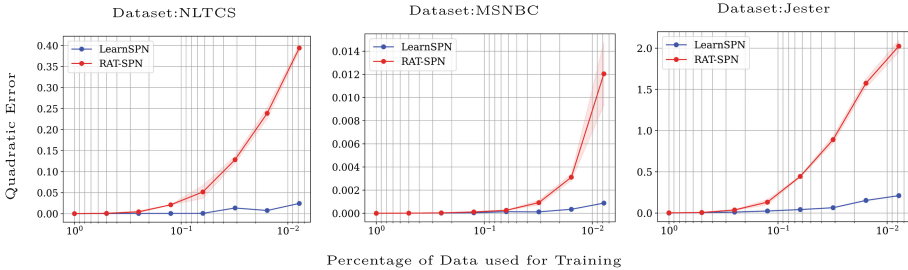


Fig. 5. Sum of quadratic differences on marginal parameters between the models with and without marginal constraints, when trained on scarce data. Constraints clearly refine the model more strongly for RAT-SPNs than for LearnSPN. Standard RAT-SPN marginals are very far from matching the empirical marginal distributions (data not shown).

The idea behind constraints on marginals is to adjust a probabilistic model to match the empirical marginals of \mathbf{X} on data \mathcal{D} . Typically, it is easier to accurately learn the marginal distributions over single variables rather than the whole joint distribution, in particular in cases when \mathcal{D} is scarce and/or incomplete. In Sects. 4.1 and 4.2 we explore how the use of constraints on empirical marginals affects the performance/behavior of learned PCs.

In Sect. 4.3, we investigate the impact of applying our method to a variety of fairness-specific classification tasks by adding fairness in the form of PPL constraints into PCs. Most common PC learning methods are not known to be inherently compatible with fairness, and being able to apply fairness constraints to PCs opens the door to utilizing these probabilistic models in areas where fairness is a priority, thus extending their domain of applicability.

Throughout the experiments, we utilize both LearnSPN [8] and RAT-SPN [29] for learning baseline PC models (one could also handcraft a PC for a purpose and use it with our approach, as we are not bound by the way the PC was obtained). We use the original implementation of RAT-SPN¹, and the implementation of [3]² for LearnSPN. For LearnSPN, statistical test significance and the Laplace smoothing parameter are set to 0.01 over all experiments. For RAT-SPN, hyperparameters that correspond to the region graph structure are set as follows: the number of recursive splits is 10, the depth of each recursive split is 2, the number of input distributions in each partition is 8, and the number of sum nodes per partition is 8; all the other hyperparameters are set to their defaults. For each experiment, RAT-SPN is trained for 20 epochs.

4.1 Scarce Datasets

We carry out this experiment on three different binary datasets, namely NLTCS, MSNBC, and Jester [22]. Our goal is to illustrate how additional information

¹ <https://github.com/cambridge-mlg/RAT-SPN>.

² <https://github.com/AICorreia/GeFs>.

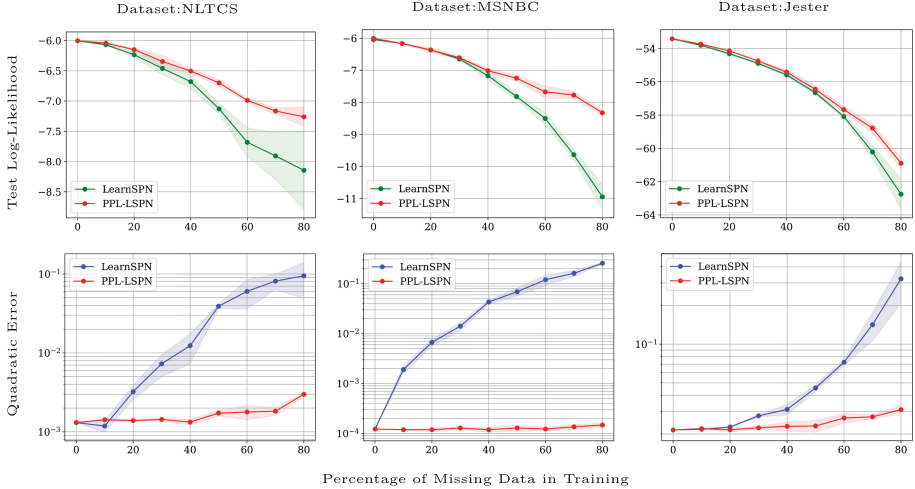


Fig. 6. LearnSPN vs. (constrained) PPL-LSPN trained on datasets with MCAR missing values. Test log-likelihood measures the joint fitness, while quadratic error shows the quality of the marginals of the model with respect to the marginals of test data, with clear superior accuracy after constraints are imposed.

pertaining to the empirical marginal distributions can be incorporated into the PC so as to compensate for data scarcity. In order to simulate scarce data, we randomly subsample each dataset with a varying number of data instances. We use this subsample to train the PC model using LearnSPN and RAT-SPN. We then improve the model using the procedure described above so as to match the empirical marginal distributions: we add PPL constraints of the form $p(X_i = 1) = \alpha_i$ for every variable X_i , which we enforce globally into the model. The test log-likelihood results are given in Figs. 3 and 4. The enhanced models obtained by applying the constrained optimization are called PPL-LSPN (variant of the LearnSPN baseline), and PPL-RSPN (variant of the RAT-SPN).

As can be seen, PPL-LSPN (Fig. 3) and PPL-RSPN (Fig. 4) slightly outperform LearnSPN (resp. RAT-SPN) as the training data become scarcer. This performance gain (in terms of testing data log-likelihood) is not similar across all datasets, which we attribute to the relative amount of information captured by the marginals. Arguably, in smaller datasets (in terms of the number of samples), matching marginals should lead to larger performance gains, as marginals encode a relatively larger amount of information. More importantly, matching marginals did not harm joint accuracy.

We argue that marginal matching is even more advantageous to PPL-RSPN compared to PPL-LSPN, as the learned marginals are far more erroneous in the case of RAT-SPN. Figure 5 displays the increase in quadratic error induced by *not* matching marginals, for both LearnSPN and RAT-SPN. Clearly, a large gap can be observed between LearnSPN and RAT-SPN on scarce data. Somewhat to our surprise, estimated marginals in RAT-SPN are far off (also when compared

to LearnSPN), making that model useless for marginal inference unless the constraints are imposed. Hence, and in particular for RAT-SPN, matching marginals leads to a strong improvement on the marginals themselves while (only but still) slightly improving the test joint likelihood.

4.2 Experiments with Missing Values

We again use the three binary datasets above (NLCTS, MSNBC, and Jester). In order to simulate missing data, we train the baseline PC (via LearnSPN) in a missing completely at random (MCAR) setting, by removing entries completely at random from the data tables. After the models are trained, we enhance the learned distribution to match the training data marginals using the proposed approach. Note that the current implementation of RAT-SPN mimics the effect of missing data with dropout layers (which is different from learning in presence of missing values); as well, the original version of RAT-SPN is not equipped to deal with missing data at training time, but can be easily tweaked for that purpose. We therefore focus these experiments on models trained with LearnSPN.

Results with MCAR data are summarized in Fig. 6. The top plots show the joint testing data log-likelihood, while the bottom plots show the difference in the testing data marginal distributions (whose gains are very clear). We can see that in every experiment, as the proportion of missing values increases, the PC enhanced using constraints outperforms the base model, which suggests that marginal matching can be considered as an effective way to deal with missing data, potentially as an alternative to data imputation.

4.3 Fairness Experiments

We investigate the impact of imposing fairness constraints in PCs. For each experiment, we assume variables \mathbf{X} which comprise a binary class/target variable $Y \in \mathbf{X}$ and a binary protected attribute $X' \in \mathbf{X}, X' \neq Y$. Our objective is to improve the distribution learned via a PC towards fairness for the protected attribute when predicting class labels. We consider statistical parity as our measure of fairness (we use this as an example; we will not debate on fairness measures, since it is not the main focus of the paper). The corresponding fairness constraint is $p(y = 1|x' = 1) = p(y = 1|x' = 0)$. It is clear that this constraint is not of the form $\sum_i \tau_i \cdot p(F_i) \leq \alpha$. It would actually induce a non-linear constraint and the convex optimization could not be directly applied. However, we can lift the optimization problem to a higher dimension by including a new unknown β where we take $p(y = 1|x' = 1) = p(y = 1|x' = 0) = \beta$. This latter can be decomposed into two separate linear constraints in the desired form:

$$\begin{aligned}
 p((y = 1) \wedge (x' = 1)) - \beta \cdot p(x' = 1) &= 0, \\
 p((y = 1) \wedge (x' = 0)) - \beta \cdot p(x' = 0) &= 0;
 \end{aligned}
 \tag{15}$$

and as long as β is fixed, the optimization can be carried out to impose the constraints in Eq. (15) to a learned PC using the proposed approach. In order to solve for β , we simply carry out an exhaustive search over candidate values between 0 and 1, retaining the best based on the performance of each resulting PC (obviously, this search procedure is reasonable for a single unknown β , or at most a few; otherwise, a smarter strategy would be required).

We consider six different classification datasets commonly used in fairness-aware machine learning, namely Adult, German Credit, Bank Marketing, Dutch Census, Credit Card Clients, and Law School [18]. As in Sect. 4.1, we refer to the variants as PPL-LSPN (for LearnSPN) and PPL-RSPN (for RAT-SPN). The details regarding the pre-processing of each dataset are provided in the appendix. The results are displayed in Tables 1 and 2. Not only PPL-LSPN and PPL-RSPN are able to achieve a “fair” distribution w.r.t the protected attribute, but they also manage to do so without losing much of their representation power compared to LearnSPN or RAT-SPN, that is, the test likelihood and 0-1 accuracy are barely affected while statistical parity is enforced by the use of constraints. We stress out that our procedure being a post-processing of the PC at hand, models already trained and potentially in use in applications could be enhanced without the need of re-training from scratch.

Table 1. Classification with LearnSPN vs. PPL-LSPN enforcing statistical parity via constraints.

Dataset	Protected Attribute	Method	Test LL	Accuracy	Statistical Parity
Adult	Sex	LearnSPN	-13.614	0.8256	0.1754
		PPL-LSPN	-13.764	0.7946	0.0
German Credit	Sex	LearnSPN	-22.802	0.6993	-0.0171
		PPL-LSPN	-23.075	0.704	0.0
Bank Marketing	Marital Status	LearnSPN	-16.448	0.8957	-0.0305
		PPL-LSPN	-16.493	0.8949	0.0
Dutch Census	Sex	LearnSPN	-9.801	0.8141	0.2520
		PPL-LSPN	-9.947	0.7359	0.0
Cr. Card Clients	Sex	LearnSPN	-22.505	0.8164	0.0185
		PPL-LSPN	-22.539	0.8035	0.0
Law School	Race	LearnSPN	-11.800	0.9076	-0.3012
		PPL-LSPN	-11.845	0.9013	0.0

Table 2. Classification with RAT-SPN vs. PPL-RSPN enforcing statistical parity via constraints.

Dataset	Protected Attribute	Method	Test LL	Accuracy	Statistical Parity
Adult	Sex	RAT-SPN	-7.767	0.8193	0.2000
		PPL-RSPN	-7.796	0.8148	0.0
German Credit	Sex	RAT-SPN	-28.752	0.745	-0.0309
		PPL-RSPN	-28.756	0.745	0.0
Bank Marketing	Marital Status	RAT-SPN	-13.736	0.8820	-0.0388
		PPL-RSPN	-13.739	0.8788	0.0
Dutch Census	Sex	RAT-SPN	-12.880	0.7888	0.2620
		PPL-RSPN	-12.923	0.7629	0.0
Cr. Card Clients	Sex	RAT-SPN	-3.998	0.7838	0.0053
		PPL-RSPN	-3.998	0.7838	0.0
Law School	Race	RAT-SPN	-7.274	0.9050	-0.2054
		PPL-RSPN	-7.294	0.9034	0.0

5 Conclusions and Future Work

We introduce a novel approach that allows to incorporate probabilistic propositional logic (PPL) constraints into a (pre-trained) probabilistic circuit (PC), so that the distribution encoded by the PC respects the constraints. We explain our design choices which allow for achieving tractable learning and inferences while ensuring that PPL constraints are satisfied. We also develop theoretical foundations that explain the feasibility of the optimization and how to reach an optimal solution in computationally tractable (polynomial) time. Experiments illustrate how we can take PCs and enhance them into better PCs that can be applied to practical scenarios, for example by applying fairness measures to the learned distribution and by (arguably) better handling missing values in the training data. The supplementary materials, which include details regarding the discretization process of datasets, can be accessed through the arXiv version [9].

We make space for a couple of reflections. The goal of this research is to enhance machine learning models with probabilistic logic assessments, in the same spirit as neurosymbolic AI. We found out that PC models are already over-parameterized: thus, one can better tune the parameters in order to satisfy external constraints. The first obvious idea is to do so via some variation of Expectation-Maximization or gradient methods, putting violation of constraints as (strong) penalties. However, it is not guaranteed that constraints are fully enforced; we therefore see that avenue as a great direction to investigate, even though the solution is likely to differ from the one described here. We managed to find a way to improve PCs a posteriori (without retraining) and efficiently (the optimization can run exactly and fast with modern convex optimization solvers). This choice comes at the expense of being able to only change the

parameters of leaf distribution nodes; this—quite surprisingly—turns out to be enough to precisely enforce the constraints globally on the joint distribution while not losing model fitness. Moreover, we have no intention to claim that we are (or not) obtaining state-of-the-art results. This is an investigation of the combination of constraints into circuits, which we consider overall successful (but obviously not without limitations). We see many possibilities with that. We are aware that the bucket size limitation is a serious complication, but creative experiments show that there may be many interesting problems to solve even under such limitation. Moreover, we know that the limitation can be mitigated by using some smarter parametrization of the local distributions: this direction is definitely worthwhile, although it may lead to a decrease in accuracy and will likely not provide the same guarantees as we currently have.

Beyond these research directions, the paper opens doors for future work, as the desire to combine probabilistic logic constraints and deep machine learning methods is immense. Possible immediate avenues include extending the applicability of constraints on continuous and mixed variables, applying constraints to new tasks such as other forms of fairness measures (for instance, equalized odds [12]) in order to improve already learned PCs, improving the trade-off between accuracy and efficiency by using different optimizers, considering extensions beyond consistent and valid PCs, and utilizing constrained PCs to guide and tune intractable models (similar to [37], but with the added benefit of including arbitrary constraints to the PC), to name but a few.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Adel, T., Balduzzi, D., Ghodsi, A.: Learning the structure of sum-product networks via an SVD-based algorithm. In: *Uncertainty in Artificial Intelligence (2015)*. <https://api.semanticscholar.org/CorpusID:15429402>
2. Van den Broeck, G., Di Mauro, N., Vergari, A.: Tractable probabilistic models: representations, algorithms, learning, and applications. <http://web.cs.ucla.edu/~guyvdb/slides/TPMTutorialUAI19.pdf> (2019), tutorial at *Uncertainty in Artificial Intelligence (UAI) 2019*
3. Correia, A., Peharz, R., de Campos, C.P.: Joints in random forests. In: *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 11404–11415 (2020)
4. Cozman, F.G., de Campos, C.P., da Rocha, J.C.F.: Probabilistic logic with independence. *Int. J. Approximate Reasoning* **49**(1), 3–17 (2008)
5. Darwiche, A.: A differential approach to inference in Bayesian networks. *J. ACM* **50**(3), 280–305 (2003)
6. Dennis, A., Ventura, D.: Learning the architecture of sum-product networks using clustering on variables. In: *Advances in Neural Information Processing Systems*, vol. 25 (2012)
7. Di Mauro, N., Vergari, A., Basile, T.M.A., Esposito, F.: Fast and accurate density estimation with extremely randomized cutset networks. In: *Ceci, M., Hollmén, J.,*

- Todorovski, L., Vens, C., Džeroski, S. (eds.) ECML PKDD 2017. LNCS (LNAI), vol. 10534, pp. 203–219. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-71249-9_13
8. Gens, R., Domingos, P.: Learning the structure of sum-product networks. In: International Conference on Machine Learning, pp. 873–880. PMLR (2013)
 9. Ghandi, S., Quost, B., de Campos, C.: Probabilistic circuits with constraints via convex optimization. arXiv preprint [arXiv:2403.13125](https://arxiv.org/abs/2403.13125) (2024)
 10. Goodfellow, I., et al.: Generative adversarial networks. *Commun. ACM* **63**(11), 139–144 (2020)
 11. Hansen, P., Jaumard, B.: Probabilistic satisfiability. In: Kohlas, J., Moral, S. (eds.) *Handbook of Defeasible Reasoning and Uncertainty Management Systems: Algorithms for uncertainty and defeasible reasoning*, vol. 5, pp. 321–367. Springer (2000). https://doi.org/10.1007/978-94-017-1737-3_8
 12. Hardt, M., Price, E., Srebro, N.: Equality of opportunity in supervised learning. In: *Advances in Neural Information Processing Systems*, vol. 29 (2016)
 13. Hsu, W., Kalra, A., Poupart, P.: Online structure learning for sum-product networks with gaussian leaves. arXiv preprint [arXiv:1701.05265](https://arxiv.org/abs/1701.05265) (2017)
 14. Kalra, A., Rashwan, A., Hsu, W.S., Poupart, P., Doshi, P., Trimonias, G.: Online structure learning for feed-forward and recurrent sum-product networks. In: *Advances in Neural Information Processing Systems*, vol. 31 (2018)
 15. Karanam, A., Mathur, S., Sidheekh, S., Natarajan, S.: Bayesian learning of probabilistic circuits with domain constraints. In: *The 6th Workshop on Tractable Probabilistic Modeling* (2023)
 16. Kingma, D.P., Welling, M.: Auto-encoding variational Bayes. arXiv preprint [arXiv:1312.6114](https://arxiv.org/abs/1312.6114) (2013)
 17. Koller, D., Friedman, N.: *Probabilistic Graphical Models: Principles and Techniques*. MIT press (2009)
 18. Le Quy, T., Roy, A., Iosifidis, V., Zhang, W., Ntoutsi, E.: A survey on datasets for fairness-aware machine learning. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **12**(3), e1452 (2022)
 19. Lee, S.W., Watkins, C., Zhang, B.T.: Non-parametric Bayesian sum-product networks. In: *ICML Workshop on Learning Tractable Probabilistic Models*, vol. 32 (2014)
 20. Liang, Y., Bekker, J., Van den Broeck, G.: Learning the structure of probabilistic sentential decision diagrams. In: *Uncertainty in Artificial Intelligence (UAI)* (2017)
 21. Liu, A., Van den Broeck, G.: Tractable regularization of probabilistic circuits. *Adv. Neural. Inf. Process. Syst.* **34**, 3558–3570 (2021)
 22. Lowd, D., Davis, J.: Learning markov network structure with decision trees. In: *2010 IEEE International Conference on Data Mining*, pp. 334–343. IEEE (2010)
 23. Molina, A., Vergari, A., Di Mauro, N., Natarajan, S., Esposito, F., Kersting, K.: Mixed sum-product networks: a deep architecture for hybrid domains. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32(1) (2018)
 24. Peharz, R.: *Foundations of sum-product networks for probabilistic modeling*. Ph.D. thesis, PhD thesis, Graz University of Technology (2015)
 25. Peharz, R., Geiger, B.C., Pernkopf, F.: Greedy part-wise learning of sum-product networks. In: Blockeel, H., Kersting, K., Nijssen, S., Železný, F. (eds.) *ECML PKDD 2013. LNCS (LNAI)*, vol. 8189, pp. 612–627. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40991-2_39
 26. Peharz, R., Gens, R., Pernkopf, F., Domingos, P.: On the latent variable interpretation in sum-product networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **39**(10), 2030–2044 (2016)

27. Peharz, R., et al.: Einsum networks: fast and scalable learning of tractable probabilistic circuits. In: International Conference on Machine Learning, pp. 7563–7574. PMLR (2020)
28. Peharz, R., Tschitschek, S., Pernkopf, F., Domingos, P.: On theoretical properties of sum-product networks. In: Artificial Intelligence and Statistics, pp. 744–752 (2015)
29. Peharz, R., et al.: Random sum-product networks: a simple and effective approach to probabilistic deep learning. In: Uncertainty in Artificial Intelligence, pp. 334–344. PMLR (2020)
30. Rahman, T., Gogate, V.: Merging strategies for sum-product networks: from trees to graphs. In: Uncertainty in Artificial Intelligence (UAI) (2016)
31. Rahman, T., Jin, S., Gogate, V.: Look ma, no latent variables: accurate cutset networks via compilation. In: International Conference on Machine Learning, pp. 5311–5320. PMLR (2019)
32. Rooshenas, A., Lowd, D.: Learning sum-product networks with direct and indirect variable interactions. In: International Conference on Machine Learning, pp. 710–718. PMLR (2014)
33. Trapp, M., Peharz, R., Ge, H., Pernkopf, F., Ghahramani, Z.: Bayesian learning of sum-product networks. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
34. Vergari, A., Choi, Y., Liu, A., Teso, S., Van den Broeck, G.: A compositional atlas of tractable circuit operations for probabilistic inference. *Adv. Neural. Inf. Process. Syst.* **34**, 13189–13201 (2021)
35. Vergari, A., Di Mauro, N., Esposito, F.: Simplifying, regularizing and strengthening sum-product network structure learning. In: Appice, A., Rodrigues, P.P., Santos Costa, V., Gama, J., Jorge, A., Soares, C. (eds.) ECML PKDD 2015. LNCS (LNAI), vol. 9285, pp. 343–358. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23525-7_21
36. Vergari, A., Molina, A., Peharz, R., Ghahramani, Z., Kersting, K., Valera, I.: Automatic Bayesian density analysis. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33(1), pp. 5207–5215 (2019)
37. Zhang, H., Dang, M., Peng, N., Van den Broeck, G.: Tractable control for autoregressive language generation. In: International Conference on Machine Learning, pp. 40932–40945. PMLR (2023)
38. Zhao, H., Poupart, P., Gordon, G.J.: A unified approach for learning the parameters of sum-product networks. In: Advances in Neural Information Processing Systems, vol. 29 (2016)



FedAR: Addressing Client Unavailability in Federated Learning with Local Update Approximation and Rectification

Chutian Jiang, Hansong Zhou, Xiaonan Zhang^(✉), and Shayok Chakraborty

Department of Computer Science, Florida State University, Tallahassee, FL, USA
{cj20cn,hz21e}@fsu.edu, {xzhang,shayok}@cs.fsu.edu

Abstract. Federated learning (FL) enables clients to collaboratively train machine learning models under the coordination of a server in a privacy-preserving manner. One of the main challenges in FL is that the server may not receive local updates from each client in each round due to client resource limitations and intermittent network connectivity. The existence of unavailable clients severely deteriorates the overall FL performance. In this paper, we propose *FedAR*, a novel client update Approximation and Rectification algorithm for FL to address the client unavailability issue. FedAR can get all clients involved in the global model update to achieve a high-quality global model on the server, which also furnishes accurate predictions for each client. To this end, the server uses the latest update from each client as a surrogate for its current update. It then assigns a different weight to each client's surrogate update to derive the global model, in order to guarantee contributions from both available and unavailable clients. Our theoretical analysis proves that FedAR achieves optimal convergence rates on non-IID datasets for both convex and non-convex smooth loss functions. Extensive empirical studies show that FedAR comprehensively outperforms state-of-the-art FL baselines including FedAvg, MIFA, FedVARP and Scaffold in terms of the training loss, test accuracy, and bias mitigation. Moreover, FedAR also depicts impressive performance in the presence of a large number of clients with severe client unavailability.

Keywords: Federated learning · Client selection · Bias mitigation

1 Introduction

Federated learning (FL) allows multiple clients to collaboratively learn a powerful global machine learning model without sharing the training data with the server. As a privacy-preserving and communication-efficient distributed learning framework, FL has garnered substantial research attention and has surged as a

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70352-2_11.

key enabler of distributed intelligence in many real-world applications, such as next-word prediction on mobile keyboards [11] and medical record analysis in digital health [4]. In the vanilla FL algorithm, known as FedAvg [21], the server distributes the current global model to all the clients in each round, which serves as the basis for running several steps of stochastic gradient descent (SGD) on the local data for each client. The local updates are then sent back to the server to update the global model. This process is iterated until the global model converges.

In FL, clients can be diverse, ranging from medical wearables and IoT devices to smartphones [38]. Many of these clients operate as low-power devices and communicate over wireless networks. This presents a challenge to FedAvg, as clients may abort training midway due to issues like low battery levels or incoming calls [2, 10, 15, 21]. As a result, clients may fail to return their trained local updates to the server, especially when the communication from the clients to the server is hampered by poor channel quality and intermittent connectivity [33–37, 39] (also referred to as *unavailable / non-participating clients* or the *partial client participation problem*). In FedAvg, the inability to receive local updates from unavailable clients can cause a serious delay and it can even discard these updates when deriving the global model to maintain learning efficiency [26, 31, 32, 41]. Missing the expected local updates introduces an undesired bias against unavailable clients [1, 31]. This will result in the global model overfitting the characteristics of consistently available clients, thereby diminishing its performance for clients that participate less frequently and reducing its overall generalization capability [5, 12, 13, 22, 40].

The primary goal of this paper is to develop and validate an efficient FL algorithm termed *Federated Learning with local update Approximation and Rectification* (FedAR), which addresses the partial client participation problem. We first study the contributions of the latest observed local updates from unavailable clients to the global update. Our observation reveals that unavailable clients with varying inactive rounds exert diverse positive influences on the global update. Motivated by this insight, we propose a novel server-side aggregation strategy that incorporates local updates from unavailable clients in the global update. More importantly, our framework does not require any additional computation at the clients or introduce any extra communication between the clients and the server. FedAR utilizes the latest update from each client observed by the server as a surrogate of its current update, which is then used in updating the global model. Moreover, we devise an innovative weighting scheme to accommodate the variable influence on the global model from local updates of clients with differing inactive rounds. We slightly magnify the contributions from unavailable clients (based on the number of inactive rounds) in addition to the contributions from the available clients, to update the global model. To achieve this, we design the weight as a mildly increasing function of the number of inactive rounds of each client. This strategy enables the server to include the local data distribution information from unavailable clients in updating the global model, thereby circumventing the bias against these clients. Lastly, unlike tradi-

tional FL, FedAR does not assume that the server is aware of the total number of clients in advance. Instead, it dynamically counts the number of clients who get involved in the global model update, which better reflects real-world application scenarios. In light of the above discussion, we summarize our key contributions in this paper as follows:

- We propose FedAR, a novel FL algorithm that addresses the client unavailability issue. FedAR unevenly weighs the contributions from both available and unavailable clients in the global model update based on the number of their inactive rounds. Moreover, FedAR does not necessitate any additional computation at the clients, nor does it demand any extra communication between the clients and the server. It does not require all clients to participate in FL in the first round either.
- We theoretically provide a convergence guarantee for FedAR for both convex and non-convex smooth loss functions on non-IID datasets across clients.
- We evaluate the performance of FedAR on three real-world datasets MNIST, CIFAR-10, and SVHN. Compared to the vanilla and the state-of-the-art FL baselines, FedAvg, MIFA, FedVARP, and Scaffold, FedAR can achieve a 75% improvement in test accuracy and a 50% reduction in training loss in the best case. Moreover, we empirically show that FedAR can better mitigate the bias against unavailable clients, as evidenced by the observation that the derived global model generates more accurate predictions for clients who have been intermittently inactive during the training process. FedAR also demonstrates impressive performance in the presence of a large number of clients with severe client unavailability.

2 Related Work

One of the main challenges of the vanilla FL algorithm, FedAvg, is the intermittent unavailability of clients. Specifically, the server will not update the global model until receiving local updates from all clients, which results in considerable training delay in the presence of client unavailability. Client sampling can be used as a remedy to this issue, where some clients are selected to participate in the global model update. The common client sample strategies include random sampling, significant sampling, and cluster sampling. Random sampling [21] selects clients at random whereas importance sampling [6, 7, 20] selects the most valuable clients in terms of data quantity, communication time, and local training results. In cluster sampling [3, 8, 9], clients are first divided into groups based on sample size, model similarity etc.; the clients in each group are then selected for global update. All these sampling strategies engage only available clients but ignore unavailable clients in the global update. Consequently, the global model biases towards the available clients that are selected repetitively [23], which would undermine the FL performance.

A body of research addresses the client unavailability issue by incorporating stale updates from unavailable clients into the training process, such as the Memory-augmented Impatient Federated Averaging (MIFA) algorithm [10] and

the Federated VAriance Reduction for Partial Client Participation (FedVARP) algorithm [14]. Their major differences with FedAR are listed in Table 1. In particular, seeking to maximize non-IID data coverage, MIFA gives equal weightage to updates from both available and unavailable clients, making it a biased scheme. Even worse, MIFA requires all clients to participate in the first training round, which is an unrealistic assumption. FedVARP allocates higher weights to the updates from available clients than to the updates from unavailable clients. It also attempts to reduce the variance to available clients caused by the partial client partition, which, however, is not empirically demonstrated. Similar to both MIFA and FedVARP, the FedAR algorithm reuses the latest observed update for each client as an approximation of its current update. Different from MIFA, FedAR formulates a novel weighting scheme to efficiently involve unavailable clients with various inactive rounds in the global model update. Moreover, FedAR does not require all clients to participate in FL in the first training round. Motivated by [30], FedAR assigns higher weights to the updates of the unavailable clients with a larger number of inactive rounds, i.e., we amplify the local updates from unavailable clients, which is contrary to FedVARP. Our experimental results show the efficacy of FedAR in terms of overall convergence, test accuracy and bias mitigation, compared to relevant baselines.

Table 1. Comparison of FedAR with MIFA and FedVARP

	MIFA	FedVARP	FedAR
Enhance the FL efficiency with uncertain availability of clients			
Issue addressed	maximize data coverage	reduce variance of available local updates	reduce bias against unavailable local updates
Rationale on local updates	all have the same contribution	available ones have higher contributions	unavailable ones can also have contributions
Solution	allocate the same weight to all local updates	allocate higher weights to available local updates	allocate higher weights to unavailable local updates with higher contributions
All clients assumption	must respond in the first round	not necessarily respond in the first round	

3 Problem Setup

We consider that a set of clients $\mathcal{N} = \{1, 2, \dots, N\}$ with restricted power and computational resources collaborate with a server to execute FL over T rounds. The datasets for local training are subject to non-IID distributions. The clients and the server iteratively communicate over wireless networks to obtain a global model w aiming at minimizing the global loss function:

$$\min f(w) = \frac{1}{N} \sum_{i=1}^N f_i(w), \quad (1)$$

where $f_i(w)$ is the loss function for client i .

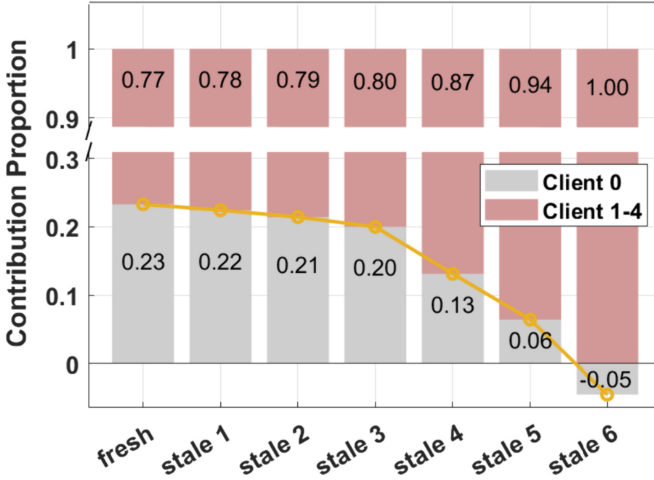


Fig. 1. Contribution of each client to the global model. “stale i ” denotes that Client 0 has been inactive for the last i rounds. “fresh” denotes that all the clients are active for all the 9 rounds. A high staleness level indicates more inactive rounds

3.1 Basic Algorithm of FL

We begin by recalling the vanilla FL setting in FedAvg. In round $t - 1$, $t \in \{1, \dots, T\}$, the server broadcasts the global model w_{t-1} to all the clients. Each client $i \in \mathcal{N}$ uses its own private dataset to execute K steps of Stochastic Gradient Descent (SGD) for the local update. For each step $k \in K$:

$$w_{t,k+1}^i = w_{t-1,k}^i - \eta_{t-1} \nabla f_i(w_{t-1,k}^i), \quad (2)$$

where η is the local learning rate and $\nabla f_i(\cdot)$ represents the gradient. Each client then sends back its local update to the server; the server aggregates all the client updates to derive the global model as:

$$w_t = \frac{1}{N} \sum_{i=1}^N w_{t,K}^i. \quad (3)$$

Problem in FedAvg. Practically, due to the limited resources of each client and the intermittent network connectivity, the server may not receive the local updates $w_{t,K}^i$ from all the clients; these clients are called *unavailable* / *non-participating* clients. Due to this, FedAvg delays or even aborts the local updates from unavailable clients during the global update, causing an undesirable bias against these unavailable clients. However, the local updates from the unavailable clients also contain valuable information, which can be useful in global model updates. We conduct a toy experiment on a simple, restricted setup to demonstrate this idea and provide motivation for our approach.

3.2 Motivation

Let us assume a standard FL setting where 5 clients (numbered 0 through 4) collaborate with a central server on a classification task using the CIFAR-10 dataset [17]. The server and clients execute a total of 9 rounds of communication. We conduct 7 different experiments, as shown by the vertical bars in Fig. 1. In all the experiments, client 1 to client 4 are always available across all the 9 rounds of communication. Client 0, conversely, becomes inactive after a certain number of rounds in each experiment. In Fig. 1, the term “stale i ” refers to client 0 being active for the initial $9 - i$ rounds and then inactive for the subsequent i rounds. For instance, “stale 3” indicates that client 0 is active from rounds 1 to 6 but inactive during rounds 7 to 9. In this case, we aggregate the most recent local updates (from the 9th round) for clients 1 to 4 and the local update from the 6th round for client 0 (last active round) to update the global model. “fresh” denotes the case where all the 5 clients were available across all the 9 rounds of communication. After the 9th round, we use the Shapley Value (SV) [25] to quantify the contribution of each local update to the global model. Shapley value is a classical concept in cooperative game theory, and it is extensively used to evaluate client contributions in FL [27–29]. We compute each client’s SV based on the global model’s test accuracy, which is obtained by different combinations of the local client updates for the different experiments. We sum up all the SV and represent the contribution of client 0 and clients 1 to 4 as a percentage; the larger the value, the greater the contribution. From Fig. 1, it can be observed that as the staleness level of client 0 increases (larger number of inactive rounds), its contribution to the global model (height of the gray bar) decreases. At stale 6, the contribution of client 0 is negative, meaning that its local update has an adverse effect on the global model. Based on the above toy experiment, we draw the following conclusions:

- The stale local updates from unavailable clients can still contribute to the global model (as evident from the gray bars in stale 1 through stale 5).
- The contribution of the stale local updates decreases with increasing staleness level, suggesting it may be necessary to assign higher weights (during the global update) to stale local updates with more inactive rounds.
- An excessively high staleness level is detrimental to the performance of the global model. In the global update, it may not be necessary to include these local updates.

Given these observations, we propose our FedAR algorithm, as detailed below.

4 FedAR Algorithm

FedAR is designed as a simple and effective algorithm by involving local updates of unavailable clients in the global model update on the server. In addition, given that a client’s unavailability leads to decreased contributions, we assign weights to different local updates accordingly. Our goal is to enhance FL performance by efficiently involving updates from all clients in the global model update. Specifically, FedAR consists of two components: *local update approximation* and *local update rectification*. In each round, the server sets a maximum waiting time for the local updates from all clients. When the maximum waiting time is reached, the server estimates the local updates that would be obtained from the unavailable clients. The weighted average over all the local updates is then performed to derive the global model for the next round. We describe our system in detail next.

Local Update Approximation. To approximate the local updates, the server maintains an update-matrix $\mathbf{G}[t] = [G_1[t]; \dots; G_i[t]; \dots; G_N[t]]$ saving its most recent observed local updates from all clients. Initially, $\mathbf{G}[0]$ is a zero matrix. In round t , $G_i[t]$ will only be replaced if the server obtains the client i ’s local update $w_{t,K}^i$. Otherwise, $G_i[t]$ will not change. Let $\mathcal{A}(t) \subset \mathcal{N}$ represent the set of available clients whose updates are successfully received by the server in round t . Mathematically, we have,

$$G_i[t] = \begin{cases} \frac{1}{\eta_t}(w_t - w_{t,K}^i) & \text{if client } i \in \mathcal{A}(t) \\ G_i[t-1] & \text{otherwise.} \end{cases} \quad (4)$$

FedAR uses $G_i[t] \in \mathbf{G}[t]$ as the estimates for local updates while deriving the global model. The global model is thus able to include the data distribution from the unavailable clients, which will help mitigate the bias against them.

Local Update Rectification. Figure 1 shows that stale local updates with different inactive rounds have various contributions to the global model, inspiring us to weigh local updates during the global update. We propose to assign weights to local updates based on the number of their inactive rounds, and by doing so, we expect to enhance the contributions from unavailable clients and further mitigate the bias.

Formally, the server maintains an update-array $\tau(t-1) = [\tau(1, t-1), \dots, \tau(i, t-1), \dots, \tau(N, t-1)]$ to record the number of inactive rounds for all clients. $\tau(0)$ is initialized as a zero array. In round t , if the update from client i is received, the server resets $\tau(i, t)$ to 0. Otherwise, the server increases $\tau(i, t-1)$ by 1 to get $\tau(i, t)$. We express $\tau(i, t)$ as:

$$\tau(i, t) = \begin{cases} 0 & \text{if client } i \in \mathcal{A}(t) \\ \tau(i, t-1) + 1 & \text{otherwise.} \end{cases} \quad (5)$$

Based on $\tau(i, t)$, we design a weight function $\psi_{i,t}$ to quantify the contribution from client i to the global update. The general expression of $\psi_{i,t}$ is given as:

$$\psi_{i,t} = \begin{cases} 0 & \text{if } \tau(i, t) \geq g(t) \\ \min([\tau(i, t) + 1]^\rho, 2) & \text{otherwise.} \end{cases} \quad (6)$$

If the client i is available at round t , i.e., $\tau(i, t) = 0$, we have $\psi_{i,t} = 1$, which aligns with FedAvg. We introduce $g(t)$ to prevent local updates with many inactive rounds from negatively impacting the global model and to remove such updates from the current global update. $g(t)$, as a function of round t , is different based on whether we are optimizing a convex loss function or a non-convex loss function. We will discuss it in more detail in Sect. 5 (Theoretical Analysis).

Since unavailable clients with more inactive rounds contribute less to the global update, we assign them higher weights to increase their contributions, as shown in Eq. (6). However, an extremely high weight $\psi_{i,t}$ will cause the unavailable clients to dominate the global model update, which would induce bias against available clients. We therefore introduce the hyperparameter $\rho \in [0, 1]$ in Eq. (6) to restrict the growth of $\psi_{i,t}$. We also set the maximum value of $\psi_{i,t}$ (ψ_{max}) to 2 to guarantee the convergence of FedAR. Please refer to the Appendix for more details on the convergence analysis.

Global Model Update. Clients arbitrarily participate in global model update in each round due to their limited resources and intermittent network connectivity. Hence, the server does not know the exact number of clients in advance; instead, it dynamically counts the clients that contribute to the global model update, i.e. those clients that are either available or unavailable but not too stale. Suppose there are N_t contributing clients in round t . With $G_i[t]$ and $\psi_{i,t}$, FedAR updates the global model as follows:

$$w_{t+1} = w_t - \frac{\eta_t}{N_t} \sum_{i=1}^N G_i[t] \psi_{i,t}. \quad (7)$$

Combined with Eq. (4), Eq. (7) ensures that the update matrix $G_i[t]$ always reflects the most recent client updates, while being able to reasonably consider the contributions of all clients when the global model is updated. In addition, although Eq. (7) seems to have all clients in the global model update, some clients do not get involved. They are either the clients that have never participated in FL, i.e., $G_i[t] = 0$, or the clients that have been inactive for many rounds, i.e., $\psi_{i,t} = 0$. Hence, Eq. (7) aligns with our idea of engaging only the contributing clients in the global model update.

Algorithm 1 shows the details of FedAR. We use the “temporary client set \mathcal{E} ” to include the clients that have ever participated in the global model update in Line 3. Initially, N_t is the number of clients in \mathcal{E} . When the client i has been inactive for many rounds, i.e., $\psi_{i,t} = 0$, it will be excluded from the global model update, i.e., $N_t = N_t - 1$ in Line 11. Ultimately, N_t counts the contributing clients as in Eq. (7).

Algorithm 1: FedAR

Input: initial w_0 , learning rate η_t , local step K , total round number T , total client number N **Output:** The derived global model w_T **Server executes:**

```

1: Initialize  $\psi_{i,1} = 1$ ,  $\tau(i, 1) = 0$ , and  $G_i[0] = 0$ ,  $\forall i$ , temporary client set  $\mathcal{E}$ 
2: for  $t=1,2, \dots, T$  do
3:    $\mathcal{E} \leftarrow \mathcal{E} \cup \{\text{new active client}\}$ ,  $N_t = |\mathcal{E}|$ .
4:   for  $i=1,2 \dots, N$  in parallel do
5:     if client  $i$  is available then
6:        $G_i[t] \leftarrow \text{DeviceUpdate}(i, w_t)$ 
7:        $\tau(i, t) = 0$ 
8:     else
9:        $\tau(i, t) = \tau(i, t) + 1$ 
10:    end if Calculate the  $\psi_{i,t}$  by Eq. (6)
11:    if  $\psi_{i,t} = 0$  then
12:       $N_t = N_t - 1$ 
13:    end if
14:  end for
15:   $w_{t+1} \leftarrow w_t - \frac{\eta_t}{N_t} \sum_{i=1}^N G_i[t] \psi_{i,t}$ 
16: end for

```

DeviceUpdate(i, w_t):

```

1:  $w_{t,0}^i \leftarrow w_t$ 
2: for  $k = 0, 1, \dots, K - 1$  do
3:    $w_{t,k+1}^i \leftarrow w_{t,k}^i - \eta_t \nabla f_i(w_{t,k}^i)$ 
4: end for
5: Return  $\frac{1}{\eta_t} (w_t - w_{t,K}^i)$ 

```

Regarding privacy enhancements in FL, FedAvg suggests that Differential Privacy (DP) can improve data privacy performance. However, our work is not primarily focused on privacy protection, and as such, an in-depth examination of this topic will not be included in our current research.

5 Theoretical Analysis of FedAR

In this section, we analyze the convergence of the proposed FedAR for convex and non-convex smooth loss functions.

5.1 Convex Loss Function

To analyze the convergence of FedAR for a convex loss function, we make the following assumptions regarding $f_i(w)$, $i = 1, 2, \dots, N$.

Assumption 1: L-smoothness. The loss function $f_i(w)$ is L-smooth. That is: for all $x, y \in \mathbb{R}$, $f(x) - f(y) \leq \langle \nabla f(y), x - y \rangle + \frac{L}{2} \|y - x\|^2$ with $L > 0$.

Assumption 2: μ -strong convex. The loss function $f_i(w)$ is μ -strong convex. That is: for all $x, y \in \mathbb{R}$, $f(x) - f(y) \geq \langle \nabla f(y), x - y \rangle + \frac{\mu}{2} \|y - x\|^2$ with $\mu > 0$.

Assumption 3: Variance bound. The variance of the unbiased estimator of $\nabla f_i(w)$ in round t is upper bounded, where $\mathbb{E}\{\|\tilde{\nabla} f_i(w) - \nabla f_i(w)\psi_{i,t}\|^2\} \leq \sigma^2$.

Theorem 1: Suppose the objective loss function $f_i(w)$ satisfies Assumptions 1 to 3, $\tau_{max} \leq g(t)$. By setting the learning rate $\eta_t = \frac{4}{\mu(t+a)}$ and constant $a = 100(\frac{L}{\mu})^{1.5}$, after T rounds, FedAR satisfies:

$$\begin{aligned} \mathbb{E}[f(\bar{w}_T)] - f(w_*) &= \mathcal{O}\left(\frac{\sigma^2(1 + \bar{\tau}_T)}{\mu K N T}\right) \\ &+ \mathcal{O}\left(\frac{F + \|w_1 - w_*\|^2 + \tau_{max}^2 L \sigma^2 N \psi_{max}}{K \mu^3 T^2}\right), \end{aligned}$$

where τ_{max} is the maximum number of $\tau(i, t)$ over all clients and rounds. $g(t) = t_0 + \frac{1}{b}t$ for a constant $t_0 > 0$ and $b > 2$, $F = LKND + L(K-1)^2 \cdot (DN^2 + \frac{\sigma^2}{K})$, $\bar{w}_T = \frac{\sum_{t=1}^T (t+a-1)(t+a-2)w_t}{W_T}$, $W_T = \sum_{t=1}^T (t+a-1)(t+a-2)$, $\bar{\tau}_T = \frac{1}{N(T-1)} \sum_{t=1}^{T-1} \sum_{i=1}^N \tau(i, t)$, and $D = \frac{1}{N} \sum_{i=1}^N \|\nabla f_i(w_*)\|^2$.

Remark 1: In Theorem 1, both the first and the second terms tend to zero as T increases, indicating that FedAR converges at the rate of $\mathcal{O}(1/T)$. The first term's convergence is related to the average inactive round number $\bar{\tau}_T$. We can find that too high a value of $\bar{\tau}_T$ will negatively impact convergence, which is consistent with our observation in Sect. 3.2 (Motivation). Also, convergence is adversely affected when most clients remain unavailable for a long time, i.e., a large $\bar{\tau}_T$. Besides, we observe that weight function ψ has a relatively negligible effect on the convergence rate. This can be attributed to our restriction on ψ in Eq. (6) to prevent it from becoming excessively large with an increase of τ . This is because a larger ψ could lead to the dominance of clients with more inactive rounds during the global model update.

5.2 Non-convex Loss Function

To analyze the convergence of FedAR for a non-convex smooth loss function, we make the following assumptions regarding $f_i(w), i = 1, 2, \dots, N$.

Assumption 4: Hessian Lipschitz. The Hessian of a twice differentiable function $f: \mathbb{R}^d \rightarrow \mathbb{R}$ is λ -Lipschitz continuous if $\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq \lambda \|x - y\|$ for all x, y .

Assumption 5: Gradient noise. The noise of the local stochastic gradients in round t is upper bounded by a constant δ : $\|\tilde{\nabla} f_i(w) - \nabla f_i(w)\| \leq \delta$.

Assumption 6: Gradient dissimilarity. $\exists \alpha > 0$ and $\beta_i > 0$: $\|\nabla f_i(w)\|^2 \leq \alpha \|\nabla f_i(w)\|^2 + \beta_i > 0$ and we define $\beta = \frac{1}{N} \sum_{i=1}^N \beta_i$.

Assumption 7: There exists a constant v_i such that $\tau(i, t) \leq v_i$ for $\forall i \in \mathcal{N}$, and define $\bar{v} = \frac{1}{N} \sum_{i=1}^N v_i$, $v_{max} = \max_{i \in \mathcal{N}} v_i$.

Theorem 2: Suppose Assumptions 1 to 7 hold, set learning rate $\eta = \sqrt{\frac{N}{KTL(1+\bar{v})}}$, $T \geq \max\{32\alpha LNK, 16LN^5K, \frac{8KNv_{max}^2(L^2+\lambda\delta N^2)}{L}\}$, and $\tau_{max} \leq g(t)$. After T rounds, FedAR satisfies:

$$\begin{aligned} \mathbb{E}[\|\nabla f(w_T)\|^2] &\leq \mathcal{O}\left(R\sqrt{\frac{L(1+\bar{v})}{TKN}}(f(w_1) - f^* + \sigma^2)\right. \\ &\quad \left. + \frac{\alpha\sigma^2\bar{v}LKN^2\psi_{max}}{T} + \frac{\sigma^2\lambda\delta N\psi_{max}}{LT} + \frac{F_1}{T}\right), \end{aligned}$$

where $g(t) = \frac{1}{4}\sqrt{\frac{L}{(L^2+\beta\lambda N)KN}} \times \max\{\sqrt{t}, \sqrt{t_0}\}$ for a constant $t_0 > 0$, $F_1 = (\alpha+1)(LKN\sigma^2\bar{v} + LKN\sigma v_{max}\sqrt{\beta + \frac{\sigma^2}{KN}}) + \frac{(L^2+\lambda\delta N^2)\sigma v_{max}}{L} + (K-1)(2\beta + \frac{\sigma^2}{K})$, and $R = \frac{8\psi_{max}^2}{4\psi_{max}^2-1}$.

Remark 2: In Theorem 2, the convergence of FedAR for a non-convex smooth loss function is dominated by the first term, which converges at the rate of $\mathcal{O}(\sqrt{1/T})$. This dominant term is mainly influenced by the initial error $f(w_1) - f^*$, the variance bound σ , and the average upper bound of inactive round number across clients \bar{v} . In addition, we observe that weight function ψ appears in the dominant term through the parameter R . Regardless of how ψ changes, the value of R tends towards a constant, and thus the impact produced by ψ is not significant. Compared to ψ , τ and N have a greater influence on the convergence via impacting the dominant term. We can draw similar conclusions as Theorem 1: as more clients continue to join the FL, more rounds are required to achieve convergence. Meanwhile, the fact that τ_{max} is a major variable affecting convergence aligns with our initial observations in Sect. 3.2 (Motivation); that is, the local updates with more inactive rounds negatively impact the global model's performance and further prevent the global model from converging. Thus, there must exist a critical value $g(t)$ as we express in Eq. (6) to exclude those clients from the global update to ensure the model convergence, i.e., the clients whose inactive round number exceeds $g(t)$ will not be considered.

Please refer to our Appendix for the proof of Theorem 1 and Theorem 2, as well as Remark 3 on Theorem 2.

6 Experiments and Evaluations

In this section, we evaluate the performance of FedAR by conducting extensive experiments on a desktop with the GeForce RTX 3060 graphic card.

6.1 Experimental Setup

System Settings. We conduct the FL experiments with one server and 100 clients. Let p_i denote the probability that client i is available during any given round. The availability of all clients is independent, with a minimum probability of p_{min} , indicating that the client availability probability varies from p_{min} to 1. This is a practical setting given that clients have their unique resource constraints and face distinct wireless environments. We examine both the challenging and mild client unavailability, where $p_{min} = 0.1$ and 0.5 , respectively. In summary, most clients are inactive for 5 - 20 rounds, with a few clients being inactive for more than 40 rounds.

Data and Model. We evaluate FedAR on three real-world datasets: MNIST [18], CIFAR-10 [17], and SVHN [24]. To ensure non-IID data distribution among all clients, we assume all datasets to be evenly distributed on all clients, and each client to contain only two classes of data. We use the logistic regression for MNIST, Lenet-5 for CIFAR-10, and Resnet-18 for SVHN as the local models. We set all experiments' initial local learning rate as $\eta_0 = 0.1$, local training step as $K = 5$, local batch size as 64, and hyperparameter as $\rho = 0.1$. We set weight decay as 0.001 during the local SGD.

Baselines. We compare FedAR with recent FL baselines: (1) *MIFA* [10]. It assigns the same weight to both available and unavailable clients; (2) *FedVARP* [14]. It assigns higher weights to available clients' updates, while the weights for unavailable clients remain unchanged; (3) *FedAvg-IS*. It engages only available clients in global update using the FedAvg algorithm. The local updates are weighted by clients' availability probabilities; (4) *FedAvg (S=50)*. It involves at most half of available clients in the global update with the FedAvg algorithm. Given 100 clients, at most 50 clients join the global update; and (5) *Scaffold* [16]. It is a FL algorithm designed to improve the quality of global model updates by applying personalized control variate adjustments to each client; it does not consider client unavailability.

6.2 Experimental Results

¹ **Overall Convergence Performance.** We evaluate the convergence performance of FedAR on different datasets in both the challenging and mild settings in Fig. 2. We find that FedAR has a similar convergence speed as FedAvg-IS, MIFA, and FedVARP. Notably, on CIFAR and SVHN datasets, the convergence speed of FedAR is markedly superior to that of Scaffold. This observation is consistent with our theoretical analysis that our designed weight function ψ has negligible negative impacts on convergence.

When $p_{min} = 0.1$, Fig. 2a shows that FedAR on CIFAR-10 reduces the training loss to 1.5 and attains the highest test accuracy of 44%, an enhancement

¹ For clear observation, we recommend viewing all figures about experimental results in color.

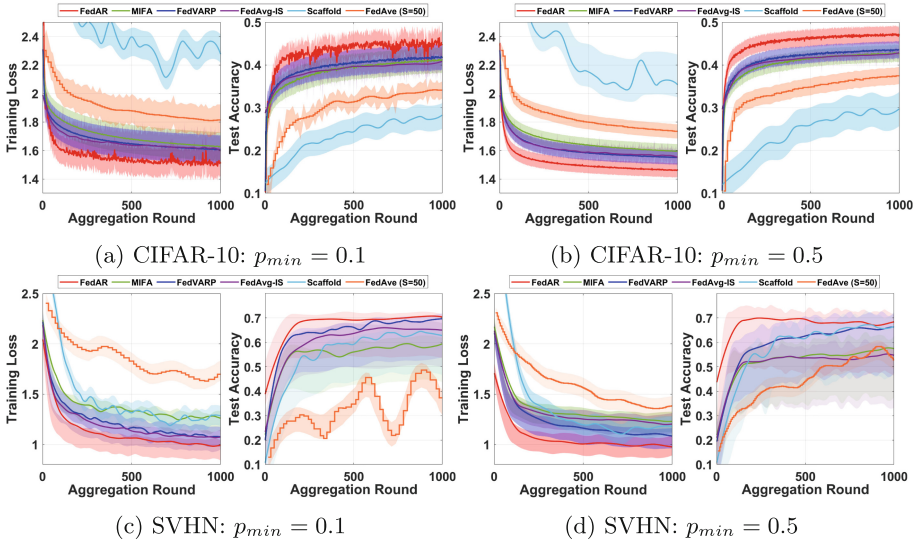


Fig. 2. Convergence, training loss and test accuracy performance

of over 3% compared to baseline algorithms. Figure 2c shows that FedAR is the only algorithm achieving a training loss below 1 and a test accuracy over 70% on SVHN. When more clients are available, i.e., $p_{min} = 0.5$, FedAR in Fig. 2b greatly boosts the test accuracy to 46% on CIFAR-10. Additionally, we find that FedAR consistently reaches a test accuracy of around 70% in most training rounds on SVHN, and outperforms all the baselines.

Table 2. P-Value analysis of FedAR performance

Dataset	Baselines				
	FedVARP	MIFA	Scaffold	FedAve(s=50)	FedAve-IS
Cifar10;p=0.1	$1.15 \cdot 10^{-163}$	$1.15 \cdot 10^{-194}$	$1.87 \cdot 10^{-194}$	$1.53 \cdot 10^{-110}$	$4.36 \cdot 10^{-180}$
Cifar10;p=0.5	0.0	0.0	$1.38 \cdot 10^{-246}$	$3.46 \cdot 10^{-316}$	$2.02 \cdot 10^{-285}$
SVHN;p=0.1	0.00029	$1.49 \cdot 10^{-54}$	$5.72 \cdot 10^{-35}$	$1.63 \cdot 10^{-60}$	$4.77 \cdot 10^{-45}$
SVHN;p=0.5	$1.34 \cdot 10^{-52}$	$5.91 \cdot 10^{-111}$	$1.71 \cdot 10^{-81}$	$2.25 \cdot 10^{-77}$	$1.96 \cdot 10^{-116}$

We also conduct statistical tests of significance using paired t-test to assess whether the improvement in performance achieved by FedAR is statistically significant. We compare the test accuracy of FedAR against each of the baselines individually for both CIFAR-10 and SVHN, and for $p_{min} = 0.1$ and $p_{min} = 0.5$. The results are illustrated in Table 2; each entry in the table denotes the p-value of the paired t-test between FedAR and the corresponding baseline (denoted in

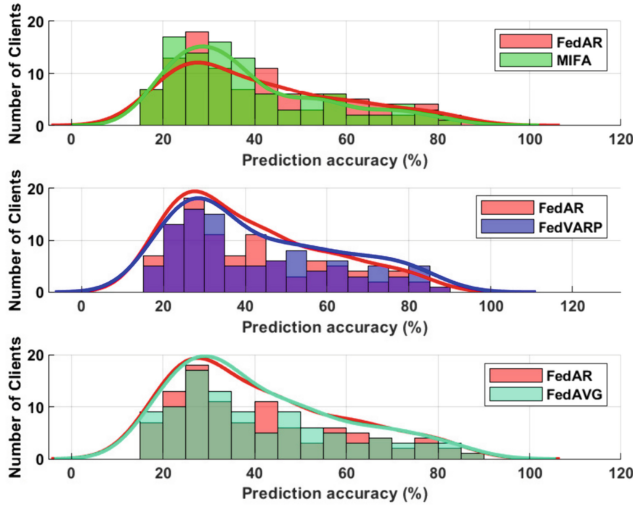


Fig. 3. Accuracy distributions

the columns) for the corresponding dataset (denoted in the rows). From the table, we find that the improvement in performance achieved by FedAR is statistically significant ($p < 0.001$) compared to all the baselines, consistently for both the datasets and both values of p_{min} . These results further corroborate the promise and potential of FedAR. FedAR also shows superior performance on the MNIST dataset with lower training losses and higher test accuracy upon convergence, as elaborated in the Appendix.

Bias Mitigation. We study the bias mitigation performance of FedAR on CIFAR-10 in the challenging setting, where $p_{min} = 0.1$. Specifically, the global model is used to make predictions for each client after convergence, and we study the consistency of the prediction accuracies across all clients. In addition to MIFA and FedVARP, we compare FedAR with the ideal situation of FedAvg, where all the clients are continuous available throughout the entire training process.

Table 3. Accuracy statistics

ALGO	Mean (%)	Var	Worst 10% (%)	Best 10% (%)
FedAR	40.9±18.1	325	20.8±4.5	67.7±8.7
MIFA	34.0±13.6	182.5	19.7±4.2	53.8±8.8
FedVARP	41.3±20.7	432.5	19.4±2.6	73.9±11.6
FedAvg	41.0±18.0	321.6	21.2±4.3	69.5±12.6

Table 3 depicts the statistics (mean \pm std and variance) of the prediction accuracy across clients. In addition, we record the prediction accuracy of the worst 10% clients and the best 10% clients, denoted by “Worst 10%” and “Best 10%” respectively [19]. From Table 3, we observe that the “Mean”, “Worst 10%”, and “Best 10%” prediction accuracy of FedAR closely align with FedAvg. This suggests that the performance of FedAR is comparable to the ideal situation of full client availability. Furthermore, FedAR achieves an average accuracy approximately 6% higher than MIFA, which requires all the clients to participate in the first training round. Although the average prediction accuracy of FedVARP is marginally higher than FedAR, it exhibits a considerably higher variance of 432.5, over 100 more than FedAR. Such a high variance indicates a significant variation in prediction accuracy across different clients in FedVARP.

To more intuitively evaluate the bias mitigation performance, we visually depict the distribution of the number of clients and its Probability Density Function (PDF) of prediction accuracy in Fig. 3. Compared to MIFA, FedAR enables a larger number of clients to achieve a prediction accuracy of 40% or higher. Additionally, within the accuracy interval between 25% and 65%, the PDF curve of FedAR surpasses that of FedVARP. Outside this interval, PDF curve of FedAR falls below that of FedVARP. This pattern indicates that the prediction accuracies in FedAR are more centralized around the mean value (40%). This explains the high variance values of FedVARP in Table 3. Furthermore, the PDF curve of FedAR almost coincides with that of FedAvg. This indicates that even under the challenging client unavailability ($p_{min} = 0.1$), FedAR maintains prediction accuracy distribution similar to the ideal full client availability situation. Both Table. 3 and Fig. 3 confirm that FedAR can effectively mitigate the bias despite severe client unavailability.

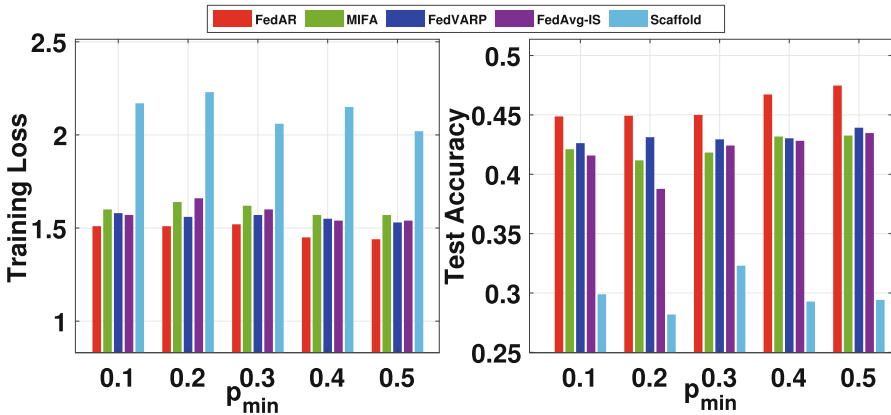


Fig. 4. Effect of p_{min}

Hyperparameter Evaluation. We study the effect of hyperparameters under the challenging setting of $p_{min} = 0.1$ on CIFAR-10. Please refer to our Appendix for the performance analysis on SVHN and the evaluation for ρ value in Eq. (6).

Minimum Client Participation Probability p_{min} . We evaluate FedAR under various client participation probability, i.e., p_{min} spanning from 0.1 to 0.5. We exclude FedAvg ($S=50$) due to its notably inferior performance compared to other baselines. As shown in Fig. 4, FedAR consistently outperforms all the baselines for every p_{min} . Additionally, we note a marginal enhancement in FedAR’s performance as p_{min} increases. When $p_{min} = 0.5$, FedAR achieves an accuracy of 47% whereas the accuracy of all the baselines is below 45%. This is because a higher participation probability reduces the average number of inactive rounds, thus positively impacting the FL performance.

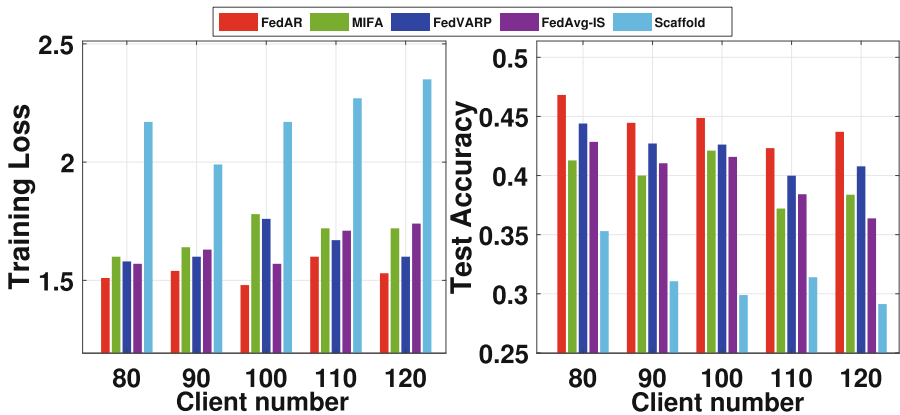


Fig. 5. Effect of N

Number of Clients N . We evaluate FedAR with varying numbers of clients from 80 to 120. As shown in Fig. 5, as N increases, all algorithms show an increasing trend in training loss and a decreasing trend in test accuracy, among which FedAR achieves the best performance. Specifically, FedAR marginally increases the training loss only from 1.5 to 1.6 when N is increased from 80 to 120. The lowest accuracy of FedAR is 43% in the case of $N = 110$. In contrast, the performance of other baselines degrades significantly with the increase in the number of clients. Except for FedVARP, the test accuracy of the rest of the baselines has fallen below 40%. The surge in the number of clients inherently leads to a rise in unavailable clients, posing challenges across all algorithms. This suggests that FedAR is more adept at handling a large number of clients, making it ideal for large-scale FL, especially in the presence of significant client unavailability.

7 Conclusion

In this paper, we propose a novel FL algorithm, FedAR, to address the client unavailability. We found that clients with different numbers of inactive rounds have diverse contributions to the current global update. Based on this observation, we design a novel weighting strategy that not only engages the unavailable clients in the global model update, but also quantifies their contributions based on the number of their inactive rounds. We theoretically prove the convergence of FedAR for both convex and non-convex smooth loss functions with non-IID data across clients. Our experimental results demonstrate that FedAR significantly outperforms competing FL baselines FedAvg, MIFA, FedVARP and Scaffold with respect to the training loss, the test accuracy, and the bias mitigation. FedAR further demonstrates remarkable performance and surpasses those baselines in large-scale FL with severe client unavailability. As part of future work, we will study the performance of FedAR under other practical challenges such as missing data and class imbalance across clients.

Acknowledgment. The work of X. Zhang is partially supported by the National Science Foundation under Grant Number: CCF-2312617. The work of S. Chakraborty is partially supported by the National Science Foundation under Grant Number: IIS-2143424 (NSF CAREER Award).

References



1. Abdelmoniem, A.M., Sahu, A.N., Canini, M., Fahmy, S.A.: Refl: Resource-efficient federated learning. In: Proceedings of the Eighteenth European Conference on Computer Systems, pp. 215–232 (2023)
2. Bonawitz, K., et al.: Towards federated learning at scale: system design. *Proc. Mach. Learn. Syst.* **1**, 374–388 (2019)
3. Briggs, C., Fan, Z., Andras, P.: Federated learning with hierarchical clustering of local updates to improve training on non-iid data. In: 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1–9. IEEE (2020)
4. Brisimi, T.S., Chen, R., Mela, T., Olshevsky, A., Paschalidis, I.C., Shi, W.: Federated learning of predictive models from federated electronic health records. *Int. J. Med. Informatics* **112**, 59–67 (2018)
5. Chen, S., Li, B.: Towards optimal multi-modal federated learning on non-iid data with hierarchical gradient blending. In: IEEE INFOCOM 2022-IEEE Conference on Computer Communications, pp. 1469–1478. IEEE (2022)
6. Cho, Y.J., Gupta, S., Joshi, G., Yağın, O.: Bandit-based communication-efficient client selection strategies for federated learning. In: 2020 54th Asilomar Conference on Signals, Systems, and Computers, pp. 1066–1069. IEEE (2020)
7. Cho, Y.J., Wang, J., Joshi, G.: Towards understanding biased client selection in federated learning. In: International Conference on Artificial Intelligence and Statistics, pp. 10351–10375. PMLR (2022)
8. Fraboni, Y., Vidal, R., Kameni, L., Lorenzi, M.: Clustered sampling: Low-variance and improved representativity for clients selection in federated learning. In: International Conference on Machine Learning, pp. 3407–3416. PMLR (2021)

9. Ghosh, A., Chung, J., Yin, D., Ramchandran, K.: An efficient framework for clustered federated learning. *Adv. Neural. Inf. Process. Syst.* **33**, 19586–19597 (2020)
10. Gu, X., Huang, K., Zhang, J., Huang, L.: Fast federated learning in the presence of arbitrary device unavailability. *Adv. Neural. Inf. Process. Syst.* **34**, 12052–12064 (2021)
11. al Hard, A., et al.: Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604* (2018)
12. Horvath, S., Laskaridis, S., Almeida, M., Leontiadis, I., Venieris, S., Lane, N.: Fjord: fair and accurate federated learning under heterogeneous targets with ordered dropout. *Adv. Neural. Inf. Process. Syst.* **34**, 12876–12889 (2021)
13. Huang, W., Ye, M., Du, B.: Learn from others and be yourself in heterogeneous federated learning. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10143–10153 (2022)
14. Jhunjhunwala, D., SHARMA, P., Nagarkatti, A., Joshi, G.: Fedvarp: tackling the variance due to partial client participation in federated learning. In: *The 38th Conference on Uncertainty in Artificial Intelligence* (2022)
15. Kairouz, P., et al.: Advances and open problems in federated learning. *Foundat. Trends Mach. Learn.* **14**(1–2), 1–210 (2021)
16. Karimireddy, S.P., Kale, S., Mohri, M., Reddi, S., Stich, S., Suresh, A.T.: Scaffold: stochastic controlled averaging for federated learning. In: *International Conference on Machine Learning*, pp. 5132–5143. PMLR (2020)
17. Krizhevsky, A.: Learning multiple layers of features from tiny images. University of Toronto (May 2012)
18. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. *Proc. IEEE* **86**(11), 2278–2324 (1998). <https://doi.org/10.1109/5.726791>
19. Li, T., Sanjabi, M., Beirami, A., Smith, V.: Fair resource allocation in federated learning. *arXiv preprint arXiv:1905.10497* (2019)
20. Luo, B., Xiao, W., Wang, S., Huang, J., Tassiulas, L.: Tackling system and statistical heterogeneity for federated learning with adaptive client sampling. In: *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pp. 1739–1748. IEEE (2022)
21. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: *Artificial Intelligence and Statistics*, pp. 1273–1282. PMLR (2017)
22. Mendieta, M., Yang, T., Wang, P., Lee, M., Ding, Z., Chen, C.: Local learning matters: Rethinking data heterogeneity in federated learning. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8397–8406 (2022)
23. Mohri, M., Sivek, G., Suresh, A.T.: Agnostic federated learning. In: *International Conference on Machine Learning*, pp. 4615–4625. PMLR (2019)
24. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning. In: *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011* (2011). http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf
25. Shapley, L.S., et al.: A value for n-person games (1953)
26. Shu, J., Zhang, W., Zhou, Y., Cheng, Z., Yang, L.T.: Flas: computation and communication efficient federated learning via adaptive sampling. *IEEE Trans. Netw. Sci. Eng.* **9**(4), 2003–2014 (2021)
27. Soltani, B., Zhou, Y., Haghighi, V., Lui, J.: A survey of federated evaluation in federated learning. *arXiv preprint arXiv:2305.08070* (2023)

28. , Song, T., Tong, Y., Wei, S.: Profit allocation for federated learning. In: 2019 IEEE International Conference on Big Data (Big Data), pp. 2577–2586. IEEE (2019)
29. Wang, G., Dang, C.X., Zhou, Z.: Measure contribution of participants in federated learning. In: 2019 IEEE International Conference on Big Data (Big Data), pp. 2597–2604. IEEE (2019)
30. Wang, S., Ji, M.: A unified analysis of federated learning with arbitrary client participation. arXiv preprint [arXiv:2205.13648](https://arxiv.org/abs/2205.13648) (2022)
31. Wang, Z., Fan, X., Qi, J., Jin, H., Yang, P., Shen, S., Wang, C.: Fedgs: federated graph-based sampling with arbitrary client availability. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, pp. 10271–10278 (2023)
32. Yan, Y., et al.: Federated optimization under intermittent client availability. *INFORMS J. Comput.* **36**(1), 185–202 (2024)
33. Yu, S., Lin, C., Zhang, X., Guo, L.: Defending against cross-technology jamming in heterogeneous IoT systems. In: IEEE 42nd International Conference on Distributed Computing Systems (ICDCS), pp. 702–712 (2022)
34. Yu, S., Zhang, X., Huang, P., Guo, L., Cheng, L., Wang, K.: AuthCTC: defending against waveform emulation attack in heterogeneous IoT environments. In: Proceedings of the 15th ACM Asia Conference on Computer and Communications Security, pp. 20–32 (2020)
35. Zhang, X., Guo, L., Li, M., Fang, Y.: Social-enabled data offloading via mobile participation—a game-theoretical approach. In: 2016 IEEE Global Communications Conference (GLOBECOM), pp. 1–6 (2016)
36. Zhang, X., Huang, P., Guo, L., Fang, Y.: Hide and seek: Waveform emulation attack and defense in cross-technology communication. In: 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), pp. 1117–1126 (2019)
37. Zhang, X., Yu, S., Zhou, H., Huang, P., Guo, L., Li, M.: Signal emulation attack and defense for smart home iot. In: *IEEE Trans. Dependable Sec. Comput.* (2022)
38. Zhou, H., Wang, S., Jiang, C., Zhang, X., Guo, L., Yuan, Y.: Waste not, want not: service migration-assisted federated intelligence for multi-modality mobile edge computing. In: Proceedings of the Twenty-fourth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing, pp.211–220 (2023)
39. Zhou, H., Yu, S., Zhang, X., Guo, L., Lorenzo, B.: DQN-based QoE Enhancement for Data Collection in Heterogeneous IoT Network. In: 2022 IEEE 19th International Conference on Mobile Ad Hoc and Smart Systems (MASS), pp.188–194 (2022)
40. Zhou, P., Xu, H., Lee, L.H., Fang, P., Hui, P.: Are you left out? an efficient and fair federated learning for personalized profiles on wearable devices of inferior networking conditions. *Proc. ACM on Interactive, Mobile, Wearable Ubiquitous Technol.* **6**(2), 1–25 (2022)
41. Zhu, L., Lin, H., Lu, Y., Lin, Y., Han, S.: Delayed gradient averaging: tolerate the communication latency for federated learning. *Adv. Neural. Inf. Process. Syst.* **34**, 29995–30007 (2021)



Selecting from Multiple Strategies Improves the Foreseeable Reasoning of Tool-Augmented Large Language Models

Yongchao Wu^(✉)  and Aron Henriksson 

Stockholm University, NOD-huset, Borgarfjordsgatan 12, 16455 Stockholm, Sweden
{yongchao.wu, aronhen}@dsv.su.se

Abstract. Large language models (LLMs) can be augmented by interacting with external tools and knowledge bases, allowing them to overcome some of their known limitations, such as not having access to up-to-date information or struggling to solve math problems, thereby going beyond the knowledge and capabilities obtained during pre-training. Recent prompting techniques have enabled tool-augmented LLMs to combine reasoning and action to solve complex problems with the help of tools. This is essential for allowing LLMs to strategically determine the timing and nature of tool-calling actions in order to enhance their decision-making process and improve their outputs. However, the reliance of current prompting techniques on a single reasoning path or their limited ability to adjust plans within that path can adversely impact the performance of tool-augmented LLMs. In this paper, we introduce a novel prompting method, whereby an LLM agent selects and executes one among multiple candidate strategies. We assess the effectiveness of our method on three question answering datasets, on which it outperforms state-of-the-art methods like ReWOO, while also being a competitive and more cost-efficient alternative to ReAct. We also investigate the impact of selecting a reasoning trajectory from different strategy pool sizes, further highlighting the risks in only considering a single strategy.

Keywords: Large language models · Tool-augmented language models · Chain-of-thought prompting · Question answering

1 Introduction

The advanced capabilities of large language models (LLMs) [9] have extended their utility beyond mere language generation tasks, paving the way for their application as autonomous agents to make decisions across diverse environments [4, 8]. Reasoning is crucial for autonomous agents in their the decision-making processes, particularly in scenarios involving tool usage to determine the appropriate timing and selection of tools for completing complex tasks. The Chain-of-Thought (CoT) prompting paradigm has become a prominent method for

enhancing the reasoning abilities of LLMs. By prompting LLMs to generate intermediate reasoning steps prior to delivering a final answer, CoT has significantly improved their performance in tasks requiring arithmetic, commonsense, and symbolic reasoning [13]. Recent prompting strategies like ReAct [17] and ReWOO [14] are two state-of-the-art (SOTA) methods that apply the CoT paradigm with tool-augmented LLMs (TA-LLMs). When tackling tasks with TA-LLMs, ReAct enhances the dynamism of CoT by generating reasoning thoughts based on environmental observations (observation-dependent reasoning) before deciding which tool to utilize in each step. Conversely, ReWOO transforms the intermediate reasoning steps of CoT into actionable plans (foreseeable reasoning), strategically determining the sequence of tool usage. The development of TA-LLMs has allowed some of the limitations of traditional LLMs to be addressed, such as restricted access to knowledge obtained during pre-training [6] and a tendency to hallucinate [5]. Unlike Retrieval-Augmented Generation (RAG) [7], which aims to reduce LLM hallucination by acquiring knowledge from a static external knowledge base that is indexed offline, TA-LLMs operate entirely online. They leverage external tools, such as software APIs, to access up-to-date information, offering greater flexibility by enabling the resolution of more complex problems through sophisticated API calls.

A limitation of current prompting techniques for TA-LLMs is that they either rely on a single reasoning path or can only adjust plans within the same reasoning trajectory. Since no individual reasoning path is infallible and can result in incorrect model output, not taking into consideration multiple reasoning trajectories may impede the performance of TA-LLMs. Hence, we introduce a novel prompting method named ReMSV (**R**easoning with **M**ultiple **S**trategies and **V**oting) that generates and considers multiple reasoning trajectories before deciding on a course of action, including which tools to use. Specifically, building on the ReWOO framework, we design a process that incorporates the roles of *Director*, *Voter*, *Worker*, and *Solver* to generate multiple candidate reasoning trajectories, selects the trajectory with the most votes, and then executes the chosen reasoning trajectory to solve a given task. To assess the effectiveness of our method, we use several benchmark datasets, namely *HotpotQA* [15], *GSM4K* [2], and *PhysicsQA* [1]. ReMSV obtains a superior performance, surpassing ReAct in *HotpotQA* and *PhysicsQA*, while consistently outperforming ReWOO across all three benchmarks. Significantly, on the *HotpotQA* dataset, ReMSV achieves a relative improvement of 34.5% and 9.1% in comparison to ReAct and ReWOO, respectively. In addition, compared to ReAct, ReMSV provides a more cost-efficient alternative that consumes significantly fewer tokens, e.g., 200% fewer in the *HotpotQA* benchmark. We also carry out experiments to investigate the influence of considering multiple reasoning trajectories, underscoring the efficacy of our approach. To summarize, our key contributions are as follows:

- We present a prompting method called ReMSV that considers multiple reasoning trajectories. By selecting one among multiple sampled strategies through voting, our method not only consistently surpasses ReWOO but also generally

outperforms **ReAct**, showcasing its effectiveness compared to current SOTA prompting strategies.

- We further analyze the proposed multi-strategy mechanism, both mathematically and empirically, showing that although **ReMSV** comes with a slightly higher cost compared to **ReWOO**, it is more cost-effective in terms of token consumption than **ReAct**, making it a viable alternative that improves performance at a slightly higher cost compared to **ReWOO**.

2 Related Work

Historically, the study of autonomous agents that can use external tools has primarily centered on reinforcement learning techniques. For example, **WebGPT** [8], which was designed to interact with web browsers to respond to complex questions, relies heavily on costly human feedback for its reinforcement learning process. Similarly, **SimpleTOD** [4], a task-focused dialogue system, requires extensive datasets derived from human feedback for its policy training. Prior to the introduction of **CoT** [13], reasoning capability was viewed as a primary constraint of LLMs that could not be addressed merely by scaling and enlarging the model size [10]. With few-shot in-context learning, **CoT** unlocks the reasoning capabilities of LLMs by incorporating sequential intermediary reasoning steps before producing the final result. Recently proposed prompting strategies [11, 14, 17] combine the reasoning and action capabilities of LLMs by converting the static intermediary reasoning phases into actionable plans that engage with the environment. Among these prompting strategies, **ReAct** [17] and **ReWOO** [14] are two approaches that achieve SOTA performance results in challenges that require several reasoning steps to conclude a final response. Specifically, **ReAct** [17] introduced an (*Obs*, *Thought*, *Action*) prompting technique, which consistently produces a *Thought* (verbal reasoning) based on environmental observations prior to executing task-specific actions. This method seamlessly connects the reasoning of an LLM with its actions, allowing it to interweave reasoning trajectories with tool-calling actions. Despite its remarkable performance, **ReAct** continuously generates observations and loops them back into the LLMs as context to generate the next (thought, action) pair. This leads to high token consumption, which can significantly increase cost and energy consumption. **ReWOO** utilizes the foreseeable reasoning capabilities of TA-LLMs to sketch a strategy for decomposing the problem into reasoning and actionable plans, without needing to resort to explicit observations. By compartmentalizing step-wise reasoning and tool-calling actions into distinct modules, it not only matches the performance of **ReAct** but also boasts a 5-fold increase in token efficiency.

A common limitation of both **ReAct** and **ReWOO** is their inability to account for multiple reasoning trajectories. Researchers have also explored the use of multi-reasoning trajectories with LLMs, such as self-consistency (**SC**) [12] and Tree-of-Thoughts (**ToT**) [16]. However, these methods were originally proposed to improve **CoT** in scenarios not involving tool calling. Moreover, these prompting strategies are computationally expensive and result in a substantial cost due

to the necessity of executing each sampled reasoning trajectory. Our proposed method aims to combine the advantages of considering multiple reasoning trajectories with a cost-effective approach that avoids executing every trajectory. To achieve this, we are introducing a new method named **ReMSV**.

3 Methods

In this section, we formalize the problem and introduce the main components of the proposed method, **ReMSV**. Unlike **ReAct** and **ReWOO**, **ReMSV** considers multiple reasoning trajectories before deciding on a course of action involving tool-calling operations. Specifically, building on the **ReWOO** framework, we formulate a procedure that integrates the functions of *Director*, *Voter*, *Worker*, and *Solver*. Specifically, *Director* creates several potential reasoning trajectories/strategies¹, *Voter* reflects on which strategy is the best before deciding on which one to execute, and *Worker* subsequently executes the selected strategy. Compared to our approach, **ReWOO** only considers a single reasoning path, which could result in an erroneous strategy formulation.

Figure 1 illustrates the differences between **ReAct**, **ReWOO**, and **ReMSV**. Moreover, unlike **SC** that executes all (n) strategies and combines the outcomes through aggregation, leading to costs n times higher than **ReWOO**, our method solely executes the voted strategy after reflection, resulting in a more cost-efficient approach. Please note that **SC** has not been applied to tool-calling scenarios in any previous studies, primarily due to its high cost. In this study, we only estimate the cost and performance of **SC** in Sects. 6.1 and 6.2.

3.1 Problem Formulation

We explore the use of an LLM as an autonomous agent for handling tasks in text-based settings using external tools. Initially, the agent is equipped with the permissible actions \mathcal{A} in the environment and a textual task directive $g \in \mathcal{G}$ from the task space \mathcal{G} . To accomplish the task g , the LLM navigates through a generated sequence of policies $[p_0, p_1, \dots, p_n]$ to execute tool-calls. At time step t , the agent adheres to policy $p = \pi(a_t|c_t)$ to perform an action, where c_t is a trajectory context which contains observations \mathcal{O} from the environment or evidence \mathcal{E} from the tool-call executions in prior steps. Current SOTA methods, such as **ReAct**, first generate an initial policy based on the initial observation of the environment $p_0 = \pi(a_1|g, o_1)$, and formulate subsequent action policies by reasoning over the environmental observation at each step in an ad-hoc fashion, i.e. it lacks an overarching strategy for solving the problem. However, when scrutinizing how humans navigate and solve complex problems, we frequently choose a course of action from several viable alternative strategies. We design a similar framework which consists of the essential components *Director*, *Voter*, *Worker and Solver*, described below and illustrated in Fig. 2.

¹ In this paper, we use the terms reasoning trajectory and strategy interchangeably.

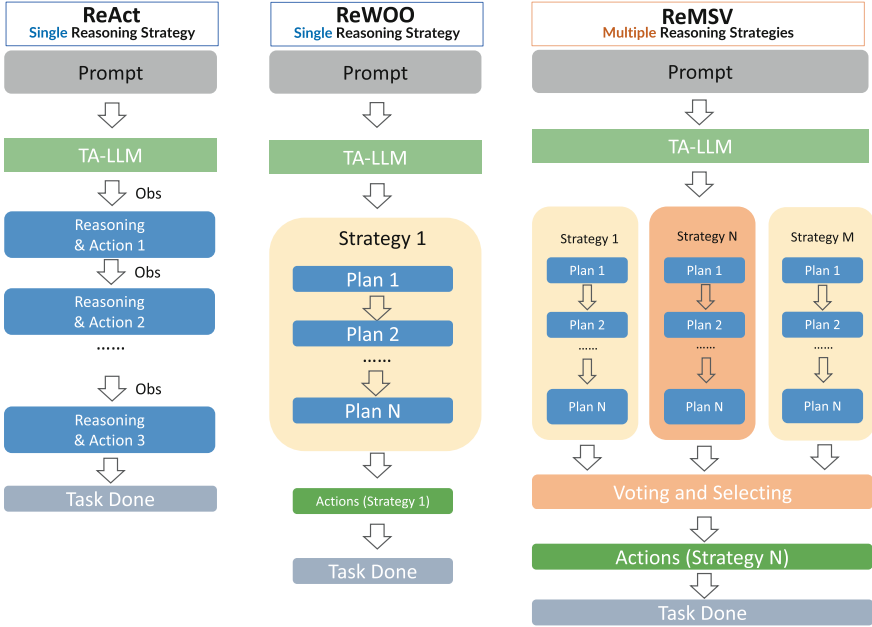


Fig. 1. A comparison of prompting techniques based on a single reasoning trajectory (ReAct, ReWOO) and multiple reasoning trajectories (ReMSV).

3.2 Method Components

Below, we describe the main components of the proposed method ReMSV, namely *Director*, *Voter*, *Worker*, and *Solver*.

Director: Strategy Making and Sampling. To produce n strategies, an LLM initially receives several pre-defined explicit CoT exemplars, illustrating the structure of a strategy $[(Plan_1, \mathcal{E}_1), (Plan_2, \mathcal{E}_2), \dots, (Plan_n, \mathcal{E}_n)]$, which involves a series of step-by-step plans to tackle the problem. Specifically, for each step $(Plan_t, \mathcal{E}_t)$ of a strategy, it contains an instruction $Plan_t$ indicating the corresponding action a_t as well as a marking token (\mathcal{E}_t) to store the execution result of action a_t , which, in turn, provides context for subsequent steps. Then, akin to the SC [12] method, we sample a variety of candidate outputs from the LLM, thereby creating a diverse collection of potential strategies.

Voter: Strategy Evaluation and Voting. Motivated by ToT [16], we incorporate a *Voter* component to assess various strategies created by the *Director*, utilizing a straightforward zero-shot voting prompt (“Analyze the strategies and conclude the most promising one to solve the problem”). We sample n candidates from the voting outputs and choose the strategy with the highest number of votes as the concluding strategy to be executed. We opt for multiple rounds (5)

of voting instead of a single vote to enhance the robustness and generalizability of the voting procedure.

Worker and Solver: Strategy Execution. We use the ReWOO framework to implement the selected strategy following the voting process. Specifically, *Workers* adhere to the plans at each step ($Plan_t, \mathcal{E}_t$) in the strategy to execute tool-calls from the action space \mathcal{A} , and the execution result will substitute the marking token \mathcal{E}_t , acting as observations or evidence. The *Solver* compiles all plans and observations to produce the final output.

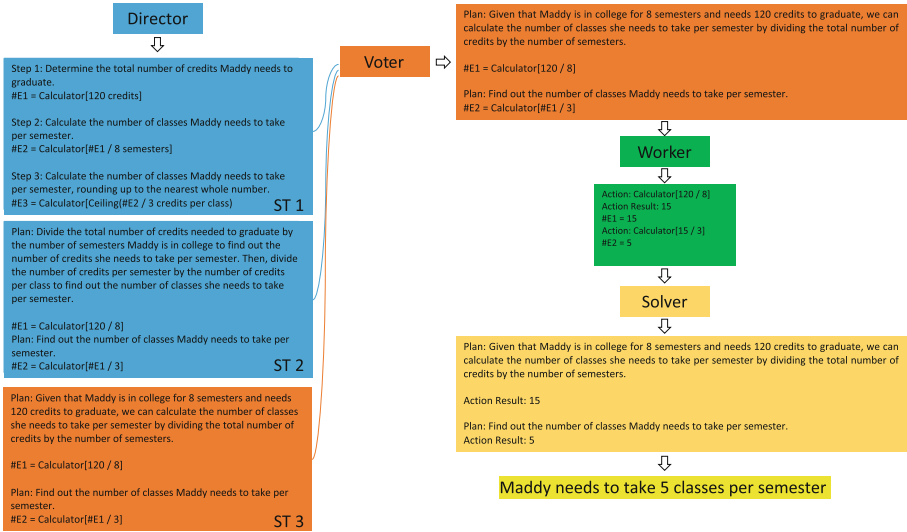


Fig. 2. The workflow of ReMSV. The question is: *Maddy is in college for 8 semesters. She needs 120 credits to graduate. If each class is 3 credits, how many classes does she need to take per semester?* The *Director* generates multiple candidate strategies, from which the *Voter* selects one, the *Worker* executes the tool-calls, and the *Solver* compiles all plans and observations to produce the final output.

4 Token Consumption Estimation

In order to compare our proposed method, ReMSV, to ReAct and ReWOO in terms of cost efficiency, we conduct a mathematical analysis to estimate their token consumption. Let $N(p)$ denote the tokens for a text p . Suppose a TA-LLM addressing a question \mathcal{Q} requires k steps of reasoning to arrive at the correct answer. Initially, the TA-LLM is provided with a context prompt \mathcal{C} and several in-context learning examples \mathcal{I} . In the process of using the ReAct prompting approach, the TA-LLM persistently produces observations \mathcal{O} , which are then

fed back into the TA-LLM as context to create subsequent pairs of thoughts (\mathcal{T}) and actions (\mathcal{A}). The token consumption of **ReAct** can be calculated as:

$$\begin{aligned} \text{Token}^{\text{ReAct}} &= N(\mathcal{C} + \mathcal{I} + \mathcal{Q}) + \sum_{i=1}^{k-1} N(\mathcal{C} + \mathcal{I} + \mathcal{Q} + \sum_{j=1}^i (\mathcal{T}_j + \mathcal{A}_j + \mathcal{O}_j)) \quad (1) \\ &= kN(\mathcal{C} + \mathcal{I} + \mathcal{Q}) + \sum_{i=1}^{k-1} (k-i)N(\mathcal{T}_i + \mathcal{A}_i + \mathcal{O}_i) \quad (2) \end{aligned}$$

ReW00 leverages the foreseeable reasoning abilities of TA-LLMs to outline a strategy for breaking down the problem into actionable plans and evidences ($\text{Plan}_n, \mathcal{E}_n$), eliminating the need for explicit observations. The token consumption of **ReW00** is:

$$\begin{aligned} \text{Token}^{\text{ReW00}} &= N(\mathcal{C}_{\text{planner}} + \mathcal{I} + \mathcal{Q}) + N(\mathcal{C}_{\text{solver}} + \mathcal{Q} + \sum_{i=1}^k (P_i + E_i)) \quad (3) \\ &\approx 2N(\mathcal{C} + \mathcal{Q}) + N\mathcal{I} + \sum_{i=1}^k N(P_i + E_i) \quad (4) \end{aligned}$$

ReMSV adds two additional components, *Director* and *Voter*, to sample and vote for m strategies. The token consumption of **ReMSV** can be calculated as:

$$\begin{aligned} \text{Token}^{\text{ReMSV}} &= N(\mathcal{C}_{\text{Director}} + \mathcal{I} + \mathcal{Q}) + m \sum_{i=1}^k N(P_i + E_i) + N(\mathcal{C}_{\text{Voter}} + \mathcal{Q}) \quad (5) \\ &\quad + N(\mathcal{C}_{\text{solver}} + \mathcal{Q} + \sum_{i=1}^k (P_i + E_i)) \quad (6) \\ &\approx 3N(\mathcal{C} + \mathcal{Q}) + N(\mathcal{I}) + (m+1) \sum_{i=1}^k N(P_i + E_i) \quad (7) \end{aligned}$$

As indicated by the formulas above, $\text{Token}^{\text{ReAct}}$ scales linearly with the size of $(\mathcal{C}, \mathcal{I}, \mathcal{Q})$ by a factor of k and quadratically with the size of $(\mathcal{T}_i, \mathcal{A}_i, \mathcal{O}_i)$ in terms of k . On the other hand, $\text{Token}^{\text{ReW00}}$ increases linearly with the size of $(\mathcal{C}, \mathcal{I}, \mathcal{Q})$ due to constant factors and linearly with the size of $(P_i + E_i)$. Similarly, $\text{Token}^{\text{ReMSV}}$ also grows linearly with the size of $(\mathcal{C}, \mathcal{I}, \mathcal{Q})$, albeit with a slightly larger constant factor, and linearly with the size of $(P_i + E_i)$, multiplied by a factor of m . The analysis indicates that with **ReAct**, token consumption escalates substantially as the number of reasoning steps grows. In contrast, **ReW00** and **ReMSV** do not suffer from this issue, maintaining efficiency in token usage regardless of the increase in reasoning steps. Nevertheless, **ReMSV** will consume more tokens than **ReW00** due to sampling m multiple strategies. We also empirically evaluate actual token consumption in the results Sect. 6.

5 Experiments

We evaluate our approach against SOTA methods **ReAct** and **ReW00** using question answering (QA) datasets related to general knowledge and arithmetic reasoning that require multiple reasoning steps.

5.1 Benchmarks

The following three datasets are used to evaluate our approach.

- *HotpotQA* is a dataset that comprises multi-hop reasoning questions, which, based on Wikipedia, require the identification and analysis across various supporting documents to generate an answer [15].
- *GSM8K* is a dataset featuring linguistically varied school math word problems, crafted by human problem composers. Solving these problems typically involves a series of elementary calculations employing basic arithmetic operations ($+$ $-$ \times \div) to obtain the final answer [2].
- *PhysicsQA* focuses on high school physics questions that test knowledge in areas such as Newton’s second law, force identification, kinematics, and so forth [1].

5.2 Baselines

The following SOTA methods, which allow LLMs to engage with external tools, serve as baselines.

- **ReAct** [17] represents a prompting strategy that combines reasoning and actions to resolve language reasoning and decision-making tasks by interacting with the external environment. Specifically, it uses a (*Thought, Act, Obs*) prompting approach to adjust action plans according to the external environment.
- **ReW00** [14] is designed with a *Plan - Work - Solve* strategy to leverage the foreseeable reasoning capabilities of LLMs without relying on explicit observations.

5.3 Action Space

We have designed the task-specific action space for TA-LLMs to interact with the following APIs. Considering the various characteristics of the benchmark datasets, action spaces are configured differently as shown in Table 1.

- **LLM-Tool**. Considering that the term “tools” in this context specifically refers to external tools, we employ LLM-tool, a separate LLM² that can process language reasoning tasks, which is consistent with the experiment setting of the ReW00 paper [14].

² GPT-3.5-Turbo is used in our experiment.

- **Wikipedia Search.** The search API for Wikipedia³ pages to query knowledge from Wikipedia’s database.
- **Google Search.** The search API from Google SERP⁴ service to query knowledge from the Internet.
- **WolframAlpha.** The complex mathematical computation result from WolframAlpha⁵.
- **Calculator.** A simple calculator⁶ to perform basic arithmetic operations.

Table 1. Action space configurations for different benchmark datasets

Dataset	No. Tools	Action Space
HotpotQA	2	LLM-tool, Wiki
GSM8K	3	LLM-tool, WolframAlpha, Calculator
PhysicsQA	5	LLM-tool, WolframAlpha, Calculator, Wiki, Google

5.4 Evaluation Metrics

Evaluation metrics commonly used to assess QA performance of LLMs [3, 13, 14, 17], including exact match (EM), character-level F_1 scores, and accuracy, are used in this study.

- **EM.** The model’s prediction is checked whether it precisely matches the correct answer in the reference data. If it does, the score is 1 (100%); if not, it is 0 (0%). EM scores are not presented for *GSM8K* and *PhysicsQA* because the inclusion of strings alongside numbers interferes with accurate matching, struggling to accurately represent true performance.
- **F_1 .** The character-level F_1 -score evaluates how accurately the model predicts individual characters compared to the ground truth, with a score of 1 indicating perfect accuracy.
- **Accuracy.** Since the output of LLMs vary syntactically, accuracy is determined by using GPT-4 to evaluate the predictions against the ground truth, giving a binary score of 1 or 0.
- **Token Consumption.** We calculate the average token consumption per question, incorporating both prompting tokens and text completion tokens, to assess the cost-effectiveness of each prompting strategy.

³ <https://www.mediawiki.org/wiki/API>.

⁴ <https://serpapi.com/search-api>.

⁵ <https://products.wolframalpha.com/api>.

⁶ https://js.langchain.com/docs/api/tools_calculator/.

5.5 Experimental Setup

For both the baselines and our method, CoT exemplars are employed to assist the LLM in generating strategies for executing tool-calls. The same task-dependent exemplar questions (released by ReAct, ReW00) were utilized across the three datasets for both the baseline methods and our approach to ensure fair comparisons. Specifically, for *HotpotQA*, six exemplars (six-shot) are used. While for *GSM8K* and *PhysicsQA*, one exemplar (one-shot) is used, which is consistent with the experiment settings in the ReW00 paper [14]. For the LLM-Agent, we employ *GPT-3.5-Turbo*⁷ as the base model. Considering the cost, we assess the benchmark performance of our method and the baselines by utilizing 500 randomly chosen samples from *HotpotQA*, *GSM8K*, and all examples (57) from *PhysicsQA*. We sample 3 strategies when conducting the general benchmarking experiment. Subsequently, we conduct additional experiments to investigate the effects of varying strategy counts (from 2 to 5) and to evaluate the efficiency of individual strategies using 50 randomly sampled questions from *HotpotQA*, *GSM8K* and *PhysicsQA*.

6 Results

We present the results of our experiments in three parts. First, we benchmark our proposed method on three QA datasets vs. the baseline methods. We then investigate the impact of the multi-strategy mechanism in an attempt to explain why the proposed method outperforms the baselines. Finally, we conduct an error analysis to identify situations in which the proposed method fails and how.

6.1 Benchmarking Prompting Methods

Table 2 presents the experimental results of the baselines and our method on *HotpotQA*, *GSM8K*, and *PhysicsQA*. Our approach outperforms ReAct on *HotpotQA* and *PhysicsQA*, while consistently outperforming ReW00 on all three datasets.

Notably, our approach delivers the best performance on *HotpotQA* with an accuracy of 54.6, achieving a relative improvement of 44.4% and 4.5% in accuracy, 38.6% and 9.6% in F_1 , and 34.5% and 9.1% in EM when compared to ReAct and ReW00, respectively. ReAct yields the best results in *GSM8K*, with our method securing the second-highest performance. Our approach consistently outperforms ReW00 across all three benchmark datasets, demonstrating that incorporating the multi-strategy mechanism enhances overall performance. Regarding token costs, ReMSV is shown to be more cost-effective in terms of token consumption than ReAct, although it does use more tokens than ReW00. More precisely, ReMSV uses roughly 50% more tokens than ReW00 across the benchmarks. However, it uses significantly fewer tokens compared to ReAct. In particular, ReMSV requires over 200% fewer tokens than ReAct in benchmarks such as *HotpotQA*. This result is in line with the cost estimations detailed in Sect. 4 that showed

⁷ <https://platform.openai.com/docs/models/gpt-3-5>. access on 1st, Sep, 2023.

that ReMSV and ReWOO scale well for complex problems requiring many reasoning steps. Moreover, employing SC, which executes all strategies, would lead to an estimated cost that is threefold higher than ReWOO and almost double that of ReMSV.

Table 2. Predictive performance of ReAct, ReWOO, and ReMSV across three QA datasets. The best performance is marked in bold, while the second best results are underlined. We also report the average number of reasoning steps for each benchmark.

Dataset	Method	Acc	F_1	EM	# Tokens	Steps
HotpotQA	ReAct	37.8	33.7	28.4	6,771	5.4
	ReWOO	<u>52.2</u>	<u>42.6</u>	<u>35.0</u>	1,447	4.1
	ReMSV	54.6	46.7	38.2	<u>2,194</u>	4.3
GSM8K	ReAct	65.2	35.8	N/A	1,662	3.2
	ReWOO	59.4	28.9	N/A	759	3.5
	ReMSV	<u>62.6</u>	<u>29.6</u>	N/A	<u>1,359</u>	3.2
PhysicsQA	ReAct	32.0	11.0	N/A	2,252	2.8
	ReWOO	<u>35.0</u>	<u>12.0</u>	N/A	908	3.0
	ReMSV	36.0	13.0	N/A	<u>1,548</u>	2.9

6.2 Impact of the Multi-strategy Mechanism

We conduct further experiments to explore the impact of the multi-strategy mechanism, particularly in terms of the size of the strategy pool and the effectiveness of individual strategies. These experiments are described below.

Effect of Strategy Pool Size. To explore the impact of varying the number of strategies created by the *Director*, we conducted an experiment on *HotpotQA* using the same 500 samples, varying the strategy pool size from two to five. Table 3 demonstrates that, in general, our method, employing between two to five strategies, surpasses ReWOO, which relies on executing a single strategy. We also observe that our method yields improved results when choosing either 3 or 4 strategies. This suggests that with too few strategies (i.e. one or two), there is a risk that a more optimal strategy might be overlooked. Conversely, with an excessive number of strategies (like five), the likelihood increases that a sub-optimal strategy might be chosen given the quality of the *Voter*. We intend to further validate this in future work.

Efficacy of Individual Strategies. We conduct an experiment using a subset of 50 examples from each of *HotpotQA*, *GSM8K* and *PhysicsQA* to explore the performance of each individual strategy in the strategy pool, i.e. if they were

Table 3. Results when considering different numbers of strategies (STs). The best performance is highlighted in bold while the second best performance is underlined. Note that **ReWOO** corresponds to considering only a single strategy.

Method	No. STs	Acc	F_1	EM
ReWOO	1	52.2	42.6	35.0
ReMSV	2	52.2	43.1	34.6
	3	<u>54.6</u>	46.7	38.2
	4	54.8	<u>46.4</u>	<u>38.0</u>
	5	53.4	44.7	36.0

Table 4. Performance for selected and not selected strategies. The best performance is marked in bold, while runner-up performance is underlined. The average outcome for not selected and all strategies is given for comparison.

Method	Acc.	F_1
1st ST (ReWOO)	46.0	27.2
Selected ST (ReMSV)	50.0	30.5
Not selected STs	47.6	29.9
Avg. all STs	<u>48.9</u>	<u>30.2</u>

to be executed, as well as the effectiveness of the voting mechanism. To provide a comprehensive comparison between **ReMSV** and **ReWOO**, we present the average performance metrics across three datasets, focusing on the shared metrics **Accuracy** and F_1 . In line with the results in Table 2, Table 4 shows that our approach using the selected strategy outperforms **ReWOO**, which essentially corresponds to selecting the first generated strategy. Additionally, the performance of the selected strategy also generally achieves superior performance to the average of the strategies that were not selected. From the data presented in Table 2 and Table 4, it is evident that **ReMSV** consistently surpasses **ReWOO**. Notably, the average results from the strategies that were not selected are also on par with or exceed the performance of **ReWOO**, which suggests that **ReMSV** effectively reduces the risk of executing sub-optimal strategies due to greedy sampling. Furthermore, the average results of both selected and non-selected strategies fall short of **ReMSV**, indicating that **ReMSV** would also outperform **SC**, which aggregates all executed strategies. Together with the cost estimation for **SC** in Sect. 6.1, without resorting to executing all strategies, **ReMSV** not only is able to surpass the performance of methods based on single trajectories, but also proves to be more cost-effective in terms of token consumption compared to **SC**. Figure 3 illustrates an instance where the single reasoning trajectory selection of **ReWOO** leads to an erroneous strategy implementation, yielding an incorrect prediction. Conversely, in this scenario, **ReMSV** casts votes across the sampled strategies and chooses one that ultimately produces the correct prediction.

6.3 Error Analysis

To gain insights into when and why our proposed method fails to produce the correct prediction, we conduct an error analysis on the incorrect predictions generated by **ReMSV**. We summarize the statistics of different sources of failure for these incorrect predictions in Table 5. Out of the 50 experimental examples in each dataset, **ReMSV** produces 21 incorrect predictions for *HotpotQA* and *GSM8K*,

Question: Maddy is in college for 8 semesters. She needs 120 credits to graduate. If each class is 3 credits, how many classes does she need to take per semester?

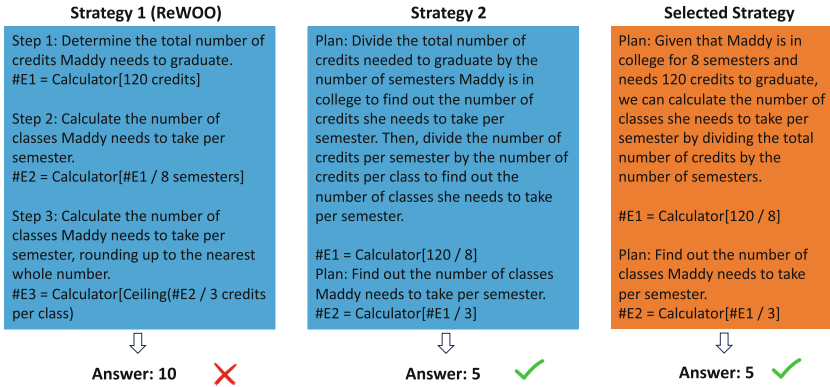


Fig. 3. An example from GSM8K illustrating different sampled strategies: the not selected strategies are colored in blue, while the selected strategy is colored in orange. (Color figure online)

Table 5. *Director* failure indicates that the strategy pools contains no successful strategy. *Worker* failure indicates that the wrong prediction is due to a failed strategy execution, resulting in an inability to produce a valid answer. *Voter* failure indicates that an unsuccessful strategy is selected when there is at least one successful strategy in the strategy pool. Note that there can be more than one source of failure.

	HotpotQA	GSM8K	PhysicsQA
Incorrect predictions	21	21	33
- <i>Director</i> failure	17	12	29
- <i>Voter</i> failure	2	2	1
- <i>Worker</i> failure	4	9	13

and 33 for *PhysicsQA*. Most incorrect predictions stem from *Director* failures, indicating that none of the generated strategies in the strategy pool leads to a correct prediction. This suggests that there is potential for improvement in the generation of strategies, although it could also mean that these particular questions are challenging to decompose based on the foreseeable reasoning capabilities of current LLMs. Specifically, ReMSV tends to incur more *Director* failures on *HotpotQA* and *PhysicsQA* than on *GSM8K*. Conversely, *Worker* failures occur more frequently in *GSM8K* and *PhysicsQA* than in *HotpotQA*. These findings suggest that for *HotpotQA* and *PhysicsQA*, ReMSV more often generates unsuccessful strategies, whereas in *GSM8K* and *PhysicsQA* – where more tools are utilized – ensuring the proper functioning of tool calls is critical. We also report *Voter* failures, where, across all benchmarks, only 1 or 2 failures are due to not selecting a successful strategy, indicating that, compared to other components

in **ReMSV**, the *Voter* performs more robustly. The overall error analysis implies that there is room for task-specific enhancements in future work. For instance, enhancing the sampling quality for tasks such as *HotpotQA* and *PhysicsQA*, or improving the stability of tool calling for tasks like *GSM8K* and *PhysicsQA*, could lead to significant improvements.

7 Discussion

In this section, we discuss the results of our experiments, focusing especially on the pros and cons of prompting techniques based on observation-dependent reasoning vs. foreseeable reasoning and single vs. multiple reasoning trajectories. We also discuss possible directions for future work.

7.1 Observation-Dependent Reasoning Vs. Foreseeable Reasoning

Prompting strategies based on observation-dependent reasoning, such as **ReAct**, continuously observe the environment – here in the form of tool-calling output – and relay these observations back to the LLM to formulate relevant reasoning and action plans. Such prompting strategies are highly responsive to environmental changes and devise action plans in an ad-hoc, dynamic manner. Consequently, while such prompting strategies perform well and allow an LLM to interact with external tools to solve complex reasoning problems, they tend to incur a relatively high cost by requiring more interactions with the LLM-Agent and hence a higher token consumption since the observation in each iteration is fed to the LLM to generate the next thought and action. Prompting methods based on foreseeable reasoning, like **ReW00** and **ReMSV**, instead devise a strategy up-front that includes both reasoning and action plans, negating the necessity to continuously provide observations and interacting with the LLM-Agent. Therefore, prompting strategies based on foreseeable reasoning are more cost-effective and can play a crucial role in the development of efficient TA-LLMs. **ReW00** and **ReMSV**, which both rely on the foreseeable reasoning capabilities of LLMs, perform on par with or outperform **ReAct**, which is based on observation-dependent reasoning. The results on *GSM8K* show, however, that in some situations, **ReAct** leads to better performance. One possible explanation could be that certain problems are not readily decomposable up-front without first obtaining more information through tool-calls. Such problems may require dynamic reasoning and are perhaps beyond the foreseeable reasoning capabilities of current LLMs.

7.2 Single Vs. Multiple Reasoning Trajectories

Both **ReAct** and **ReW00** rely on a single reasoning trajectory, which might misguide the TA-LLMs into producing incorrect predictions. Prompting methods based on multiple reasoning trajectories, like **ToT**, on the other hand, are computationally expensive and incur a relatively high cost by needing to execute each sampled reasoning trajectory. **ReMSV** is a prompting method that aims to combine

the advantages of prompting methods based on single and multiple reasoning trajectories, respectively. It does so by generating and taking into consideration multiple reasoning trajectories, while operating in a cost-efficient manner by only selecting – based on LLM voting – and executing a single reasoning trajectory. Its cost-effectiveness stems also from leveraging the economical attributes of the foreseeable reasoning capabilities of LLMs, i.e. by generating strategies up-front rather than in an ad-hoc and dynamic fashion. The experiments demonstrate that **ReMSV** surpasses **ReAct** on *HotpotQA* and *PhysicsQA*, and consistently outperforms **ReWOO** across all three datasets. This clearly validates the efficacy of incorporating multiple reasoning trajectories, albeit in a cost-efficient manner. While **ReMSV** exploits the cost-efficient properties of **ReWOO**, there is a slightly higher cost of **ReMSV** that stems from generating a pool of strategies. However, as the experimental results show, it leads to improved predictive performance. Generally speaking, there is a likely trade-off to be made between the cost-efficiency of prompting based on single reasoning trajectories and the predictive performance of prompting based on multiple reasoning trajectories. While **ReMSV** attempts to strike a good balance in this regard, it may be possible to obtain higher predictive performance at a higher cost. One alternative in this direction is to execute all generated strategies and aggregate the results in a late fusion fashion. Although we did not fully evaluate this approach, the results in Table 4 indicate that this method would, in fact, not lead to higher predictive performance since the predictive performance of the selected strategy is higher than the average of the strategies that were not selected, as well as the average of all generated strategies. That said, there is certainly room for developing more sophisticated ensemble methods in this context.

8 Conclusions, Future Work, and Ethical Statement

In this paper, we propose a new prompting method, **ReMSV**, that leverages the foreseeable reasoning capabilities of LLMs to generate multiple strategies up-front, one of which is selected through LLM voting and strategically executed. **ReMSV** is benchmarked on three question-answering datasets and, overall, surpasses SOTA methods such as **ReAct** and **ReWOO**, and offers a more cost-effective solution compared to **ReAct**.

The present multi-reasoning trajectories of **ReMSV** operate solely on the strategy level. A compelling direction for future research could be to implement plan sampling at each reasoning step within the strategy to generate more diverse and better strategies. By generating more accurate and diverse strategies, ensemble methods are also more likely to be successful. Another avenue for future research might be to enhance the voting process. One possibility for improving the voting could be to incorporate few-shot learning into the voting mechanism to render it more task-aware.

Allowing LLMs to communicate with external environments can introduce certain dangers, such as potential data breaches or the generation of harmful actions. To mitigate these concerns in our experiments, we have imposed boundaries on interactions, restricting them to specific action spaces like Wikipedia and

WolframAlpha and confining the task to question answering on public benchmarks that does not access private or confidential information.

Acknowledgement. The computations and data handling were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06725. Acknowledgment is also extended to the ReWOO project⁸ for providing the code base used to conduct the experiments. (⁸<https://github.com/billxbf/ReWOO>)

References

1. Beatty, I.D., Gerace, W.J., Leonard, W.J., Dufresne, R.J.: Designing effective questions for classroom response system teaching. *Am. J. Phys.* **74**(1), 31–39 (2006)
2. Cobbe, K., et al.: Training verifiers to solve math word problems. arXiv preprint [arXiv:2110.14168](https://arxiv.org/abs/2110.14168) (2021)
3. Hao, S., Gu, Y., Ma, H., Hong, J.J., Wang, Z., Wang, D.Z., Hu, Z.: Reasoning with language model is planning with world model. arXiv preprint [arXiv:2305.14992](https://arxiv.org/abs/2305.14992) (2023)
4. Hosseini-Asl, E., McCann, B., Wu, C.S., Yavuz, S., Socher, R.: A simple language model for task-oriented dialogue. *Adv. Neural. Inf. Process. Syst.* **33**, 20179–20191 (2020)
5. Ji, Z., et al.: Survey of hallucination in natural language generation. *ACM Comput. Surv.* **55**(12), 1–38 (2023)
6. Komeili, M., Shuster, K., Weston, J.: Internet-augmented dialogue generation. arXiv preprint [arXiv:2107.07566](https://arxiv.org/abs/2107.07566) (2021)
7. Lewis, P., et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Adv. Neural. Inf. Process. Syst.* **33**, 9459–9474 (2020)
8. Nakano, R., et al.: Webgpt: Browser-assisted question-answering with human feedback. arXiv preprint [arXiv:2112.09332](https://arxiv.org/abs/2112.09332) (2021)
9. OpenAI, R.: Gpt-4 technical report ([arxiv: 2303.08774](https://arxiv.org/abs/2303.08774)). View in Article (2023)
10. Rae, J.W., et al.: Scaling language models: Methods, analysis & insights from training gopher. arXiv preprint [arXiv:2112.11446](https://arxiv.org/abs/2112.11446) (2021)
11. Shinn, N., Cassano, F., Labash, B., Gopinath, A., Narasimhan, K., Yao, S.: Reflexion: language agents with verbal reinforcement learning **4**. arXiv preprint [arXiv:2303.11366](https://arxiv.org/abs/2303.11366) (2023)
12. Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., Zhou, D.: Self-consistency improves chain of thought reasoning in language models. arXiv preprint [arXiv:2203.11171](https://arxiv.org/abs/2203.11171) (2022)
13. Wei, J., et al.: Chain-of-thought prompting elicits reasoning in large language models. *Adv. Neural. Inf. Process. Syst.* **35**, 24824–24837 (2022)
14. Xu, B., Peng, Z., Lei, B., Mukherjee, S., Liu, Y., Xu, D.: Rewoo: Decoupling reasoning from observations for efficient augmented language models. arXiv preprint [arXiv:2305.18323](https://arxiv.org/abs/2305.18323) (2023)
15. Yang, Z., et al.: Hotpotqa: a dataset for diverse, explainable multi-hop question answering. arXiv preprint [arXiv:1809.09600](https://arxiv.org/abs/1809.09600) (2018)
16. Yao, S., et al.: Tree of thoughts: Deliberate problem solving with large language models. arXiv preprint [arXiv:2305.10601](https://arxiv.org/abs/2305.10601) (2023)
17. Yao, S., et al.: React: Synergizing reasoning and acting in language models. arXiv preprint [arXiv:2210.03629](https://arxiv.org/abs/2210.03629) (2022)



Estimating Direct and Indirect Causal Effects of Spatiotemporal Interventions in Presence of Spatial Interference

Sahara Ali^{1,2}, Omar Faruque¹, and Jianwu Wang¹(✉)

¹ University of Maryland Baltimore County, Baltimore, MD, USA
{sali9,omarf1,jianwu}@umbc.edu

² University of North Texas, Denton, TX, USA
<https://bda1.umbc.edu>

Abstract. Spatial interference (SI) occurs when the treatment at one location affects the outcomes at other locations. Accounting for spatial interference in spatiotemporal settings poses further challenges as interference violates the stable unit treatment value assumption, making it infeasible for standard causal inference methods to quantify the effects of time-varying treatment at spatially varying outcomes. In this paper, we first formalize the concept of spatial interference in the case of time-varying treatment assignments by extending the potential outcome framework under the assumption of no unmeasured confounding. We then propose our deep learning based potential outcome model for spatiotemporal causal inference. We utilize latent factor modeling to reduce the bias due to time-varying confounding while leveraging the power of U-Net architecture to capture global and local spatial interference in data over time. Our causal estimators are an extension of average treatment effect (ATE) for estimating direct (DATE) and indirect effects (IATE) of spatial interference on treated and untreated data. Being the first of its kind deep learning based spatiotemporal causal inference technique, our approach shows advantages over several baseline methods based on the experiment results on two synthetic datasets, with and without spatial interference. Our results on real-world climate dataset also align with domain knowledge, further demonstrating the effectiveness of our proposed method.

Keywords: Spatiotemporal Causal Inference · Deep Learning · Spatial Interference

1 Introduction

Quantifying the effects of an entity, process or state, referred as the cause, on another entity, process or state, referred as the outcome, is an active research area with wide applications in epidemiology, economics, political and environmental science [29, 40]. In many real-world scenarios, the effect of a treatment or intervention is not static but evolves dynamically over space and time, presenting a challenge for accurate estimation of time-varying treatment effects under

spatial interference. The concept of spatial interference stems from spatial statistics where interventions or events at one location not only impact outcomes at that specific location but also propagate to neighboring or distant locations [30]. This spatial dependency violates the assumptions of independence and stable unit treatment value (SUTVA) [18], underlying many conventional causal inference techniques, which restricts every unit in the data to have treatment applied to only its own outcome. When spatial interference is present, the treatment at one unit spills over and influences the outcome of neighboring units. Oftentimes, spatial interference is confused with spatial confounding. Confounding occurs when the past values of some covariates influence the current values of treatment and outcome leading to spurious correlations and biases in the outcome values. We refer to such covariates as confounders [28]. In Fig. 1, we illustrate four possible spatial and temporal interactions in data at two locations $s1$ and $s2$ over timesteps t and $t-1$. Here, X is the treatment variable, Z is a covariate, and Y is the outcome. In this paper, we present the phenomenon in Fig. 1d, i.e., how time-varying treatment affects potential outcome in the presence of time-varying confounding and spatial interference. Though the real-world observational data comprises intricate complexities of both spatial and temporal confounding, we limit our scope to observed temporal confounders, whereas identifying confoundedness in space or through unobserved confounders is beyond the scope of this paper.

We rely our work on three major arguments. (1) Traditional linear models used for estimating causal effects rely heavily on strong parametric assumptions. Conversely, neural networks, being nearly non-parametric in nature, offer the advantage of capturing the diverse treatment effects observed across individual units while minimizing bias [20]. (2) Even though causal effect estimation in temporal and spatial settings have been investigated previously (see Sect. 5), there exists limited work that handles both of these tasks simultaneously. (3) We further argue propensity score based techniques are computationally expensive and unable to handle continuous time-varying treatments [2, 4].

Our Contributions. (1) We extend the potential outcome framework to spatiotemporal setting by introducing STCINet: a spatiotemporal causal inference network based on U-Net architecture with double attention to learn causal relations with spatially interfering treatments. (2) We propose an autoencoder based factor model to reduce the time-varying confounding effect of spatial data by adapting the factor model introduced in [3] and extending it to spatiotemporal domain. (3) We establish the case of spatial interference in time-varying data by evaluating our method on synthetic datasets based on diffusion phenomenon. (4) We establish a promising research direction for spatiotemporal causal inference in climate science by quantifying the direct and indirect effects of atmospheric processes on Arctic sea ice melt. Our implementation code can be accessed on GitHub.¹

The rest of the paper is organized as follows. Section 2 enlists the causal assumptions and notations followed throughout this paper. Section 3 explains the

¹ <https://tinyurl.com/stcinet>.

overall architecture and individual modules of our proposed method. Section 4 mentions the data generation process, experimental configurations, evaluation methods and empirical results of our model on synthetic and real world data. Section 5 highlights the related work in causal inference. Lastly, we conclude our paper in Sect. 6 and mention some potential extensions of this work.

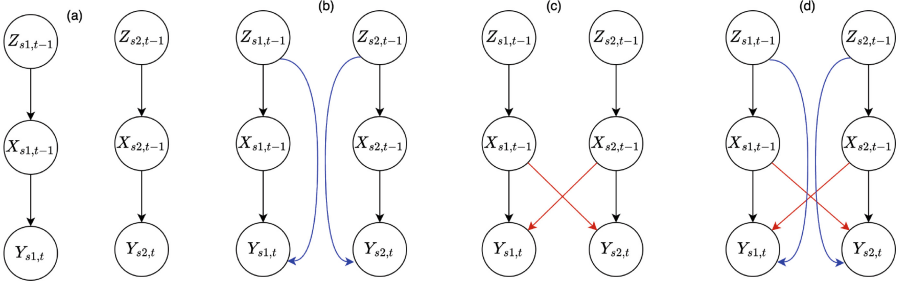


Fig. 1. Different scenarios of causation (black), confounding (blue) and interference (red) in spatiotemporal data. (a) No confounding, no interference, only temporal causation; (b) No interference, only temporal confounding and temporal causation; (c) No temporal confounding, only spatial interference and temporal causation; (d) Temporal confounding, spatial interference, temporal causation. (Color figure online)

2 Preliminaries

2.1 Notations and Definitions

Given spatiotemporal data over $N \times M$ region and spanning over T timesteps, $X^t = X_{i,j \in [N,M]}^t$ represents the treatment variable at timestep $t \in T$, $Z^t = Z_{i,j \in [N,M]}^t$ represents a set of time-varying covariates and $Y^t = Y_{i,j \in [N,M]}^t$ represents the outcome, such that at every location $\{X_{i,j}^t, Z_{i,j}^t, Y_{i,j}^t\} \in \mathbb{R}$. When intervened on the value of treatment X , the corresponding updated value for the same (i, j) location is represented by $\hat{X}_{i,j}$, where $\hat{X}_{i,j} = \text{update_factor} \times X_{i,j}$. At any given time t , $Y_{i,j}(\hat{X}_{i,j})$ is the potential outcome under intervened treatment \hat{X} , and $Y_{i,j}(X_{i,j})$ is the potential outcome under un-intervened treatment X (also called placebo effect). Further, at any given timestep t , \bar{X}^t represents the set of historic values of X and \bar{Z}^t represents the set of historic values of Z , such that $\bar{X}^t = (X^1, X^2, X^3, \dots, X^{t-1})$ and $\bar{Z}^t = (Z^1, Z^2, Z^3, \dots, Z^{t-1})$.

Outcome under no Spatial Interference (Fig.1a). Assuming Y^{t+l} to be dependent on X^t and an unknown noise term ε_y , a simple case of data-generation process under no spatial interference and only time-varying lagged dependence can be given by $Y_{i,j}^{t+l} = \beta X_{i,j}^t + \varepsilon_y$, where $X_{i,j}^t = \gamma Z_{i,j}^t + \varepsilon_x$. Here, β and γ represent the causal coefficients of X and Z and l is the temporal lag.

Outcome under Spatial Interference (Fig.1c). Next, we assume that the outcome Y at every location is also influenced by the treatment applied at neighboring locations. The data-generation process will now comprise spatial interference and time-varying lagged dependence, given by $Y_{i,j}^{t+l} = \beta_1 X_{i,j}^t + \beta_2 \tilde{X}^t + \varepsilon_y$, where, \tilde{X} represents mean of m neighborhood values of $X_{i,j}^t$ at any given timestep, such that $\tilde{X} = \mathbb{M}(X_{(i-m,j-m),\dots,(i+m,j+m)})$, excluding $X_{i,j}^t$ itself in the mean computation. Here, β_2 represents the causal coefficient of \tilde{X} .

Outcome under Spatial Interference and Temporal Confounding (Fig.1d). Here, we extend the data-generation process to settings where Z is the confounder, i.e., both X and Y are dependent on Z , given by $Y_{i,j}^{t+l} = \beta_1 X_{i,j}^t + \beta_2 \tilde{X}^t + \beta_3 Z_{i,j}^t + \varepsilon_y$, and $X_{i,j}^t = \gamma Z_{i,j}^t + \varepsilon_x$.

Under the setting of spatial interference and temporal confounding, we present three different metrics for estimating the causal effects of X on Y in the presence of confounder Z ; (1) in the treated sub-region, (2) in the untreated sub-region where spillover is observed and (3) finally in the overall spatial region of a given dataset.

Definition 1: Direct (Spatial) Average Treatment Effect [35]. Given observed X and intervened \hat{X} , referring to time-varying treatment values, at time-step t and a spatial location $s \in \mathbb{S}$, such that $\mathbb{S} \subset N \times M$, the direct treatment effect τ_{date} refers to the difference in potential outcomes Y under \hat{X} and X , observed directly on the treated location s . The direct average treatment effect τ_{date} is given by:

$$\tau_{date} = \frac{\sum_{s \in \mathbb{S}} (Y_s(\hat{X}_s, Z) - Y_s(X_s, Z))}{|\mathbb{S}|} \tag{1}$$

Definition 2: Indirect (Spatial) Average Treatment Effect [22]. The indirect spatial treatment effect τ_{iate} refers to the difference in potential outcomes under \hat{X} and X , observed on the untreated region $s' \in \mathbb{S}'$, such that $\mathbb{S}' \subset N \times M$ and $s' \neq s$, where s refers to the treated region. The indirect average treatment effect τ_{iate} is given by:

$$\tau_{iate} = \frac{\sum_{s' \in \mathbb{S}'} (Y_{s'}(\hat{X}_s, Z) - Y_{s'}(X_s, Z))}{|\mathbb{S}'|} \tag{2}$$

Definition 3: Lagged Average Treatment Effect [2]. Combining the net effect of direct and indirect treatment over the entire region $\mathbb{N} = N \times M$, the overall average treatment effect τ_{late} at a temporal lag of l is given by:

$$\tau_{late} = \frac{\sum_{k \in \mathbb{N}} (Y_k^{t+l}(\hat{X}_k^t, Z_k^t) - Y_k^{t+l}(X_k, Z_k))}{|\mathbb{N}|} \tag{3}$$

2.2 Assumptions

Extending our understanding developed in Sect. 2.1, we assume that the outcome is generated by treatment, covariates and noise, given by $Y = F(X, Z) + \varepsilon_y$,

where F is an unknown and non-linear function. Further, the following assumptions hold for the method and experiments proposed in the remaining paper:

Assumption 1: No Unmeasured Confounding [13]. We assume there is no unmeasured or unobserved confounding other than the confounding caused by observed covariates Z . Further, the confounding is only limited to temporal scale such that $Y_{i,j}^t = F(X, Z_{i,j}^t, \bar{Z}_{i,j})$ and $Y_{i',j'}^t \neq F(X, Z_{i',j'}^t, \bar{Z}_{i',j'})$ where $(i, j) \neq (i', j')$.

Assumption 2: Consistency [6]. If the historic values of treatment are $\bar{X}^t = \bar{x}^t$, then the potential outcome under the treatment is the same as the observed (factual) outcome $Y(\bar{x}^t) = Y$.

Assumption 3: SUNTVA. We replace the stable unit treatment value assumption with its variant - Stable Unit Neighborhood Treatment Value Assumption (SUNTVA), introduced by [9], to accommodate spatial interference. Under SUNTVA, for each location s , there exist a neighborhood \mathbb{M}_s , such that the outcome is influenced by X_s as well as by the neighborhood of treatment $X_{\mathbb{M}_s}$, such that $Y_s = Y_s(X_s, X_{\mathbb{M}_s})$.

In addition, we assume positivity [33], such that at each timestep t , each treatment has a non-zero probability of being assigned to the outcome in the treated region. Since we refer to a spatiotemporal setting, we relax the spatial subscript (i, j) moving forward, to mention the time-varying aspect of data, implying $Y_{i,j}^t \rightarrow Y^t$ unless specified otherwise.

3 Spatio-Temporal Causal Inference Network (STCINet)

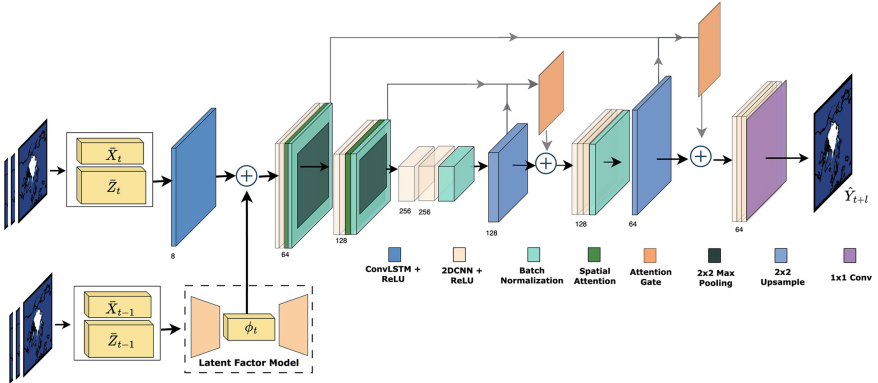


Fig. 2. Overall architecture of proposed spatiotemporal causal inference network (STCINet).

Here, we present our proposed technique to perform causal inference under time-varying confounding and spatial interference. The overall architecture is

given in Fig. 2, where we first divide the spatiotemporal data into current data (X^t, Z^t) and history data $(\bar{X}^{t-1}, \bar{Z}^{t-1})$. The current data passes through a Convolutional Long Short Term Memory (ConvLSTM) layer, to learn the lagged representation of treatment and covariates, whereas the historic data is fed to the Latent Factor Model (LFM), through which the uncorrelated latent representations of treatment and covariates are learned, assuming the covariates also comprise some observed confounders. We employ the LFM technique as opposed to the propensity score weighting (IPTW) [31] owing to the promising results of factor models observed in recent literature [3, 4]. The latent representation for ϕ^t is then combined with the output of ConvLSTM. This combined set of features is fed as the input to a U-Net based model. With the aid of attention gating, the custom U-Net model learns the local and global spatial variations in data, occurring in response to the treatment application, and finally, the U-Net predicts the future values of outcome Y^{t+l} after a lag of l timestep. At test time, the trained model is fed treated (intervened) and untreated (observed) values of treatment variable to estimate direct, indirect and overall causal effects using the metrics of τ_{date} , τ_{iate} and τ_{late} . Below, we explain the functionality of each of these modules in detail.

3.1 Latent Factor Model for Temporal Confounding

Latent factor models are designed to uncover the hidden structure or underlying factors that influence the observed data [10]. Latent variables are not directly measured or observed but are inferred from the observed data patterns. To observe the sole influence of current treatment on the future outcome, in the presence of time-varying confounders, we propose a latent factor model (LFM), inspired by [3] that learns the distribution of treatment and covariates over time and de-correlates the entangled relationship so the outcome becomes independent of the confoundedness, i.e., $Y^t \perp \bar{Z}^{t-1} | X^t$. Our implementation of the latent factor model is given in Fig. 3b, where we design an autoencoder based model that takes in historic spatiotemporal values of treatment and covariates just before the timestep where treatment is applied, i.e., $t - 1$.

The encoder part of the LFM comprise of one ConvLSTM layer, one 2D convolution (CNN) layer followed by dropout and the second 2D CNN layer followed by batch normalization. While the purpose of ConvLSTM and CNN layers is to learn the time-varying spatial features of historic data, dropout and normalization is applied to reduce overfitting and improve the generalization of the encoder. The output of the encoder is given by $\phi^t = \text{Enc}((\bar{X}^{t-1}, \bar{Z}^{t-1}))$.

The decoder comprise of one fully connected layer with ReLU activation and the second fully connected layer with linear activation. Finally, we reshape the outcome back the dimensions of X and Z . We use the Mean Squared Error (MSE) as the reconstruction loss for the decoder. The reconstructed output of the decoder is given by $(\hat{X}^{t-1}, \hat{Z}^{t-1}) = \text{Dec}(\phi^t)$.

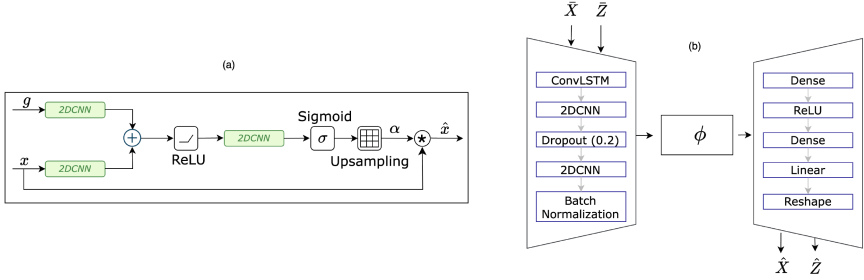


Fig. 3. Sub-modules of STCINet: (a) Attention gating mechanism to identify patterns of spatial interference, (b) Latent factor model for deconfounding covariate and treatment history.

3.2 Double Attention Mechanism

Attention mechanism allows the model to selectively focus on different parts of the input by assigning attention weights to each element in the input based on its contribution in predicting the outcome [37]. The selective focus helps the model to ignore less important parts of the input thereby improving the model’s predictive performance. Attention mechanism has previously shown promising contribution in causal discovery problems [24], whereas employing attention in U-Net has shown to increase model’s sensitivity to local and global variations [25]. In our model, we apply attention at two stages; (i) in the STCINet downsampling block and (ii) in the STCINet upsampling block. We refer to the overall role of attention as double attention mechanism for STCINet.

Spatial Attention. Spatial attention is added to the downsampling part of STCINet, where our goal is to enable the model to selectively attend to specific spatial regions that are most affected by treatment assignment. This is done by performing max pooling and average pooling separately on the output of downsampling blocks. Both the pooling outcomes are concatenated and passed through sigmoid activation to get per pixel attention weights. These weights are applied to the downsampling block’s output by element-wise tensor multiplication.

Attention Gating. In the STCINet upsampling block, we incorporate the attention-gating (AG) mechanism introduced by [19] for our spatial interference task. As shown in Fig. 3a, our AG module takes in two inputs, x and g , referring to the input and the gating signal, respectively. The key idea is to assign weights to local regions within x that align with the location of global features in gating signal g . Here, spatial regions are selected by analysing the contextual information provided by the gating signal g which is collected from a coarser scale. In case of STCINet, g represent the skip-connection from the downsampling blocks whereas x represents the upsampled output from the previous upsampling layer. Both the inputs first pass through 2D CNN layers, to align their depth (filters) and dimensions (height, width). We then perform element-wise addition of

the transformed inputs, followed by ReLU activation and 1×1 2D convolution, to reduce the number of trainable parameters in gating operation. Finally, we use sigmoid activation to retrieve attention coefficients $\alpha \in [0, 1]$ and upsample them back to the original dimension of x . The output of the attention gate is the element-wise product of attention coefficients and original input x , given by \hat{x} . Through attention gating, we filter out the noise while retaining global patterns of spatial variations.

3.3 U-Net for Spatial Interference

Our proposed potential outcome prediction model is based on a U-Net architecture [32]. It comprises three modules: downsampling blocks, upsampling blocks and a bottleneck block that acts as a bridge between the two. What distinguishes a U-Net architecture from a transformer based model is the use of skip connections between different upsampling and downsampling layers. In case of STCINet, these skip connections help retain the causal context in data.

Downsample Block. Our STCINet comprises two downsampling blocks. Each block consists of two 2D CNN layers, followed by a spatial attention layer, batch normalization layer and a 2D max pooling layer. The second block follows the same architecture with the difference of input shape. In every successive layer of the downsampler, we increment the output channels by a multiplicative factor of 2, as shown in Fig. 2. All CNN layers use the same 3×3 kernel size filters. The ReLU activation function is used in all the downsampling layers. This part of our model helps learn low-level spatiotemporal dependencies in the data and identifies patterns needed for predicting spatial maps.

Upsample Block. This block learns from the low-level (downsampled) features and helps reconstruct the spatial map in the same dimension as the input but at a future timestep. Similar to the downsampler, the upsampler comprises two upsampling blocks. Every block comprises a 2×2 upsampling layer using the nearest interpolation method and a 2×2 kernel size filter. Just before the the skip connection is built, we concatenate the output of each upsampling block with the feature map generated by the corresponding downsampling block and pass it to the attention gate, as shown in Fig. 2. The output from attention gate is further concatenated with the output from the previous layer and finally passed through three 2D CNN layers. The output channel size of every CNN layer is reduced by a factor of 2 in order to regain the initial input dimension. Finally, a 1×1 convolution with linear activation is applied to the upsampler’s output to generate the predicted spatial map.

Custom Weighted Loss. To jointly train the LFM module with the U-Net module, we customize the overall objective function to be a weighted sum of the two losses from LFM and U-Net. We further multiply the outcome of this custom loss with a $N \times M$ weight map \mathbb{W} . This element-wise multiplication is done to give treated units higher weightage than non-treated units. The purpose of subregion weighting is to help the model focus on treated areas irrespective of their overall spread. Our custom-loss function is given in Eq. 4:

$$L_{tot} = \mathbb{W} \odot (\lambda_1 L_{lfm} + \lambda_2 L_{unet}) \quad (4)$$

$$\text{where, } L_{lfm} = \frac{\sum_{\mathbb{N}} ((X, Z) - \hat{\phi})^2}{|\mathbb{N}|}, L_{unet} = \frac{\sum_{\mathbb{N}} (Y - \hat{Y})^2}{|\mathbb{N}|}$$

λ_1 and λ_2 are hyperparameters to give weightage to the losses from LFM and U-Net, and $\lambda_1 + \lambda_2 = 1$.

Treatment Effect Estimation. Once the STCINet model is trained using custom loss, the trained model is used to make factual and counterfactual predictions by feeding observed and intervened treatment values to the model. The corresponding outcome predictions are used for estimating direct, indirect and lagged averaged treatment effects using the metrics defined in Sect. 2.1.

4 Experiments

4.1 Synthetic Dataset

To test our method for tracking information flow in spatiotemporal data, we generate two variants of synthetic datasets to mimic a dominant physical process found in many geo-science applications, that is, diffusion. Diffusion is a physical process that describes the movement of particles or substances from regions of higher concentration to regions of lower concentration. Following this concept, we generate three spatiotemporal variables X, Y and Z . Where Z is an independent variable with spatial and temporal autocorrelations. X is dependent on Z and Y is dependent on both X and Z . The synthetic data is generated in Python using NumPy and SciPy libraries. The detailed description of our data generation process is given at GitHub². We utilize the causal coefficients α, β , and γ to incorporate causal influence in these variables. All diffusion coefficients (D_x, D_y, D_z) are set to 0.01. We keep the temporal lag as 1 for all temporal dependencies, whereas the time step size (dt) is 0.1.

To model the spatial diffusion of each variable in the dataset, we perform a Laplacian operation ∇^2 on each of them. Further, we employ a time-stepping loop to iteratively update the variables over multiple time steps. At each time step, the Laplacian of X, Y , and Z is computed to model diffusion. The variables are then updated using their respective diffusion equations, incorporating time lags and dependencies between variables. For each time step t from 1 to T , we update the variables using following equations:

$$Z_{i,j}^t = Z_{i,j}^{t-1} + dt \times (D_z \times \nabla^2 Z) \quad (5)$$

$$X_{i,j}^t = X_{i,j}^{t-1} + dt \times (D_x \times \nabla^2 X + \alpha \times \nabla^2 Z_{i,j}^{t-1}) \quad (6)$$

$$Y_{i,j}^t = Y_{i,j}^{t-1} + dt \times (D_y \times \nabla^2 Y + \beta \times \nabla^2 Z_{i,j}^{t-1} + \gamma \times \nabla^2 X_{i,j}^{t-1}) \quad (7)$$

We intervene on the treatment variable X by applying an *update_factor* = 0.6 to a specific sub-region $i = [10 : 15], j = [10 : 15]$ of X at time step t to create

² <https://tinyurl.com/stcinet>.

the intervened scenario. The corresponding counterfactual outcome values \hat{Y} are then generated by:

$$\hat{Y}_{i,j}^t = \hat{Y}_{i,j}^{t-1} + dt \times (D_y \times \nabla^2 \hat{Y} + \beta \times \nabla^2 Z_{i,j}^{t-1} + \gamma \times \nabla^2 \hat{X}_{i,j}^{t-1}) \quad (8)$$

To incorporate a spillover effect of X on the untreated regions, we add the mean of the per-pixel neighborhood of X (excluding the pixel itself) in Y . A visualization of our synthetically generated data at different timesteps is given in Fig. 4.

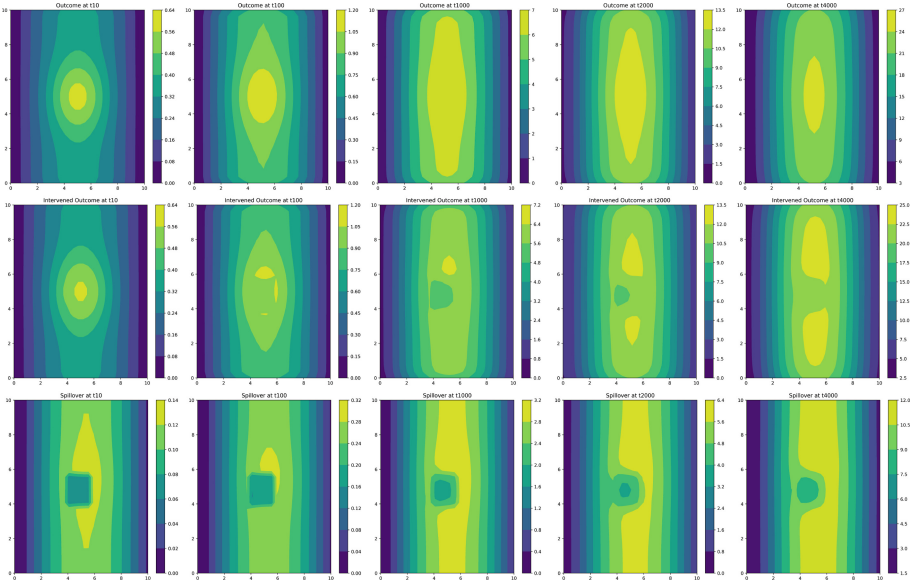


Fig. 4. Potential outcome variable Y at timesteps 10, 100, 1000, 2000 and 4000. Top row: Outcome under no intervention at different timesteps. Middle row: Intervened outcome at different timesteps. Bottom row: Spillover effects at different timesteps, which is the difference in intervened outcomes with and without the spatial interference.

4.2 Evaluation Metrics

We provide the average treatment effect estimations for both synthetic and real-world datasets by reporting the Rooted Precision in Estimation of Heterogeneous Treatment (PEHE) scores based on the treatment effect metrics defined in Eqs. 1, 2, 3. This metric is commonly used in machine learning literature for calculating the average error across the predicted ATEs [14]. Additionally, we report the Root Mean Squared Error (RMSE) to evaluate the model’s predictive performance. Both PEHE and RMSE can only be calculated for synthetic data which

has ground truth information. Since it is a spatiotemporal 3D dataset, we customized the RMSE and PEHE metrics for our spatiotemporal models evaluation and report their respective error ε using the following formulae:

$$\varepsilon_{RMSE} = \sqrt{\frac{\sum_I \sum_J (Y[i, j] - \hat{Y}[i, j])^2}{|\mathbb{N}|}}, \sqrt{\varepsilon_{PEHE}} = \sqrt{\frac{\sum_I \sum_J (\tau[i, j] - \hat{\tau}[i, j])^2}{|\mathbb{N}|}}$$

where τ is the average treatment effect.

4.3 Experimental Setup

All our experiments are performed using the Amazon Web Services (AWS) cloud-based Elastic Compute Cloud (EC2) accelerated computing instances with high frequency 2.5 GHz (base) Intel Xeon Scalable Processor, 32 vCPUs and 64 GBs of GPU memory. Our STCINet model is trained using Keras Functional API with a Tensorflow backend and has around 293,000 trainable parameters. We trained our model using Adam optimizer with exponential decay of $e^{-0.1}$ in the learning rate after 10 epochs. The model was trained on 60 epochs with an early stopping criteria and batch size of 64 for all experiments. After hyperparameter tuning, we found the best performance with loss weightages as $\lambda_1 = 0.25$ and $\lambda_2 = 0.75$. The dataset is not split into training and test sets as our goal is to get outcome predictions on intervened treatment variables which automatically fulfills the unseen data requirement at test time.

4.4 Ablation Study

We test the performance of multiple variants of our proposed method on the synthetic datasets to identify the optimal configurations for spatiotemporal causal inference under spatial interference and temporal confounding. These variants and their corresponding ATE and PEHE scores are given in Table 1 for data with and without spatial interference. Here, *STCINet*[†] refers to our predictive model without the *LFM* or attention modules, *STCINet* – *NA* refers to the spatiotemporal causal inference model with no attention (NA) mechanism, *STCINet* – *SA* refers to the spatiotemporal causal inference model with spatial attention (SA) mechanism, *STCINet* – *AG* refers to the spatiotemporal causal inference model with attention gating (AG) mechanism and *STCINet* refers to the spatiotemporal causal inference model with *LFM*, spatial attention and attention gating mechanism.

Observing the results on data with spatial interference in Table 1, we see that *STCINet*[†] yields lower *PEHE* error than *STCINet* for direct effect estimation (DATE). The exception of *STCINet*[†] can be attributed to the fact that direct treatment effect is only estimated on the treated region where spillovers are easy to capture or non-existent, therefore we see the simplest variant performing the best on it. In case of no spatial interference, we observe that *STCINet* gives the

lowest (best) *PEHE* error on direct, indirect and overall lagged treatment effects as compared to all its variants. It is also interesting to note that *STCINet-SA* yields the second best results for IATE and LATE errors, in capturing treatment effects in the absence of spillover or interference, demonstrating the potential of spatial attention. We discuss the comparison with state-of-the-art (SOTA) methods in the next section.

Table 1. Related work comparison and ablation study of our proposed model on the synthetic data without and with spatial interference. **Bold** → best results, underline → second best results.

Model	$DATE(\sqrt{\varepsilon_{PEHE}})$	$IATE(\sqrt{\varepsilon_{PEHE}})$	$LATE(\sqrt{\varepsilon_{PEHE}})$	ε_{RMSE}
Data without Spatial Interference				
Deconfounder [4]	1.8658	0.2008	0.3890	3.1450
G-Net [21]	1.3136	0.0365	0.0785	2.0510
Weather2Vec [35]	1.1575	0.0333	0.0784	<u>0.1640</u>
STCINet [†]	1.3129	0.0365	0.0785	0.1660
STCINet-NA	1.3120	0.0363	0.0783	0.1730
STCINet-SA	1.2877	<u>0.0337</u>	<u>0.0752</u>	0.2200
STCINet-AG	1.3140	0.0364	0.0785	0.1270
STCINet	<u>1.2665</u>	<u>0.0337</u>	0.0744	0.1690
Data with Spatial Interference				
Deconfounder [4]	2.6580	0.2969	0.5599	6.4580
G-Net [21]	1.4123	0.0382	0.0940	6.7750
Weather2Vec [35]	0.5103	0.0606	0.0863	0.3320
STCINet [†]	<u>1.1959</u>	<u>0.0311</u>	<u>0.0693</u>	0.2930
STCINet-NA	1.6182	0.0454	0.0971	0.2390
STCINet-SA	1.6178	0.0448	0.0970	1.2790
STCINet-AG	1.6179	0.0453	0.0972	0.6250
STCINet	1.5264	0.0249	0.0684	<u>0.2790</u>

4.5 Comparison with Baseline Methods

Here, we compare STCINet with three state-of-the-art methods as our baselines, which are divided into the following two categories.

Temporal Deconfounding Methods. These methods perform causal inference on time-series data in the presence of time-varying confounders. We consider two such works, (i) Latent factor model modified for single treatment (Deconfounder) [4] and (ii) Deep learning based inverse propensity score method (G-Net) [21] for calculating effects of time-varying continuous treatment. By reducing the spatial dimension, we apply these methods on our synthetic datasets.

Spatial Causal Inference Method. Here, we consider a recent spatial causal inference method (Weather2vec) introduced for non-local spatial confounding. We consider the Weather2vec-AVG variant (see details in [35]) that can capture immediate neighborhood interference on the treated regions. To implement this spatial method, we reduce the temporal dimension by considering all data as independent samples irrespective of their temporal sequence. We present the results from these baselines in Table 1.

In case of direct treatment effect estimation (DATE) for both synthetic datasets, we notice Weather2Vec gives the lowest PEHE error which shows its potential of estimating interference effects on the treated region, however, the method fails to capture the interference on untreated region on data with spatial interference, quantified by indirect treatment effects (IATE). This shows the inability of Weather2Vec to capture spatial interference or spillover effects, further highlighting the significance of STCINet for overcoming the limitation of Weather2Vec. In case of IATE on data without spatial interference, we see a marginal difference in STCINet’s and Weather2Vec results. Here it is important to note that this data does not possess spatial interference making it viable for Weather2Vec to have good estimations. Overall, our model yields lowest PEHE error for both datasets in case of lagged average treatment effect (LATE), that accounts for temporal lag in effect estimation over the entire spatial region. It is also interesting to see that G-Net’s performance, despite being a time-series method, is comparable to our model, however, Deconfounder performs poorly in all scenarios yielding the worst performance overall.

4.6 Case Study on Real-World Arctic Data

Here, we present a case study on real-world Arctic data where we estimate the causal influence of atmospheric processes on the Pan-Arctic sea ice concentrations (SIC), which have seen a continuous decline since 1979. This accelerated ice loss is prominently visible in Summers (JJA - June, July, August) where the minimum sea ice has reduced by more than 50% of what it was in 1979 [15]. While identifying the true causes of ice melt is a complex task due to multiple thermodynamic feedbacks, a recent study suggests that one of the drivers of early melt in two Arctic sub-regions, namely East Siberian Sea and Laptav Sea, is the increase of downward longwave radiations (LWDN), with high ice melt observed in 1990 and 2003 [16]. Another study suggests that the sum of sensible and latent heat flux (HFX) plays an important role in Arctic’s energy budget and has bidirectional causal links with LWDN and SIC [17]. Using STCINet and the data provided by [17], we estimate the effect of these regional LWDN radiations on SIC on both regional and Pan-Arctic level at a lead time of one month. We include HFX, considering it a potential confounder in our study. The region of interest for applying treatment are the Laptav and East Siberian seas, as shown in Fig. 5a. We set *update_factor* (see details in Sect. 2.1) to be -0.05 for LWDN as our intervened treatment, which implies 5% reduction in original LWDN values. Our trained STCINet model predicts an average of 4% annual increase and a 44% summer (JJA) increase in SIC in the Laptav and

East Siberian seas if LWDN values were reduced by 5% in that region. The direct treatment effect is visible in Fig. 5b where we see a 42-year average difference (increase) in Laptav and East Siberian SIC. Our findings not only align with literature on the negative role of longwave radiations on sea ice melt, but also quantify the relations by estimating the direct and lagged average causal effects. More importantly, our model is able to capture the anomalous behavior in 1990 (Fig. 5c) and 2003 (Fig. 5d) where we observe that the effects of LWDN are not just restrained to the region of interest, but also spatially interfere with other Pan-Arctic regions influencing regional SIC [16]. Through this case study, we demonstrate the potential of STCINet to provide insights into complex spatial and temporal relations of atmosphere and the ocean.

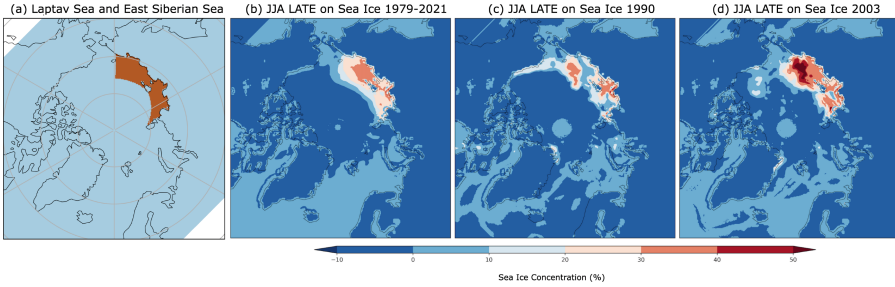


Fig. 5. Case study on climate data when longwave radiations are reduced by 5%. (a) Region of applying treatment, (b) Lagged Average treatment effect (LATE) on Summer SIC for 1979-2021, (c) LATE on Summer SIC for 1990, (d) LATE on Summer SIC for 2003.

5 Related Work

The existing work in spatiotemporal causal inference is still foundational with much focus on the theoretical aspects. Wang et al. proposed causal inference framework for panel data with spatial and temporal interference under stable unit (SUTVA) assumption [38]. Papadogeorgou et al. extended the potential outcome framework for point-process treatment and stochastic intervention [26]. Christiansen et al. proposed a non-parametric hypothesis test to develop causal models for multivariate spatiotemporal data [5]. Owing to the intricate nature of spatiotemporal causal variations, the problem is often broken down to either time-series by fixing a region of interest, or spatial causal inference in time-invariant settings [8, 34]. Here, we present the relevant literature in both domains.

Causal inference with Temporal Confounding. The challenge of confounding bias in time-series causal inference has received attention in recent literature. Notably, the techniques such as instrumental variables [36], propensity score matching [31], and recurrent neural networks [4, 23] address confounding when

estimating causal effects of time-invarying treatments. Few studies have explicitly addressed the joint challenge of estimating time-varying treatment effects while accounting for confounding bias in time-series data. Bica et al. attempted to bridge this gap, exploring methodologies that integrate time-series analysis with causal inference frameworks to disentangle multi-treatment effects from confounders in dynamic systems [4]. Their work struggles with single-cause treatment effect estimation. Recently, G-Net was proposed to tackle confounding of time-varying treatment using Long Short Term Memory (LSTM) model [21]. These methods majorly work on binary treatments and there remains a notable gap in methodologies capable of effectively addressing both time-varying treatment effects and confounding bias in time-series data due to the strong ignorability condition.

Causal inference for Spatial Interference. Most of the existing methods for spatial causal inference are basically spatial statistical techniques to study the interactions between spatial units in the presence of spatial confounding, spatial interference, or both [1, 29]. For instance, Graham et al. used Poisson regression with spatial predictors to model spatial confounding and interference, but their approach does not focus on quantifying causal relations in data [12]. Reich et al. and Giffin et al. both explored the utilization of spatial structure and generalized propensity scores to accommodate unmeasured confounding and interference [11, 29]. Wang et al. introduced a design-oriented framework for spatial experiments involving interference [39], while they later extended this to encompass spillover effects in panel data [38]. In fact, Di et al. first introduced a spatial hierarchical Difference-in-Differences model for policy evaluation [7], which Wang et al. delved further into design-based inference for spatial experiments considering unknown interference [39]. Most recently, Papadogeorgou et al. suggested a parametric method that concurrently tackles interference and bias arising from local and neighborhood unmeasured spatial confounding [27]. These spatial methods are majorly an extension of difference-in-difference technique, which is inapplicable in time-varying domain, or propensity score methods which are computationally expensive methods and infeasible for continuous treatment effects estimation.

Overall, there are several limitations of existing methods making them infeasible to offer generalized solutions for spatiotemporal data. Some of the limitations include: (i) limiting work to specific applications, for instance, point process or panel data, (ii) mistaking spatial confounding with spatial interference, (iii) limiting scope to binary treatments, and (iv) inability to handle continuous or time-varying treatment assignment on spatial data. Our proposed STCINet model overcomes these limitations offering a state-of-the-art technique to perform causal inference on spatiotemporal data.

6 Conclusion

In this paper, we presented our deep learning based potential outcome model for spatiotemporal causal inference. We utilized latent factor modeling to reduce

the bias due to time-varying confounding while leveraging the power of U-Net architecture and attention mechanism for capturing global and local spatial interference in data over time. Through empirical study on two synthetic datasets, we compared our method with state-of-the-art spatial and temporal inference methods to quantify direct (DATE), indirect (IATE), and lagged effects (LATE) on spatiotemporal data. We further provided a case study on real-world climate dataset and demonstrated the effectiveness of our proposed approach on quantifying the direct (sub-regional) and indirect (regional) effects of longwave radiations on sea ice concentration, paving paths for atmospheric scientists to adopt data driven methods to unravel important climate patterns. In the future, we would extend our work to estimate spatiotemporal causal inference in the presence of latent and spatially varying confounders.

Acknowledgement. This work is supported by NSF grants: CAREER: Big Data Climate Causality (OAC-1942714) and HDR Institute: HARP - Harnessing Data and Model Revolution in the Polar Regions (OAC-2118285).

References



1. Akbari, K., Winter, S., Tomko, M.: Spatial causality: a systematic review on spatial causal inference. *Geogr. Anal.* **55**(1), 56–89 (2023)
2. Ali, S., Faruque, O., Wang, J.: Quantifying causes of arctic amplification via deep learning based time-series causal inference. arXiv preprint [arXiv:2303.07122](https://arxiv.org/abs/2303.07122) (2023)
3. Bica, I., Alaa, A., Van Der Schaar, M.: Time series deconfounder: Estimating treatment effects over time in the presence of hidden confounders. In: *International Conference on Machine Learning*, pp. 884–895. PMLR (2020)
4. Bica, I., Alaa, A.M., Jordon, J., van der Schaar, M.: Estimating counterfactual treatment outcomes over time through adversarially balanced representations. arXiv preprint [arXiv:2002.04083](https://arxiv.org/abs/2002.04083) (2020)
5. Christiansen, R., Baumann, M., Kuemmerle, T., Mahecha, M.D., Peters, J.: Toward causal inference for spatio-temporal data: conflict and forest loss in Colombia. *J. Am. Stat. Assoc.* **117**(538), 591–601 (2022)
6. Cole, S.R., Hernán, M.A.: Constructing inverse probability weights for marginal structural models. *Am. J. Epidemiol.* **168**(6), 656–664 (2008)
7. Di Gennaro, D., Pellegrini, G., et al.: Policy evaluation in presence of interferences: A spatial multilevel did approach (2016)
8. Ebert-Uphoff, I., Deng, Y.: Causal discovery from spatio-temporal data with applications to climate science. In: *2014 13th International Conference on Machine Learning and Applications*, pp. 606–613. IEEE (2014)
9. Forastiere, L., Airoidi, E.M., Mealli, F.: Identification and estimation of treatment and interference effects in observational studies on networks. *J. Am. Stat. Assoc.* **116**(534), 901–918 (2021)
10. Ghojogh, B., Ghodsi, A., Karray, F., Crowley, M.: Factor analysis, probabilistic principal component analysis, variational inference, and variational autoencoder. *Tutorial and Survey* (2021)
11. Giffin, A., Reich, B.J., Yang, S., Rappold, A.G.: Generalized propensity score approach to causal inference with spatial interference. *Biometrics* **79**(3), 2220–2231 (2022)

12. Graham, D.J., McCoy, E.J., Stephens, D.A.: Quantifying the effect of area deprivation on child pedestrian casualties by using longitudinal mixed models to adjust for confounding, interference and spatial dependence. *J. R. Stat. Soc. Ser. A Stat. Soc.* **176**(4), 931–950 (2013)
13. Hernán, M.A., Robins, J.M.: Estimating causal effects from epidemiological data. *J. Epidemiol. Commun. Health* **60**(7), 578–586 (2006)
14. Hill, J.L.: Bayesian nonparametric modeling for causal inference. *J. Comput. Graph. Stat.* **20**(1), 217–240 (2011)
15. Holland, M.M., Landrum, L., Bailey, D., Vavrus, S.: Changing seasonal predictability of arctic summer sea ice area in a warming climate. *J. Clim.* **32**(16), 4963–4979 (2019)
16. Huang, Y., Dong, X., Xi, B., Deng, Y.: A survey of the atmospheric physical processes key to the onset of arctic sea ice melt in spring. *Clim. Dyn.* **52**(7), 4907–4922 (2019)
17. Huang, Y., Kleindessner, M., Munishkin, A., Varshney, D., Guo, P., Wang, J.: Benchmarking of data-driven causality discovery approaches in the interactions of arctic sea ice and atmosphere. *Front. Big Data* **4**, 642182 (2021)
18. Imbens, G.W., Rubin, D.B.: *Causal inference in statistics, social, and biomedical sciences*. Cambridge University Press (2015)
19. Jetley, S., Lord, N.A., Lee, N., Torr, P.H.: Learn to pay attention. arXiv preprint [arXiv:1804.02391](https://arxiv.org/abs/1804.02391) (2018)
20. Koch, B., Sainburg, T., Geraldo, P., Jiang, S., Sun, Y., Foster, J.G.: Deep learning of potential outcomes. arXiv preprint [arXiv:2110.04442](https://arxiv.org/abs/2110.04442) (2021)
21. Li, Ret al.: G-net: a recurrent network approach to g-computation for counterfactual prediction under a dynamic treatment regime. In: *Machine Learning for Health*, pp. 282–299. PMLR (2021)
22. Lok, J.J.: Defining and estimating causal direct and indirect effects when setting the mediator to specific values is not feasible. *Stat. Med.* **35**(22), 4008–4020 (2016)
23. Moraffah, R., et al.: Causal inference for time series analysis: problems, methods and evaluation. *Knowl. Inform. Syst.*, 1–45 (2021)
24. Nauta, M., Bucur, D., Seifert, C.: Causal discovery with attention-based convolutional neural networks. *Mach. Learn. Knowl. Extraction* **1**(1), 19 (2019)
25. Oktay, O., et al.: Attention u-net: Learning where to look for the pancreas. arXiv preprint [arXiv:1804.03999](https://arxiv.org/abs/1804.03999) (2018)
26. Papadogeorgou, G., Imai, K., Lyall, J., Li, F.: Causal inference with spatio-temporal data: estimating the effects of airstrikes on insurgent violence in iraq. *J. R. Stat. Soc. Ser. B Stat Methodol.* **84**(5), 1969–1999 (2022)
27. Papadogeorgou, G., Samanta, S.: Spatial causal inference in the presence of unmeasured confounding and interference. arXiv preprint [arXiv:2303.08218](https://arxiv.org/abs/2303.08218) (2023)
28. Pearl, J.: Simpson’s paradox, confounding, and collapsibility. *Causality: Models Reasoning Inference*, 173–200 (2009)
29. Reich, B.J., Yang, S., Guan, Y., Giffin, A.B., Miller, M.J., Rappold, A.: A review of spatial causal inference methods for environmental and epidemiological applications. *Int. Stat. Rev.* **89**(3), 605–634 (2021)
30. Ripley, B.D.: *Statistical inference for spatial processes*. Cambridge university press (1988)
31. Robins, J.M., Hernan, M.A., Brumback, B.: Marginal structural models and causal inference in epidemiology. *Epidemiology*, 550–560 (2000)
32. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F.

- (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
33. Rubin, D.B.: Causal inference using potential outcomes: design, modeling, decisions. *J. Am. Stat. Assoc.* **100**(469), 322–331 (2005)
 34. Runge, J., et al.: Inferring causation from time series in earth system sciences. *Nat. Commun.* **10**(1), 1–13 (2019)
 35. Tec, M., Scott, J.G., Zigler, C.M.: Weather2vec: representation learning for causal inference with non-local confounding in air pollution and climate studies. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, pp. 14504–14513 (2023)
 36. Thams, N., Søndergaard, R., Weichwald, S., Peters, J.: Identifying causal effects using instrumental time series: Nuisance iv and correcting for the past. arXiv preprint [arXiv:2203.06056](https://arxiv.org/abs/2203.06056) (2022)
 37. Vaswani, A., et al.: Attention is all you need. *Adv. Neural Inform. Processing Syst.* **30** (2017)
 38. Wang, Y.: Causal inference under temporal and spatial interference. arXiv preprint [arXiv:2106.15074](https://arxiv.org/abs/2106.15074) (2021)
 39. Wang, Y., Samii, C., Chang, H., Aronow, P.: Design-based inference for spatial experiments with interference. arXiv preprint [arXiv:2010.13599](https://arxiv.org/abs/2010.13599) (2020)
 40. Yao, L., Chu, Z., Li, S., Li, Y., Gao, J., Zhang, A.: A survey on causal inference. *ACM Trans. Knowl. Dis. Data (TKDD)* **15**(5), 1–46 (2021)



Continuous Geometry-Aware Graph Diffusion via Hyperbolic Neural PDE

Jiaxu Liu¹ , Xinpeng Yi² , Sihao Wu¹, Xiangyu Yin¹, Tianle Zhang¹, Xiaowei Huang¹, and Shi Jin²

¹ Department of Computer Science, University of Liverpool, Liverpool, UK
{jiaxu.liu, sihao.wu, xiangyu.yin, tianle.zhang, xiaowei.huang}@liverpool.ac.uk

² National Mobile Communications Research Laboratory, Southeast University, Nanjing, China
{xyi, jinshi}@seu.edu.cn

Abstract. While Hyperbolic Graph Neural Network (HGNN) has recently emerged as a powerful tool dealing with hierarchical graph data, the limitations of scalability and efficiency hinder itself from generalizing to deep models. In this paper, by envisioning depth as a continuous-time embedding evolution, we decouple the HGNN and reframe the information propagation as a partial differential equation, letting node-wise attention undertake the role of diffusivity within the Hyperbolic Neural PDE (HPDE). By introducing theoretical principles *e.g.*, field and flow, gradient, divergence, and diffusivity on a non-Euclidean manifold for HPDE integration, we discuss both implicit and explicit discretization schemes to formulate numerical HPDE solvers. Further, we propose the Hyperbolic Graph Diffusion Equation (HGDE) – a flexible vector flow function that can be integrated to obtain expressive hyperbolic node embeddings. By analyzing potential energy decay of embeddings, we demonstrate that HGDE is capable of modeling both low- and high-order proximity with the benefit of local-global diffusivity functions. Experiments on node classification and link prediction and image-text classification tasks verify the superiority of the proposed method, which consistently outperforms various competitive models by a significant margin.

Keywords: Continuous GNN · Hyperbolic Space · Neural ODE

1 Introduction

Graphs play a vital role in various disciplines, including social network analysis [12], bioinformatics [48], and computer vision [37]. The advent of Graph Neural Networks (GNNs, [23]) has significantly enhanced the analysis of these structures to capture complex relationships between nodes in a graph. However, traditional GNNs operate within the borders of Euclidean space, which may not be sufficiently expressive for data with inherent hierarchical or complex structures. To improve, this paper delves into the realm of hyperbolic geometry, a Riemannian

manifold demonstrated to be particularly effective for embedding hierarchical data [16, 18]. We focus on the development of HGNNs [6], which leverage the unique properties of hyperbolic space to enhance the embedding of GNNs.

The principal challenge confronted by HGNNs is their architectural design, which primarily consists of combinations of aggregation and transformation within layers. This fusion presents a unique problem, particularly the difficulty of training attention weights and manifold parameters (*e.g.*, curvature of the hyperbolic manifold) layer-wise in a **deeply** layered scheme. With such challenge, we pose our initial questions: **Q1**: *Considering hyperbolic space slows down layer-wise attention and propagation [11, 25], how to develop a deeply-layered attentive HGNN?* **Q2**: *How to incorporate high-order info to benefit a deeply layered scheme?* **Q3**: *Deep GNNs suffer from embedding smoothing, how should the node smoothness be measured when there is no defined metric for hyperbolic smoothness. And how to tackle over-smoothing within hyperbolic manifold constraints?*

Motivated by above questions, in this paper, we propose to decouple the functions within layers of HGNNs so as to deal with each of them separately. Unlike traditional decoupling-GNN approaches [15, 35] that aggregate all information from the neighbors, we view information propagation as a distillation process, such that unimportant information is filtered out and significant information is weighted and contributes to the continuous variation of embeddings. More explicitly, by letting the transformation layer manifest as an encoder-decoder scheme, the aggregation layer is re-envisioned to solve the partial differential equation (Neural ODE/PDE, [8]) - essentially, the graph diffusion equation [4] in hyperbolic space, which essentially simulates an infinitely deep HGNN with single layer parameters. In specific, in response to **Q1**, we consider the PDE reformulation and developed Hyperbolic-PDE (HPDE) solvers, which only leverage single-layer parameters. To answer **Q2**, we formulate the Hyperbolic Graph Diffusion Equation (HGDE), a low-high order vector flow function that can be integrated by HPDE. Tackling **Q3**, we firstly introduce the hyperbolic adaptation of Dirichlet energy and augmented HGDE with a hyperbolic residual, powered by Poincaré midpoint. Deconstructions above introduce extensive mathematical principles, including for instance: manifold vector field, flow, gradient, divergence, diffusivity, numerical HPDE solvers and hyperbolic residuals for bounding embedding energy decay. Through these concepts, we open new pathways to fully exploit the unique potential of hyperbolic space in the contextual analysis of graph-based data. In summary, the contributions of this paper are listed as follows.

(I) We present the geometric intuition for designing projective numerical integration methods that solve hyperbolic ODE/PDE, and examine the connection to Riemannian gradient descent methods. Focusing on fixed-grid solvers, we derive both hyperbolic generalizations of explicit schemes (Euler, Runge-Kutta) and implicit schemes (Adams-Moulton).

(II) We formulate the HGDE, which acts as the *vector flow* of the HPDE, and thereby induces concepts such as gradient, divergence and diffusivity within HGDE. The proposed framework is flexible and efficient for generating expressive (endowed by the depth) hyperbolic graph embeddings.

(III) We instantiate the diffusivity function as a mixed-order multi-head attention to account for both homophilic (local) and heterophilic (global) relations. Besides, we introduce hyperbolic residual technique to benefit the optimization and prevent over-smoothing.

Through extensive experiments and comparison with the state-of-the-art on multiple real-world datasets, we show that HGDE framework can not only learn comparably high-quality node embeddings as Euclidean models on non-hierarchical datasets, but outperform all compared hyperbolic models variants on highly-hierarchical datasets with improved efficiency and accuracy. [The code and appendix can be found in https://github.com/ljxw88/HyperbolicGDE.](https://github.com/ljxw88/HyperbolicGDE)

2 Preliminaries

Riemannian Geometry and Hyperbolic Space. A Riemannian manifold \mathcal{M} of n -dimension is a topological space associated with a metric tensor g , denoted as (\mathcal{M}, g) , which extends curved surfaces to higher dimensions and can be locally approximated by \mathbb{R}^n . At any point $\mathbf{x} \in \mathcal{M}$, the tangent space $\mathcal{T}_{\mathbf{x}}\mathcal{M} \cong \mathbb{R}^n$ represents the first-order approximation of a small perturbation around \mathbf{x} , isomorphic to Euclidean space. The Riemannian metric g on the manifold determines a smoothly varying positive definite inner product on the tangent space, enabling the definition of diverse properties *e.g.* geodesic length, angles, and curvature.

The hyperbolic space \mathbb{H}^n is a smooth Riemannian manifold with a constant negative sectional curvature $\kappa < 0$. Its coordinates can be represented via various isometric models. [3] established the equivalence of hyperbolic and Euclidean geometry through the utilization of the n -dimensional *Poincaré ball model*, which equips an open ball $\mathbb{D}_{\kappa}^n = (\mathcal{D}_{\kappa}^n, g^{\mathbb{D}})$, with point set $\mathcal{D}_{\kappa}^n = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}\| < -\frac{1}{\kappa}\}$ and Riemannian metric $g_{\mathbf{x}}^{\mathbb{D}} = (\lambda_{\mathbf{x}}^{\kappa})^2 \mathbf{I}_n$, where the conformal factor $\lambda_{\mathbf{x}}^{\kappa} = \frac{2}{1+\kappa\|\mathbf{x}\|^2}$. The Poincaré metric tensor induces various geometric properties *e.g.* distances $d_{\mathbb{D}}^{\kappa}(\mathbf{x}, \mathbf{y})$, inner products $\langle \mathbf{u}, \mathbf{v} \rangle_{\mathbf{x}}^{\kappa}$, geodesics $\gamma_{\mathbf{x} \rightarrow \mathbf{y}}(t)$ and more [26]. Geodesics also induce the definition of *exponential* and *logarithmic* maps [13]. At point $\mathbf{x} \in \mathbb{D}_{\kappa}^n$, the exponential map $\exp_{\mathbf{x}}^{\kappa} : \mathcal{T}_{\mathbf{x}}\mathbb{D}_{\kappa}^n \rightarrow \mathbb{D}_{\kappa}^n$ essentially maps a small perturbation of \mathbf{x} by $\mathbf{v} \in \mathcal{T}_{\mathbf{x}}\mathbb{D}_{\kappa}^n$ to $\exp_{\mathbf{x}}^{\kappa}(\mathbf{v}) \in \mathbb{D}_{\kappa}^n$, so that $t \in [0, 1] : \exp_{\mathbf{x}}^{\kappa}(t\mathbf{v})$ is the geodesic from \mathbf{x} to $\exp_{\mathbf{x}}^{\kappa}(\mathbf{v})$. The logarithmic map $\log_{\mathbf{x}}^{\kappa} : \mathbb{D}_{\kappa}^n \rightarrow \mathcal{T}_{\mathbf{x}}\mathbb{D}_{\kappa}^n$ is defined as the inverse of $\exp_{\mathbf{x}}^{\kappa}$. Finally, the parallel transport $\mathcal{PT}_{\mathbf{x} \rightarrow \mathbf{y}} : \mathcal{T}_{\mathbf{x}}\mathbb{D}_{\kappa}^n \rightarrow \mathcal{T}_{\mathbf{y}}\mathbb{D}_{\kappa}^n$ moves a tangent vector $\mathbf{v} \in \mathcal{T}_{\mathbf{x}}\mathbb{D}_{\kappa}^n$ along the geodesic to $\mathcal{T}_{\mathbf{y}}\mathbb{D}_{\kappa}^n$ while preserving the metric tensor. For closed-form expression of above operations, please refer to Appendix B.

Diffusion Equations. The process of generating representations of individual data points through information flows can be characterized by an an-isotropic *diffusion* process, a concept borrowed from physics used to describe heat diffusion on Riemannian manifold. Denote the manifold as \mathcal{M} , and let $z(t)$ denote a family of functions on $\mathcal{M} \times [0, \infty)$ and $z(u, t)$ be the density at location $u \in \mathcal{M}$ and times t . The general framework of diffusion equations is expressed as a PDE

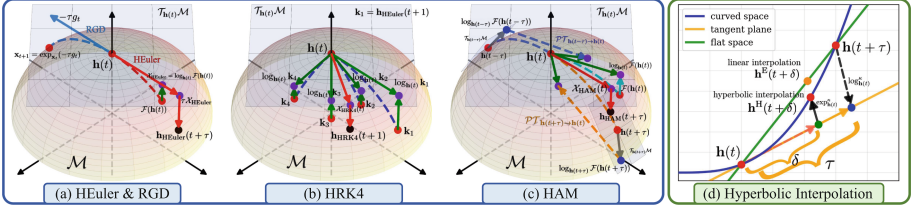


Fig. 1. (a–c) Illustration of various numerical integration methods with comparison to RGD. In each time-step, an explicit scheme calibrates the vector field within only the tangent space of time t , while an implicit scheme requires multiple tangent spaces to estimate future slopes, thus requiring parallel transport for aligning the directions of vectors in different spaces. (d) Illustration of hyperbolic interpolation method.

$$\partial z(u, t) / \partial t = \text{div}(a(z(u, t)) \nabla z(u, t)), \quad t > 0 \tag{1}$$

where $a(\cdot)$ defines the *diffusivity* function controlling the diffusion strength between any location pair at time t . The gradient operator $\nabla : \mathcal{M} \rightarrow \mathcal{TM}$ describes the steepest change at point $u \in \mathcal{M}$. $\text{div}(\cdot) : \mathcal{TM} \rightarrow \mathcal{M}$ is the divergence operator that summarizes the flow of the diffusivity-scaled vector field $(a(\cdot) \nabla)$. Equation (1) can be physically viewed as a variation of heat based on time at the location i , identical to the heat that flows through that point from the surrounding areas.

Graph Diffusion Equation. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote an undirected graph with the node set \mathcal{V} and the edge set \mathcal{E} . Let $\mathbf{x} = \{\mathbf{x}_i \in \mathbb{R}^d\}_{i=1}^{|\mathcal{V}|}$ be the node features and $\mathbf{z}(t)$ be node embeddings at time t . Process Eq. (1) can be re-written as

$$\partial \mathbf{z}_i(t) / \partial t = \text{div}(\mathbf{A}(\mathbf{z}(t)) \nabla \mathbf{z}_i(t)), \tag{2}$$

where \mathbf{A} is generally realised by a time-independent $n \times n$ attention matrix [4, 5], consistent with the flow of *heat flux* in/out node i . The formulation of Eq. (2) as a PDE allows leveraging vast existing numerical integration methods to solve the continuous dynamics.

3 Hyperbolic Numerical Integrators

Consider the continuous form of ODE/PDE specified by a neural network parameterized by θ , expressed as

$$d\mathbf{h}(t) / dt = f_\theta(\mathbf{h}(t), t), \quad \mathbf{h}(0) = \mathbf{h}_0 \tag{3}$$

where the time step $t = [0, T]$. Equation (3) essentially tells that the *rate of change* of $\mathbf{h}(t) \in \mathbb{R}^n$ at each time step is given by the vector field $f_\theta : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}^n$. Equation (3) is integrated to obtain $\mathbf{h}(T)$. In our context, we are interested in formulating a PDE recipe that is aware of hyperbolic geometry.

Definition 1. A *time-dependent manifold vector field* is a mapping $\mathcal{X} : \mathcal{M} \times \mathbb{R} \rightarrow \mathcal{TM}$, which assigns each point in \mathcal{M} at t a tangent vector. The particle's time-evolution according to \mathcal{X} is then given by the following PDE

$$d\mathbf{h}(t)/dt = \mathcal{X}_\theta(\mathbf{h}(t), t). \quad (4)$$

Definition 2. A *vector flow* is a mapping generated by vector field, i.e. $\mathcal{F} \equiv \pi(\mathcal{X})$, where $\pi : \mathcal{M} \rightarrow \mathcal{M}$ is a smooth projection of vector field to manifold of their local coordinates. Vice versa, if π is a diffeomorphism, then $\mathcal{X} \equiv \pi^{-1}(\mathcal{F})$.

In hyperbolic geometry, where π and π^{-1} are properly defined exp and log maps, our concern lies in the particle's location on the manifold subsequent to integration, i.e. integrate through the path defined by flow \mathcal{F} . This can be achieved via the spirit of projective method [17]. In the following, we derive numerical solvers for estimating the integral of field \mathcal{X} or flow \mathcal{F} w.r.t. time t using, respectively, the explicit and implicit schemes.

3.1 Hyperbolic Projective Explicit Scheme

In an explicit scheme, the state at the next time step is computed directly from the current state and its derivatives. In this part, we derive the hyperbolic generalization of the explicit scheme. To illustrate high-level ideas, we introduce both *single step method* and *multi-step method*. We also discuss the geometric intuition and strong analogy between one-step explicit scheme and Riemannian gradient descent (RGD).

H-Explicit Euler (HEuler). Consider a small time step τ . Iteratively, we seek an approximation for $\mathbf{h}(t + \tau)$ based on $\mathbf{h}(t)$ and vector field $f(\cdot)$. In Euclidean space, the explicit Euler method is written as

$$\mathbf{h}(t + \tau) \approx \mathbf{h}(t) + \tau f_\theta(\mathbf{h}(t), t), \quad (5)$$

which is a discrete version of Eq. (3). Similarly in hyperbolic space, we discretize Eq. (4), and have the stepping function formulated as

$$\mathbf{h}_{\text{HEuler}}(t + \tau) = \exp_{\mathbf{h}(t)}^\kappa(\tau \mathcal{X}_{\text{HEuler}}(t)), \quad (6)$$

where the vector field \mathcal{X} gives the direction at time t according to flow $\mathcal{F}_\theta^\kappa$

$$\mathcal{X}_{\text{HEuler}}(t) = \log_{\mathbf{h}(t)}^\kappa(\mathcal{F}_\theta^\kappa(\mathbf{h}(t), t)) \in \mathcal{T}_{\mathbf{h}(t)}\mathbb{D}_\kappa^n. \quad (7)$$

Geometric Intuition. The equation in Eq. (5) signifies a transition from $\mathbf{h}(t)$ in the direction of f by a distance proportional to τ . In hyperbolic space, where $\mathbf{h}(t) \in \mathbb{D}_\kappa^n$ and we presume $\mathcal{X}^\kappa : \mathbb{D}_\kappa^n \rightarrow \mathcal{T}\mathbb{D}_\kappa^n$, the transition follows the geodesic dictated by the direction of \mathcal{X}^κ . Recall the definition of exponential map: given $x \in \mathbb{D}_\kappa$, $\exp_x^\kappa(v)$ takes $v \in \mathcal{T}_x\mathbb{D}_\kappa$ and returns a point in \mathbb{D}_κ reached by moving from x along the geodesic determined by the tangent vector v . Thus Eqs. (6–7)

can be essentially viewed as a geometric transportation of points on manifold along the curve defined by \mathcal{F} .

Connection to RGD. As visualized in Fig. 1(a), the explicit Euler can be viewed as reversed RGD, where the direction $\mathcal{X}_{\text{HEuler}}(t)$ plays similar role as the Riemannian gradient g_t at $\mathbf{h}(t)$. Similar to RGD, when (\mathcal{M}, ρ) is Euclidean space $(\mathbb{R}^n, \mathbf{I}_n)$, then Eq. (6) converges to Eq. (5) since we have $\exp_h^\kappa(v) \xrightarrow{\kappa \rightarrow 0} h + v$. This property is useful on developing higher-order integrators.

H-Runge-Kutta (HRK). With a similar geometric intuition, we derive the hyperbolic extension of the Runge-Kutta method. Define the s -order HRK stepping function

$$\mathbf{h}_{\text{HRK}}(t + \tau) = \exp_{\mathbf{h}(t)}^\kappa(\tau \mathcal{X}_{\text{HRK}}(t)), \tag{8}$$

where the vector field is estimated by

$$\mathcal{X}_{\text{HRK}}(t) = \left(\sum_{i=1}^s \phi_i \log_{\mathbf{h}(t)}^\kappa(\mathbf{k}_i) \right) / \sum_{i=1}^s \phi_i. \tag{9}$$

In Eq. (9), \mathbf{k} denotes the vector flow functions, $\{\phi_i\}$ are coefficients determined by the order. Specifically for 4th order Runge-Kutta (HRK4), we have $\{\phi_{1\dots4}\} = \{1, 3, 3, 1\}$ derived from Taylor series expansion as in [8]. The vector flows $\mathbf{k}_{1\dots4}$ are respectively formulated by

$$\mathbf{k}_1 = \mathbf{h}_{\text{HEuler}}(t + \tau), \tag{Eq. (6)} \tag{10}$$

$$\mathbf{k}_2 = \mathcal{F}_\theta^\kappa(\exp_{\mathbf{h}(t)}^\kappa(\tau \mathcal{X}_{\mathbf{k}_2}), t + \tau/3), \text{ where } \mathcal{X}_{\mathbf{k}_2} = \log_{\mathbf{h}(t)}^\kappa(\mathbf{k}_1)/3.$$

$$\mathbf{k}_3 = \mathcal{F}_\theta^\kappa(\exp_{\mathbf{h}(t)}^\kappa(\tau \mathcal{X}_{\mathbf{k}_3}), t + 2\tau/3), \text{ where } \mathcal{X}_{\mathbf{k}_3} = \log_{\mathbf{h}(t)}^\kappa(\mathbf{k}_2) - \log_{\mathbf{h}(t)}^\kappa(\mathbf{k}_1)/3.$$

$$\mathbf{k}_4 = \mathcal{F}_\theta^\kappa(\exp_{\mathbf{h}(t)}^\kappa(\tau \mathcal{X}_{\mathbf{k}_4}), t + \tau), \text{ where } \mathcal{X}_{\mathbf{k}_4} = \log_{\mathbf{h}(t)}^\kappa(\mathbf{k}_1) - \log_{\mathbf{h}(t)}^\kappa(\mathbf{k}_2) + \log_{\mathbf{h}(t)}^\kappa(\mathbf{k}_3).$$

As illustrated in Fig. 1(b), this method approximates the solution to the PDE within a small interval, considering not only the derivative at the initial time (as in Eq. (5)), but also at intermediate points and the end of the interval.

3.2 Hyperbolic Projective Implicit Scheme

In an implicit scheme, the state of the next iteration is computed by incorporating its own value. This requires solving a linear system to obtain $\mathbf{h}(t + \tau)$ based on $\mathbf{h}(t)$. In below, we illustrate a hyperbolic generalization of the implicit solver.

H-Implicit Adams-Moulton (HAM). Adams numerical integration methods are introduced as families of multi-step methods. With order $s = 0$, Adams methods are identical to the Euler’s method. Principally, there are two types of Adams methods, namely, Adams-Bashforth (explicit) and Adams-Moulton (implicit). Our emphasis is on the latter.

The implicit nature of AM requires the initialization of first several steps with a different method. We use the hyperbolic Runge-Kutta (Eq. (8)) for initialization. With the input $\mathbf{h}(t) \in \mathbb{D}_\kappa^n$ and flow \mathcal{F}^κ , define the warm up

$$\mathbf{h}_{\text{HAM}}(i\tau) = \mathbf{h}_{\text{HRK4}}(i\tau), \quad 0 \leq i < s_{\min} \quad (11)$$

where s_{\min} is the min order. During the whole warm up process, we maintain a queue \mathbf{q} of tangent vectors and the points spanning the tangent space. In each time step of Eq. (11), we push $[q_0 = \mathcal{X}_{\text{RK4}}(i\tau), q_1 = \mathbf{h}(i\tau)]$ to the head of \mathbf{q} . When $\text{len}(\mathbf{q}) \geq s_{\min}$, we start the time-stepping

$$\mathbf{h}_{\text{HAM}}(t + \tau) = \exp_{\mathbf{h}(t)}^{\kappa}(\tau \mathcal{X}_{\text{HAM}}(t)), \quad (12)$$

where the vector field is expressed as

$$\begin{aligned} \mathcal{X}_{\text{HAM}}(t) = & \phi_0 \mathcal{PT}_{\mathbf{h}(t+\tau) \rightarrow \mathbf{h}(t)}(\log_{\mathbf{h}(t+\tau)}^{\kappa}(\mathcal{F}_{\theta}^{\kappa}(\mathbf{h}(t + \tau), t + \tau))) \\ & + \sum_{i=1}^s \phi_i \mathcal{PT}_{\mathbf{q}_{i,1} \rightarrow \mathbf{h}(t)}(\mathbf{q}_{i,0}). \end{aligned} \quad (13)$$

The order $s = \min(\text{len}(\mathbf{q}), s_{\max})$, $\{\phi_i\}$ are coefficients determined by the order, which are typically within a pre-defined look-up table. As illustrated in Fig. 1(c), since the reference point $\mathbf{h}(t)$'s stored in \mathbf{q} are different, the parallel transport \mathcal{PT} is leveraged for aligning tangent spaces for different slopes. When $\mathbf{h}_{\text{HAM}}(t + \tau)$ is accepted as converged, $[\log_{\mathbf{h}(t)}^{\kappa}(\mathbf{h}_{\text{HAM}}(t + \tau)), \mathbf{h}(t)]$ is pushed to \mathbf{q} for the next iteration and the last element is popped if $\text{len}(\mathbf{q})$ reaches s_{\max} . We refer readers to Appendix C for detailed explanation of the algorithms.

3.3 Interpolation on Curved Space

Fixed grid PDE solvers typically use their own internal step sizes τ to advance the solution of the PDE. For certain time step t , given $\mathbf{h}(t)$ and $\mathbf{h}(t + \tau)$, we may want to obtain the solution at time point $t + \delta$ where $0 < \delta < \tau$. Since δ does not lie on the grid defined by $\{0, \tau\}$, interpolation methods are invoked to estimate $\mathbf{h}(t + \delta)$. For hyperbolic geometry that $\mathbf{h} \in \mathbb{D}_{\kappa}^n$, define the interpolation

$$\mathbf{h}(t + \delta) = \exp_{\mathbf{h}(t)}^{\kappa} \left(\delta \log_{\mathbf{h}(t)}^{\kappa}(\mathbf{h}(t + \tau)) / \tau \right). \quad (14)$$

Proposition 1 (Proved in Appendix D). *For any step size $0 < \delta < \tau$, the interpolation $\mathbf{h}(t + \delta)$ via Eq. (14) is on the geodesic between $\mathbf{h}(t)$ and $\mathbf{h}(t + \tau)$ on the manifold, and $\frac{d_{\mathbb{D}}^{\kappa}(\mathbf{h}(t), \mathbf{h}(t + \delta))}{d_{\mathbb{D}}^{\kappa}(\mathbf{h}(t), \mathbf{h}(t + \tau))} = \frac{\delta}{\tau}$ where $d_{\mathbb{D}}^{\kappa}$ is the geodesic length.*

4 Diffusing Graphs in Hyperbolic Space

4.1 Hyperbolic Graph Diffusion Equation

We study the diffusion process of graphs with node representation residing in hyperbolic geometry. Given the diffusion time $t \in [0, T]$, embedding space $\mathbb{D}_{\kappa_t}^d$ with learnable curvature κ_t at time t , node embedding $\mathbf{z}_*(t) \in \mathbb{D}_{\kappa_t}^d$ and $\mathcal{C}(\cdot)$ being the correlated coordinates of certain node, we formulate the vector flow $\mathcal{F}_{\theta}^{\kappa}$ of the i th representation as

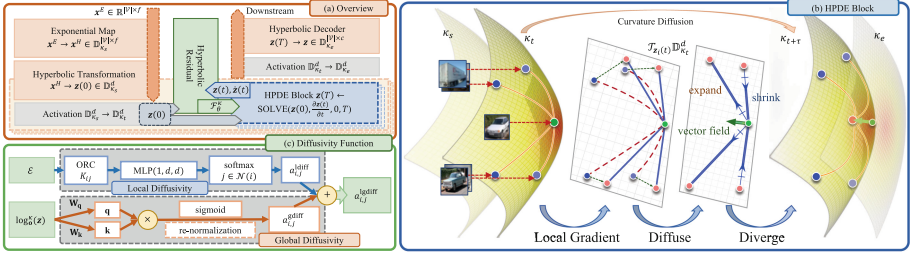


Fig. 2. Schematic of HGDE. (a) The pipeline of our method includes hyperbolic projection, feature transformation, and HPDE block that integrates the GDE. After that, a decoder is applied to the embeddings for specific downstream tasks. (b) The visualization of the diffusion process within the HPDE block: first, map local gradients of \mathbf{z}_i onto the tangent space, calculate the diffusivity, and diverge to obtain the vector flow (green arrow), then perform one-step integration on the manifold with the guidance of continuous curvature diffusion. (c) The details of attention-powered local-global diffusivity function. (Color figure online)

$$\underbrace{\exp^{\kappa_t}_{\mathbf{z}_i(t)}}_{\text{divergence}} \left(\sigma \left[\underbrace{\sum_{j \in \mathcal{C}(i)} a(\mathbf{z}_i(t), \mathbf{z}_j(t))}_{\text{diffusivity}} \underbrace{\log^{\kappa_t}_{\mathbf{z}_i(t)}(\mathbf{z}_j(t))}_{\text{gradient}} \right] \right), \quad (15)$$

where σ can be either identity/non-linear activation. With initial state encoded by learnable feature transformation ψ , *i.e.* $\mathbf{z}(0) = \psi(\mathbf{x}) \in \mathbb{D}_{\kappa}^d$, the final state can be numerically estimated by our proposed HPDE integrators, *i.e.* $\mathbf{z}_i(T) = \text{HPDESolve}(\mathbf{z}_i(0), \frac{\partial \mathbf{z}_i(t)}{\partial t}, 0, T)$. In matrix form, the vector flow is expressed as

$$\mathcal{F}_{\theta}^{\kappa}(\mathbf{z}(t), t) = \exp^{\kappa_t}_{\mathbf{z}(t)} \left(\sigma[\mathbf{S}(\mathbf{z}(t))\nabla\mathbf{z}(t)] \right), \quad (16)$$

where $\mathbf{S}(\mathbf{z}(t)) = (a(\mathbf{z}_i(t), \mathbf{z}_j(t)))$ is a normalized $|\mathcal{V}| \times |\mathcal{V}|$ similarity matrix, and $\nabla\mathbf{z}(t)_{i,j} := \log^{\kappa_t}_{\mathbf{z}_i(t)}(\mathbf{z}_j(t))$. In below, we discuss the key ingredients of Eq. (15, 16).

Gradient. The gradient of a function $z(u, t)$ at location u in a discrete space can be approximated as the difference between the function values at neighboring points. In graph space, let \mathbf{z}_i and $\{\mathbf{z}_j\}_{j \in \mathcal{C}(i)}$, respectively, denote the target node and the correlated positions of i that can be modeled by edge connectivity or self-attention. The graph diffusion process [4, 5] treats nodes as Euclidean representations, such that the analogy of gradient operator $(\nabla\mathbf{z}(t))_{i,j} : \mathbb{R}^d \rightarrow \mathbb{R}^d$ is expressed as $\mathbf{z}_j(t) - \mathbf{z}_i(t)$. However, when nodes are embedded in Riemannian manifolds, the gradient of a node is no longer the difference between itself and neighboring points. Instead, we take vectors in the tangent space at \mathbf{z}_i that are obtained by taking the derivative of \mathbf{z} in all possible directions, *i.e.* $(\nabla\mathbf{z}(t))_{i,j} : \mathbb{D}_{\kappa}^d \rightarrow T\mathbb{D}_{\kappa}^d$ that can be formulated as $\log^{\kappa_t}_{\mathbf{z}_i(t)}(\mathbf{z}_j(t))$. One recovers the discrete Euclidean gradient as the curvature $\kappa \rightarrow 0$.

Diffusivity. The diffusivity scales the gradient, with either isotropic or anisotropic behavior. For graph diffusion, the **isotropic** formula is presented

by the normalized adjacency matrix [23], where $a_{i,j} = \frac{1}{\sqrt{d_i d_j}}$ iff. $(i, j) \in \mathcal{E}$ and d is the degree.

Alternatively, the **anisotropic** approach incorporates the attention mechanism [33] to account for the asymmetric relationship between pairs of nodes. This paper considers *local*, *global* and *local-global* schemes based on structure information. Define the schemes

$$\begin{cases} a_{i,j}^{\text{ldiff}} = \text{normalize}_{j \in \mathcal{N}(i)} (f_\theta(\mathbf{z}_i(t), \mathbf{z}_j(t))) & \text{(local scheme)} \\ a_{i,j}^{\text{gdiff}} = \beta \text{normalize}_{j \in \mathcal{V}} (g_\phi(\mathbf{z}_i(t), \mathbf{z}_j(t))) + \frac{1-\beta}{\sqrt{d_i d_j}} & \text{(global scheme)} \\ a_{i,j}^{\text{lgdiff}} = \beta \text{normalize}_{j \in \mathcal{V}} (g_\phi(\mathbf{z}_i(t), \mathbf{z}_j(t))) + (1-\beta) a_{i,j}^{\text{ldiff}} & \text{(local-global scheme)} \end{cases}$$

where f_θ/g_ϕ are learnable functions that compute the diffusivity weight between node pair $(i, j) \in \mathcal{E}$. β can be constant or trainable parameters that adjust the emphasis on homophilic (local attention) and heterophilic (high-order, global attention) relations. In comparison, the local attention scheme implicitly incorporates the graph information since only neighboring elements are considered based on $\mathcal{N}(i)$. Whereas for global attention, it neglects the graph topology and hence requires manual incorporation.

Low-Order Local Diffusivity. A straightforward approach is to leverage the formula of graph attention [34], which is extended to the hyperbolic space by [6], where the weights are calculated tacitly in the tangent space. An alternative method to consider is the Oliver-Ricci Curvature (ORC) [27] attention, introduced in [38, 40] to drive message propagation. This approach is not limited by the non-Euclidean property of node feature, as it computes attention weight via the ORC value derived from the graph topology, thus allowing adoption without leveraging tangent space.

High-Order Global Diffusivity. Propagation of high-order node pairs results in exponentially increasing complexity compared to f_θ . [36, 42] introduced a series of scalable and efficient node-level transformers. With a similar notion in the hyperbolic space, we first project the embeddings onto the tangent space of the origin. Subsequently, the weights can be obtained using existing graph transformer architectures. We adopt energy-constrained transformers [36] with a sigmoid kernel, which performs well in most scenarios.

* Figure 2(c) presents the high-level schematic of *diffuse*. The implementation and algorithmic details are delegated to Appendix C.

Divergence. For simplicity, we assume any $\mathbf{x}_i \in \mathbb{R}^d$ to be scalar-valued. The divergence at a point \mathbf{z}_i is a measure of how much the vector field $\mathcal{X} = \{\nabla \mathbf{z}(t)\}_{i,j} \in \mathcal{C}(i)$ is expanding or contracting at \mathbf{z}_i . In a Euclidean space, the divergence would indeed be the sum of the components of the gradient, *i.e.*, $\text{div}_i = \sum_j (\nabla \mathbf{z}(t))_{i,j}$, producing a scalar (with dimensionality $\mathcal{T}_{\mathbf{z}_i} \mathbb{D}^d \cong \mathbb{R}^d$). In our context, we are interested in how \mathbf{z}_i is varied in the manifold rather than in the tangent space; thus an exponential map is applied to the sum of gradients on $\mathcal{T}_{\mathbf{z}_i} \mathbb{D}^d$, giving $\text{div}_i = \exp_{\mathbf{z}_i(t)}(\sum_j a_{i,j} (\nabla \mathbf{z}(t))_{i,j})$. This also satisfies the form of $\mathcal{F}_\theta^\kappa$ in Definition 2, and thus can be numerically integrated through HPDESolve.

Continuous Curvature Diffusion. Equation (16) implicitly guides the manifold towards its optimal geometry for embedding $\mathbf{z}(t)$ as the manifold parameter κ_t also accumulates and is updated during backpropagation. Similar to the attention parameters θ , we let κ be time independent based on the assumption that $\lim_{\tau \rightarrow 0} \frac{\kappa_{t+\tau} - \kappa_t}{\tau} = 0$.

4.2 Convergence of Dirichlet Energy

Definition 3 Given the node embedding $\{\mathbf{z}_i \in \mathbb{D}_\kappa^d\}_{i=1}^{|\mathcal{V}|}$, the hyperbolic Dirichlet energy is defined as

$$f_{\text{DE}}^\kappa(\mathbf{z}) = \frac{1}{2} \sum_{(i,j) \in \mathcal{E}} d_{\mathbb{D}}^\kappa \left(\exp_{\circ}^\kappa \left(\frac{\log_{\circ}^\kappa(\mathbf{z}_i)}{\sqrt{1+d_i}} \right), \exp_{\circ}^\kappa \left(\frac{\log_{\circ}^\kappa(\mathbf{z}_j)}{\sqrt{1+d_j}} \right) \right)^2, \quad (17)$$

where $d_{i/j}$ denotes the node degree of node i/j . The distance $d_{\mathbb{D}}^\kappa(\mathbf{x}, \mathbf{y})$ between two points $\mathbf{x}, \mathbf{y} \in \mathbb{D}$ is the geodesic length; we detail the closed form expression in Appendix B.

Definition 3 introduces a node-similarity measure to quantify over-smoothness in hyperbolic space. f_{DE}^κ of node representation can be viewed as the weighted sum of distance between normalized node pairs. [25, Prop. 4] proved that hyperbolic energy f_{DE}^κ diminishes after message passing, and multiple aggregations result in converging towards zero energy, indicating reduced embedding expressiveness that could potentially cause over-smoothing. Also as proved in [43, Prop. 2] that over-smoothing is an intrinsic property of first-order continuous GNN. In a continuous diffusion process, where each iteration can be viewed as a layer in HGNNs, as supported by Fig. 3, we also observe a convergence of hyperbolic Dirichlet energy of $\mathbf{z}(t)$ w.r.t. time t .

Residual-Empowered Flow. Empirically, studies in multi-layer GNNs [15, 24] demonstrated the efficacy of adding residual connections to the initial layer. It is also claimed in [45] that using residual connections for both initial and previous layers can prevent the Dirichlet energy from reaching a lower energy limit, thus avoiding over-smoothing. Building upon these studies, we define the hyperbolic residual empowered vector flow

$$\mathcal{F}_\theta^\kappa(\mathbf{z}(t), t) = \mu_{\mathbb{D}}^\kappa(\{\dot{\mathbf{z}}(t), \mathbf{z}(t), \mathbf{z}(0)\}; \{\eta\}_{j=1}^J), \quad (18)$$

where $\dot{\mathbf{z}}(t) = \exp_{\mathbf{z}(t)}^{\kappa_t}(\sigma[\mathbf{S}(\mathbf{z}(t))\nabla\mathbf{z}(t)])$ is the manifold dynamic as in Eq. (16). $\{\eta\}_{j=1}^J$ are the weight coefficients. $\mu_{\mathbb{D}}^\kappa$ is the node-wise hyperbolic averaging. We instantiate it via *Möbius Gyromidpoint* [32] for its trade-off between computational cost and precision. Define

$$\mu_{\mathbb{D}}^\kappa(\{\mathbf{z}\}_{j=1}^J; \{\eta\}_{j=1}^J) = \left(\frac{1}{2} \otimes_{\kappa} \left(\frac{\sum_j \eta_j \lambda_{\mathbf{z}_i}^{\kappa(j)} \mathbf{z}_i^{(j)}}{\sum_j |\eta_j| (\lambda_{\mathbf{z}_i}^{\kappa(j)} - 1)} \right) \right)_{i=1}^{|\mathcal{V}|}. \quad (19)$$

This operation ensures the point set constraint of \mathbb{D} for the residual flow. We recover the arithmetic mean as $\kappa \rightarrow 0$. During diffusion, Eq. (18) retains at least a portion of the initial and prior embeddings. Since the initial embedding possesses high energy, the residual connection mitigates energy degradation and retains the energy of the final iteration at the same level as the preceding iterations.

5 Empirical Results

5.1 Experiment Setup

Datasets. Under homophilic setting, we consider 5 datasets for node classification and link prediction: DISEASE, AIRPORT (transductive datasets, provided in [6] to investigate the tree-likeness modeling), PUBMED, CITESEER and CORA ([39] widely used citation networks), which are summarized in the table in Appendix A. Additionally, we report the Gromov’s hyperbolicity δ given by [16] for each dataset. A graph is more hyperbolic as $\delta \rightarrow 0$ and is a tree when $\delta = 0$.

For heterophilic datasets, we evaluate node classification on three heterophilic graphs, respectively, CORNELL, TEXAS and WISCONSIN [29] from the WebKB dataset (webpage networks). Detailed statistics are summarized in Appendix A. We use the original fixed 10 split datasets. In addition, we report the homophily level \mathcal{H} of each dataset, a sufficiently low $\mathcal{H} \leq 0.3$ means that the dataset is more heterophilic when most of neighbours are not in the same class.

Baselines. We compare our models to (1) *Euclidean-hyperbolic* baselines, (2) *discrete-continuous depth* baselines and (3) *heterophilic relationship* baselines. For (1), we compare against feature-based models, Euclidean, and hyperbolic graph-based models. Feature-based models: without using graph structure, we feed node feature directly to MLP and HNN [14]; Euclidean graph-based models: GCN [23], GAT [34], GraphSAGE [19], and SGC [35]; Hyperbolic graph-based models: HGCN [6], κ GCN [1], LGCN [44] and HyboNet [10]. For (2), we compare our models on citation networks with the discrete-continuous depth models. Discrete depth: GCNII [7], C-DropEdge [20]; Discrete-decouple: HyLa-SGC [41]; Continuous depth: GDE [30], GRAND and BLEND [4, 5]. For (3), we compare to the prevalent GNNs: GCN, GAT, HGCN, HyboNet, and those optimized for heterophilic relationships: H2GCN [46], GCNII, GraphSAGE and GraphCON [31]. The test results are partially derived from the above works. For fairness, we compare to models with no more than 16 layers/iterations. Please refer to Appendix A for more details regarding the compared baselines. We detail the parameter settings for model and evaluation metric in Appendix C.

5.2 Experiment Results

Euclidean-Hyperbolic Baselines. We investigate our methods with different solvers with $\tau = 1$, *i.e.* HGDE-E (multi-step explicit integrator, HRK4) and HGDE-I (multi-step implicit integrator, HAM). The experimental results are summarized in Tables 1 and 2. (1) Our proposed models outperform previous

Table 1. Test accuracy (%) for node classification task.

Dataset	DISEASE	AIRPORT	PUBMED	CITeseer	CORA
δ	0	1	3.5	5	11
MLP	32.5 \pm 1.1	60.9 \pm 3.4	72.4 \pm 0.2	59.5 \pm 0.9	51.6 \pm 1.3
HNN	45.5 \pm 3.3	80.6 \pm 0.5	69.9 \pm 0.4	59.5 \pm 1.2	54.7 \pm 0.6
GCN	69.7 \pm 0.4	81.6 \pm 0.6	78.1 \pm 0.4	70.3 \pm 0.4	81.5 \pm 0.5
GAT	70.4 \pm 0.4	82.7 \pm 0.4	78.2 \pm 0.4	71.6\pm0.8	83.0 \pm 0.5
SAGE	69.1 \pm 0.6	82.2 \pm 0.5	77.5 \pm 2.4	67.5 \pm 0.7	79.9 \pm 2.5
SGC	69.5 \pm 0.2	80.6 \pm 0.2	78.8 \pm 0.2	71.4 \pm 0.8	81.3 \pm 0.5
HGCN	82.8 \pm 0.8	89.2 \pm 1.3	80.3\pm0.3	68.0 \pm 0.6	79.9 \pm 0.2
κ GCN	82.1 \pm 1.1	84.4 \pm 0.4	78.3 \pm 0.6	71.1 \pm 0.6	80.8 \pm 0.6
LGCN	84.4 \pm 0.8	90.9\pm1.0	78.8 \pm 0.5	71.1 \pm 0.3	83.3\pm0.5
HyboNet	96.0\pm1.0	90.9 \pm 1.4	78.0 \pm 1.0	69.8 \pm 0.6	80.2 \pm 1.3
HGDE-E	92.1\pm1.6	95.1\pm0.4	81.2\pm0.5	74.1\pm0.5	84.4\pm0.7
HGDE-I	90.9\pm2.5	93.9\pm0.8	81.0\pm0.3	73.5\pm0.7	84.0\pm0.4

Table 2. Test ROC AUC (%) results for link prediction task.

Dataset	DISEASE	AIRPORT	PUBMED	CITeseer	CORA
δ	0	1	3.5	5	11
MLP	69.9 \pm 3.4	68.9 \pm 0.5	83.3 \pm 0.6	93.7 \pm 0.6	83.3 \pm 0.6
HNN	70.2 \pm 0.1	80.6 \pm 0.5	94.7 \pm 0.1	93.3 \pm 0.5	90.9 \pm 0.4
GCN	64.7 \pm 0.5	89.3 \pm 0.4	89.6 \pm 3.7	82.6 \pm 1.9	90.5 \pm 0.2
GAT	69.8 \pm 0.3	90.9 \pm 0.2	91.5 \pm 1.8	86.5 \pm 1.5	93.2 \pm 0.2
SAGE	65.9 \pm 0.3	90.4 \pm 0.5	86.2 \pm 0.8	92.1 \pm 0.4	85.5 \pm 0.5
SGC	65.1 \pm 0.2	89.8 \pm 0.3	94.1 \pm 0.1	91.4 \pm 1.7	91.5 \pm 0.2
HGCN	91.2 \pm 0.6	96.4 \pm 0.1	95.1 \pm 0.1	96.6\pm0.1	93.8\pm0.1
κ GCN	92.0 \pm 0.5	92.5 \pm 0.5	94.9 \pm 0.3	95.1 \pm 0.6	92.6 \pm 0.4
LGCN	96.6\pm0.6	96.0 \pm 0.6	96.6\pm0.1	95.8 \pm 0.4	93.6 \pm 0.4
HyboNet	96.8\pm0.4	97.3\pm0.3	95.8 \pm 0.2	96.7\pm0.8	93.6 \pm 0.3
HGDE-E	96.2\pm0.5	98.2\pm0.2	96.6\pm0.2	96.7\pm0.7	94.1\pm0.4
HGDE-I	95.6 \pm 0.5	97.6\pm0.5	96.2\pm0.7	96.4 \pm 0.7	94.5\pm0.8

Euclidean and hyperbolic models in four out of five datasets, suggesting that graph learning in hyperbolic space through topological diffusion is beneficial. (2) Hyperbolic models typically exhibit poor performance on datasets that are less hyperbolic (*e.g.*, CORA), while our method surprisingly exceeds Euclidean

Table 3. Discrete-continuous depth GNN comparison.

Type	Model	CORA	CITESEER	PUBMED
Discrete	GCNII	84.6\pm0.8	72.9 \pm 0.5	80.2 \pm 0.4
	C-DropEdge	82.6 \pm 0.9	71.0 \pm 1.0	77.8 \pm 1.0
Decouple (Hyp PosEnc)	HyLa-SGC	82.5 \pm 0.5	72.6 \pm 1.0	80.3 \pm 0.9
Continuous	GDE	83.8 \pm 0.5	72.5 \pm 0.5	79.9 \pm 0.3
	GRAND	82.9 \pm 0.7	73.6 \pm 0.3	81.0\pm0.4
Continuous (Hyp PosEnc)	BLEND	84.2\pm0.6	74.4\pm0.7	80.7 \pm 0.7
Continuous (Hyp Embed)	HGDE(4)	83.4 \pm 0.5	73.0 \pm 0.3	80.2 \pm 0.6
	HGDE(8)	83.7 \pm 0.6	73.5 \pm 0.7	80.8 \pm 0.4
	HGDE(12)	84.2\pm0.6	74.1\pm0.5	81.2\pm0.5
	HGDE(16)	84.4\pm0.7	73.8\pm0.7	80.9\pm0.3

Table 4. Heterophilic relationship GNN comparison.

Type	\mathcal{H}	TEXAS 0.11	WISCONSIN 0.21	CORNELL 0.30
Euclidean	GCN	55.1 \pm 5.2	51.8 \pm 3.1	60.5 \pm 5.3
	GAT	52.2 \pm 6.6	49.4 \pm 4.1	61.9 \pm 5.1
Hyperbolic	HGCN	55.7 \pm 6.3	48.1 \pm 6.1	62.1 \pm 3.7
	HyboNet	60.0 \pm 4.1	51.2 \pm 3.3	62.3 \pm 3.5
High-Order GNNs	H2GCN	84.9\pm7.2	87.7\pm5.0	82.7\pm5.3
	GCNII	77.6 \pm 3.8	80.4 \pm 3.4	77.9 \pm 3.8
	SAGE	82.4 \pm 6.1	81.2 \pm 5.6	76.0 \pm 5.0
	GraphCON	85.4\pm4.2	87.8\pm3.3	84.3\pm4.8
Ours	HGDE	85.9\pm2.8	86.2\pm2.4	85.0\pm5.3

GAT on datasets with lower δ , indicating the necessity of curvature diffusion in adapting to datasets with scarce hierarchical structures and modeling long-term dependency via the local-global diffusivity function. (3) HGDE and other hyperbolic models achieve superior performance compared to Euclidean counterparts in link prediction due to the larger embedding space in hyperbolic geometry, which better preserves structural dependencies and allows for improved node arrangement. (4) HGDE-E generally outperforms HGDE-I with lower memory

Table 5. Evaluation on image (CIFAR/STL) and text (20News) classification (Left) and Memory & Runtime comparison (Right). \star indicate OOM.

Dataset		MLP	LabelProp	ManiReg	GCN-kNN	GAT-kNN	DenseGAT	GLCN	HGDE
CIFAR	100 labels	65.9 \pm 1.3	66.2	67.0 \pm 1.9	66.7 \pm 1.5	66.0 \pm 2.1	\star	66.6 \pm 1.4	68.9\pm2.1
	500 labels	73.2 \pm 0.4	70.6	72.6 \pm 1.2	72.9 \pm 0.4	72.4 \pm 0.5	\star	72.7 \pm 0.5	74.0\pm1.8
	1000 labels	75.4 \pm 0.6	71.9	74.3 \pm 0.4	74.7 \pm 0.5	74.1 \pm 0.5	\star	74.7 \pm 0.3	76.3\pm0.9
STL	100 labels	66.2 \pm 1.4	65.2	66.5 \pm 1.9	66.9 \pm 0.5	66.5 \pm 0.8	\star	66.4 \pm 0.8	66.9\pm1.3
	500 labels	73.0\pm0.8	71.8	72.5 \pm 0.5	72.1 \pm 0.8	72.0 \pm 0.8	\star	72.4 \pm 1.3	72.5 \pm 0.2
	1000 labels	75.0 \pm 0.8	72.7	74.2 \pm 0.5	73.7 \pm 0.4	73.9 \pm 0.6	\star	74.3 \pm 0.7	75.1\pm0.6
20News	1000 labels	54.1 \pm 0.9	55.9	56.3 \pm 1.2	56.1 \pm 0.6	55.2 \pm 0.8	54.6 \pm 0.2	56.2 \pm 0.8	56.3\pm0.9
	2000 labels	57.8 \pm 0.9	57.6	60.0 \pm 0.8	60.6 \pm 1.3	59.1 \pm 2.2	59.3 \pm 1.4	60.2 \pm 0.7	61.0\pm1.0
	4000 labels	62.4 \pm 0.6	59.5	63.6 \pm 0.7	64.3 \pm 1.0	62.9 \pm 0.7	62.4 \pm 1.0	64.1 \pm 0.8	64.1\pm0.8
T/τ	Model (with Att)		Memory ($\times 10^6$)		Runtime (ms)				
2	HGDCN		4045		9.25				
	HGDCN (LocalAtt)		4246		1310.31				
	LGCN		4630		16.64				
	HyboNet		4368		14.90				
	HGDE		62		13.66				
4	HGDCN		10255		29.77				
	HGDCN (LocalAtt)		10578		4086.08				
	LGCN		12675		40.88				
	HyboNet		11931		35.23				
	HGDE		73		20.28				
8	HGDCN		22674		67.24				
	HGDCN (LocalAtt)		OOM		\star				
	LGCN		23712		160.75				
	HyboNet		23403		122.55				
	HGDE		112		34.11				
16	All Baselines		OOM		\star				
	HGDE		188		61.95				

consumption and better precision, indicating that a larger τ may be necessary for implicit solvers. To align with multi-layer GNN schema (step-size is analogous to depth), we employ HGDE with HRK4 ($\tau = 1$) for further evaluation.

Discrete-Continuous Depth Baselines. In Table 3, we compare our models with discrete and continuous-depth baselines. We observe that our method with $T \in [12, 16]$ achieves competitive results with the state-of-the-art models. Notably, HGDE models consistently outperform discrete models and continuous models with Euclidean embeddings, highlighting the benefits of utilizing hyperbolic embeddings in a continuous-depth framework. Compared to position encoding approaches (*e.g.*, HyLa, BLEND), HGDE exhibits superior performance, indicating the feasibility of using hyperbolic space embeddings directly over initial position encoding. Interestingly, we find HGDE models performs better when increasing T up to 12, but slightly worse at $T = 16$. This may due to the capacity of Poincaré ball or potential over-smoothing. Overall, the results underscore the effectiveness of the proposed HGDE models in harnessing the power of hyperbolic space for graph data modeling.

Heterophilic Relationship Baselines. We show that HGDE is also capable in managing heterophilic relationship. In Table 4, HGDE achieves the highest scores on the TEXAS and CORNELL and a competitive score on WISCONSIN. This shows that hyperbolic space is beneficial in learning hierarchical heterophilic relationships. It also reflects the flexibility of HGDE as a hyperbolic vector flow for embedding high-order structures, with our model, powered by HPDE, outperforming other baselines on average.

Image and Text Classification. We follow the experiment setup in [36] and conduct additional experiments on the CIFAR, STL, and 20NEWS datasets to evaluate HGDE in multiple scenarios with limited label rates. We employ the SimCLR [9] extracted embedding as provided in [36] for image classification. For the pre-processed 20NEWS [28] for text classification, we take 10 topics and regard words with TF-IDF > 5 as features. For graph-based models, we use k NN to construct a graph over input features. For HGDE (hyperbolic), we map the initial feature to \mathbb{D}_κ via $\exp_\circ^\kappa(\cdot)$ before the embedding process. As depicted in Table 5(Left), HGDE consistently surpasses its opponents, including MLP, LabelProp [47], ManiReg [2], GCN- k NN, GAT- k NN, DenseGAT, and GLCN [21]. Across all datasets, HGDE outperforms the Euclidean models, underscoring its proficiency in understanding the potential hierarchical structure of image embeddings [22] and text embeddings. Furthermore, HGDE exhibits good performance compared to static graph-based baselines *e.g.*, GAT- k NN and GLCN, which underlines the distinct advantage of the evolving diffusivity mechanism in understanding the potential hierarchical structure of image/text embeddings.

5.3 Ablation Study

Efficacy of Hyperbolic Residual. Figure 3 visualizes the convergence of hyperbolic energy through iterations. We observe that, without residuals, the averaged energy rapidly decreases to near-zero values, supporting the hypothesis that, without residual connections, the embedding can evolve to an overly smoothed state that is potentially low in expressiveness. However, with hyperbolic residuals, for all three integrators, the average energy decreases over the first few iterations and then appears to stabilize around a certain value above zero. This behavior is consistent across both datasets, suggesting that the system is able to converge to a stable state with non-zero energy (Fig. 4).

Efficacy of Diffusivity Function. Figure 5 visualizes sampled node embeddings and their edge diffusivity on CORA. The blue edges are inherently determined by the graph structure. Red ones are determined by global attention, showing that a^{lgdiff} accounts for high-order relations. The bar graph shows the average accuracy on various datasets produced by HGDE with different diffusivity functions. We find that anisotropic approaches generally outperform the isotropic approach, suggesting the necessity of directional information in the diffusion process. Although the performance degrades on CITESEER when using a^{lgdiff} , there are significant improvements on other graphs, certifying the benefit of higher-order proximity induced by local-global diffusivity.

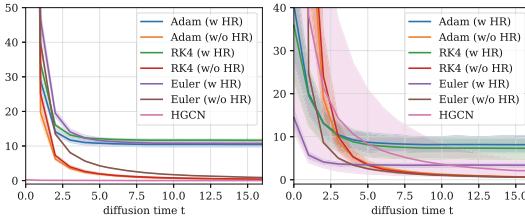


Fig. 3. Hyperbolic Dirichlet energy $f_{DE}^K(\cdot)$ variation through t on CORA (left) and CITESEER (right). Models are compared with different integrators w or w/o hyperbolic residual.

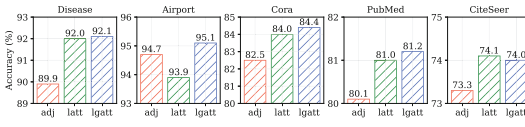


Fig. 4. Averaged node classification performance comparison of models with different diffusivity functions on various datasets.

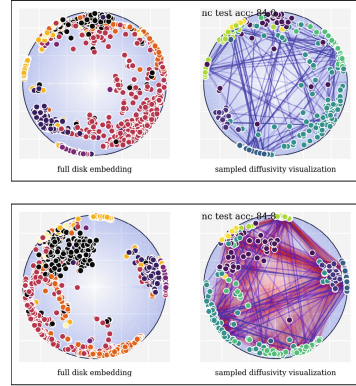


Fig. 5. CORA diffusivity (400 node sampled from \mathbb{D}_K^2 embeddings) produced by a^{ldiff} (left) and a^{lgdiff} (right), blue and red lines denote local and global attention; bolder lines indicate more attentiveness. (Color figure online)

Parameter Efficiency. In Table 5 (Right), we provide an additional comparison of peak GPU memory usage and per-epoch running time on the CORA. We tested HGDE-E where all models have a 16 hidden dim. Our model significantly outperforms the other baselines in both training time (for ≥ 4 layers) and memory consumption. The memory reduction is primarily due to the utilization of sparse attention, and the advantages of a weight-tied network (requiring only single-layer parameters) as a nature of HPDE. The training time efficiency is achieved by eliminating layer-wise feature transformation, implementing weight-tying, and applying scattered-agg for hyperbolic representation.

6 Conclusion

We developed multiple numerical integrators for HPDE, and proposed the first hyperbolic continuous-time embedding diffusion framework – HGDE. Being capable of capturing both low and high order proximity, HGDE outperforms both Euclidean and hyperbolic baselines on various datasets. The effectiveness of HGDE was further validated by the ablation studies on hyperbolic energy and diffusivity functions. The superiority of HGDE underscores the potential of developing PDE-based non-Euclidean models.

Limitation. While HGDE presents strong performance in modeling graph data, hyperbolic spaces may not always be optimal, particularly for data without clear

hierarchical structures. For instance, HGDE is difficult to beat natural Euclidean deep models (*e.g.* GCNII) on the non-hierarchical CORA. Moreover, a higher memory complexity and lower training time only tells the efficiency rather than scalability of HGDE, since our models are evaluated with fixed number of parameters (which is natural for ODE-based models), increasing T is not necessarily *scaling up*. Future work include addressing these limitations and exploring the scalability and generalizability of HGDE in diverse real-world settings.

Acknowledgments. XH is financially supported by the U.K. EPSRC through End-to-End Conceptual Guarding of Neural Architectures [EP/T026995/1]. JL is supported by Liverpool-CSC scholarship [202208890034].

References

1. Bachmann, G., Bécigneul, G., Ganea, O.: Constant curvature graph convolutional networks. In: International Conference on Machine Learning, pp. 486–496. PMLR (2020)
2. Belkin, M., Niyogi, P., Sindhvani, V.: Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. *J. Mach. Learn. Res.* **7**(11) (2006)
3. Beltrami, E.: Teoria fondamentale degli spazii di curvatura costante: memoria. F. Zanetti (1868)
4. Chamberlain, B., Rowbottom, J., Gorinova, M.I., Bronstein, M., Webb, S., Rossi, E.: Grand: graph neural diffusion. In: International Conference on Machine Learning, pp. 1407–1418. PMLR (2021)
5. Chamberlain, B., Rowbottom, J., Eynard, D., Di Giovanni, F., Dong, X., Bronstein, M.: Beltrami flow and neural diffusion on graphs. In: Advances in Neural Information Processing Systems, vol. 34, pp. 1594–1609 (2021)
6. Chami, I., Ying, Z., Ré, C., Leskovec, J.: Hyperbolic graph convolutional neural networks. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
7. Chen, M., Wei, Z., Huang, Z., Ding, B., Li, Y.: Simple and deep graph convolutional networks. In: International Conference on Machine Learning, pp. 1725–1735. PMLR (2020)
8. Chen, R.T., Rubanova, Y., Bettencourt, J., Duvenaud, D.K.: Neural ordinary differential equations. In: Advances in Neural Information Processing Systems, vol. 31 (2018)
9. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International Conference on Machine Learning, pp. 1597–1607. PMLR (2020)
10. Chen, W., et al.: Fully hyperbolic neural networks. arXiv preprint [arXiv:2105.14686](https://arxiv.org/abs/2105.14686) (2021)
11. Choudhary, N., Reddy, C.K.: Towards scalable hyperbolic neural networks using Taylor series approximations. arXiv preprint [arXiv:2206.03610](https://arxiv.org/abs/2206.03610) (2022)
12. Clauset, A., Moore, C., Newman, M.E.: Hierarchical structure and the prediction of missing links in networks. *Nature* **453**(7191), 98–101 (2008)
13. Ganea, O., Bécigneul, G., Hofmann, T.: Hyperbolic entailment cones for learning hierarchical embeddings. In: International Conference on Machine Learning, pp. 1646–1655. PMLR (2018)

14. Ganea, O., Bécigneul, G., Hofmann, T.: Hyperbolic neural networks. In: *Advances in Neural Information Processing Systems*, vol. 31 (2018)
15. Gasteiger, J., Bojchevski, A., Günnemann, S.: Predict then propagate: graph neural networks meet personalized pagerank. arXiv preprint [arXiv:1810.05997](https://arxiv.org/abs/1810.05997) (2018)
16. Gromov, M.: Hyperbolic groups. In: Gersten, S.M. (ed.) *Essays in Group Theory*. Mathematical Sciences Research Institute Publications, vol. 8, pp. 75–263. Springer, New York (1987). https://doi.org/10.1007/978-1-4613-9586-7_3
17. Hairer, E.: Solving differential equations on manifolds, **8** (2011). <https://www.unige.ch/~hairer/poly-sde-mani.pdf>
18. Hamann, M.: On the tree-likeness of hyperbolic spaces. In: *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 164, pp. 345–361. Cambridge University Press (2018)
19. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: *Advances in Neural Information Processing Systems*, vol. 30 (2017)
20. Huang, W., et al.: Towards deepening graph neural networks: a GNTK-based optimization perspective. arXiv preprint [arXiv:2103.03113](https://arxiv.org/abs/2103.03113) (2021)
21. Jiang, B., Zhang, Z., Lin, D., Tang, J., Luo, B.: Semi-supervised learning with graph learning-convolutional networks. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11313–11320 (2019)
22. Khrulkov, V., Mirvakhabova, L., Ustinova, E., Oseledets, I., Lempitsky, V.: Hyperbolic image embeddings. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6418–6428 (2020)
23. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint [arXiv:1609.02907](https://arxiv.org/abs/1609.02907) (2016)
24. Li, G., Muller, M., Thabet, A., Ghanem, B.: DeepGCNs: can GCNs go as deep as CNNs? In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9267–9276 (2019)
25. Liu, J., Yi, X., Huang, X.: DeephGCN: recipe for efficient and scalable deep hyperbolic graph convolutional networks (2024)
26. Nickel, M., Kiela, D.: Poincaré embeddings for learning hierarchical representations. In: *Advances in Neural Information Processing Systems*, vol. 30 (2017)
27. Ollivier, Y.: Ricci curvature of Markov chains on metric spaces. *J. Funct. Anal.* **256**(3), 810–864 (2009)
28. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011)
29. Pei, H., Wei, B., Chang, K.C.C., Lei, Y., Yang, B.: Geom-GCN: geometric graph convolutional networks. arXiv preprint [arXiv:2002.05287](https://arxiv.org/abs/2002.05287) (2020)
30. Poli, M., Massaroli, S., Park, J., Yamashita, A., Asama, H., Park, J.: Graph neural ordinary differential equations. arXiv preprint [arXiv:1911.07532](https://arxiv.org/abs/1911.07532) (2019)
31. Rusch, T.K., Chamberlain, B., Rowbottom, J., Mishra, S., Bronstein, M.: Graph-coupled oscillator networks. In: *International Conference on Machine Learning*, pp. 18888–18909. PMLR (2022)
32. Ungar, A.A.: A gyrovector space approach to hyperbolic geometry. *Synth. Lect. Math. Stat.* **1**(1), 1–194 (2008)
33. Vaswani, A., et al.: Attention is all you need. In: *Advances in Neural Information Processing Systems*, vol. 30 (2017)
34. 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)
35. Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., Weinberger, K.: Simplifying graph convolutional networks. In: *International Conference on Machine Learning*, pp. 6861–6871. PMLR (2019)

36. Wu, Q., Yang, C., Zhao, W., He, Y., Wipf, D., Yan, J.: Difformer: scalable (graph) transformers induced by energy constrained diffusion. arXiv preprint [arXiv:2301.09474](https://arxiv.org/abs/2301.09474) (2023)
37. Yan, S., Xiong, Y., Lin, D.: Spatial temporal graph convolutional networks for skeleton-based action recognition. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
38. Yang, M., Zhou, M., Pan, L., King, I.: κ HGCN: tree-likeness modeling via continuous and discrete curvature learning (2023)
39. Yang, Z., Cohen, W., Salakhudinov, R.: Revisiting semi-supervised learning with graph embeddings. In: International Conference on Machine Learning, pp. 40–48. PMLR (2016)
40. Ye, Z., Liu, K.S., Ma, T., Gao, J., Chen, C.: Curvature graph network. In: International Conference on Learning Representations (2019)
41. Yu, T., De Sa, C.: Random Laplacian features for learning with hyperbolic space. arXiv preprint [arXiv:2202.06854](https://arxiv.org/abs/2202.06854) (2022)
42. Yun, S., Jeong, M., Kim, R., Kang, J., Kim, H.J.: Graph transformer networks. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
43. Zhang, Y., Gao, S., Pei, J., Huang, H.: Improving social network embedding via new second-order continuous graph neural networks. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 2515–2523 (2022)
44. Zhang, Y., Wang, X., Shi, C., Liu, N., Song, G.: Lorentzian graph convolutional networks. In: Proceedings of the Web Conference 2021, pp. 1249–1261 (2021)
45. Zhou, K., et al.: Dirichlet energy constrained learning for deep graph neural networks. In: Advances in Neural Information Processing Systems, vol. 34, pp. 21834–21846 (2021)
46. Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L., Koutra, D.: Beyond homophily in graph neural networks: current limitations and effective designs. In: Advances in Neural Information Processing Systems, vol. 33, pp. 7793–7804 (2020)
47. Zhu, X., Ghahramani, Z., Lafferty, J.D.: Semi-supervised learning using Gaussian fields and harmonic functions. In: Proceedings of the 20th International Conference on Machine Learning (ICML-03), pp. 912–919 (2003)
48. Zitnik, M., Sosič, R., Feldman, M.W., Leskovec, J.: Evolution of resilience in protein interactomes across the tree of life. *Proc. Natl. Acad. Sci.* **116**(10), 4426–4433 (2019)



SpanGNN: Towards Memory-Efficient Graph Neural Networks via Spanning Subgraph Training

Xizhi Gu¹, Hongzheng Li¹, Shihong Gao², Xinyan Zhang¹, Lei Chen²,
and Yingxia Shao¹(✉)

¹ Beijing University of Posts and Telecommunications, Beijing, China
{guxizhi, Ethan_Lee, xianxx492, shaoyx}@bupt.edu.cn

² Hong Kong University of Science and Technology, Guangzhou, China
sgaoar@connect.ust.hk, leichen@cse.ust.hk

Abstract. Graph Neural Networks (GNNs) have superior capability in learning graph data. Full-graph GNN training generally has high accuracy, however, it suffers from large peak memory usage and encounters the Out-of-Memory problem when handling large graphs. To address this memory problem, a popular solution is mini-batch GNN training. However, mini-batch GNN training increases the training variance and sacrifices the model accuracy. In this paper, we propose a new memory-efficient GNN training method using spanning subgraph, called SpanGNN. SpanGNN trains GNN models over a sequence of spanning subgraphs, which are constructed from empty structure. To overcome the excessive peak memory consumption problem, SpanGNN selects a set of edges from the original graph to incrementally update the spanning subgraph between every epoch. To ensure the model accuracy, we introduce two types of edge sampling strategies (i.e., variance-reduced and noise-reduced), and help SpanGNN select high-quality edges for the GNN learning. We conduct experiments with SpanGNN on widely used datasets, demonstrating SpanGNN's advantages in the model performance and low peak memory usage.

1 Introduction

Graph Neural Networks (GNNs) achieve the state-of-the-art performance on graph learning tasks, such as node classification [16, 19, 27], link prediction [23, 39] and graph classification [9, 37]. They have been widely used in various domains, like social network analysis [10, 26], recommendation [17, 34, 35], healthcare [1, 7], short-term load forecasting [25] and bio-informatics [11, 21]. Most of GNNs [16, 19, 27, 36] follow the message passing paradigm [13], which exploits graph topology and node/edge features simultaneously. In this paradigm, the edge related memory consumption predominantly influences the amount of peak GPU memory [30]. The edge calculation of GNNs involves four operations, which are collection, messaging, aggregation, and feature updating. The four operations require

the storage of intermediate results (e.g., the updated embedding of edge feature and the aggregation of feature embedding), which are used for the gradient calculation in the subsequent backward propagation process. According to the existing empirical studies [30], the peak memory consumption can be up to 100 times of the size of dataset itself. As a result, the high memory usage of the edge calculation restricts the GNNs scaling to large graphs.

Since the edge calculation is the main factor of high memory usage, an intuitive idea is to reduce the number of edges for training. Sampling is a standard technique to generate graphs with few edges. It has been well studied in the mini-batch training. Many works [6, 8, 16, 37, 38] use various sampling techniques to create mini-batches, which are subgraphs rooted from a limited number of target nodes. Although mini-batch training is scalable and memory-efficient, it brings in non-negligible training variance and heavily compromises model accuracy. Full-graph GNN training is more accurate than mini-batch training [18]. However, the existing complex sampling methods cannot be efficiently adopted to the full-graph GNN training. The sampling step is time-consuming and becomes the efficiency bottleneck for GNN training on large graphs [30]. Unlike the sampling technique, DropEdge [22] randomly drops edges of the original graph during the full-graph training. It not only reduces the size of peak memory, but also is scalable to large graphs. Nonetheless, DropEdge also suffers from a prominent model accuracy loss as the edge drop ratio increases, especially on large graphs. This limitation arises because DropEdge treats all edges equally and ignores the inherent structure of the original graph. Therefore, *how to develop a memory-efficient and accurate full-graph GNN learning method remains unsolved.*

In this paper, we propose SpanGNN to achieve memory-efficient full-graph GNN training while guaranteeing the model accuracy. First, SpanGNN trains GNN models across a sequence of spanning subgraphs, which are constructed from empty structure. Each spanning subgraph contains significantly fewer edges than those present in the original graphs, thus effectively reducing the peak memory footprint. Furthermore, in each training epoch, SpanGNN selects a set of edges from the original graph to incrementally update the spanning subgraph that used in the previous epoch. Meanwhile, the updated spanning graph always satisfies the sparsity constraint defined by the edge ratio α (See the definition in Sect. 2.2). Second, to guarantee the model accuracy and training efficiency, we propose a fast quality-aware edge selection method for SpanGNN. We analyze the training variance and gradient noise that inherent in the spanning subgraph training framework, and propose variance-reduced sampling and gradient-noise reduced sampling strategies, respectively, to help SpanGNN selects a set of high-quality edges and guarantee the model accuracy. However, it is expensive to directly apply the above two sampling strategies over large graphs, we introduce a two-step sampling method to speed up the edge selection process. Extensive experiments demonstrate that SpanGNN is capable of saving over 40% of GPU memory usage without compromising training performance.

Our main contributions are summarized as follows:

- We propose SpanGNN that supports memory-efficient and accurate full-graph GNN training on large graphs. The new method reduces the peak memory usage significantly during the training, meanwhile achieving high model accuracy.
- We introduce a fast quality-aware edge selection method to alleviate the negative impacts caused by spanning subgraph training and ensure training efficiency.
- We analyze the connection between SpanGNN and curriculum learning [3]. With the help of quality-aware edge selection, SpanGNN selects edges that are highly beneficial to the learning in priority, and then gradually uses edges with low benefits.
- Experimental results on widely used datasets demonstrate that SpanGNN reduces peak memory usage effectively while guaranteeing that the model accuracy is almost equivalent to the one of full graph training.

2 Preliminary

2.1 Graph Neural Networks

The general matrix formulation of GNN models is as follows:

$$Z^{(l)} = PH^{(l-1)}W^{(l-1)}, \quad (1)$$

$$H^{(l)} = \sigma(Z^{(l)}), \quad (2)$$

where $Z^{(l)}$, $H^{(l)}$, and $W^{(l)}$ represent the intermediate embedding matrix, feature embedding matrix and trainable weight matrix at l -th layer, respectively. σ is a non-linear activation function, like ReLU. P is the propagation matrix that is transformed from the graph adjacency matrix.

During the backward propagation, the gradient of the loss with respect to $W^{(l-1)}$ is as follows:

$$\nabla_{W^{(l-1)}}L = \frac{\partial L}{\partial W^{(l-1)}} = (H^{(l-1)})^T P^T \delta^{(l)}, \quad (3)$$

where $\delta^{(l)}$ denotes the gradient of the loss with respect to $Z^{(l)}$. Then, $W^{(l-1)}$ is updated as follows:

$$W^{(l-1)} = W^{(l-1)} - \eta \nabla_{W^{(l-1)}}L, \quad (4)$$

where η denotes the learning rate of training.

2.2 Spanning Subgraph GNN Training

Given a graph $G = (V, E)$, a spanning subgraph $G_s = (V_s, E_s)$ generated from G is a subgraph with vertex set V , i.e., $V_s = V$ and $E_s \subset E$ [32]. We define edge ratio α between G_s and G is $\frac{|E_s|}{|E|}$. α represents the degree of edge reduction of a

spanning subgraph against the corresponding original graph. The smaller α is, the more edges are deleted, and less memory is demanded for training over the spanning subgraph.

Spanning subgraph GNN training make GNNs only propagation along the subgraph G_s . Therefore, the key GNN operations (Eqs. 1-3) are rewritten as:

$$\tilde{Z}^{(l)} = \tilde{P}H^{(l-1)}W^{(l-1)}, \quad (5)$$

$$\tilde{H}^{(l)} = \sigma(\tilde{Z}^{(l)}), \quad (6)$$

$$\nabla_{W^{(l-1)}}\tilde{L} = \frac{\partial\tilde{L}}{\partial W^{(l-1)}} = (\tilde{H}^{(l-1)})^T\tilde{P}^T\tilde{\delta}^{(l)}, \quad (7)$$

where \tilde{P} is the propagation matrix that is transformed from the spanning subgraph. Spanning subgraph GNN training results in approximated node embedding matrix $\tilde{H}^{(l)}$ and the approximated embedding gradients $\tilde{\delta}^{(l)}$. The model accuracy is affected by these approximated intermediate results as well. We will discuss the main factors that influence the model accuracy in Sect. 4.

3 SpanGNN: Memory-Efficient Full-Graph GNN Learning

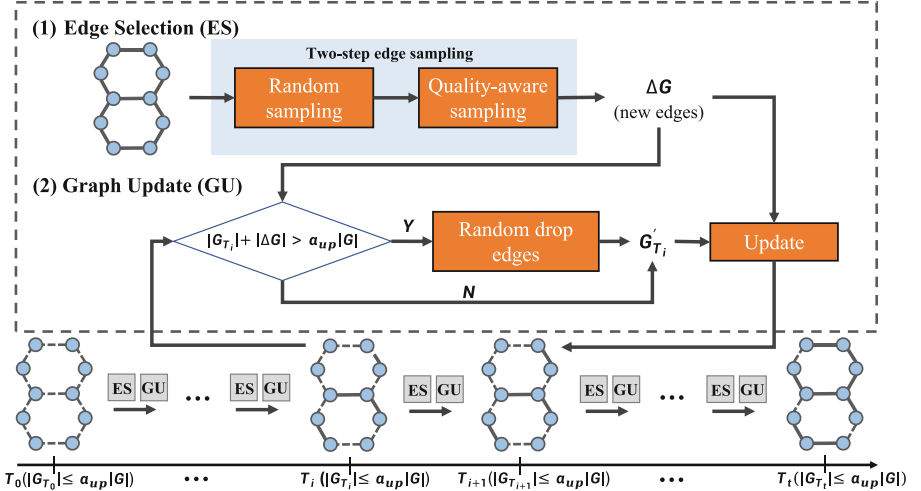


Fig. 1. The framework of SpanGNN.

Figure 1 illustrates the overview of SpanGNN. It starts with an empty spanning subgraph G_{T_0} and progressively includes more edges during the training. For every training epoch T_i , SpanGNN selects a set of edges from the original

graph G , updates the spanning subgraph $G_{T_{i-1}}$ with the selected edge set, and generates a new spanning subgraph G_{T_i} . Furthermore, to limit the peak memory usage, SpanGNN guarantees that, in each training epoch, the edge ratio $\frac{|E_{T_i}|}{|E|}$ does not exceed α_{up} , which is preset by users. According to the definition of edge ratio, the parameter α_{up} implies the upper bound of peak memory usage during the training. Therefore, SpanGNN is able to control the maximal size of peak memory flexibly and is memory-efficient. The pseudocode of the framework is shown in Sect. 2.2 of our technical report [15].

Edge Selection. Edges in the graph contribute differently to the GNN training, so it is important to pick out the most beneficial edges for the training to guarantee the model accuracy. Weighted sampling is a standard approach to select important edges in priority. In this paper, we analyze two types of factors that influence the model accuracy and propose quality-aware edge selection approach in Sect. 4. The new edge selection approach adopts variance-reduced sampling strategy and gradient-noise reduced sampling strategy to select high-quality edges. However, directly sampling from the entire graph with non-uniform probability distribution is time-consuming. We further introduce the two-step edge sampling method to speed up the edge selection. In addition, using the quality-aware edge selection approach, the training process of SpanGNN aligns with the principles of curriculum learning (discussed in Sect. 5), therefore, SpanGNN has high accuracy.

Graph Update. In order to continuously satisfy the edge ratio constraint (i.e., $\frac{|G_{T_i}|}{|G|} \leq \alpha_{up}$), we introduce an edge drop step in the graph update. In each training epoch T_i , if SpanGNN detects that the new spanning subgraph will violate the edge ratio constraint, it first randomly drops a set of edges in $G_{T_{i-1}}$, then adds the selected edge set to the subgraph $G_{T_{i-1}}$; otherwise, the selected edge set is directly added into the subgraph $G_{T_{i-1}}$. The edge drop step helps SpanGNN ensure the memory-efficiency. More importantly, it improves the diversity of the trained spanning subgraphs and enhances the model accuracy as well.

4 Fast Quality-Aware Edge Selection

In this section, we first introduce two types of edge sampling strategies, which are variance-reduced sampling strategy and gradient noise-reduced sampling strategy. Then, we introduce the two-step edge sampling method that optimizes the efficiency of edge selection over large graphs.

4.1 Variance-Minimized Sampling Strategy

The Variance of Aggregated Embedding. Since the edges are probability selected in SpanGNN, the spanning subgraph can be treated as a sampled subgraph from the original graph, and the variance of aggregated embeddings in SpanGNN affects the model accuracy and should not be ignored. Similar to the

existing works [38], the unbiased estimator of aggregated embeddings without activation and the variance of embeddings estimator can be defined as:

$$\xi = \sum_{(l)} \sum_e \frac{b_e^{(l)}}{p_e} \mathbb{1}_e^{(l)} \quad (8)$$

$$\text{Var}(\xi) = \sum_e \frac{(\sum_l b_e^{(l)})^2}{p_e} - \sum_e (\sum_l b_e^{(l)})^2, \quad (9)$$

where p_e denotes the probability of an edge to be sampled, $b_e^{(l)} = P_{v,u} \tilde{x}_u^{(l-1)} + P_{u,v} \tilde{x}_v^{(l-1)}$, P is the propagation matrix (e.g., the normalized adjacency matrix in GCN), \tilde{x} is feature matrix after linear operation, and $\mathbb{1}_e = 1$ if e is in E_s .

Variance Minimization. To minimize the variance of the aggregated embeddings estimator, we follow the strategy used in GraphSAINT [38]. By using the Cauchy-Schwarz inequality, the variance of aggregated embeddings is minimized when $p_e \propto |\sum_l b_e^{(l)}|$, which can be simplified as:

$$p_e \propto P_{v,u} + P_{u,v} = \frac{1}{\text{deg}(u)} + \frac{1}{\text{deg}(v)}. \quad (10)$$

The probability p_e defined in Eq. 10 interprets that if two nodes u, v are connected and they have few neighbors, then edge between u and v are more likely to be sampled and to reduce the variance. In other words, such edges will contain more information for the nodes, which is more conducive to the training of the node.

4.2 Gradient Noise-Reduced Sampling Strategy

The Noise of Gradient. As mentioned before, with the spanning subgraphs, the final learned embeddings change compared to the exact ones. Therefore, the results of loss function and gradient change as well. We define the noise of gradient as the change between $\nabla_{W^{(l-1)}} L_s$ that is calculated by original graph training and spanning subgraph training. We formulate the noise of gradient as below:

$$G_{noise} = \nabla_{W^{(l-1)}} \tilde{L} - \nabla_{W^{(l-1)}} L. \quad (11)$$

Gradient Noise Reduction. The noise of gradient slows down the convergence and affects the model accuracy. To solve the problem, we derive a probability distribution for edge sampling that can reduce the upper bound of gradient noise. The probability of an edge $e(u, v)$ is formulated as:

$$p_{v,u} = \frac{\|P_{*,u}\|_2}{\sum_{(v,u) \in E} \|P_{*,u}\|_2}, \quad (12)$$

where $P_{*,u}$ denotes the vector of $node_u$'s propagation matrix.. The larger the sampling probability of the edge, the smaller the gradient noise in the training process.

Next, we theoretically analyze the above probability and dig into the upper bound of the expected gradient noise, which is summarized in Theorem 1.

Theorem 1. Upper bound of the expected gradient noise. *Given the square of Frobenius norm $\|P\|_F^2$, $\|H^{(l)}\|_F^2$, $\|\delta^{(l)}\|_F^2$ are bounded by some constants B, C, D and the L_2 -norm $\|H^{(l)}W^{(l)}\|$ is bounded by constant ξ . Assume that the activation function σ is ρ_σ -Lipschitz and the gradient $\nabla_{Z^{(l)}}L$ is ρ_Z -Lipschitz, then we have:*

$$E[\|G_{noise}\|_F^2] \leq (2BD\rho_\sigma + 4BC\rho_Z)E\left[\|\tilde{Z}^{(l)} - Z^{(l)}\|_F^2\right] + 4BCD. \quad (13)$$

According to E.q. 13, the upper bound of the expected gradient noise is decided by $\|\tilde{Z}^{(l)} - Z^{(l)}\|_F^2$, i.e., the expected value of the difference of the hidden layer embedding. We further analyze the upper bound of of this difference.

Theorem 2. Upper bound of the expected hidden embeddings' difference. *Given the entire edge set E and the selected edge subset E_s , we derive the following inequation:*

$$E_{E_s}\left[\|\tilde{Z}_{V,*} - Z_{V,*}\|_F^2\right] \leq \frac{1}{|E_s|} \sum_{(v,u) \in E} \frac{1}{p(v,u)} \|P_{*,u}\|_2^2 \xi^2. \quad (14)$$

The detailed proof of the above two theorems are presented in Sect.4.2 of our technical report [15]. As illustrated in E.q. 14, we find the upper bound of hidden embeddings' difference is related to edge sampling probability $p_{v,u}$. By combining Eq. 13 and removing the constants that are hard to calculate, we can formulate a gradient noise optimization problem and minimize the value of the noise by using the following constraint:

$$s.t. \quad \sum_{(v,u) \in E} p(v,u) = 1 \quad (15)$$

Based on E.q. 14 and Eq. 15, by using the Lagrange function, we derive the edge sampling probability as defined in E.q. 12, and the probability can reduce the gradient noise in the spanning subgraph training.

4.3 Two-Step Edge Sampling Method

The probabilities defined by Eqs. 10 and 12 are non-uniform. It is a challenge to fast sample non-uniform distribution in large sample space [24]. An efficient sampling method, Alias sampling [28], requires massive memory and entails the high cost of building data structures. In this paper, we propose a simple but effective approximate sampling method – two-step edge sampling to speed up the quality-aware edge selection process.

In the first step, SpanGNN reduces the sample space by randomly sampling an edge set, denoted as e_t , in iteration T_t . This step confines the final selected edges focusing on the edge set e_t rather than the entire edge set E . In the second step, SpanGNN samples e'_t from the edge set e_t according to the probability defined by E.q. 10 and 12. The pseudocode of quality-aware edge selection with two-step sampling is given in Sect. 4.3 of our technical report [15]. Additionally, we show more details about parameter sensitive analysis of two-step sampling in Sect. 6.6 of our technical report [15].

Here, we discuss the advantages of two-step sampling with a memory-efficient non-uniform sampling method, which first constructs a cumulative probability array, then uses random numbers to select elements. The time complexity entailed by the first step of random sampling from the entire edge set is $O(|e|)$. The time complexity of the second step of weighted sampling is about $O(|e| + |e'| \log(|e|))$. Therefore, the total time complexity of the two-step sampling is $O(|e|) + O(|e'| \log(|e|))$. However, if we directly sample $|e'|$ edges from entire edge set, the time complexity is $O(|E| + |e'| \log(|E|))$. In practice, $|e|$ is typically several to ten times $|e'|$, while $|E|$ can be up to a hundred times larger than $|e|$. Therefore, the time cost of the two-step sampling is lower than that of direct sampling.

5 Connection to Curriculum Learning

In this section, we analyze the connection between SpanGNN and curriculum learning. To our knowledge, curriculum learning increases the robustness of the learned model against noisy training samples by training samples from easy to hard. An intuitive explanation is that curriculum learning spends less time with the harder (noisy) samples to achieve better robustness.

SpanGNN incorporates the principles of curriculum learning by constructing different graph structures (i.e., spanning subgraphs) during the learning process. SpanGNN not only mirrors the educational strategy of progressing from easy to hard lessons, but also aligns with the model’s need to first grasp fundamental concepts before tackling more challenging tasks. The detailed discussion is as follows.

First, in SpanGNN, the empty graph at the beginning can be regarded as the simplest ‘course’. In the training process, edges are gradually added to the graph. This progressive learning process helps the model master basic structural information first, and then learn more complex graph structures, which helps the model to learn more robust and avoid overfitting.

Second, through the Quality-aware Edge Selection, we prioritize edges that are more significant for model training, to help minimize feature variance and reduce gradient noise. Edges with smaller feature variance mean that the aggregated features are more consistent. Also, edges with less gradient noise mean that they can help the model learn more stable. This is similar to the ‘from easy to hard’ in curriculum learning, where we initially learn the data that will be more beneficial for subsequent learning.

6 Experimental Studies

In this section, we start with the descriptions of experimental settings, which cover the datasets and configurations used in the experiments. Then, we evaluate the performance of SpanGNN by comparing it with full-graph training methods, conduct ablation studies to verify the effectiveness of the proposed techniques, and study the efficiency of SpanGNN. Finally, we also compare SpanGNN with mini-batch training methods to demonstrate that generally SpanGNN is able to achieve high accuracy. In addition, due to the limited space, we put the results of parameter sensitivity in Sect. 6.6 of our technical report [15].

6.1 Experimental Setups

Table 1. Dataset statistics

Dataset name	Dataset attributions			
	<i>#Nodes</i>	<i>#Edges</i>	<i>Features</i>	<i>Classes</i>
Ogbn-proteins	132,534	79,122,504	8	112
Reddit	232,965	114,615,892	602	41
Amazon	1,598,960	264,339,468	200	107
Ogbn-products	2,449,029	126,167,053	100	47

Environments and Datasets. We implemented SpanGNN with PyTorch 2.0.1, and the code is released¹. We evaluate the performance of SpanGNN using two common GNN models including GCN [19] and SAGE [16] with the mean aggregator. All experiments are conducted on NVIDIA RTX A6000. We use four large graph datasets. Table 1 lists the summary of the datasets.

Performance Metrics and Evaluation Protocol. Accuracy is used to measure the effectiveness of SpanGNN on Reddit and Ogbn-products datasets, F1-score is used on Amazon, and AUC-ROC is used on Ogbn-proteins. All performance metrics are calculated on the validation set and the results are the average of three times experiments. Furthermore, we conduct experiments under different edge ratios to verify the memory-efficiency of SpanGNN.

Baselines. 1) **Full-graph.** It is a naive full-graph training method, but consumes heavy GPU memory. 2) **DropEdge.** It has good scalability for training on large graphs. 3) **GraphSAGE, ClusterGCN** and **GraphSAINT** are selected as the representations of the mini-batch training.

Additionally, SpanGNN equipped with variance-minimized sampling and gradient noise-reduced sampling respectively are denoted by **SpanGNN-F** and **SpanGNN-G**.

¹ <https://github.com/guxizhi/SpanGNN>.

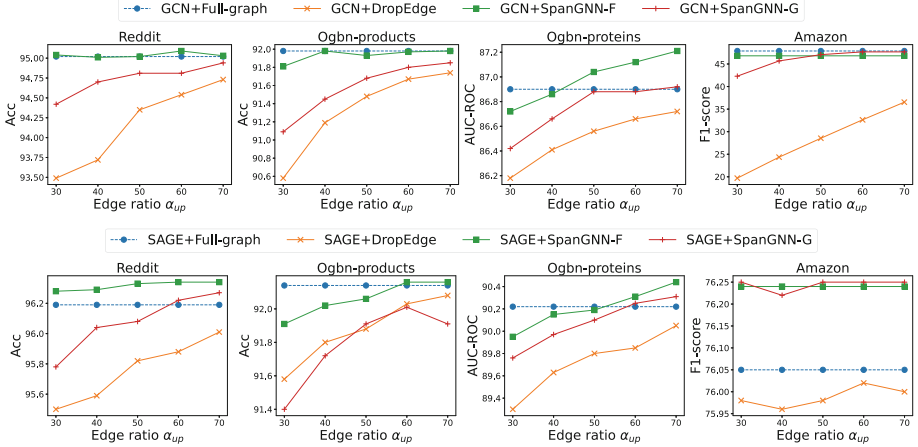


Fig. 2. The performance of the training methods on GCN (Up-side) and SAGE (Down-side) with various edge ratios.

6.2 Performance of SpanGNN

Comparison of Model Accuracy. Figure 2 illustrates the model accuracy of SpanGNN, Full-graph and DropEdge on GCN and SAGE with various edge ratios α_{up} , which are set from 0.3 to 0.7. We see that SpanGNN-F’s performance is similar to or even better than Full-graph. For example, on Reddit, the accuracy of SpanGNN-F is higher than the one of Full-graph with SAGE, regardless of α_{up} . On Ognb-proteins, as α_{up} gets larger, the AUC-ROC of SpanGNN-F gradually exceeds the one of Full-graph. The exception is on Amazon, where SpanGNN-G is better than SpanGNN-F as α_{up} gets larger. This is because SpanGNN-F’s sampling probability on Amazon is extremely skewed, and it is caused by the fact that few edges are connected by two low-degree nodes. These minority edges are given larger weight during selection, causing them to be selected repeatedly in every edge selection. It is hard to obtain enough edges for the spanning subgraph (i.e., the edge ratio of a spanning subgraph is hard to reach α_{up}) and results in a decrease of F1-score. This problem also reduces the size of peak memory usage. In Fig. 3, we see that the peak memory usage of SpanGNN-F is stable with respect to different edge ratios on Amazon.

Compared to SpanGNN, DropEdge suffers from the decrease in model performance more seriously. On Reddit, DropEdge losses the accuracy by up to 1.5% on GCN and by up to 0.8% on SAGE. Even worse, DropEdge severely damages the model’s F1-score on Amazon by more than 25%. Overall, SpanGNN is better at ensuring model’s performance compared to DropEdge.

Comparison of Peak Memory Usage. Here we compare the peak memory usage among SpanGNN, Full-graph, DropEdge. As shown in Fig. 3, we see that reducing the number of edges effectively reduces the size of peak memory by

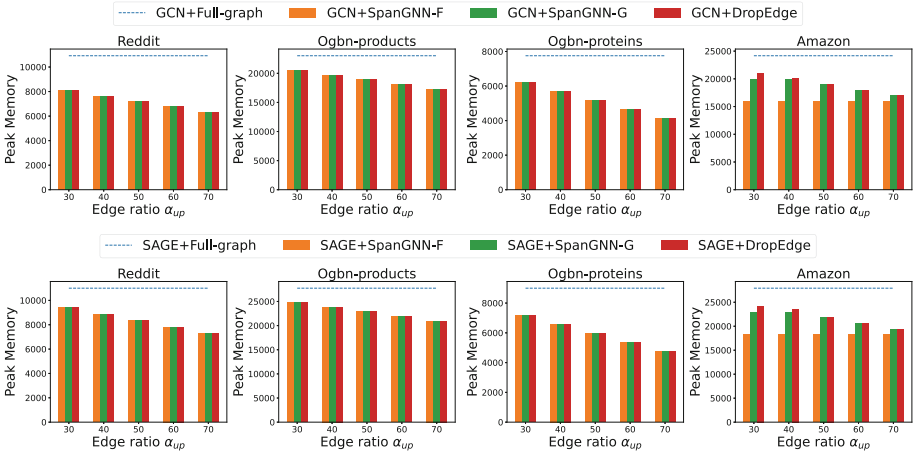


Fig. 3. Peak Memory Usage on GCN(Up-side) and SAGE(Down-side).

comparing SpanGNN and Full-graph. There is no significant difference between SpanGNN and DropEdge, since they drop the same size of edges. In addition, the percentage of peak memory saved is also independent of the model. By using only 30% edges, SpanGNN and DropEdge can reduce the peak memory usage by 42%, 25% and 47% on Reddit, Ognb-products and Ognb-proteins, respectively. Note that on Amazon, due to the actual edge ratio cannot achieve α_{up} , which is discussed in the ‘‘Comparison of model accuracy’’, SpanGNN-F has less peak memory overhead.

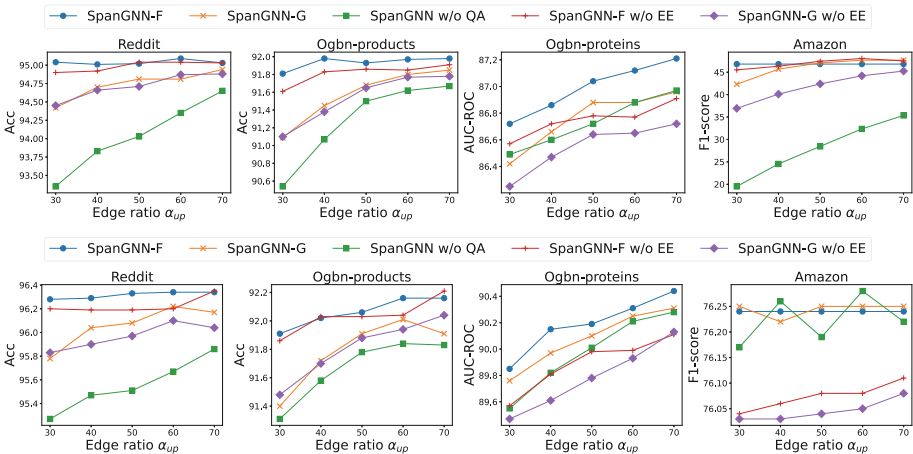


Fig. 4. Ablation studies on GCN(Up-side) and SAGE(Down-side)

6.3 Ablation Studies

Effectiveness of the Framework. In order to verify the effectiveness of the principles of curriculum learning used by SpanGNN, we compare the model accuracy between SpanGNN and SpanGNN w/o EE. Instead of the empty graph, SpanGNN w/o EE is initialized by the graph with $\alpha_{up}|G|$ edges, which are selected by quality-aware edge selection. The results are shown in Fig. 4.

The results indicate that SpanGNN improves the model performance on various datasets by adopting the curriculum learning principles. On Ogbn-proteins, it is clear that SpanGNN outperforms SpanGNN w/o EE and the improvement is around 0.2%. On other datasets, depending on the based model and the value of α_{up} , SpanGNN is generally better than or equal to SpanGNN w/o EE.

Effectiveness of Quality-Aware Edge Selection. In order to verify the effectiveness of variance-minimized sampling and gradient noise-reduced sampling strategies, we compare the model performance among SpanGNN-G, SpanGNN-F, and SpanGNN w/o QA. Here SpanGNN w/o QA applies random sampling instead of quality-aware sampling. The results are shown in Fig. 4.

Generally, SpanGNN-G and SpanGNN-F have better performance than SpanGNN w/o QA. The advantage can reach 1.5% on Reddit and even more than 20% on Amazon. In certain cases, SpanGNN w/o QA might outperform SpanGNN. As discussed in the ‘‘Comparison of model accuracy’’, on Amazon, the spanning subgraph in SpanGNN is easy to contain fewer edges than the required ones defined by α_{up} because of the skewed sampling probability. However, SpanGNN w/o QA can successfully reach α_{up} , and contain sufficient edges. Therefore, SpanGNN does not always perform better than SpanGNN w/o QA on SAGE. Overall, we conclude that the variance-minimized sampling and the gradient noise-reduced sampling generally plays important roles in improving performance of SpanGNN.

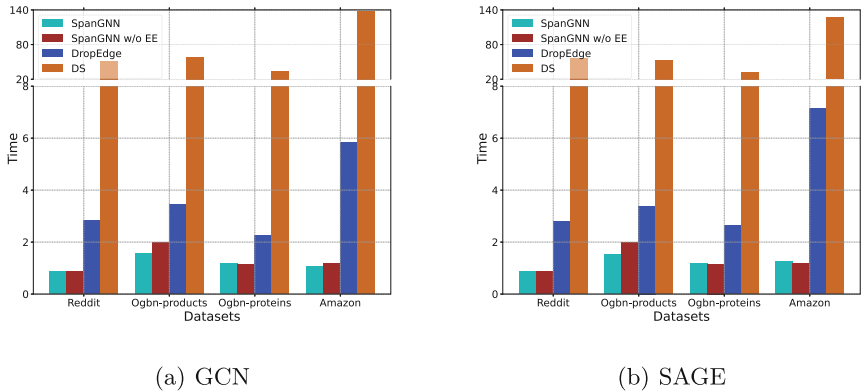


Fig. 5. The average time cost of generating spanning subgraphs.

6.4 Efficiency of SpanGNN

In this section, we demonstrate the efficiency of SpanGNN by comparing the average time cost of generating spanning subgraphs. Figure 5 illustrates the results of SpanGNN, SpanGNN w/o EE, DropEdge and direct sampling from the entire graph based on quality-aware edge selection (DS). To guarantee the fairness of the comparison, all methods have the same edge ratio (i.e., $\alpha_{up} = 0.3$).

As we can see, SpanGNN is more efficient than other frameworks, and integrating curriculum learning principle does not destroy the execution efficiency. Specifically, compared to DropEdge, SpanGNN speeds up from 1.95x to 5.65x on different datasets. As analyzed in Sect. 4.3, the time complexity of generating spanning subgraphs in SpanGNN is proportional to the number of first-step sampling edges e . Due to dropping a lot of edges each iteration (i.e., $\alpha_{up} = 0.3$), DropEdge entails much more time cost than SpanGNN. When compared to DS, SpanGNN speeds up from 26.99x to 132.08x. This is because the time complexity of DS is proportional to the number of edges in the entire graphs, which can reach hundreds of times that of e .

Table 2. The comparison of model performance with mini-batch training methods. Note that SpanGNN’s results are determined by taking the best one among different edge ratios.

GNN	Model	Reddit	Ogbn-products	Amazon	Ogbn-proteins
		Acc	Acc	F1-score	AUC-ROC
GCN	SpanGNN-G	95.26	<u>91.50</u>	47.79	<u>87.11</u>
	SpanGNN-F	<u>95.46</u>	91.68	46.78	87.19
	GraphSAGE	91.99	90.18	28.73	71.31
	ClusterGCN	92.05	89.94	<u>46.86</u>	79.30
	GraphSAINT	96.53	90.14	7.50	80.12
SAGE	SpanGNN-G	96.51	<u>91.33</u>	<u>76.29</u>	<u>90.36</u>
	SpanGNN-F	<u>96.62</u>	91.90	76.26	90.49
	GraphSAGE	94.55	90.57	72.99	82.35
	ClusterGCN	94.74	90.55	77.43	83.54
	GraphSAINT	97.46	90.15	75.21	85.35

6.5 Performance of SpanGNN Compared to Mini-batch Training

In this section, we compare SpanGNN with different mini-batch training methods in terms of model performance. The memory usages of mini-batch training is related to the batch size, and it is more flexible than SpanGNN in terms of memory consumption. However, as shown in Table 2, generally SpanGNN achieves better performance than the mini-batch methods. On Ogbn-products

and Ogbn-proteins, SpanGNN always outperforms the mini-batch methods and its improvements can achieve 1.7% and 7.0% respectively. On other datasets, SpanGNN is either the best one or the second best but very close to the best one. Therefore, compared to the mini-batch training, SpanGNN achieves high model performance.

7 Related Work

7.1 Memory-Efficient Graph Neural Networks

Mini-batch training is an effective approach to reduce the memory consumption. Existing works do a lot of exploration on sampling methods with mini-batch training approach. The works [5, 16] apply node-level sampling to select a set of nodes in neighbors. In this way, it reduce the number of each node’s neighbors in the phase of aggregation. However, it can not resolve the problem of ‘neighbor explosion’ when GNNs goes deeper. The works [8, 42] apply layer-level sampling to select a fixed number of nodes in each GNN layer. Since this type of sampling methods use fixed number of nodes in the each layer, it can alleviate the ‘neighbor explosion’. However, FastGCN [8] suffers from unbalanced receptive fields. LADIES [42] tracks each node’s neighbors in the previous layer and calculates an importance estimator, but causes much overhead. The works [6, 38] apply subgraph-level sampling to limit the aggregation field to a subgraph. ClusterGCN [6] partitions the graph into a set of clusters and then randomly combines partitions to be a mini-batch. GraphSAINT [38] directly forms subgraphs with overlapping nodes among mini-batches. However, compared to Full-graph training, mini-batch training incurs information loss.

Even though almost no work explicitly discusses using spanning subgraph to reduce peak memory usage during GNNs training, there exist some close works. DropEdge [22] randomly removes a certain percentage of edges from the original input graph in each epoch. It alleviates the problem of over-smoothness [2, 14] and over-fitting and can be considered as a strategy that uses spanning subgraph. TADropEdge [12] additionally considers the factors of graph structure. They analyze the graph connectivity and gives larger weight to keep inter-cluster edges in GNNs training. NeuralSparse [41] applies a deep neural network to learn how to sparse graphs with the feedback of downstream prediction tasks. It improves generalization ability by removing potentially task-irrelevant edges. SGCN [20] also considers sparsification as an optimization problem and applies ADMM-based solution to solve it. But these works focus on improving the prediction results, overlooking the problem of peak memory usage. In addition, some other works directly delete or change the structure of GNNs model. SGC [33] reduces redundant calculations by deleting the non-linear activation function between GCN layers. PPRGo [4], by calculating the influence matrix, avoids the overhead of collecting multi-hop neighbors. However, they fundamentally change the characteristics of GNNs, and cannot directly be applied to existing models.

7.2 Curriculum Learning on GNN

Recently, curriculum learning is introduced into GNNs training and achieves performance improvement in GNN models. CurGraph [29] introduces curriculum learning to train GNNs with graphs in ascending order of difficulty. This method uses the informax technique for graph-level embeddings and a neural density estimator to model the embedding distributions. After calculating the difficulty scores of graphs, it first exposes GNN models to easy graphs and moves on to harder ones. They focus on the prediction of graphs. CLnode [31] defines the difficulty of samples at the level of the node and applies various pacing functions to train GNNs from easy-to-hard. It measures nodes' difficulty from the perspective of neighborhoods and features. RCL [40] considers that connections of nodes significantly affect the curriculum learning. It distinguishes the level of difficulty for edges and gradually incorporates more information at the level of edges. However, neither CLnode nor RCL takes the memory usage into account, and they need to train GNNs on the entire graphs. Differently, our work sets an upper bound of the number of edges and designs difficulty scoring function by fully considering the impact of the spanning subgraph.

8 Conclusion

In this paper, we proposed SpanGNN that carries out GNNs training on large-scale graphs efficiently by using spanning subgraphs and integrating the principles of curriculum learning. SpanGNN consists of two main components to limit the memory overhead and ensure the model performance. Quality-aware edge selection samples beneficial edges for spanning subgraph GNN training and follows the manner of curriculum learning to add edges for training. Graph update determines the size of the spanning subgraph at each epoch to control the peak memory. Overall, we provide an efficient large-scale GNN training method that can reduce memory overhead and maintain the model performance.

Acknowledgement. This work is supported by the National Natural Science Foundation of China (Nos. 62272054, 62192784), Beijing Nova Program (No. 20230484319), and Xiaomi Young Talents Program. Lei Chen's work is partially supported by National Science Foundation of China (NSFC) under Grant No. U22B2060, the Hong Kong RGC GRF Project 16213620, RIF Project R6020-19, AOE Project AoE/E-603/18, Theme-based project TRS T41-603/20R, CRF Project C2004-21G, China NSFC No. 61729201, Guangdong Basic and Applied Basic Research Foundation 2019B151530001, Hong Kong ITC ITF grants MHX/078/21 and PRP/004/22FX, Microsoft Research Asia Collaborative Research Grant and HKUST-Webank joint research lab grants.

References

1. Ahmedt-Aristizabal, D., Armin, M.A., Denman, S., Fookes, C., Petersson, L.: Graph-based deep learning for medical diagnosis and analysis: Past, present and future. *Sensors* **21**(14), 4758 (2021)
2. Bause, F., Moustafa, S., Langguth, J., Gansterer, W.N., Kriege, N.M.: On the Two Sides Of Redundancy in Graph Neural Networks (2024)
3. Bengio, Y., Louradour, J., Collobert, R., Weston, J.: Curriculum learning. In: *ICML*, pp. 41–48 (2009)
4. Bojchevski, A., et al.: Scaling graph neural networks with approximate pagerank. In: *KDD*, pp. 2464–2473 (2020)
5. Chen, J., Zhu, J., Song, L.: Stochastic training of graph convolutional networks with variance reduction. In: *ICML*, vol. 80, pp. 942–950 (2018)
6. Chiang, W.L., Liu, X., Si, S., Li, Y., Bengio, S., Hsieh, C.J.: Cluster-GCN: an efficient algorithm for training deep and large graph convolutional networks. In: *KDD*, pp. 257–266 (2019)
7. Choi, E., et al.: Learning the graphical structure of electronic health records with graph convolutional transformer. In: *AAAI*, pp. 606–613 (2020)
8. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: *NIPS*, pp. 3844–3852 (2016)
9. Duvenaud, D.K., et al.: Convolutional networks on graphs for learning molecular fingerprints. *NIPS* **28** (2015)
10. Fan, W., et al.: Graph neural networks for social recommendation. In: *WWW*, pp. 417–426 (2019)
11. Fout, A., Byrd, J., Shariat, B., Ben-Hur, A.: Protein interface prediction using graph convolutional networks. In: *NIPS*. **30**, 6533–6542 (2017)
12. Gao, Z., Bhattacharya, S., Zhang, L., Blum, R.S., Ribeiro, A., Sadler, B.M.: Training robust graph neural networks with topology adaptive edge dropping. arXiv preprint [arXiv:2106.02892](https://arxiv.org/abs/2106.02892) (2021)
13. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: *ICML*, pp. 1263–1272 (2017)
14. Graziani, C., Drucks, T., Bianchini, M., franco scarselli, Gärtner, T.: No PAIN no gain: more expressive GNNs with paths. In: *NeurIPS 2023 Workshop: New Frontiers in Graph Learning* (2023). <https://openreview.net/forum?id=q2xXh4M9Dx>
15. Gu, X., Li, H., Gao, S., Zhang, X., Chen, L., Shao, Y.: SpanGNN: towards memory-efficient graph neural networks via spanning subgraph training (2024)
16. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. *NIPS* **30** (2017)
17. Huang, T., Dong, Y., Ding, M., Yang, Z., Feng, W., Wang, X., Tang, J.: MixGCF: an improved training method for graph neural network-based recommender systems. In: *KDD*, pp. 665–674 (2021)
18. Jia, Z., Lin, S., Gao, M., Zaharia, M., Aiken, A.: Improving the accuracy, scalability, and performance of graph neural networks with roc. In: *MLSys* **2**, 187–198 (2020)
19. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint [arXiv:1609.02907](https://arxiv.org/abs/1609.02907) (2016)
20. Li, J., Zhang, T., Tian, H., Jin, S., Fardad, M., Zafarani, R.: SGCN: a graph sparsifier based on graph convolutional networks. In: *PAKDD*, pp. 275–287 (2020)
21. Rao, J., Zhou, X., Lu, Y., Zhao, H., Yang, Y.: Imputing single-cell RNA-seq data by combining graph convolution and autoencoder neural networks. *IScience* **24**(5), 102393 (2021)

22. Rong, Y., Huang, W., Xu, T., Huang, J.: DropEdge: towards deep graph convolutional networks on node classification. In: ICLR (2019)
23. Schlichtkrull, M., Kipf, T.N., Bloem, P., Van Den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: ESWC, pp. 593–607 (2018)
24. Shao, Y., Huang, S., Li, Y., Miao, X., Cui, B., Chen, L.: Memory-aware framework for fast and scalable second-order random walk over billion-edge natural graphs. *VLDB J.* **30**(5), 769–797 (2021)
25. Sun, C., Ning, Y., Shen, D., Nie, T.: Graph neural network-based short-term load forecasting with temporal convolution. *Data Sci. Eng.* **9**(2), 113–132 (2023)
26. Tan, Q., Liu, N., Hu, X.: Deep representation learning for social network analysis. *Frontiers Big Data* **2**, 2 (2019)
27. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: ICLR, pp. 1–12 (2018)
28. Walker, A.J.: An efficient method for generating discrete random variables with general distributions. *TOMS* **3**(3), 253–256 (1977)
29. Wang, Y., Wang, W., Liang, Y., Cai, Y., Hooi, B.: CurGraph: curriculum learning for graph classification. In: WWW, pp. 1238–1248 (2021)
30. Wang, Z., Wang, Y., Yuan, C., Gu, R., Huang, Y.: Empirical analysis of performance bottlenecks in graph neural network training and inference with GPUs. *Neurocomputing* **446**, 165–191 (2021)
31. Wei, X., Gong, X., Zhan, Y., Du, B., Luo, Y., Hu, W.: CLNode: curriculum learning for node classification. In: WSDM, pp. 670–678 (2023)
32. West, D.B.: Introduction to Graph Theory. Prentice Hall, 2 edn. (September 2000)
33. Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., Weinberger, K.: Simplifying graph convolutional networks. In: ICML, pp. 6861–6871 (2019)
34. Wu, S., Sun, F., Zhang, W., Xie, X., Cui, B.: Graph neural networks in recommender systems: a survey. *Comput. Surv.* **55**(5), 1–37 (2022)
35. Xiao, S., Zhu, D., Tang, C., et al.: Combining graph contrastive embedding and multi-head cross-attention transfer for cross-domain recommendation. *Data Sci. Eng.* **8**, 247–262 (2023). <https://doi.org/10.1007/s41019-023-00226-7>
36. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? In: ICLR, pp. 1–17 (2019)
37. Ying, Z., You, J., Morris, C., Ren, X., Hamilton, W., Leskovec, J.: Hierarchical graph representation learning with differentiable pooling. *NIPS* **31** (2018)
38. Zeng, H., Zhou, H., Srivastava, A., Kannan, R., Prasanna, V.K.: Graphsaint: Graph sampling based inductive learning method. In: ICLR. OpenReview.net (2020)
39. Zhang, M., Chen, Y.: Link prediction based on graph neural networks. *NIPS* **31**, 1–11 (2018)
40. Zhang, Z., Wang, J., Zhao, L.: Curriculum learning for graph neural networks: which edges should we learn first. *NIPS* **36** (2024)
41. Zheng, C., et al.: Robust graph representation learning via neural sparsification. In: ICML, pp. 11458–11468. PMLR (2020)
42. Zou, D., Hu, Z., Wang, Y., Jiang, S., Sun, Y., Gu, Q.: Layer-dependent importance sampling for training deep and large graph convolutional networks. In: NIPS, pp. 1–11 (2019)



AKGNet: Attribute Knowledge Guided Unsupervised Lung-Infected Area Segmentation

Qing En¹ and Yuhong Guo^{1,2(✉)}

¹ School of Computer Science, Carleton University, Ottawa, Canada
qingen@cunet.carleton.ca

² Canada CIFAR AI Chair, Amii, Canada
yuhong.guo@carleton.ca

Abstract. Lung-infected area segmentation is crucial for assessing the severity of lung diseases. However, existing image-text multi-modal methods typically rely on labour-intensive annotations for model training, posing challenges regarding time and expertise. To address this issue, we propose a novel attribute knowledge guided framework for unsupervised lung-infected area segmentation (AKGNet), which achieves segmentation solely based on image-text data without any mask annotation. AKGNet conducts text attribute knowledge learning, attribute-image cross-attention fusion, and high-confidence based pseudo-label exploration simultaneously. It learns statistical information and captures spatial correlations between image and text attributes in the embedding space, iteratively refining the mask to enhance segmentation. Specifically, we introduce a text attribute knowledge learning module by extracting attribute knowledge and deploying it for feature representation learning, enabling the model to learn statistical information and adapt to different attributes. Moreover, we devise an attribute-image cross-attention module by exploiting the correlations between attributes and images in the embedding space to capture spatial dependency information, thus selectively focusing on relevant regions. Finally, a self-training mask improvement process is employed by generating pseudo-labels using high-confidence predictions and enhancing the mask and segmentation iteratively. Experimental results on a benchmark medical image dataset demonstrate the superior performance of our proposed method compared to state-of-the-art segmentation techniques in unsupervised scenarios.

Keywords: Image-Text Model · Unsupervised Medical Image Segmentation

1 Introduction

Medical image analysis is crucial for diagnosing lung diseases, particularly pneumonia and tuberculosis [2, 8, 23]. A fundamental task within this domain is lung-infected region segmentation, aimed at identifying infected regions in medical images [10, 17]. With the advancements in deep neural networks, many fully-supervised medical image segmentation methods have emerged, often relying on

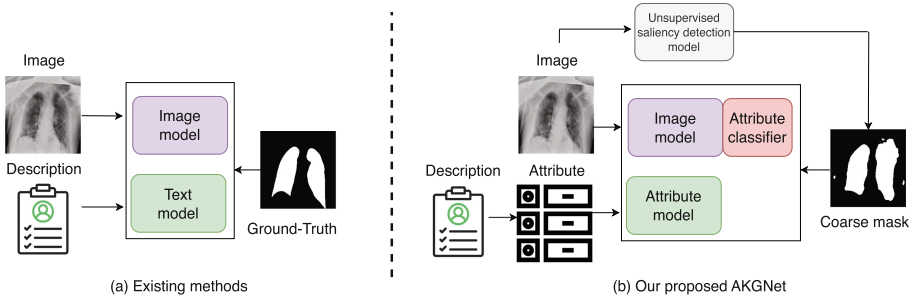


Fig. 1. Illustration of the proposed framework. (a) Existing methods require mask annotations to train the model to achieve image-text lung infection region segmentation. (b) Our proposed AKGNet achieves image-text based unsupervised lung infection region segmentation without mask annotation by mining valuable text attribute knowledge and exploiting statistical information.

densely-labeled masks for model training [4, 22]. Additionally, some approaches utilize image-text pairs to develop more expressive image-text models, thus enhancing segmentation performance in the medical domain [14].

While existing fully supervised methods have shown progress, they typically depend on manual annotations for model training, which involves meticulous delineation of infected regions by domain experts [25]. In light of this, this paper investigates an untouched task: image-text based unsupervised lung-infected area segmentation, aiming to effectively achieve lung-infected area segmentation without mask annotations by autonomously learning feature representations from existing medical images and text data, thereby capturing implicit relationships. The challenges in image-text based unsupervised lung-infected area segmentation can be summarized in two aspects: (1) The absence of mask annotations deprives the model of a direct and reliable training target, which is particularly challenging for complex and heterogeneous lung infections. It is inappropriate to directly train existing fully-supervised models with automated coarse masks as training targets, as they ignore valuable knowledge in text data and are prone to noise from the coarse masks. (2) Redundant information in textual descriptions may hinder proper model training, while integrating textual and image information in an unsupervised manner introduces additional difficulties. Drawing from human cognitive science, humans can extract regularities from the environment over time [24]. We observe that text descriptions contain valuable text attribute knowledge, which can significantly enhance the understanding and interpretation of visual data. The text attribute knowledge encompasses critical details such as pathological findings and clinical observations, which can provide additional statistical information for segmentation model learning [15]. Motivated by this observation, we propose to address the challenges mentioned above for unsupervised segmentation by extracting and leveraging valuable text attribute knowledge from textual descriptions to enhance segmentation.

In this paper, we propose a novel framework, AKGNet, for segmenting lung-infected areas based on image-text data without mask annotations, as shown in Fig. 1. AKGNet can leverage text attribute knowledge to learn statistical information, facilitate attribute-image cross-attention to capture spatial correlations between image and text attributes, and iteratively refine masks by exploring high-confidence based pseudo-labels. Specifically, we introduce a text attribute knowledge learning module that extracts attribute knowledge and integrates it into feature representations. It acquires useful statistical information that adapts to various attributes by deploying attribute classifiers using mask-guided features and attribute targets. Moreover, we develop an attribute-image cross-attention module that exploits correlations between attributes and images in the embedding space. This captures useful spatial dependency information, enabling the model to selectively focus on relevant regions while filtering out irrelevant areas. Furthermore, a self-training mask refinement process is utilized to improve the produced masks by generating pseudo-labels from high-confidence predictions. This iterative approach enhances segmentation results through a proposed self-training loss. The unsupervised segmentation loss, derived from generated coarse masks, is integrated with the other losses to jointly optimize AKGNet. The main contributions of our paper can be summarized as follows:

- We propose a novel AKGNet for unsupervised lung-infected area segmentation based on image-text data without mask annotations.
- AKGNet enables simultaneous learning of text attribute knowledge, cross-attention fusion between attributes and images, and exploration of high-confidence pseudo-labels. It efficiently captures statistical information and spatial correlations between image and text attributes in the embedding space, iteratively refining the masks to improve segmentation.
- Experimental results on a benchmark medical image dataset show that the proposed AKGNet can effectively perform unsupervised lung-infected area segmentation.

2 Related Work

2.1 Medical Image Segmentation

Medical image segmentation aims to assign specific labels to each pixel within an image, encompassing organ classification and lesion area delineation. Existing segmentation models typically deploy two types of architectures: Convolutional Neural Networks (CNN) and Transformer architectures [4, 13, 18, 22, 26]. UNet [22] is a widely used CNN-based model known for its efficient encoder-decoder structure and effective segmentation results. Additionally, models incorporating attention mechanisms [9, 21, 30] and denser connections [31] based on UNet have shown promise in improving medical image segmentation. Inspired by the success of natural scene segmentation, transformer-based methods employ global attention mechanisms like self-attention and trans-attention to capture medical image characteristics [3, 7, 26]. Some methods combine CNN and transformer

structures to leverage global and local semantic information [4, 13]. While previous approaches focus solely on image data and require dense labeling, our work concentrates on image-text based unsupervised lung-infected area segmentation.

2.2 Vision-Language Based Segmentation

Recently, vision-language models have made significant strides in multi-modal visual recognition tasks [20, 29]. The CLIP model [20] notably stands out, utilizing extensive text-image pairs to train its transformer-based image and text encoders. This training process maximizes the similarity between positive image and text embeddings while minimizing the similarity of embeddings from negative pairs. Moreover, various vision-language models have been proposed to enhance natural and medical image segmentation by incorporating text descriptions as prompt information [19]. Referring image segmentation methods add decoders on top of the pre-trained image and text encoders to integrate vision and language information, facilitating segmentation [16, 27]. Similarly, image-text based medical image segmentation methods utilize textual descriptions as a reference to fuse visual and linguistic information through hybrid CNN and transformer architectures [12, 14]. However, these methods rely on mask annotation information and do not fully exploit valuable textual descriptions. In contrast, our proposed method achieves image-text based medical image segmentation without mask annotations, leveraging text attribute information mined from text descriptions.

3 Method

In this section we introduce the proposed AKGNet framework for unsupervised segmentation of lung-infected areas. We first provide an overview of the architecture of AKGNet in Sect. 3.1, followed by the coarse mask generation process detailed in Sect. 3.2. Next, we describe the proposed text attribute knowledge learning module in Sect. 3.3 and present the attribute-image cross-attention module in Sect. 3.4. Subsequently, the self-training mask refining process is presented in Sect. 3.5. Finally, we outline the overall loss function utilized for training AKGNet in Sect. 3.6.

3.1 Overall Framework

For unsupervised segmentation of lung-infected areas, our objective is to train a robust segmentation model using a limited set of N image-text pairs, denoted as $\mathcal{D} = \{(I^n, T^n)\}_{n=1}^N$, where $I^n \in \mathbb{R}^{1 \times H \times W}$ represents the n -th input image, and T^n denotes the corresponding input textual description. Here H and W denote the height and width of the input image, respectively.

The overall architecture of the proposed AKGNet is illustrated in Fig. 2. AKGNet comprises an image encoder $f_I : \mathbb{R}^{1 \times H \times W} \rightarrow \mathbb{R}^{c \times h \times w}$, M attribute classifiers $\{e_m : \mathbb{R}^{c \times h \times w} \rightarrow \mathbb{R}^{a_m}\}_{m=1}^M$, a text attribute encoder $f_A : \mathbb{W}^L \rightarrow$

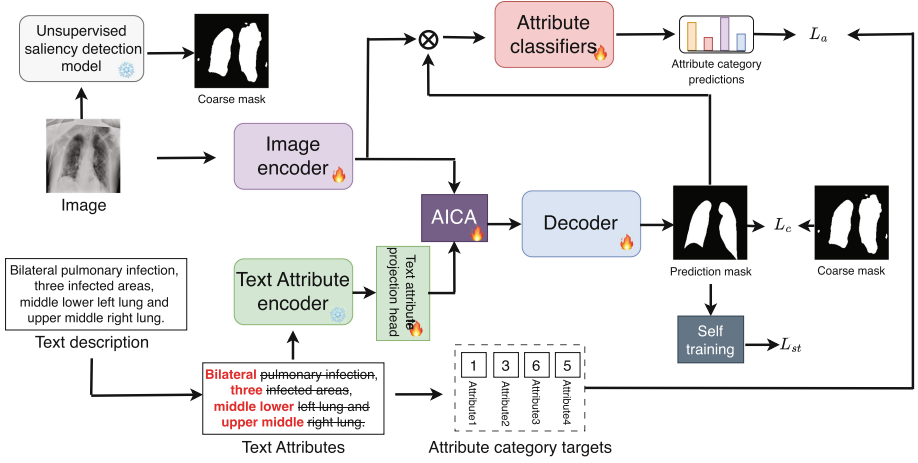


Fig. 2. An overview of the proposed AKGNet. First, a coarse mask is generated. Next, text attribute knowledge is extracted from text descriptions to construct the training targets for the text attribute classifiers. Then, the mask-guided image features extracted from the image encoder are fed into the classifiers to compute L_a . The attribute-image fusion features generated by the AICA module are fed into the image decoder to generate a prediction mask, which is used to compute L_c with the coarse mask. Finally, the self-training mask refining process is implemented by computing L_{st} .

$\mathbb{R}^{d \times L}$, a text attribute projection head $f_{proj} : \mathbb{R}^{d \times L} \rightarrow \mathbb{R}^{c \times L}$, and an image decoder $g_I : \mathbb{R}^{c \times h \times w} \rightarrow \mathbb{R}^{1 \times H \times W}$. Here, c , h , and w denote the number of channels, height, and width of feature embeddings, respectively; \mathbb{W} denotes the word dictionary space, L denotes the maximum number of words of the combined text attribute values A extracted for each image, and d represents the dimension of the attribute embedding; M denotes the number of text attributes, e_m refers to the m -th auxiliary attribute classifier, and a_m indicates the number of categories for the m -th auxiliary classifier. The image embedding and attribute embedding are denoted as $x_I = f_I(I)$ and $x_A = f_A(A)$, respectively. The output of the decoder is used to produce the predicted segmentation mask P .

AKGNet first generates coarse masks of lung-infected areas by using an unsupervised lung saliency detection model, providing estimations for the infected region. Next, the text attribute knowledge learning (TAKL) module is employed to acquire statistical information and adapt to various attributes by extracting attribute knowledge and deploying it to learn feature representations. Then, the attribute-image cross-attention (AICA) module is used to capture spatial dependency information by exploiting the correlations between attributes and images in the embedding space. Furthermore, a self-training process is utilized to improve the segmentation mask by generating pseudo-labels from high-confidence predictions and hence enhance the segmentation model.

3.2 Coarse Mask Generation

Given that the infected region typically resides within the interior of the lungs, we generate a coarse mask that specifically targets the lung region. This provides an unsupervised indication of the potential presence of the infected area. Specifically, given an input image I , we utilize an unsupervised lung saliency detection model N_{sal} [1] to produce a coarse mask $\hat{Y} \in \mathbb{R}^{1 \times H \times W}$ as follows:

$$\hat{Y} = \mathbb{1}[\sigma(N_{sal}(I)) > \tau]. \quad (1)$$

Here, τ denotes a predefined threshold for generating a binary coarse mask, $\sigma(\cdot)$ represents the sigmoid function, and $\mathbb{1}[\cdot]$ denotes the indicator function. As a result, the training dataset \mathcal{D} is expanded to $\mathcal{D} = \{(I^n, T^n, \hat{Y}^n)\}_{n=1}^N$, containing image-text pairs as inputs, along with a coarse mask serving as the coarse segmentation target.

3.3 Text Attribute Knowledge Learning Module

Although the coarse mask estimates potential infection areas, it lacks precise supervisory information and may contain noise. Despite the availability of fully-supervised language-driven medical image segmentation methods that leverage textual descriptions to enhance segmentation through multi-modal learning, they often underutilize the valuable text attribute information within these descriptions. Therefore, we introduce a text attribute knowledge learning (TAKL) module to extract text attribute knowledge from textual descriptions and perform attribute classification. It computes an attribute classification loss by leveraging mask-guided image features and attribute category targets. This module can effectively integrate attribute knowledge into the image feature representation learning, exploiting the underlying statistical information.

Attribute Knowledge Extraction. Initially, we extract the text attributes A from the textual description T . This attribute information from the training set is then utilized to establish the categories for the attributes, which serve as the training targets for the attribute classifiers. Specifically, for a textual description ‘*Bilateral pulmonary infection, three infected areas, middle lower left lung and upper middle right lung*’, we first split it into three parts according to commas to obtain $\{‘Bilateral pulmonary infection’, ‘three infected areas’, ‘middle lower left lung and upper middle right lung’\}$. It consists of descriptions for four attributes: unilateral/bilateral infection, number of infected areas, and the location of the infection within the lung (left or right). For each textual description T , we use its four attribute values to form the corresponding attribute description A ; e.g. ‘*Bilateral pulmonary infection, three infected areas, middle lower left lung and upper middle lower right lung*’ \rightarrow ‘*Bilateral, three, middle lower, upper middle*’. We construct M sets of attribute categories $\{C_m\}_{m=1}^M$ based on the attribute values extracted from the training set. These text attributes, their IDs and values are summarized in Table 1. Consequently, the augmented dataset

Table 1. Attribute knowledge from the Qata-COV19 dataset. Attribute ID: the ID of the attribute (M=4 in total). Text Attribute Meaning: the meaning of the attribute. Text Attribute Values: the set of constituent elements of the attribute. Different attribute IDs are used to construct different classifiers, while different attribute values are used to construct the categories of the corresponding classifiers.

Attribute ID	Text Attribute Meaning	Text Attribute Values
m = 1	unilateral or bilateral of lung infection	unilateral, bilateral
m = 2	number of infected areas	one, two, three, four, five, six
m=3	location of the infected area in the left part	all, upper, middle, lower, upper middle, middle lower, no
m = 4	location of the infected area in the right part	all, upper, middle, lower, upper middle, middle lower, no

$\mathcal{D} = \{(I^n, \hat{Y}^n, A^n, \{C_m^n\}_{m=1}^M)\}_{n=1}^N$ includes attribute values and attribute category information, wherein A^n supplants T^n in each sample, offering a more nuanced representation. Here, C_m^n denotes the category value for the m -th attribute of the n -th sample.

Mask-Guided Attribute Knowledge Classification. After obtaining the attribute category values as prediction targets, a conventional approach involves classifying the intermediate image features produced by the segmentation model. However, this method solely focuses on the image encoder, overlooking the image decoder responsible for mask generation. Hence, we propose to perform mask-guided attribute knowledge classification. This technique utilizes the generated prediction masks to isolate foreground regions within the intermediate features, which are then employed for attribute knowledge classification.

Specifically, given an input image I and the corresponding text attribute value description A , we initially generate the image embedding $x_I \in \mathbb{R}^{c \times h \times w}$ and the prediction mask $P \in \mathbb{R}^{1 \times H \times W}$ as previously described. Next, we produce the masked image feature embedding $x_{MI} \in \mathbb{R}^{c \times h \times w}$, which encapsulates features pertinent to the foreground of the input image, as follows:

$$x_{MI} = x_I \mathbb{1}[\sigma(P) > \alpha]. \quad (2)$$

Here, α denotes a predefined threshold for generating a binary prediction mask. Therefore, the mask-guided attribute classification loss is defined as follows:

$$L_a = \sum_{m=1}^M L_{ce}(e_m(x_{MI}), C_m), \quad (3)$$

where L_{ce} represents the cross-entropy loss function and $\{e_m\}_{m=1}^M$ denotes the M attribute classifiers.

This module extracts valuable attribute knowledge from textual descriptions, which can be deployed to assist feature representation learning through auxiliary

attribute classifications. Moreover, the proposed L_a leverages both the prediction masks and the intermediate features. By enhancing the model’s ability to adapt to diverse informative attributes, it fosters the segmentation model’s acquisition of statistical information directly pertinent to the segmentation task.

3.4 Attribute-Image Cross-Attention Module

For unsupervised lung-infected area segmentation, it is critical to effectively exploit the image and attribute correlations to enable the model to focus on relevant regions and ignore irrelevant regions [11]. To this end, we introduce the attribute-image cross-attention (AICA) module to compute the correlations between attributes and images in the embedding space. It aims to effectively capture spatial dependency information, allowing the model to selective focus on relevant regions while filtering out irrelevant areas.

The AICA module takes the image embedding x_I and the attribute embedding x_A as input, generating the attribute-image fusion feature $x_{AI} \in \mathbb{R}^{c \times h \times w}$ as output. We first pass the attribute embedding x_A through the text attribute projection head f_{proj} to align the embedding dimension d with the channel dimension c of the image features. Then we utilize a learnable parameter matrix $\gamma \in \mathbb{R}^{L \times (hw)}$ to yield the projected attribute embedding $x_{proA} \in \mathbb{R}^{c \times h \times w}$ as follows:

$$x_{proA} = Reshape(f_{proj}(x_A) \times \gamma), \quad (4)$$

where γ transforms the output of f_{proj} into the same size as x_I , and the reshape operation rearranges the features into a tensor with dimension $c \times h \times w$.

Next, we calculate the pairwise dot product between the transformed image embeddings and attribute embeddings to obtain a spatial attention map $S \in \mathbb{R}^{h \times h \times w}$:

$$S = \text{softmax}(\phi(x_I)^T \theta(x_{proA})), \quad (5)$$

where $\text{softmax}(\cdot)$ represents the softmax function; ϕ and θ represent two transformation functions, implemented using two 1×1 convolution layers followed by a reshaping operation (i.e., $c \times h \times w \rightarrow c \times hw$). Meanwhile, we utilize another transformation function, φ (a 1×1 convolution layer followed by a reshaping operation), on the projected attribute embedding x_{proA} and conduct a matrix multiplication operation between it and the transpose of S . Finally, we scale it by a learnable parameter β and conduct an element-wise summation with the image embedding x_I to produce the attribute-image fusion feature $x_{AI} \in \mathbb{R}^{c \times h \times w}$:

$$x_{AI} = \beta S \varphi(x_{proA}) + x_I. \quad (6)$$

x_{AI} serves as input to the image decoder g_I for generating the predicted segmentation mask P . In this scenario, the attribute-image fusion features enable a comprehensive contextual understanding by merging spatial details from medical images with semantic cues from textual descriptions. This fusion equips the model to leverage complementary insights from both modalities, thus improving segmentation accuracy and efficiently capturing infected areas in medical images.

3.5 Self-training Mask Refinement

While the aforementioned modules leverage text attribute knowledge to assist in model training, the resulting segmentation masks may still lack precision. Unsupervised segmentation of lung infection regions poses challenges due to the missing of ground-truth labels and the necessity to enhance segmentation accuracy. Thus, we introduce a self-training mask refinement process to enhance the predicted masks by generating pseudo-labels from high-confidence predictions. Through this process, the model iteratively enhances its segmentation performance, gradually learning from its predictions and adjusting to data intricacies.

Specifically, we select high-probability predictions from the prediction mask P and filter out low-probability ones to obtain credible self-training pseudo-labels $\bar{Y} \in \mathbb{R}^{1 \times H \times W}$:

$$\bar{Y} = \mathbb{1}[\sigma(P) > \delta]. \quad (7)$$

Here, δ represents a predefined threshold to discard noisy pseudo-labels. Subsequently, we formulate a self-training segmentation loss based on these refined pseudo-labels and prediction masks:

$$L_{st} = L_{seg}(P, \bar{Y}). \quad (8)$$

L_{seg} denotes the loss function commonly used in medical image segmentation, consisting of the cross-entropy loss function and the Dice loss function:

$$L_{seg}(P, Y) = \frac{1}{2} * L_{ce}(P, Y) + \frac{1}{2} * L_{dice}(P, Y), \quad (9)$$

where P indicates the prediction mask, and Y indicates the target labels. With the proposed self-training segmentation loss L_{st} , the model iteratively enhances its segmentation performance by updating based on high-confidence pseudo-labels, leading to more refined segmentation masks. Furthermore, this iterative refinement strategy effectively guides the model's attention to complex areas where confident predictions can be made, facilitating the convergence of segmentation masks towards improved accuracy.

3.6 Loss Function

The overall loss function on each sample for training the proposed unsupervised AKGNet framework contains three terms:

$$L_{total} = \lambda_c L_c + \lambda_a * L_a + \lambda_{st} * L_{st}, \quad (10)$$

where λ_c , λ_a and λ_{st} are hyperparameters controlling the trade-off between different loss components. Here L_c represents a coarse segmentation loss calculated from the coarse masks in Sect. 3.2:

$$L_c = L_{seg}(P, \hat{Y}). \quad (11)$$

L_a indicates the attribute classification loss defined in Eq. (3), and L_{st} is the self-training segmentation loss defined in Eq. (8). During training, the weights of the attribute encoder are fixed to pre-trained initial values, while the weights of the other components are updated.

4 Experimental Results

4.1 Experimental Settings

Datasets and Evaluation Metrics. Our proposed framework is evaluated using the QaTa-COV19 dataset [5, 14], which consists of lung X-ray images paired with corresponding textual descriptions. The images are compiled by researchers from Qatar University and Tampere University, while textual descriptions are provided by Li et al. [14]. Following [14], we allocate 5716 samples for training, 1429 for validation, and 2113 for testing. We rectify errors in the textual descriptions, including spelling mistakes. We employ the Dice coefficient and Jaccard coefficient to evaluate the segmentation performance of our framework. The Dice coefficient measures twice the intersection over the sum of the sizes of the segment, while the Jaccard coefficient considers the intersection over the union of the segments. In addition, Param(M) and Flops(G) are used to evaluate the model size and computational complexity, respectively, with Param(M) indicating the number of parameters in millions and Flops(G) representing the number of floating-point operations per second in billions.

Implementation Details. We use the encoder and decoder of UNet [22], which is widely used in medical image segmentation, as our image encoder and decoder. The BERT-embedding model [6] is employed as the attribute encoder. The text attribute projection head is a one-dimensional convolutional layer with a kernel size 3. The weights of the attribute encoder are initialized with pre-trained BERT models [6], while those of other components are randomly initialized. The number of attribute classifiers, denoted as M , is set to 4. Each attribute classifier comprises two fully connected layers with intermediate ReLU functions, and average pooling is conducted on the features before they are fed into the fully connected layers. We adopt the unsupervised lung saliency detection model [1], represented as N_{sal} , for extracting coarse masks. The input image size is 224×224 , with random rotation and flipping. We employ the Adam optimizer with a learning rate $1e-4$ and set the batch size to 12. τ and α are set to 0.5, while δ is set to 0.7. λ_c , λ_a and λ_{st} are set to 1.0, 0.9 and 0.3, respectively.

4.2 Comparison Results

We initially compared AKGNet with two state-of-the-art medical image segmentation methods, UNet [22] and LViT [14], on the QaTa-COV19 dataset under the same experimental setup of image-text unsupervised segmentation for inductive scenarios. These methods were re-implemented and trained under identical settings as AKGNet, utilizing the generated coarse masks. Additionally, we conducted a comparison in a fully supervised experimental setup involving LAVT [28] and LViT [14] with textual descriptions and UNet [22] without textual descriptions. The comparison results, summarized in Table 2, indicate that our proposed AKGNet outperforms the other methods in the same unsupervised scenarios. It achieves a Dice value of 53.8 and a Jaccard value of 41.8

Table 2. Quantitative comparison results on the QaTa-COV19 dataset. We report the results in terms of Dice and Jaccard. We also report the number of model parameters (Param) and the computational complexity (Flops). GT denotes using ground-truth masks as training targets for L_c . CM denotes using the generated coarse masks as the training target for L_c . AKGNet-T represents our transductive results, while AKGNet-I represents our inductive results.

Method	Text	Mask	Label ratio	Param (M)	Flops (G)	Dice (%) \uparrow	Jaccard (%) \uparrow
UNet [22]	×	GT	100%	14.8	50.3	79.0	69.5
LAVT [28]	✓	GT	100%	118.6	83.8	79.3	69.9
LViT [14]	✓	GT	100%	29.7	54.1	83.6	75.1
UNet [22]	×	CM	0%	14.8	50.3	45.1	32.5
LViT [14]	✓	CM	0%	29.7	54.1	49.6	37.3
AKGNet-T	✓	CM	0%	16.8	50.7	53.8	41.8
AKGNet-I	✓	CM	0%	16.8	50.7	55.5	43.7

under the transductive setting, where only testing data is used for unsupervised training, and yields a Dice value of 55.5 and a Jaccard value of 43.7 under the inductive setting. Although LViT incorporates textual descriptive information and employs a hybrid CNN and transformer structure, its results are inferior to ours, accompanied by higher parameter count and computational complexity. These results show that our proposed AKGNet is able to achieve the best segmentation results in unsupervised experimental scenarios while taking into account less computational overhead.

4.3 Ablation Studies

Impact of Different Components. We summarize the impact of different components on the performance in terms of Dice and Jaccard values in Table 3. We initially directly use the generated coarse masks as the segmentation results, yielding a Dice value of 35.2. In the inductive scenario, the experimental result improves to a Dice value of 44.1 when utilizing the coarse masks as targets to train the segmentation model. Further enhancements are observed when incorporating auxiliary L_a or incorporating the AICA module, resulting in Dice values of 48.6 and 49.2, respectively. Moreover, the performance increases to 50.3 when text attributes instead of the original text descriptions are used as the input to the text attribute encoder. The model achieves an even better segmentation result with a Dice value of 52.9 when L_a is excluded, and L_{st} is added. Ultimately, our full model achieves the highest performance, reaching a Dice value of 55.5. Similar trends are observed in the transductive scenario and in terms of Jaccard values. These findings underscore the effectiveness of the various components integrated in AKGNet.

Table 3. Ablation study on the QaTa-COV19 dataset in terms of Dice and Jaccard. L_c : using the proposed coarse segmentation loss. L_a : using the proposed attribute classification loss. L_{st} : using the proposed self-training segmentation loss. CM: using the generated coarse masks as the training target for L_c . ArT: using extracted text attributes (\checkmark) instead of the original text descriptions ($-$) as input to the text attribute encoder. AICA: using the proposed attribute-image cross-attention module.

CM	L_c	L_a	ArT	AICA	L_{st}	Transductive		Inductive	
						Dice \uparrow	Jaccard \uparrow	Dice \uparrow	Jaccard \uparrow
\checkmark	$-$	$-$	$-$	$-$	$-$	35.2	23.9	$-$	$-$
\checkmark	\checkmark	$-$	$-$	$-$	$-$	45.1	32.5	44.1	31.6
\checkmark	\checkmark	\checkmark	$-$	$-$	$-$	47.6	34.6	48.6	36.0
\checkmark	\checkmark	\checkmark	$-$	\checkmark	$-$	48.9	36.2	49.2	36.8
\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	$-$	49.7	38.8	50.3	39.4
\checkmark	\checkmark	$-$	\checkmark	\checkmark	\checkmark	51.6	39.2	52.9	40.7
\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	53.8	41.8	55.5	43.7

Impact of Different Attributes in L_a . We summarize the impact of different attributes in L_a on the performance for the transductive scenario in terms of Dice value in Fig. 3 (a). The experimental results demonstrate that considering all attributes yields the best performance, achieving a Dice value of 53.8. Removing the classification loss for any individual attributes leads the model to disregard crucial statistical information during training, resulting in decreased segmentation accuracy. When omitting the first attribute (unilateral/bilateral) or the second attribute (number), the Dice values decrease from 53.8 to 52.7 and 52.1, respectively. Similarly, ignoring the information on the left or right position (third and fourth attributes) also results in performance reduction. Furthermore, when both information on the left and right positions are ignored, the model’s performance further decreases to a Dice value of 51.3. These experimental results underscore the importance of simultaneously exploiting classifiers for multiple attributes to learn statistical information for assisting the segmentation task.

Impact of Using Mask-Guided Intermediate Features or Original Intermediate Features. We summarize the impact of using mask-guided intermediate features or original intermediate features as the attribute classifiers’ inputs in the transductive scenario in terms of Dice value in Fig. 3 (b). Utilizing the mask-guided features yields significantly better results than using the original features, with an improvement of 1.1% in terms of Dice value. This improvement can be attributed to the fact that using mask-guided features is directly relevant to the segmentation task, and employing the prediction mask as guidance enables both the image encoder and image decoder to be trained effectively, allowing the model to generate better prediction masks while learning attribute statistics. Importantly, our proposed architecture can achieve promising results using either of these two strategies, underscoring the effectiveness of

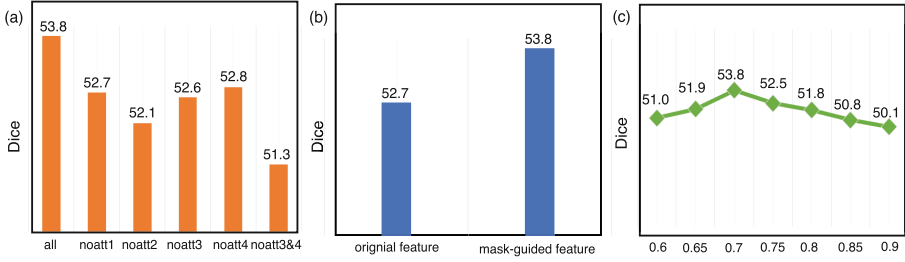


Fig. 3. Ablation study on the (a) impact of different attributes in L_a ; (b) impact of using mask-guided intermediate features or original intermediate features; (c) impact of the threshold in self-training δ . We report the Dice values.

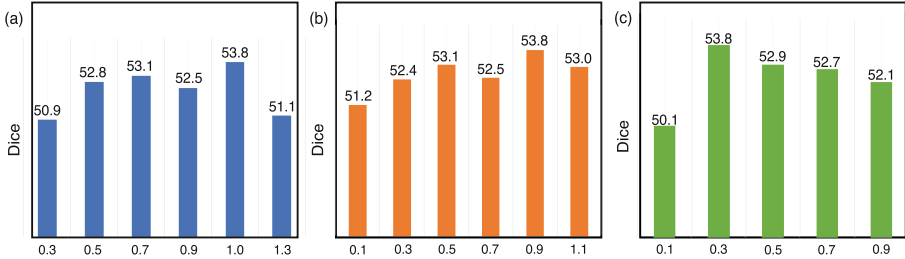


Fig. 4. Hyperparameter analysis: (a) λ_c ; (b) λ_a ; (c) λ_{st} . We report the Dice values.

our proposed method, which does not rely solely on features guided by prediction masks.

Impact of Threshold δ in Self-training. We summarize the impact of the threshold in self-training, denoted as δ , in the transductive scenario in terms of Dice value in Fig. 3 (c). This parameter governs the confidence level in the self-training process for obtaining high-confidence pseudo-labels, where larger values indicate that the prediction masks are filtered through higher confidence to obtain pseudo-labels. We tested a set of different δ values. Experimental results reveal that the best performance is achieved when the δ value is set to 0.7, yielding a Dice value of 53.8. Deviation from this value, either by increasing or decreasing it, leads to a decrease in experimental results. This is because excessively high confidence levels result in too few pseudo-labels, while overly low confidence levels introduce too much noise in the pseudo-labels, both of which are detrimental to achieving optimal results through self-training.

Impact of Hyperparameter Values. We summarize the impact of the weights of different loss terms, i.e., the values of hyperparameters $\{\lambda_c, \lambda_a, \lambda_{st}\}$, on the segmentation performance in the transductive scenario in terms of Dice value in Fig. 4. When adjusting one weight, all other weights remain fixed. Firstly, concerning the weight of the coarse segmentation loss L_c (Fig. 4 (a)), the best

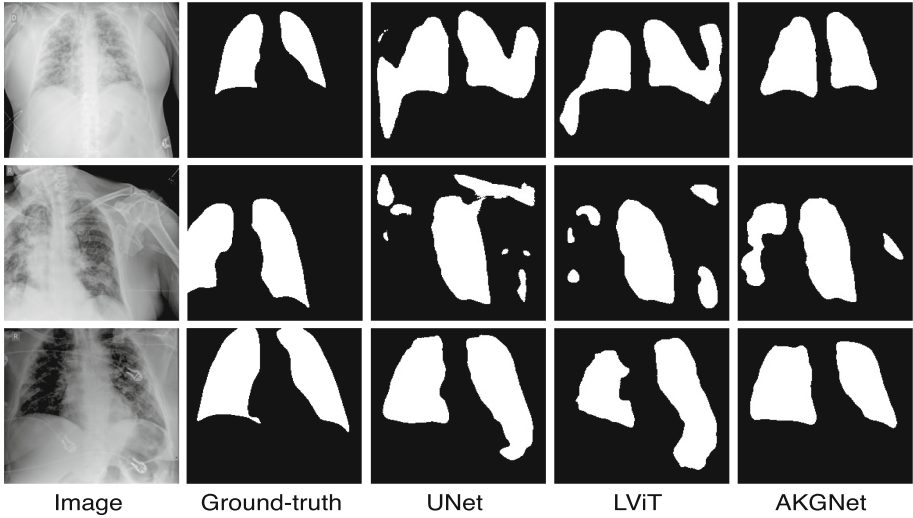


Fig. 5. Visualization examples of different methods under the image-text unsupervised segmentation experimental scenario. The left two columns present input images and ground-truths. The remaining three columns present the segmentation results of UNet, LViT, and our proposed AKGNet.

result is obtained when λ_c is set to 1.0, yielding a Dice value of 53.8. The results progressively deteriorate as the weight is reduced, indicating its crucial role in providing the model with a task-related training objective. Conversely, increasing λ_c leads to decreased segmentation effectiveness, suggesting that excessive noise within L_c could have an adverse impact. Secondly, for the attribute classification loss L_a (Fig. 4 (b)), the model achieves its best results when λ_a is set to 0.9. Conversely, reducing or increasing this weight may cause the model to disregard statistical information or overly focus on categorical information, resulting in inferior outcomes. Finally, concerning the self-training segmentation loss L_{st} (Fig. 4 (c)), the optimal result is obtained when λ_{st} is set to 0.3. Setting λ_{st} to a larger value (e.g., 0.9) slightly degrades the result, suggesting that an excessively large weight during self-training could overly bias the model towards current predictions, diminishing its efficacy. Conversely, reducing it to 0.1 diminishes the extent to which the model undergoes self-training.

4.4 Qualitative Evaluation Results

To further illustrate the effectiveness of AKGNet, we present qualitative results in Fig. 5, comparing our framework with the state-of-the-art methods under unsupervised settings on the QaTa-COV19 dataset. Visualizations reveal that AKGNet outperforms other methods, particularly in scenarios with asymmetric infected regions or when these regions are near the image edge. UNet, when used without textual guidance, tends to produce numerous mis-segmentations. While

LViT utilizes textual description information, its performance in unsupervised scenarios falls short compared to AKGNet.

5 Conclusion

In this paper, we introduced a novel AKGNet framework for image-text based unsupervised lung-infected areas segmentation. AKGNet uses a text attribute knowledge learning module to exploit text attribute knowledge and learn statistical information by facilitating model adaptation to various text attributes. Additionally, an image-attribute cross-attention module is used to capture spatial dependencies between images and text attributes by exploiting their correlations in the embedding space. Furthermore, a self-training mask refinement process is employed to accelerate model convergence towards refined masks. Experimental results demonstrate the effectiveness of the proposed framework, surpassing existing segmentation methods in unsupervised scenarios.

References

1. de Almeida, P.A.C., Borges, D.L.: A deep unsupervised saliency model for lung segmentation in chest x-ray images. *Biomed. Signal Process. Control* **86**, 105334 (2023)
2. Anwar, S.M., Majid, M., Qayyum, A., Awais, M., Alnowami, M., Khan, M.K.: Medical image analysis using convolutional neural networks: a review. *J. Med. Syst.* **42**, 1–13 (2018)
3. Cao, H., Wang, Y., Chen, J., Jiang, D., Zhang, X., Tian, Q., Wang, M.: Swin-unet: Unet-like pure transformer for medical image segmentation. In: *European Conference on Computer Vision (ECCV) Workshops* (2023). https://doi.org/10.1007/978-3-031-25066-8_9
4. Chen, J., et al.: Transunet: Transformers make strong encoders for medical image segmentation. *arXiv preprint arXiv:2102.04306* (2021)
5. Degerli, A., Kiranyaz, S., Chowdhury, M.E., Gabbouj, M.: Osegnet: Operational segmentation network for covid-19 detection using chest x-ray images. In: *2022 IEEE International Conference on Image Processing (ICIP)* (2022)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)* (2019)
7. Dong, B., Wang, W., Fan, D.P., Li, J., Fu, H., Shao, L.: Polyp-pvt: Polyp segmentation with pyramid vision transformers. *arXiv preprint arXiv:2108.06932* (2021)
8. Duncan, J.S., Ayache, N.: Medical image analysis: progress over two decades and the challenges ahead. *IEEE Trans. Pattern Anal. Mach. Intell.* **22**(1), 85–106 (2000)
9. Keshwani, D., Kitamura, Y., Ihara, S., Iizuka, S., Simo-Serra, E.: TopNet: topology preserving metric learning for vessel tree reconstruction and labelling. In: Martel, A.L., et al. (eds.) *MICCAI 2020*. LNCS, vol. 12266, pp. 14–23. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59725-2_2
10. Fan, D.P., et al.: Inf-net: automatic Covid-19 lung infection segmentation from ct images. *IEEE Trans. Med. Imaging* **39**(8), 2626–2637 (2020)
11. Fu, J., Liu, J., Tian, H., Li, Y., Bao, Y., Fang, Z., Lu, H.: Dual attention network for scene segmentation. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019)

12. Lee, G.E., Kim, S.H., Cho, J., Choi, S.T., Choi, S.I.: Text-guided cross-position attention for segmentation: Case of medical image. In: International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI) (2023)
13. Li, Z., et al.: Ttfens: a cnn-transformer hybrid network for medical image segmentation. In: International Conference on Artificial Neural Networks (ICANN) (2022)
14. Li, Z., et al.: Lvit: language meets vision transformer in medical image segmentation. *IEEE Transactions on Medical Imaging* (2023)
15. Liu, F., You, C., Wu, X., Ge, S., Sun, X., et al.: Auto-encoding knowledge graph for unsupervised medical report generation. *Advances in Neural Information Processing Systems (NeurIPS)* (2021)
16. Lüddecke, T., Ecker, A.: Image segmentation using text and image prompts. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022)
17. Mansoor, A., et al.: Segmentation and image analysis of abnormal lungs at ct: current approaches, challenges, and future trends. *Radiographics* **35**(4), 1056–1076 (2015)
18. Milletari, F., Navab, N., Ahmadi, S.A.: V-net: fully convolutional neural networks for volumetric medical image segmentation. In: International Conference on 3D Vision (3DV) (2016)
19. Poudel, K., Dhakal, M., Bhandari, P., Adhikari, R., Thapaliya, S., Khanal, B.: Exploring transfer learning in medical image segmentation using vision-language models. arXiv preprint [arXiv:2308.07706](https://arxiv.org/abs/2308.07706) (2023)
20. Radford, A., et al.: Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning (ICML) (2021)
21. Rahman, M.M., Marculescu, R.: Medical image segmentation via cascaded attention decoding. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) (2023)
22. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
23. Shen, D., Wu, G., Suk, H.I.: Deep learning in medical image analysis. *Annu. Rev. Biomed. Eng.* **19**, 221–248 (2017)
24. Sherman, B.E., Graves, K.N., Turk-Browne, N.B.: The prevalence and importance of statistical learning in human cognition and behavior. *Curr. Opin. Behav. Sci.* **32**, 15–20 (2020)
25. Tajbakhsh, N., Jeyaseelan, L., Li, Q., Chiang, J.N., Wu, Z., Ding, X.: Embracing imperfect datasets: a review of deep learning solutions for medical image segmentation. *Med. Image Anal.* **63**, 101693 (2020)
26. Wang, H., et al.: Mixed transformer u-net for medical image segmentation. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2022)
27. Wang, Z., et al.: Cris: clip-driven referring image segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022)
28. Yang, Z., Wang, J., Tang, Y., Chen, K., Zhao, H., Torr, P.H.: Lavt: language-aware vision transformer for referring image segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022)
29. Zhang, S., et al.: Large-scale domain-specific pretraining for biomedical vision-language processing. arXiv preprint [arXiv:2303.00915](https://arxiv.org/abs/2303.00915) (2023)

30. Zhang, Z., Fu, H., Dai, H., Shen, J., Pang, Y., Shao, L.: ET-Net: a generic edge-attention guidance network for medical image segmentation. In: Shen, D., et al. (eds.) MICCAI 2019. LNCS, vol. 11764, pp. 442–450. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32239-7_49
31. Zhou, Z., Rahman Siddiquee, M.M., Tajbakhsh, N., Liang, J.: Unet++: A nested u-net architecture for medical image segmentation. In: Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support (2018)



Diffusion Model in Normal Gathering Latent Space for Time Series Anomaly Detection

Jiashu Han¹, Shanshan Feng^{2,3,4}, Min Zhou⁵, Xinyu Zhang¹, Yew Soon Ong^{2,6}, and Xutao Li¹(✉)

¹ Harbin Institute of Technology, Shenzhen, China

zhangxinyu45@stu.hit.edu.cn, lixutao@hit.edu.cn

² Centre for Frontier AI Research, A*STAR, Singapore, Singapore

asysong@ntu.edu.sg

³ Institute of High-Performance Computing, A*STAR, Singapore, Singapore

⁴ The Singapore Institute of Manufacturing Technology, A*STAR, Singapore, Singapore

⁵ Huawei Technologies, Shenzhen, China

zhoumin27@huawei.com

⁶ Nanyang Technological University, Singapore, Singapore

Abstract. Generative models have been widely used in time series anomaly detection, effectively identifying abnormal states within the data. Among these, diffusion models stand out for their powerful generative capabilities and have been increasingly applied to anomaly detection tasks, showcasing advantages in handling complex time series data. However, existing approaches employ diffusion models directly in the numerical space, which leads to several limitations, particularly in failing to reconstruct normal time series. To address these issues, we propose NGLS-Diff, an innovative approach that uses a diffusion model within a normal gathering latent space to enhance anomaly detection capabilities. This method introduces a novel latent space that captures the distributions of normal temporal patterns, thus rectifying the drawbacks of previous diffusion models. By operating the diffusion process in the normal gathering latent space, our approach significantly enhances the model's ability to detect anomalies within normal time series data. Extensive experiments conducted on four real-world datasets demonstrate the significant performance improvements of our NGLS-Diff compared to various methods, validating its effectiveness in time series anomaly detection.

Keywords: Time Series Anomaly Detection · Diffusion Model · Representation Learning

1 Introduction

Recently, diverse equipment and technologies continuously produce extensive data monitored by sensors across areas such as industrial machinery and space

J. Han and S. Feng—The first two authors contributed equally.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024
A. Bifet et al. (Eds.): ECML PKDD 2024, LNAI 14943, pp. 284–300, 2024.

https://doi.org/10.1007/978-3-031-70352-2_17

exploration [8, 11, 36]. Identifying anomalies within these vast datasets is critical for ensuring security, preventing financial losses, and sustaining efficiency [4]. The core challenge of current research lies in identifying rare anomalies amidst a vast amount of normal data. Consequently, there is a growing emphasis on developing unsupervised methods that can effectively handle time complexities and distinguish anomalies from normal data points.

Effectively detecting time series anomalies requires learning the normal temporal patterns from complex time distributions [51]. Abnormal time patterns are characterized by uncertainty and infrequency [7]. Conversely, normal patterns exhibit periodicity and occur more frequently. The conventional methods for detecting time series anomalies involve reconstructing the time series using generative models [3, 10, 22], where anomalies are identified by examining the discrepancies before and after reconstruction at various time points [6]. In search of more robust and accurate methods, researchers have already begun exploring advanced generative techniques applied to time series anomaly detection tasks.

Recently, diffusion models [13] have garnered widespread attention in the anomaly detection community [14, 25]. Diffusion models distinguish themselves from other generative models for anomaly detection in time series data due to their stable training and robustness. Some researchers have begun exploring approaches based on diffusion models to tackle time-series anomaly detection tasks. Diffusion models demonstrate impressive data generation capabilities in time series anomaly detection, producing samples that closely resemble real data, thereby enhancing the accuracy of anomaly detection. However, due to the latent variables generated by the diffusion model lack of semantic information [18, 33], using them for time series anomaly detection in numerical space may lead to two key challenges: (1) Direct reconstruction in numerical space may encounter obstacles because the diffusion model cannot learn the underlying features of the time series; (2) Models operating in numerical space are often more susceptible to outliers in the data, which degrades the quality of anomaly detection.

To handle the above challenges, we introduce a distinctive methodology that employs diffusion models within a specialized latent space tailored for time series anomaly detection. The diffusion model exhibits stronger generative capabilities in latent space [37] but has not been studied for detecting time series anomalies. Additionally, latent space analysis of time series effectively extracts crucial temporal features, simplifying the data’s structure and reducing its complexity [30]. Inspired by [12] and [47], we adeptly identify the spatial distribution of normal patterns that commonly occur across multiple time series patches. Utilizing latent states sampled from the distribution of normal patterns as input enhances the generative capability of the diffusion model. It also minimizes the disruptions caused by time series abnormal patterns during the generation process.

In this paper, we develop a novel time series anomaly detector, NGLS-Diff, which leverages a normal gathering latent space for learning normal temporal patterns and then conducts diffusion model generation operations within it. The latent space, specifically designed for normal pattern gathering, prioritizes learning potential representations of time series’ normal patterns that occur

more frequently across multiple patches. To preserve the long-term dependencies and global characteristics of normal patterns, the latent space captures shared features among these types of latent states. Utilizing information from normal patterns assists the diffusion model in reconstructing time series within the structural latent space, thereby facilitating anomaly detection.

The main contributions of this work are summarized as follows:

- We employ the diffusion model to reconstruct time series within a latent space tailored for gathering normal patterns, thereby enhancing time series anomaly detection. To our knowledge, this is the first work to utilize the diffusion model in latent spaces, rather than conventional numerical spaces, for identifying time series anomalies.
- We introduce a novel normal gathering latent space, which is designed for learning the distribution of normal temporal patterns. The latent space can mitigate the interference of abnormal patterns in the generation process.
- We conduct extensive experiments on four real-world benchmark datasets to demonstrate that the NGLS-Diff significantly outperforms current state-of-the-art methods. We also conduct ablation studies to analyze the contribution of NGLS-Diff’s components. Code is available at <https://github.com/h-jiashu/NGLS-Diff>.

2 Related Work

2.1 Time Series Anomaly Detection

In the early days, time series anomaly detection primarily relied on statistical methods, such as ARIMA models [29], exponential smoothing [9], the three-sigma rule [34], and so on. With the rise of machine learning, researchers began to employ machine learning techniques for time series anomaly detection tasks, such as One-Class SVM [44], Isolation Forest [24], Opprentice [23], and so on.

Nowadays, deep learning has become the common approach. The deep learning models for time series anomaly detection can mainly be categorized into two types: prediction-based and reconstruction-based. Prediction-based models compare the predicted time series with the actual values to determine anomalies. Malhotra et al. [26] utilized stacked LSTM networks to detect anomalies in time series data. Li et al. [19] extracted anomalies from the prediction error sequence using a dynamic thresholding method. Jhin et al. [17] utilized a neural network based on neural controlled differential equations, which not only detects anomalies but also predicts the occurrence of anomalies in advance.

Reconstruction-based models determine anomalies by reconstructing time series data and comparing the differences between the original and reconstructed sequences. OmniAnomaly [41] utilizes the temporal dependency and randomness of multivariate time series to model the series, thereby performing anomaly detection. USAD [2] generates time series data by adversarially training two autoencoders. Interfusion [21] captures inter-metric and temporal dependencies in multivariate time series using hierarchical Variational AutoEncoder, enabling

precise anomaly detection and interpretation. Anomaly Transformer [48] detects anomalies based on the differences in associations between normal points and abnormal points. DCdetector [50] distinguishes anomalies by comparing the results of different reconstruction methods, achieving state-of-the-art results. Compared to conventional reconstruction models, our model focuses on learning the distribution of normal temporal patterns rather than representing the whole information of the time series.

2.2 Diffusion Model for Time Series Analysis

The currently popular and powerful generative model based on the diffusion model was initially proposed by Ho and Jain [13]. Song et al. [40] introduced the Diffusion Denoising Implicit Model (DDIM), significantly accelerating the sampling efficiency of the diffusion model.

Although the diffusion model was initially applied to content generation in the computer vision field, its robust generative capability has led to its widespread adoption in time series analysis. D³VAE [20] enhances and optimizes time series forecasting by integrating a Bidirectional Variational Auto-Encoder equipped with diffusion, denoise, and disentanglement features; Rasul et al. [35] proposed a multivariate probabilistic time series forecasting model that integrates a diffusion model, sampling by estimating the data distribution gradient at each time step.

In recent years, many models using the diffusion model for time series anomaly detection tasks have emerged. Wang et al. [43] address drift from non-stationary environments through dynamic decomposition and reconstruction, effectively handling anomalies in long-period multivariate time series using data-time mix attention and noise diffusion. DiffAD [45] uses a density ratio-based strategy and interpolation combined with diffusion mode for data generation to perform anomaly detection; D³R [43] combines dynamic decomposition with diffusion model to address the drift issue in anomaly detection. Differing from other diffusion models, in NGLS-Diff, the diffusion model utilizes temporal latent states from the normal gathering latent space rather than numerical values.

3 Problem Formulation

The task of multivariate time series anomaly detection is conducted on time series $X_{1:T} \in R^{T \times D}$ generated by multiple sensors, where T represents the length of the time series, and D represents the number of dimensions. The task aims to detect outliers in multivariate time series.

Our approach NGLS-Diff belongs to the reconstruction-based anomaly detection category. It employs an unsupervised learning approach to learn the representation of multivariate time series. The model processes this information to produce reconstructed outcomes, represented as \hat{X} . Anomaly detection is subsequently conducted by point-wise comparison between the original series (X)

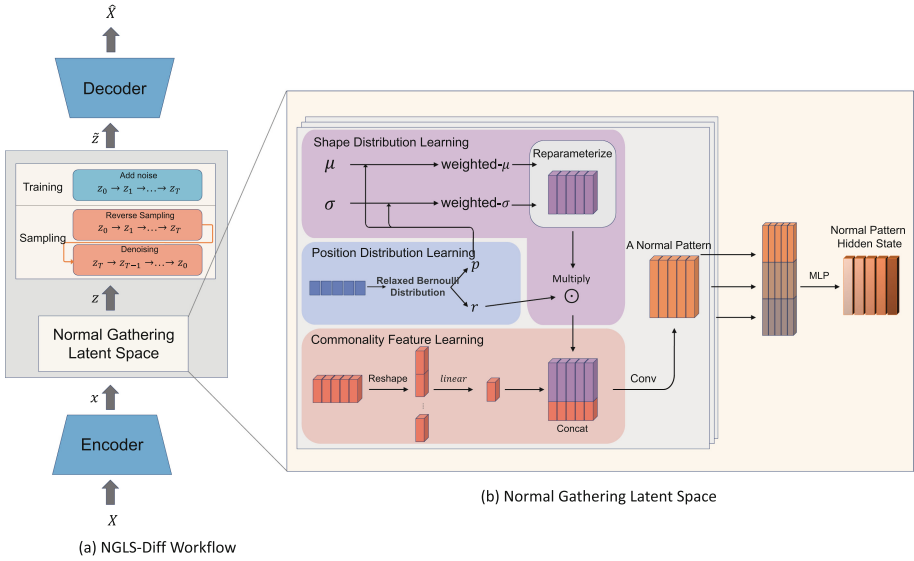


Fig. 1. (a) NGLS-Diff workflow. (b) Normal gathering latent space architecture.

and the reconstructed series (\hat{X}) across the time dimension. Anomalies are identified based on the significant deviations between X and \hat{X} , thereby determining the anomaly detection outcomes. Hence, the essence of effective time series anomaly detection depends on highlighting discrepancies between original and reconstructed data at specific anomalous instances.

4 Methodology

4.1 Overview

The workflow of NGLS-Diff is shown in Fig. 1(a). NGLS-Diff primarily consists of three parts: the autoencoder (AE), the normal gathering module, and the diffusion model. Firstly, the encoder within the AE maps time series data from the numerical space to a latent space, facilitating feature extraction from the time series information. Secondly, the normal gathering module extracts normal time series patterns from the latent space and constructs the normal gathering latent space. Thirdly, the diffusion model employs latent states in normal gathering latent space for reconstruction purposes. Finally, the reconstructed latent states are mapped back to the numerical space through the AE’s decoder.

4.2 Autoencoder

Autoencoder’s encoder maps the original time series to a basic latent space using several linear layers. We use x to represent time series in the basic latent space:

$$x = \text{Encoder}(X) \tag{1}$$

where the shape of x is $T \times d$ and d represents the feature dimension of the time series in the latent space. We compress the feature dimensions of the time series while keeping the time dimension unchanged. At the end of the model, the decoder in AE is used to map the time latent state back to the numerical space.

4.3 Normal Gathering Latent Space

Normal gathering latent space learns the distribution of normal patterns for time series and provides temporal hidden states for the diffusion model’s reconstruction processes. Normal temporal patterns can be represented by three aspects: shape, location, and global characteristics. Therefore, as shown in Fig. 1(b), there are three learning parts included in the normal gathering latent space: **Position Distribution Learning**, **Shape Distribution Learning**, and **Commonality Feature Learning**.

The latent space learns the distribution of normal temporal patterns from the output of the Encoder. Additionally, because time series patches can reflect richer local semantic information, we divide x into long non-overlapping patches $\{x_i\}$. Each patch x_i with a shape of $l_p \times d$, where l_p is the length of each patch. And $\{x_i\}$ has n_p patches. Since there may be multiple normal temporal patterns in time series, we have set up a multi-layer latent space network, with each layer learning a distinct normal temporal pattern independently. We use the superscript p to represent "pattern", and the subscript p to represent "patch".

Position Distribution Learning. The strength of a normal temporal pattern’s performance may vary in different positions of the time series. Latent Space needs to learn the position distribution of normal patterns to express the performance at different patches in the time series. The posterior distribution of normal patterns p is $Q(z_{pos}^p|x)$. We use a temporal convolutional neural network (CNN) to compress the time dimension of the patch to 1 to represent the patch position. Thus, the distribution of normal pattern positions is equivalent to the probability of normal patterns occurring on each patch:

$$y_i = \text{CNN}(x_i), \tag{2}$$

$$\hat{z}_{pos}^p \sim Q(z_{pos}^p|\{y_i\}), \tag{3}$$

where y_i represents the learned representation for the i -th patch and the shape of y_i is $1 \times d$. \hat{z}_{pos}^p is the sample of distribution. The range of \hat{z}_{pos}^p is $(0, 1)$, and \hat{z}_{pos}^p follows a Relaxed Bernoulli distribution.

We can view z_{pos}^p as the distribution of weights for normal pattern p on the patches. Therefore, we can calculate the probability of the normal pattern:

$$p^p = \text{Prob}(\hat{z}_{pos}^p), r^p = \text{Resample}(\hat{z}_{pos}^p), \tag{4}$$

where p^p indicates the probability values of pattern p distributed across the patch, interpreted as pattern weights. r^p facilitates reparameterized sampling in variational inference.

Shape Distribution Learning. Learning the distribution of normal patterns requires latent space to control the shape of normal patterns. The normal gathering latent space needs to learn the distribution of shapes exhibited by normal patterns on each patch. The posterior distribution of the normal pattern p on the time series is denoted as $Q_{pattern}(p|x)$. Following the VAE approach, we compute the mean and variance of the distribution of pattern p :

$$\mu_{shape}^p = \Phi_{\mu}^p(x), \sigma_{shape}^p = \Phi_{\sigma}^p(x), \quad (5)$$

where Φ_{μ}^p and Φ_{σ}^p represent the neural network formed by the linear layer and activation function. Combining the weight coefficients p^p learned in the Position Distribution Learning, we can obtain the $\mu_{weighted}$ and $\sigma_{weighted}$ for each patch, where $\mu_{weighted} = p^p \times \mu_{shape}^p$ and $\sigma_{weighted} = p^p \times \sigma_{shape}^p$. We sample ϵ from the distribution of standard pattern shapes and then transform it to obtain a sample z in the latent space:

$$\hat{z}_{shape}^p = \mu_{weighted} + \sigma_{weighted} \odot \epsilon, \quad (6)$$

where $\epsilon \sim N(0, 1)$. The weighted result $w\text{-}\hat{z}_{shape}^p$ is obtained by multiplying r^p with \hat{z}_{shape}^p .

Commonality Feature Learning. The Commonality Feature Module is used to learn the common characteristics of normal patterns across various patches. Common characteristics enable the normal gathering latent space to capture the overall pattern of the normal patterns, which is crucial for understanding the global behavior and long-term dependencies of the sequence.

We concatenate $\{x_i\}$ along the feature dimension and utilize a linear layer to learn a shared normal pattern latent state z_{com}^p with dimensions $n_p \times d_c$, where d_c is the channel demison of z_{com}^p . The obtained common latent state is duplicated n times and concatenated along the channel dimension of $w\text{-}\hat{z}_{shape}^p$. Finally, a convolutional neural network is used to integrate the commonalities and characteristics of normal patterns contained in each patch:

$$z^p = \text{CNN}\left(\text{Concat}(w\text{-}\hat{z}_{shape}^p, z_{com}^p)\right). \quad (7)$$

After completing the learning for each normal pattern, latent space needs to process the distribution of each normal pattern synchronously. We obtain the distribution of all normal patterns on the time series, denoted as $z = [z^{p_0}, z^{p_1}, \dots, z^{p_n}]$. After concatenating the learned latent states for different normal patterns along the channel dimension, we use a Multi-Layer Perceptron (MLP) for fusion. In the normal gathering latent space, the representation of the time series hidden states is as follows:

$$z = \text{MLP}\left(\text{Concat}([z^{p_i}])\right). \quad (8)$$

4.4 Diffusion Model

We use the Differentiable Diffusion Implicit Models (DDIM) [40] to perform diffusion operations on the normal pattern hidden states. DDIM is a variant of the diffusion model that has a more faithful reconstruction error through inverse sampling. DDIM is primarily composed of a forward diffusion process and a denoising process.

During the training phase, the diffusion incrementally process adds Gaussian noise to this latent state z over a series of steps $\tau = 1, 2, \dots, T$:

$$z_\tau = \sqrt{\alpha_\tau} z_{\tau-1} + \sqrt{1 - \alpha_\tau} \epsilon_\tau, \tag{9}$$

where z_τ is the state of z at step τ , ϵ_τ is a sample from a standard Gaussian distribution, and α_τ are coefficients that control the amount of noise added at each step.

The sampling phase, which is learned by the model, aims to reconstruct the original latent state z_0 from the noisy image z_T . This involves learning a parameterized function ϵ_θ that predicts the noise added at each step. The reverse process can be described as:

$$z_{\tau-1} = \sqrt{\alpha_{\tau-1}} \left(\frac{z_\tau - \sqrt{1 - \alpha_\tau} \epsilon_\theta(z_\tau, \tau)}{\sqrt{\alpha_\tau}} \right) + \sqrt{1 - \alpha_{\tau-1} - \sigma_\tau^2 \epsilon_\theta(z_\tau, \tau)} + \sigma_\tau \epsilon_\tau. \tag{10}$$

DDIM can be considered as the Euler integration of a specific ordinary differential equation. Referring to the DDIM reconstruction methodology, in the sampling stage, we initially use inverse sampling to derive z_τ from z_0 , and then progressively sample back to obtain the reconstructed \tilde{z}_0 . Leveraging DDIM’s low reconstruction error, we can more accurately reconstruct the normal patterns of the time series.

The output of the diffusion model is then mapped back to the numerical space through the Decoder:

$$\hat{X} = \text{Decoder}(\tilde{z}) \tag{11}$$

4.5 Objective Function

The normal gathering latent space module is trained by maximizing the Evidence Lower Bound (ELBO). In this module, we model the distribution of normal patterns through two distinct approaches: the position distribution follows a Relaxed Bernoulli distribution, while the shape distribution adheres to a normal distribution. Thus, the ELBO is given by:

$$\mathcal{L}_{\text{NGLS}} = -\mathbb{E}_{z_p \sim Q^{P,S}(z|x)} [G(x|z_p)] + \mathbb{KL} \left[Q^{P,S}(z|x) \parallel p(z|x) \right], \tag{12}$$

where $G(x|z_p)$ represents the conditional distribution of x given the latent states z_p . The KL-divergence term is further decomposed based on the distribution types assumed for position and shape:

$$\begin{aligned} \mathbb{KL}\left[Q^{P,S}(z|x)||p(z|x)\right] = & \mathbb{KL}\left[Q^P(z_{\text{pos}}|x)||\text{RelBern}(z_{\text{pos}})\right] \\ & + \mathbb{KL}\left[Q^S(z_{\text{shape}}|x)||\mathcal{N}(0,1)\right], \end{aligned} \quad (13)$$

where $Q^P(z_{\text{pos}}|x)$ and $Q^S(z_{\text{shape}}|x)$ are the posterior distributions of position and shape for normal patterns, respectively.

The Diffusion model’s training is defined by the following loss function:

$$\mathcal{L}_{\text{Diffusion}} = \mathbb{E}_{t,z_0,\epsilon} \left[\|\epsilon - \epsilon_{\theta}(z_t, t)\|^2 \right], \quad (14)$$

where θ denotes the model parameters, z_0 represents the initial latent states generated by normal gathering latent space, ϵ is a noise vector sampled from the standard normal distribution, t indicates the time step and z_t is the noisy latent state at time t .

The comprehensive loss function, integrating both the normal gathering latent space and diffusion model components, is formulated as:

$$\mathcal{L} = \mathcal{L}_{\text{NGLS}} + \mathcal{L}_{\text{Diffusion}}. \quad (15)$$

This loss function enables the simultaneous training and optimization of both model components within the NGLS-Diff framework.

Anomaly Score. We use mean-square error to calculate the anomaly score: $AnomalyScore = \text{MSE}(X, \hat{X})$. Subsequently, we label the time points with the top-k highest scores as anomalies, setting the corresponding label $y = 1$.

5 Experiments

5.1 Experimental Setup

To verify the performance of the proposed method, we evaluate it on four real-world datasets, comparing it against fourteen baseline methods.

Datasets. The used datasets are presented as follows:

- **SWaT** [27]: Secure water treatment dataset, a 51-dimensional sensor-based dataset, originates from critical infrastructure systems.
- **SMAP** [16]: Soil moisture active passive satellite dataset, sourced from NASA, presents soil samples and telemetry information.
- **MSL** [16]: Mars science laboratory, provided by NASA, captures sensor and actuator data from the Mars rover.
- **PSM** [1]: Pooled server metrics dataset, from eBay Server Machines.

Baselines. We perform extensive comparisons between our model and a diverse set of fourteen baseline methods, categorized as follows:

- **Classic Methods:** OCSVM [42], Isolation Forest (I.F) [24].
- **Autoregression-Based Model:** LSTM [16].
- **Density-Estimation Models:** MMPCACD [49], LOF [5], DAGMM [53], THOC [39].
- **Clustering-Based Method:** Deep-SVDD [38].
- **Reconstruction-Based Models:** BeatGAN [52], LSTM-VAE [32], Omni-Anomaly [41], Interfusion [21], Anomaly Transformer (A.T) [48], DCdetector [50].

Implementation Details. To ensure a fair comparison, we follow the widely-adopted non-overlapped sliding windows protocol [39] and fix the sliding window size to 100 for all datasets [48]. The input to latent space is the time series latent states processed by a proposed encoder constructed using 3 linear layers, with the number of latent states as 32. The dimension of the shared latent state is defined as 10. For the diffusion model, the schedule is set to ‘linear’ mode with a time step of 100. We employ the Adam optimizer with an initial learning rate of 0.01. All experiments are conducted on a single NVIDIA 4090 GPU in PyTorch.

Evaluation Criteria. We use adjusted evaluation methods widely recognized in recent studies, notably the point adjustment method [39, 41, 46]. This approach considers all timestamps in an anomalous segment as correctly detected if any point within it is identified as an anomaly, better reflecting the model’s ability to detect contextual and collective anomalies. This adjusted evaluation is crucial for real-world applications, where anomalies are not isolated but occur over periods or in patterns. It allows for a more realistic assessment of a model’s performance in handling the intricacies of time series data, focusing on the recognition and understanding of anomaly dynamics.

Following the evaluation method of DCdetector [50], we employ additional evaluation metrics beyond traditional F1-score for evaluating time series anomaly detection: the affiliation precision/recall pair [15] and Volume Under the Surface (VUS) [31]. The affiliation metrics improve traditional precision and recall by focusing on the relationship between true events and predictions, enhancing clarity and resistance to manipulation. Aff-P and Aff-R denote affiliation-based precision and recall metrics, respectively. On the other hand, VUS refines AUC measures to assess time series anomalies more effectively, being independent of any threshold. This includes Range-AUC-ROC and Range-AUC-PR (RAR and RAP), alongside the volumes under the ROC and PR curves (VROC and VRR), offering a more comprehensive evaluation framework.

5.2 Main Results

We conducted a comparative analysis of our model, NGLS-Diff, against 14 models using four real-world datasets. The results are reported in Table 1, where a

Table 1. Quantitative evaluation results on multiple datasets. The best results are highlighted in bold, and the second-best results are underlined.

Dataset	SWaT			MSL			SMAP			PSM		
Method/Metric	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
OCSVM [42]	0.4528	0.5035	0.4768	0.5924	0.8649	0.7031	0.5284	0.6129	0.5675	0.6315	0.7828	0.6990
IF [24]	0.4871	0.4423	0.4636	0.5562	0.8497	0.6723	0.5633	0.5612	0.5622	0.7931	0.9220	0.8527
LSTM [16]	0.8476	0.8522	0.8498	0.8459	0.8570	0.8514	0.8740	0.8132	0.8425	0.7649	0.9019	0.8277
MMPACD [49]	0.8325	0.6937	0.7567	0.8361	0.6323	0.7200	0.8527	0.7956	0.8231	0.7724	0.7812	0.7767
LOF [5]	0.7141	0.6723	0.6925	0.4920	0.8471	0.6224	0.5783	0.5746	0.5764	0.5764	0.8935	0.7007
DAGMM [53]	0.9041	0.5694	0.6987	0.8856	0.6475	0.7480	0.8752	0.5748	0.6938	0.9311	0.7157	0.8093
THOC [39]	0.8274	0.8590	0.8429	0.8949	0.9124	0.9035	0.9237	0.9022	0.9128	0.8855	0.9293	0.9068
Deep-SVDD [38]	0.8063	0.8641	0.8341	0.9052	0.7941	0.8460	0.8837	0.5849	0.7039	0.9501	0.8789	0.9131
BeatGAN [52]	0.6592	0.8627	0.7473	0.9083	0.8524	0.8794	0.9013	0.5825	0.7076	0.9011	0.9247	0.9127
LSTM-VAE [32]	0.7740	0.9122	0.8374	0.8211	0.8107	0.8158	0.9175	0.7032	0.7961	0.7629	0.9148	0.8319
OmniAnomaly [41]	0.8256	0.8643	0.8445	0.9130	0.8769	0.8945	0.9356	0.8421	0.8863	0.9002	0.7832	0.8376
Interfusion [21]	0.8199	0.8578	0.8384	0.8030	0.9317	0.8625	0.8767	0.9027	0.8895	0.8199	0.9512	0.8806
A.T [48]	0.9410	0.9797	0.9599	0.9198	0.9629	<u>0.9409</u>	0.9362	0.9947	0.9646	0.9734	0.9770	0.9752
DCdetector [50]	0.9309	0.9996	<u>0.9641</u>	0.9202	0.9466	0.9332	0.9436	0.9888	<u>0.9657</u>	0.9710	0.9809	<u>0.9759</u>
NGLS-Diff(Ours)	0.9752	0.9665	0.9708	0.9081	0.9957	0.9499	0.9530	0.9842	0.9684	0.9698	0.9912	0.9804

higher F1 score indicates better overall performance. As expected, our proposed NGLS-Diff outperforms all baselines by considerable margins, demonstrating its effectiveness in time series anomaly detection. Classic methods obtain relatively low scores, suggesting that classic methods are not effective for complex anomaly detection tasks. There is a large variance in performance among density-estimation models. Generally, the reconstruction-based methods perform the best. Moreover, the Anomaly Transformer and DCdetector approaches yield remarkable detection performance. Overall, NGLS-Diff consistently achieves the highest performance, underscoring its superior anomaly detection capabilities.

To verify the robustness of our model under different metrics, we compared our model with two strong baselines (Anomaly Transformer and DCdetector) using the new metrics [50]. The comparison results are shown in Table 2. As can be seen, NGLS-Diff generally achieves the best performance scores of the six measure metrics, which comprehensively validates its effectiveness.

5.3 Results of Normal Gathering Latent Space

In this section, we investigate the efficacy of the normal gathering latent space in NGLS-Diff. We set up three comparison schemes: the conventional DDIM model, the AE-DDIM model, and the NGLS model. The conventional DDIM model reconstructs time series directly in the original numerical space, while AE-DDIM establishes a latent space using an autoencoder without normal gathering. NGLS incorporates normal gathering latent space and directly uses a decoder to map potential states from latent space back to the numerical space, excluding the DDIM diffusion process.

The experimental results are presented in Table 3. As observed, NGLS-Diff achieved a 20.57% average improvement compared to DDIM, indicating that

Table 2. Results across multiple metrics. The best results are highlighted in bold, and the second-best results are underlined.

Dataset	Method	Aff-P	Aff-R	RAR	RAP	VROC	VPR
SWaT	A.T	0.6401	0.9390	0.9443	0.9257	0.9451	0.9263
	DCdetector	0.5248	<u>0.9804</u>	<u>0.9672</u>	<u>0.9413</u>	<u>0.9708</u>	0.9444
	NGLS-Diff	<u>0.6064</u>	0.9839	0.9801	0.9442	0.9756	<u>0.9402</u>
MSL	A.T	<u>0.5241</u>	0.9593	<u>0.9016</u>	0.8801	<u>0.8862</u>	<u>0.8667</u>
	DCdetector	0.5095	<u>0.9671</u>	0.9015	<u>0.8803</u>	0.8788	0.8604
	NGLS-Diff	0.5605	0.9761	0.9154	0.8823	0.9158	0.8828
SMAP	A.T	<u>0.5122</u>	<u>0.9891</u>	<u>0.9642</u>	<u>0.9451</u>	<u>0.9586</u>	<u>0.9369</u>
	DCdetector	0.5149	0.9852	0.9590	0.9404	0.9386	0.9227
	NGLS-Diff	0.5157	0.9897	0.9681	0.9452	0.9588	0.9372
PSM	A.T	<u>0.5584</u>	<u>0.8196</u>	<u>0.9162</u>	<u>0.9327</u>	<u>0.8944</u>	<u>0.9169</u>
	DCdetector	0.5414	0.8003	0.9094	0.9246	0.8651	0.8915
	NGLS-Diff	0.5784	0.8702	0.9404	0.9485	0.9055	0.9236

Table 3. Efficacy of latent space. The best results are highlighted in bold, and the second-best results are indicated with an underline.

Dataset	SWaT			MSL			SMAP			PSM		
Method/Metric	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
DDIM	0.7006	0.9600	0.8100	0.7836	0.5049	0.6141	0.8675	0.5628	0.6826	0.9690	0.9127	0.9400
AE-DDIM	0.9477	0.8274	<u>0.8834</u>	0.8894	0.9121	<u>0.9006</u>	0.9016	0.9351	<u>0.9180</u>	0.9718	0.9298	<u>0.9503</u>
NGLS	0.9572	0.8078	0.8761	0.8764	0.8827	0.8795	0.8601	0.8057	0.8320	0.9897	0.9072	0.9466
NGLS-Diff	0.9752	0.9665	0.9708	0.9081	0.9957	0.9499	0.9530	0.9842	0.9684	0.9698	0.9912	0.9804

DDIM’s performance in the latent space is superior to the numerical space for anomaly detection. NGLS-Diff demonstrated a 5.43% average improvement over AE-DDIM, providing evidence that NGLS-Diff creates a latent space suitable for time series anomaly detection tasks within the framework of DDIM.

5.4 Ablation Study

To verify the effectiveness of each module in NGLS-Diff, we conduct ablation studies. We further evaluate NGLS-Diff with its variants by removing each part in NGLS-Diff: (1)**w/o SD**: Removes the shape distribution learning module. (2)**w/o PD**: Removes the position distribution learning module. (3)**w/o CF**: Removes the commonality feature learning module. The results are summarized in Table 4.

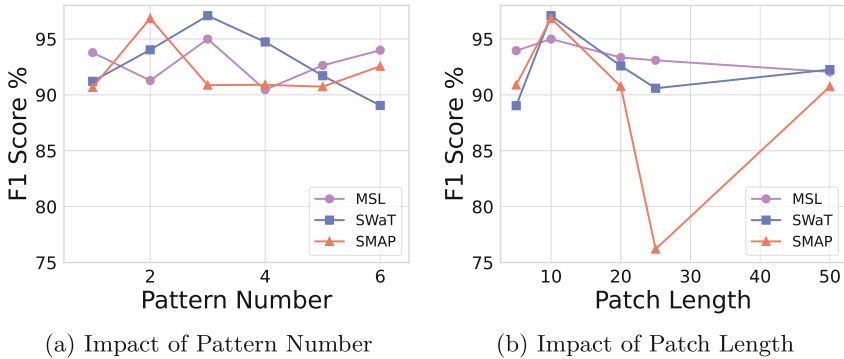
As expected, each module in the latent space construction process contributed to NGLS-Diff’s effectiveness. The shape learning module had the most significant impact, highlighting the importance of capturing the shape characteristics of normal patterns within the latent space. Conversely, the position

Table 4. Ablation results (F1-score) of NGLS-Diff on benchmark datasets.

Dataset	SWaT	MSL	SMAP	PSM
w/o SD	0.9012	0.9186	0.8153	0.9266
w/o PD	0.9425	0.9371	0.8558	0.9674
w/o CF	0.9372	0.9226	0.8046	0.9521
NGLS-Diff	0.9708	0.9499	0.9684	0.9804

distribution module had a smaller influence, suggesting that the intensity of normal patterns across different patches is relatively consistent over time. The commonality feature module also provided a notable improvement in the model’s performance, indicating that there are evident commonalities and characteristics among the same type of normal patterns across different patches. These findings confirm the effectiveness and essential nature of each module in NGLS-Diff.

5.5 Analysis and Discussion

**Fig. 2.** Sensitivity Analysis of Key Hyperparameters in NGLS-Diff.

Parameter Sensitivity Analysis. This section primarily focuses on the study of NGLS-Diff’s sensitivity to two hyperparameters: pattern number and patch length. Figure 2 shows the effects of varying the pattern number and patch length on the F1 score. Pattern number denotes the number of normal pattern types NGLS-Diff is expected to identify. The pattern number affects the dimensions of some vectors in NGLS-Diff’s construction. From Fig. 2(a), we can observe that there exists an optimal value for the pattern number, and this optimal value would vary for different datasets. A pattern number set to 1, implying no differentiation among normal patterns, results in reduced NGLS-Diff efficiency.

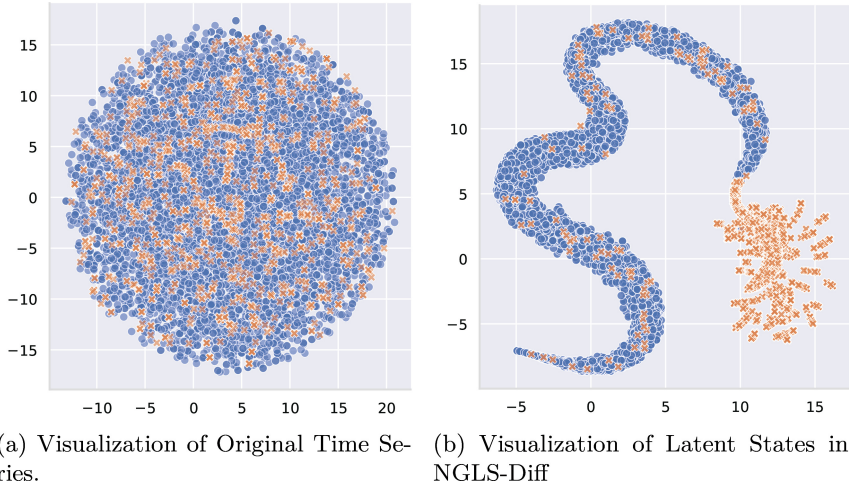


Fig. 3. Visualization of Multivariate Time Series After UMAP. Orange Points are anomaly time points. Blue Points are normal time points. (Color figure online)

Patch length is a parameter used to slice the initial latent states of time series in the latent space and determines the length and the number of time series patches. Different patch lengths affect the amount of contextual information included in the patches. Based on Fig. 2(b), for most datasets, a patch length of 10 results in the optimal F1 score.

Visualization. To further investigate the mechanism of latent space, we visualize the input data and the latent states (i.e., the output of normal gathering latent space) by UMAP [28] dimensionality reduction. Figure 3(a) illustrates the reduced two-dimensionality representation of the original multivariate time series. In this representation, normal and anomalous time points are mixed, making them indistinguishable in the original numerical space. In Fig. 3(b), the reduced dimensionality representation of the time series in the normal gathering latent space constructed by latent space shows a clear boundary between normal and anomalous time points. This observation confirms that the NGLS effectively discerns the distribution of normal patterns and characteristics hidden within. These insights remain obscured in the original data space but are distinct in the latent space, enabling effective differentiation between normal and abnormal patterns. Such clear numerical distinctions support the diffusion model in more accurately reconstructing time series using normal patterns, thus bolstering its anomaly detection capabilities.

6 Conclusion

For time series anomaly detection, existing diffusion-based methods reconstruct time series in the original numerical space, which can be problematic due to insufficient temporal semantics and outlier interference. To enhance the generative capability of the diffusion model, we propose NGLS-Diff, which uses a normal gathering latent space for normal pattern learning. By fusing positional distribution, shape distribution, and common characteristics of normal temporal patterns, NGLS-Diff gathers normal patterns, avoids anomalous influences, and enhances time series reconstruction. Extensive experiments conducted on multiple real-world datasets demonstrate the performance improvement of this proposed solution compared to various strong baselines.

Acknowledgments. This study was supported in part by NSFC under Grants 62376072, 62272130, and in part by Shenzhen Science and Technology Program Nos. JCYJ20210324120208022 and KCXFZ20211020163403005.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Abdulaal, A., Liu, Z., Lancewicki, T.: Practical approach to asynchronous multivariate time series anomaly detection and localization. In: SIGKDD, pp. 2485–2494 (2021)
2. Audibert, J., Michiardi, P., Guyard, F., Marti, S., Zuluaga, M.A.: USAD: unsupervised anomaly detection on multivariate time series. In: SIGKDD, pp. 3395–3404 (2020)
3. Bashar, M.A., Nayak, R.: TAnoGAN: time series anomaly detection with generative adversarial networks. In: SSCI, pp. 1778–1785 (2020)
4. Bhumika, Das, D.: Deep learning based urban anomaly prediction from spatiotemporal data. In: ECML PKDD, pp. 242–257 (2022)
5. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. In: SIGMOD, pp. 93–104 (2000)
6. Cazzonelli, L., Kulbach, C.: Detecting anomalies with autoencoders on data streams. In: ECML PKDD, pp. 258–274 (2022)
7. Deng, A., Hooi, B.: Graph neural network-based anomaly detection in multivariate time series. In: AAAI, pp. 4027–4035 (2021)
8. Fournier-Viger, P., He, G., Zhou, M., Nouioua, M., Liu, J.: Discovering alarm correlation rules for network fault management. In: International Conference on Service-Oriented Computing, pp. 228–239. Springer (2020). https://doi.org/10.1007/978-3-030-76352-7_24
9. Gardner Jr, E.S.: Exponential smoothing: the state of the art. *J. forecast.* **4**(1), 1–28 (1985)
10. Geiger, A., Liu, D., Alnegheimish, S., Cuesta-Infante, A., Veeramachaneni, K.: TadGAN: time series anomaly detection using generative adversarial networks. In: big data, pp. 33–43 (2020)

11. Golmohammadi, K., Zaiane, O.R.: Time series contextual anomaly detection for detecting market manipulation in stock market. In: DSAA, pp. 1–10 (2015)
12. Gupta, K., Singh, S., Shrivastava, A.: PatchVAE: learning local latent codes for recognition. In: CVPR, pp. 4746–4755 (2020)
13. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: NeurIPS, pp. 6840–6851 (2020)
14. Hu, T., Zhang, J., Yi, R., Du, Y., Chen, X., Liu, L., Wang, Y., Wang, C.: Anomaly-Diffusion: few-shot anomaly image generation with diffusion model. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol.38, pp. 8526–8534 (2024)
15. Huet, A., Navarro, J.M., Rossi, D.: Local evaluation of time series anomaly detection algorithms. In: SIGKDD, pp. 635–645 (2022)
16. Hundman, K., Constantinou, V., Laporte, C., Colwell, I., Soderstrom, T.: Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding. In: SIGKDD, pp. 387–395 (2018)
17. Jhin, S.Y., Lee, J., Park, N.: Precursor-of-anomaly detection for irregular time series. In: SIGKDD, pp. 917–929 (2023)
18. Li, B., Zhou, M., Zhang, S., Yang, M., Lian, D., Huang, Z.: BSAL: a framework of Bi-component structure and attribute learning for link prediction. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2053–2058 (2022)
19. Li, T., Comer, M.L., Delp, E.J., Desai, S.R., Mathieson, J.L., Foster, R.H., Chan, M.W.: Anomaly scoring for prediction-based anomaly detection in time series. In: IEEE Aerospace Conference, pp. 1–7 (2020)
20. Li, Y., Lu, X., Wang, Y., Dou, D.: Generative time series forecasting with diffusion, denoise, and disentanglement. *Adv. Neural Inf. Proc. Syst.* **35**, 23009–23022 (2022)
21. Li, Z., Zhao, Y., Han, J., Su, Y., Jiao, R., Wen, X., Pei, D.: Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding. In: SIGKDD, pp. 3220–3230 (2021)
22. Lin, S., Clark, R., Birke, R., Schönborn, S., Trigoni, N., Roberts, S.: Anomaly detection for time series using VAE-LSTM hybrid model. In: ICASSP, pp. 4322–4326 (2020)
23. Liu, D., et al.: Opprentice: Towards practical and automatic anomaly detection through machine learning. In: IMC, pp. 211–224 (2015)
24. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation forest. In: ICDM, pp. 413–422 (2008)
25. Livernoche, V., Jain, V., Hezaveh, Y., Ravanbakhsh, S.: On diffusion modeling for anomaly detection. In: ICLR (2024)
26. Malhotra, P., et al.: Long short term memory networks for anomaly detection in time series. In: Esann, pp. 89 (2015)
27. Mathur, A.P., Tippenhauer, N.O.: SWat: a water treatment testbed for research and training on ICS security. In: CySWater Workshop, pp. 31–36 (2016)
28. McInnes, L., Healy, J., Melville, J.: Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint* (2018)
29. Nelson, B.K.: Time series analysis using autoregressive integrated moving average (ARIMA) models. *Acad. Emerg. Med.* **5**(7), 739–744 (1998)
30. Nguyen, N., Quanz, B.: Temporal latent auto-encoder: a method for probabilistic multivariate time series forecasting. In: AAAI, pp. 9117–9125 (2021)
31. Paparrizos, J., Boniol, P., Palpanas, T., Tsay, R.S., Elmore, A., Franklin, M.J.: Volume under the surface: a new accuracy evaluation measure for time-series anomaly detection. *Proc. VLDB Endowment* **15**(11), 2774–2787 (2022)

32. Park, D., Hoshi, Y., Kemp, C.C.: A multimodal anomaly detector for robot-assisted feeding using an LSTM-based variational autoencoder. *IEEE Rob. Autom. Lett.* **3**(3), 1544–1551 (2018)
33. Preechakul, K., Chatthee, N., Wizadwongsa, S., Suwajanakorn, S.: Diffusion autoencoders: toward a meaningful and decodable representation. In: *CVPR* (2022)
34. Pukelsheim, F.: The three sigma rule. *Am. Stat.* **48**(2), 88–91 (1994)
35. Rasul, K., Seward, C., Schuster, I., Vollgraf, R.: Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In: *ICML*, pp. 8857–8868 (2021)
36. Ren, H., et al.: Time-series anomaly detection service at Microsoft. In: *SIGKDD*, pp. 3009–3017 (2019)
37. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: *CVPR*, pp. 10684–10695 (2022)
38. Ruff, L., et al.: Deep one-class classification. In: *ICML*, pp. 4393–4402 (2018)
39. Shen, L., Li, Z., Kwok, J.: Timeseries anomaly detection using temporal hierarchical one-class network. *Adv. Neural Inf. Proc. Syst.* **33**, 13016–13026 (2020)
40. Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. In: *ICLR* (2020)
41. Su, Y., Zhao, Y., Niu, C., Liu, R., Sun, W., Pei, D.: Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In: *SIGKDD*, pp. 2828–2837 (2019)
42. Tax, D.M., Duin, R.P.: Support vector data description. *Mach. Learn.* **54**, 45–66 (2004)
43. Wang, C., et al.: Drift doesn’t matter: Dynamic decomposition with diffusion reconstruction for unstable multivariate time series anomaly detection. In: *NeurIPS* (2023)
44. Wang, Y., Wong, J., Miner, A.: Anomaly intrusion detection using one class SVM. In: *SMC Information Assurance Workshop*, pp. 358–364 (2004)
45. Xiao, C., Gou, Z., Tai, W., Zhang, K., Zhou, F.: Imputation-based time-series anomaly detection with conditional weight-incremental diffusion models. In: *SIGKDD*, pp. 2742–2751 (2023)
46. Xu, H., et al.: Unsupervised anomaly detection via variational auto-encoder for seasonal KPIs in web applications. In: *WWW*, pp. 187–196 (2018)
47. Xu, J., et al.: Multi-VAE: learning disentangled view-common and view-peculiar visual representations for multi-view clustering. In: *CVPR*, pp. 9234–9243 (2021)
48. Xu, J., Wu, H., Wang, J., Long, M.: Anomaly transformer: time series anomaly detection with association discrepancy. In: *ICLR* (2022)
49. Yairi, T., Takeishi, N., Oda, T., Nakajima, Y., Nishimura, N., Takata, N.: A data-driven health monitoring method for satellite housekeeping data based on probabilistic clustering and dimensionality reduction. *IEEE Trans. Aerosp. Electron. Syst.* **53**(3), 1384–1401 (2017)
50. Yang, Y., Zhang, C., Zhou, T., Wen, Q., Sun, L.: Dcdetector: dual attention contrastive representation learning for time series anomaly detection. In: *SIGKDD*, pp. 3033–3045 (2023)
51. Yao, Y., Ma, J., Feng, S., Ye, Y.: SVD-AE: an asymmetric autoencoder with SVD regularization for multivariate time series anomaly detection. *Neural Netw.* **170**, 535–547 (2024)
52. Zhou, B., Liu, S., Hooi, B., Cheng, X., Ye, J.: BeatGAN: anomalous rhythm detection using adversarially generated time series. In: *IJCAI*, pp. 4433–4439 (2019)
53. Zong, B., et al.: Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: *ICLR* (2018)



Permutation Dependent Feature Mixing for Multivariate Time Series Forecasting

Rikuto Yamazono¹(✉) and Hirotake Hachiya^{1,2}

¹ Wakayama University, 930, Sakaedani, Wakayama-city, Wakayama 640-8510, Japan
s256279@wakayama-u.ac.jp

² Center for AIP, RIKEN, Nihonbashi 1-chome Mitsui Building, 15th floor, 1-4-1,
Nihonbashi, Chuo-ku, Tokyo 103-0027, Japan

Abstract. Multivariate time series forecasting is critical in finance and meteorology, influencing decision-making. Though effective in capturing long-range dependencies in natural language processing, traditional Transformer models face challenges when applied to time series data, including computational inefficiency and the loss of positional encoding effects. Time-Series Mixer (TSMixer) addresses these issues by efficiently blending the temporal and feature dimensions in multivariate time series data, thereby facilitating sequential dependent feature extraction. However, the current feature mixing in TSMixer applies a common multi-layer perception across all time steps, leading to time-invariant, non-adaptive feature exchange that does not allow for accurate extraction of historical information. Therefore, we propose incorporating adaptive frequency components and event proximity as additional information vectors into the Feature Mixing component of TSMixer to improve its capacity to interpret complex feature interrelations. Our research validates the effectiveness of these enhancements through experiments with various real-world multivariate time series datasets, including weather and traffic data, emphasizing its potential across different scenarios. Codes are available at <https://github.com/rikuter67/FAM-EPAM>.

Keywords: Multivariate Time Series · TSMixer · Time Series Forecasting

1 Introduction

Time series forecasting is indispensable across various sectors, shaping crucial decision-making processes. Traditional methods, such as ARIMA [4, 5], rely on predefined models to capture trends and cycles of historical series. While they are effective for stationary series, their fixed structure and inability to capture the dynamic dependencies among multiple features would result in poor performance with real-world data.

Deep neural networks with recurrent architectures, i.e., RNNs [12, 15], and memory cells, i.e., LSTMs [7, 10], enable to capture dynamic and temporal dependencies by encoding significant information into latent vectors extracted from

historical series. However, RNNs with a one-step recurrent connection face limitations on long-term dependencies due to gradient vanishing and exploding issues. While LSTMs with gates and cells can capture longer-term dependencies, their effectiveness is still constrained due to the finite capacity of the memory cells and the complexity of their computational processes.

Transformers [11, 14, 16, 18] introduce an attention mechanism, e.g., self-attention, which allows the model to directly encode important information into the sequence vectors themselves based on the relationship, e.g., co-occurrence within the sequence. This mechanism enables models to process sequences in parallel and provides an enhanced capacity to capture complex and long-term dependencies. However, the inherent permutation invariance of the attention mechanisms poses significant challenges for processing time series data despite the success in natural language processing, which employs a heuristic extension, called positional encoding, to add information about the order of time steps.

In light of this, recent studies have suggested the potential of linear models, which are inherently sensitive to the order of the inputs, for sequential data [17]. Among linear models, the TSMixer (Time-Series Mixer) [6] model employs repeated MLPs (Multi-layer perceptions) to mix time and feature information alternately, encoding useful information into sequence vectors from complex multivariate time series data. However, TSMixer’s feature mixing approach uses a common MLP across all time steps, leading to time-invariant, non-adaptive feature mixing, hindering the accurate extraction of historical information.

To address this issue, we propose enhancements to the Feature Mixing component of TSMixer. Firstly, we propose a Frequency-Aware Mixer (FAM), which adds an adaptive frequency component of each feature to the feature-mixing, enabling the adjustment of the strength of feature mixing based on the adaptive time cycles. Secondly, we propose an Event Proximity-Aware Mixer (EPAM), which adds the proximity to the principal observation (event) as an additional component of the feature mixing, enabling the strength of the feature mixing to be adjusted based on the relation to the representative events. These enhancements will enable the model to more accurately grasp the complex interrelations among features.

The main contributions of this paper are summarized as follows:

1. We propose to enhance the Feature Mixing component of TSMixer, the state-of-the-art multivariate time-series forecasting method, to allow for time-dependent and adaptive mixing by introducing principal frequency components, called Frequency-Aware Mixer (FAM) and the distance to principal time-step, called Event Proximity-Aware Mixer (EPAM) as additional information vectors.
2. We demonstrate the effectiveness of the proposed method over existing state-of-the-art multivariate time-series forecasting methods through extensive comparative experiments on various real-world datasets.

After this introductory section, the rest of this paper is organized as follows. Section 2 describes the formulation and reviews related works. Section 3 details

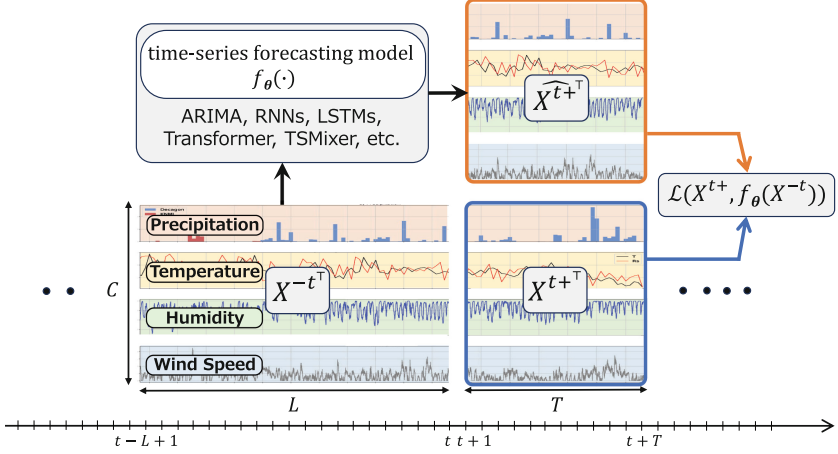


Fig. 1. Illustration of the time series forecasting process based on the formulation using weather-related multivariate data. $f_{\theta}(\cdot)$ transforms L -step of observations X^{-t} to generate T -step future observations \widehat{X}^{t+} , which is then evaluated against the ground truth X^{t+} using the loss function $\mathcal{L}(\cdot, \cdot)$ to compute the discrepancy. Multivariate data are adapted from [8].

the proposed method. Section 4 describes the experimental evaluation and discussion; Sect. 6 presents the conclusion.

2 Formulation and Related Works

This section formulates the problem of multivariate time series forecasting and reviews its related works.

2.1 Formulation

Let X_{tc} denote the c -th observation at time t and $:$ denote all elements at the corresponding axis. Let $X^{-t} \in \mathbb{R}^{L \times C}$ be L -step history of observations where $X_{t:}^{-t} \in \mathbb{R}^{1 \times C}$ be a vector of C observations at time step t and $X_{:c}^{-t} \in \mathbb{R}^{L \times 1}$ be a vector of the c -th observation over L -step. Meanwhile, let $X^{t+} \in \mathbb{R}^{T \times C}$ be T -step future observations starting from the next step of X^{-t} as follows:

$$\begin{aligned} X^{-t} &= [X_{t-L+1:}, \dots, X_{t-1:}, X_{t:}], \\ X^{t+} &= [X_{t+1:}, X_{t+2:}, \dots, X_{t+T:}]. \end{aligned} \quad (1)$$

The task of multivariate time-series forecasting is to obtain a model $f(\cdot)$ to predict future observations X^{t+} given its history X^{-t} as follows:

$$\widehat{X}^{t+} = f_{\theta}(X^{-t}), \quad (2)$$

Parameter θ of the model is tuned to minimize the loss function $\mathcal{L}(\cdot)$ averaged over training data \mathcal{D}_{tr} as follows:

$$\min_{\theta} \frac{1}{|\mathcal{D}_{\text{tr}}|} \sum_{t=L}^{|\mathcal{D}_{\text{tr}}|} \mathcal{L}(X^{t+}, f_{\theta}(X^{-t})), \quad (3)$$

where \mathcal{D}_{tr} is defined as follows:

$$\mathcal{D}_{\text{tr}} \equiv \left\{ (X^{-t}, X^{t+}) \right\}_{t=L}^{N_{\text{tr}}-T}, \quad (4)$$

where N_{tr} is the number of steps in the training sequence. Similarly, the validation and test data are defined as follows:

$$\begin{aligned} \mathcal{D}_{\text{val}} &\equiv \left\{ (X^{-t}, X^{t+}) \right\}_{t=N_{\text{tr}}}^{N_{\text{tr}}+N_{\text{val}}-T}, \\ \mathcal{D}_{\text{te}} &\equiv \left\{ (X^{-t}, X^{t+}) \right\}_{t=N_{\text{tr}}+N_{\text{val}}}^{N_{\text{tr}}+N_{\text{val}}+N_{\text{te}}-T}, \end{aligned} \quad (5)$$

where N_{val} and N_{te} are the numbers of steps in the validation and test sequence, respectively—there is no overlap between training, validation, and test sequences. As Fig. 1 illustrates the formulation overview.

2.2 Attention Mechanisms in Time Series Forecasting

Attention mechanisms have been applied to time series forecasting [11, 14, 16, 18], enabling models to dynamically encode important information into the sequence vectors X^{-t} based on the relationships between observation vectors at different time steps. More specifically, in the attention mechanism, affinity weight $W^{\text{att}} \in \mathbb{R}^{L \times L}$ is computed based on the similarity between query vectors $Q \in \mathbb{R}^{L \times C}$ and key vectors $K \in \mathbb{R}^{L \times C}$. Next, query Q is transformed through an interpolation of value vectors $V \in \mathbb{R}^{L \times C}$, as follows:

$$\begin{aligned} W^{\text{att}} &= \text{softmax}\left(\frac{(QW^Q)(KW^K)^{\top}}{\sqrt{C}}\right), \\ Q' &= \text{Attention}(Q, K, V) = W^{\text{att}}(VW^V), \end{aligned} \quad (6)$$

where W^Q , W^K , and $W^V \in \mathbb{R}^{C \times C}$ are trainable linear projection matrices. This weight W^{att} captures the dependencies across the time series, making attention mechanisms particularly useful for identifying intricate temporal relationships.

There are several extensions to overcome the limitation of Transformer for multivariate time-series forecasting, such as Autoformer [16], Informer [18], and PatchPST [11]. Autoformer incorporates an autocorrelation mechanism to capture long-term dependency and a deep decomposition architecture that sequentially decomposes the time series data into trend, seasonal, and random components during forecasting. Informer introduces ProbSparse self-attention, which probabilistically selects important queries with higher attention scores to reduce

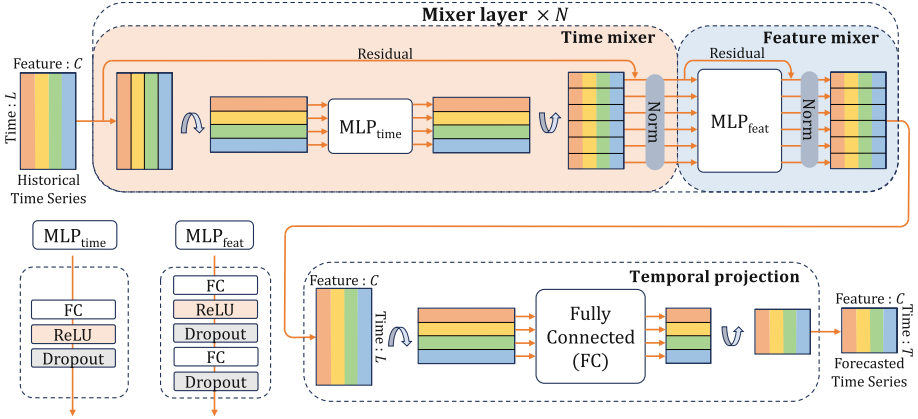


Fig. 2. Architecture of TSMixer, consisting of N repeated mixer layers and a temporal projection. Note that time and feature-mixing MLPs in each mixer layer are shared across all features and all time steps.

computational complexity enabling the efficient capture of long-term dependency. PatchTST divides the time series into patch chunks treated as time steps in the attention mechanism and applies single embedding and attention mechanism individually to each multivariate series, enabling efficient multivariate time-series forecasting.

However, due to the attention mechanisms' inherent permutation invariance property, Transformer models face challenges when directly applied to time series data, where the order of time steps critically impacts forecasting accuracy. Currently, Transformer models attempt to address this issue through positional encoding, which aims to inject sequence information into the model. Yet, there is a concern that the effect of positional encoding may diminish as the attention mechanism is applied repeatedly.

2.3 TSMixer: An All-MLP Architecture for Time Series Forecasting

Recent research [17] has highlighted that simple linear models can be highly effective for time series forecasting, surpassing Transformer-based models, i.e., Autoformer [16], Informer [18] and FEDformer [19]. As illustrated in Fig. 2, TSMixer applies MLPs alternatively in time and feature domains. TSMixer consists of three main components: Time-mixer, Feature-mixer, and Temporal Projection as follows:

Time-Mixer encodes temporal information, e.g., long-term dependencies, into the history X^{-t} by blending across the time direction $X_{:c}^{-t}$ as follows:

$$\begin{aligned} \text{TM}(X_{:c}^{-t}) &= \text{Drop}\left(\sigma\left((X_{:c}^{-t})^\top W_{\text{TM}} + \mathbf{b}_{\text{TM}}\right)\right), \\ X_{:c}^{-t} &\leftarrow \text{Norm}\left(X_{:c}^{-t} + \text{TM}(X_{:c}^{-t})^\top\right), \end{aligned} \quad (7)$$

where $W_{\text{TM}} \in \mathbb{R}^{L \times L}$ and $\mathbf{b}_{\text{TM}} \in \mathbb{R}^{1 \times L}$ are trainable weight and bias, respectively. $\sigma(\cdot)$, $\text{Drop}(\cdot)$, and $\text{Norm}(\cdot)$ represent an activation function, i.e., ReLU, a dropout operation, and a normalization operation, i.e., 2D batch normalization applied over the $L \times C$ plane along the batch dimension, respectively. Note that the same time-mixing MLPs are shared across all types of features.

Feature-Mixer encodes information regarding the relationship among different observations, e.g., co-occurrence of observations, into the history X^{-t} by blending across the feature direction $X_{t:}^{-t}$ as follows:

$$\begin{aligned} U_{t:} &= \text{Drop}\left(\sigma\left(X_{t:}^{-t} W_{\text{FM}_1} + \mathbf{b}_{\text{FM}_1}\right)\right), \quad \text{FM}(X_{t:}^{-t}) = \text{Drop}\left(U_{t:} W_{\text{FM}_2} + \mathbf{b}_{\text{FM}_2}\right), \\ X_{t:}^{-t} &\leftarrow \text{Norm}\left(X_{t:}^{-t} + \text{FM}(X_{t:}^{-t})\right), \end{aligned} \quad (8)$$

where $W_{\text{FM}_1} \in \mathbb{R}^{C \times H}$ and $W_{\text{FM}_2} \in \mathbb{R}^{H \times C}$ are trainable weights, and $\mathbf{b}_{\text{FM}_1} \in \mathbb{R}^{1 \times H}$ and $\mathbf{b}_{\text{FM}_2} \in \mathbb{R}^{1 \times C}$ are trainable biases. $U \in \mathbb{R}^{L \times H}$ represents the hidden variables with the the number H of nodes. Note that the same feature-mixing MLPs are shared across all time steps.

Temporal Projection compresses the L -step history $X_{:c}^{-t}$ to the length of future prediction period, i.e., T -step, using a fully-connected layer as follows:

$$\widehat{X_{:c}^{t+}} = \left((X_{:c}^{-t})^\top W_{\text{TP}} + \mathbf{b}_{\text{TP}}\right)^\top, \quad (9)$$

where X^{-t} represents the output of the Mixer Layer, $W_{\text{TP}} \in \mathbb{R}^{L \times T}$ and $\mathbf{b}_{\text{TP}} \in \mathbb{R}^{1 \times T}$ are trainable weight and bias.

The permutation-sensitive properties of the time-mixing MLPs in TSMixer empower the model to effectively capture the dynamic relationships among observations along the time direction, enhancing the prediction performance for multivariate time series data. On the other hand, a limitation exists in the feature-mixing MLPs where the same MLPs are used across time direction, and thus, the identical transformation is applied to vector $X_{t:}^{-t}$ regardless of position in the sequence. This permutation-invariant feature mixing avoids capturing significant relationships between observations, e.g., co-occurrence of observations related to trend and seasonal cycles, and potentially degrades the prediction performance.

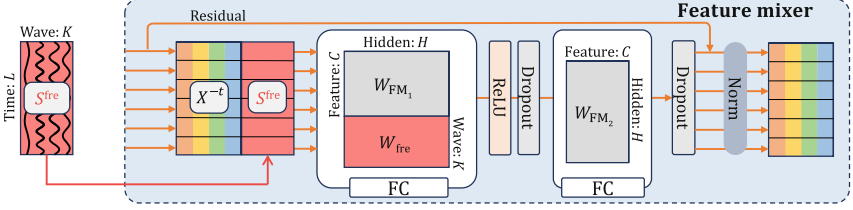


Fig. 3. Architecture of frequency-aware mixer (FAM), a permutation-sensitive extension of Feature Mixing MLPs in TSMixer (in Fig. 2). A matrix $S^{\text{fre}} \in \mathbb{R}^{L \times K}$ contains K different waveforms along the time axis, linearly integrated by weight W_{fre} .

3 Proposed Method

To allow for time-dependent and adaptive feature mixing, we propose to enhance the feature mixing by introducing principal frequency components, called Frequency-Aware Mixer (FAM), and the distance to the principal time step, called Event Proximity-Aware Mixer (EPAM), as additional information vectors.

3.1 Frequency-Aware Mixer (FAM)

We propose Frequency-Aware Mixer (FAM) which incorporates an adaptive frequency component into the first layer of feature-mixing MLPs (in Eq. 8 and Fig. 2) as depicted in Fig. 4 and as follows:

$$U_t = \text{Drop} \left(\sigma \left(X_{t:}^{-t} W_{\text{FM}_1} + \underline{S}_{t:}^{\text{fre}} W_{\text{fre}} + \mathbf{b}_{\text{FM}_1} \right) \right),$$

$$S_{tk}^{\text{fre}} = a_k \cos \left(\frac{2\pi p_k}{N_{\text{tr}}} t \right) + b_k \sin \left(\frac{2\pi p_k}{N_{\text{tr}}} t \right), \quad k = 1, 2, \dots, K, \quad (10)$$

where the frequency component $\underline{S}_{t:}^{\text{fre}} W_{\text{fre}}$ is a linear integration of K different waveforms along time-axis, $S^{\text{fre}} \in \mathbb{R}^{L \times K}$ using weight $W_{\text{fre}} \in \mathbb{R}^{K \times H}$. a_k , b_k , and p_k are trainable parameters tuning the amplitude of cos and sin waves and the frequency for the k -th waveform, respectively.

To prepare initial waveforms $\underline{S}_{t:}^{\text{fre}}$ representing training time-series data \mathcal{D}_{tr} , we apply a Fourier transform to the sequence $[X_{0c}, X_{1c}, \dots, X_{(N_{\text{tr}}-1)c}]$ of each observation c and extract $N_{\text{tr}}/2$ waves. Among the waves whose periods do not exceed the history length L , we select m waves with the largest power spectra and set their amplitudes and frequencies as the initial values of a , b , and p for each observation c —there are total $K = mC$ waves.

The feature-mixing in FAM is sensitive to permutations as the frequency components may vary with the time-step t . This allows for flexible feature mixing based on the inherent periodic characteristics of time series, e.g., seasonality, trend cycles, and cyclical cycles. This could potentially enhance the capacity of the model to capture the temporal dynamics of the data and improve the prediction performance.

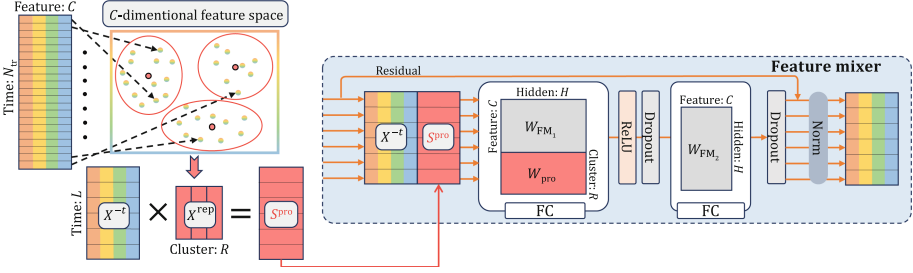


Fig. 4. Architecture of event proximity-aware mixer (EPAM), a temporal characteristics-sensitive extension of Feature Mixing MLPs in TSMixer (in Fig. 2). A matrix $S^{\text{pro}} \in \mathbb{R}^{L \times R}$ contains R different representative observations (events) along the time axis, linearly integrated by weight W_{pro} .

3.2 Event Proximity-Aware Mixer (EPAM)

We propose Event Proximity-Aware Mixer (EPAM) which incorporates an adaptive proximity component into the first layer of feature-mixing MLPs (in Eq. 8 and Fig. 2) as depicted in Fig. 4 and as follows:

$$\begin{aligned}
 U_t &= \text{Drop} \left(\sigma \left(X_t^{-t} W_{\text{FM}_1} + \frac{S_t^{\text{pro}} W_{\text{pro}}}{\|S_t^{\text{pro}}\|} + \mathbf{b}_{\text{FM}_1} \right) \right), \\
 S_t^{\text{pro}} &= X_t^{-t} X^{\text{rep}},
 \end{aligned} \tag{11}$$

where the proximity component $S_t^{\text{pro}} W_{\text{pro}}$ is a linear integration of proximities to R different representative observations (events), $S^{\text{pro}} \in \mathbb{R}^{L \times R}$, using weight $W_{\text{pro}} \in \mathbb{R}^{R \times H}$. $X^{\text{rep}} \in \mathbb{R}^{C \times R}$ is a set of R representative observation vectors and S_t^{pro} is the inner product (similarity) between an observation vector X_t^{-t} at time-step t and representative vectors X^{rep} .

To prepare representative vectors X^{rep} , we apply a clustering method, e.g., k-means, into C -dimensional vectors across all training time steps, $\{X_t\}_{t=0}^{N_{\text{tr}}-1}$ and set R cluster centroids as X^{rep} .

The feature mixing in EPAM is also sensitive to permutations as the proximity components may vary with the time-step t . This allows for flexible feature mixing based on the natural variability of time series due to the occurrence of various types of events, e.g., holiday and weather events, etc., potentially enhancing the capacity of the model to capture the fluctuation pattern of the data and improve the prediction performance.

3.3 Entire Architecture and Training

The architecture of the proposed method, i.e., $f_{\theta}(X^{-t})$, is a variant of TSMixer depicted in Fig. 2 where its feature mixer component is replaced with our proposed permutation-sensitive feature mixer: FAM (in Fig. 3 or TPAM (in Fig. 4. We refer to the combination of TSMixer with our proposed feature mixers as TSMixer + FAM and TSMixer + TPAM, respectively.

For training the entire architecture, we used mean squared error (MSE) as the loss function $\mathcal{L}(\cdot)$ in Eq. 3 as follows:

$$\mathcal{L}(X^{t+}, f_{\theta}(X^{-t})) = \frac{1}{TC} \|X^{t+} - f_{\theta}(X^{-t})\|_F^2, \quad (12)$$

where $\|\cdot\|_F$ is Frobenius norm.

We use early stopping with 5-epoch patience based on the validation loss computed using the validation data described in Table 1 and select the best model with the minimum validation loss.

4 Experimental Evaluation

In this section, we show the effectiveness of the proposed method through experiments on seven popular multivariate long-term forecasting benchmarks such as weather, electricity, and traffic.

4.1 Setting and Comparative Methods

We set the length of history observations as $L = 512$ following the work [11], and the length of future prediction observations as $T \in \{96, 192, 336, 720\}$.

We compared the performance of prediction with the state-of-the-art multivariate time series forecasting methods: Transformer-based and MLP-mixer-based models as follows:

- Transformer-based models: we used codes with default settings provided in following githubs:
 - Autoformer [16]: <https://github.com/thuml/Autoformer>
 - Informer [18]: <https://github.com/zhouhaoyi/Informer2020>
 - PatchTST [11]: <https://github.com/yuqinie98/PatchTST>
- MLP-mixer-based models:
 - TSMixer [6]: we used the basic version of TSMixer provided in the github <https://github.com/google-research/google-research/tree/master/tsmixer> and settings described in the work [6].
 - TMix-Only: we eliminated the feature mixer component (in Fig. 2) from the above TSMixer following the work [6].
 - TSMixer + FAM (proposed method, Sect. 3.1): we set the number of waves for each observation type as $m = 3$ for datasets with fewer observation types, i.e., ETT and Weather, and $m = 1$ for datasets with more types, i.e., Electricity and Traffic. We utilized `numpy.fft.rfft` function for the implementation of Fourier transform. Other settings are same as TSMixer. In addition, we applied reversible instance normalization (RevIN) into the each input X^{-t} and output \widehat{X}^{t+} of the model [9].
 - TSMixer + EPAM (proposed method in Sect. 3.2): we set the number of representative observations as $R = 5$. We utilized `sklearn.cluster` module for k-means clustering. Other settings are same as TSMixer + FAM.

Table 1. Details of the datasets used in the experiments

	ETT _{h1/h2}	ETT _{m1/m2}	Weather	Electricity	Traffic
No. of obs. C	7	7	21	321	862
Time steps	17,420	69,680	52,696	26,304	17,544
Time cycle	1 h	15 min	10 min	1 h	1 h
Data split train:valid:test	12:4:4 [month]		70:10:20 [%]		

4.2 Datasets

We used seven real-world multi-variate time series datasets: ETT [13], Weather [3], Electricity [2], and Traffic [1], provided by the work of Autoformer [16] in <https://github.com/thuml/Autoformer>.

More specifically, Electricity Transformer Temperature (ETT) datasets contain two-year sequences of loads and oil temperature collected from electricity transformers every 1 h and 15 min. Weather dataset contains one-year sequences of 21 meteorological indicators, such as air temperature and humidity, recorded every 10 min. Electricity dataset contains three-year sequences of electricity consumption of 321 customers, collected every hour. Finally, Traffic dataset contains two-year sequences of road occupancy rates at 862 different places, recorded every hour. Table 1 summarizes the details of the datasets.

As a preprocessing step for the data, we divided the sequences from each dataset into training, validation, and test subsequences, as described in Table 1. We then calculated the mean and standard deviation for each subsequence $[X_{0c}, X_{1c}, \dots]$ for each observation c and used these values to normalize the corresponding subsequences. Then, we used these normalized subsequences as training \mathcal{D}_{tr} (in Eq. 4), validation \mathcal{D}_{val} , test \mathcal{D}_{te} (in Eq. 5) data.

4.3 Result

The experimental results are shown in Table 2. The performance is measured using MSE of test data \mathcal{D}_{te} as follows:

$$\text{MSE}(\mathcal{D}_{\text{te}}) = \frac{1}{TC|\mathcal{D}_{\text{te}}|} \sum_{(X^{-t}, X^{t+}) \in \mathcal{D}_{\text{te}}} \|X^{t+} - f_{\theta}(X^{-t})\|_{\text{F}}^2. \quad (13)$$

In principle, multivariate models that simultaneously consider the relationships between time and features are expected to offer higher flexibility and performance in time series forecasting compared to univariate models, which only independently consider the time series of individual features. However, Table 2 demonstrates that the performance of Autoformer and Informer in multivariate models is inferior to that of the univariate model. Furthermore, the performance of TSMixer is equivalent to that of TMix-Only, which lacks a feature mixer, indicating that the co-occurrences in feature direction provided by a feature mixer, are not necessarily important for prediction, as reported in the work [6].

Table 2. Performance comparison in multivariate time series forecasting. The performance is measured using the MSE computed from each test data \mathcal{D}_{te} (in Eq. 13). Those performance surpassing TSMixer is indicated in red among multivariate models. In addition, the best performance among all models is indicated in bold.

Dataset		Univariate Model		Multivariate Model				
Model		TMix-Only	PatchTST	Autoformer	Informer	TSMixer	+FAM	+EPAM
ETTh1	96	0.3643	0.3721	0.6110	0.8437	0.3645	0.3644	0.3626
	192	0.3995	0.4106	0.5105	0.9920	0.4011	0.4010	0.3948
	336	0.4231	0.4216	0.5767	1.3062	0.4249	0.4245	0.4104
	720	0.4543	0.4473	0.6887	1.3983	0.4562	0.4535	0.4342
ETTh2	96	0.2698	0.2749	0.4826	0.3512	0.2730	0.2707	0.2734
	192	0.3366	0.3385	0.5602	0.3805	0.3386	0.3347	0.3344
	336	0.3602	0.3302	0.7877	0.3832	0.3631	0.3639	0.3654
	720	0.4229	0.3839	0.9875	0.6024	0.4219	0.4269	0.4520
ETTh1	96	0.2879	0.2909	0.4769	0.6620	0.2861	0.2858	0.2852
	192	0.3256	0.3349	0.5854	0.7321	0.3266	0.3254	0.3326
	336	0.3561	0.3636	0.6663	0.6024	0.3568	0.3587	0.3691
	720	0.4160	0.4166	0.6939	0.6474	0.4167	0.4202	0.4192
ETTh2	96	0.1678	0.1652	0.2866	0.3290	0.1677	0.1654	0.1962
	192	0.2184	0.2226	0.3288	0.6741	0.2185	0.2222	0.2299
	336	0.2817	0.2735	0.3809	0.8493	0.2815	0.2782	0.2953
	720	0.4015	0.3593	0.4683	0.9752	0.4175	0.4074	0.5512
Weather	96	0.1491	0.1482	0.3799	0.4160	0.1489	0.1474	0.1480
	192	0.1888	0.1938	0.3174	0.7115	0.1894	0.1897	0.1913
	336	0.2396	0.2468	0.3487	0.9766	0.2370	0.2429	0.2363
	720	0.3148	0.3136	0.3888	1.1191	0.3124	0.3219	0.3087
Electricity	96	0.1307	0.1289	0.2265	0.3316	0.1300	0.1288	0.1290
	192	0.1497	0.1468	0.2203	0.3574	0.1487	0.1491	0.1460
	336	0.1636	0.1659	0.2203	0.3848	0.1633	0.1633	0.1604
	720	0.1940	0.1997	0.2441	0.4170	0.1948	0.1939	0.1925
Traffic	96	0.3795	0.4104	0.6804	1.2582	0.3798	0.3758	0.3791
	192	0.3979	0.4125	0.6732	1.3366	0.3982	0.3936	0.3959
	336	0.4152	0.4232	0.7119	1.4528	0.4147	0.4085	0.4171
	720	0.4509	0.4614	0.7401	1.4859	0.4497	0.4487	0.4498

On the other hand, Table 2 shows that the proposed methods, TSMixer+FAM and TSMixer+EPAM, which enhance the feature-mixer with permutation sensitivity, outperform TSMixer in various datasets and future prediction steps, i.e., T . This indicates the potential of the proposed methods for permutation-dependent feature mixing in adaptively modeling relationships between features

and reveals the potential importance of feature mixing in multivariate time series forecasting.

Table 3. Comparison of parameter counts and averaged inference time per instance, measured using the test data \mathcal{D}_{te} in Traffic dataset.

Traffic		
Model	# of params	Average inference time (ms)
TMix-Only	903,292	1.557
TSMixer	1,795,112	1.838
+FAM	2,245,948	3.769
+EPAM	1,797,722	1.831

Table 3 presents a comparison of parameter counts and the average time per inference, measured using the test data \mathcal{D}_{te} in Traffic dataset with the most types of observations as shown in Table 1. As the table shows, multivariate models tend to have larger parameters as the number of observations C increases compared to univariate models. Furthermore, in the case of the TSMixer+FAM, model, datasets with a large number of observations result in a significantly higher number of extracted frequencies K (in Eq. 10), leading to a much longer inference time compared to other models. In the future, it will be necessary to adopt strategies such as feature grouping to reduce the number of additional vectors while maintaining predictive accuracy.

4.4 Analysis

In addition, Fig. 5 depicts examples of forecasts by TSMixer, FAM, and EPAM, for T (Temperature), WV (Wind Velocity), rain(precipitation), SWDR (Short Wave Downward Radiation), and CO2 (CO2 concentration) of Weather dataset with forecasting length $T = 720$ —the history $X_{:c}^{-t}$ and ground truth $X_{:c}^{t+}$ (Eq. 1) in blue and the prediction $\widehat{X}_{:c}^{t+}$ (in Eq. 2) in orange.

Figure 5 shows that compared to TSMixer, TSMixer+FAM is able to more closely follow the true values with its periodic predictions such as in WV and CO2. This outcome underscores the impact of FAM’s dynamic frequency-dependent feature mixing, which is further enhanced by the incorporation of additional frequency information into the feature mixing process. Therefore, in fields such as multivariate time series forecasting involving frequency components, like temporal traffic patterns, FAM’s ability to accurately track waveforms is considered effective.

Similarly, EPAM is able to more closely follow the true values with its finer-grained predictions. This fine-grained prediction reflects the impact of EPAM’s approach of integrating distance information between major events and each historical point into the feature mixing. Therefore, in multivariate time series

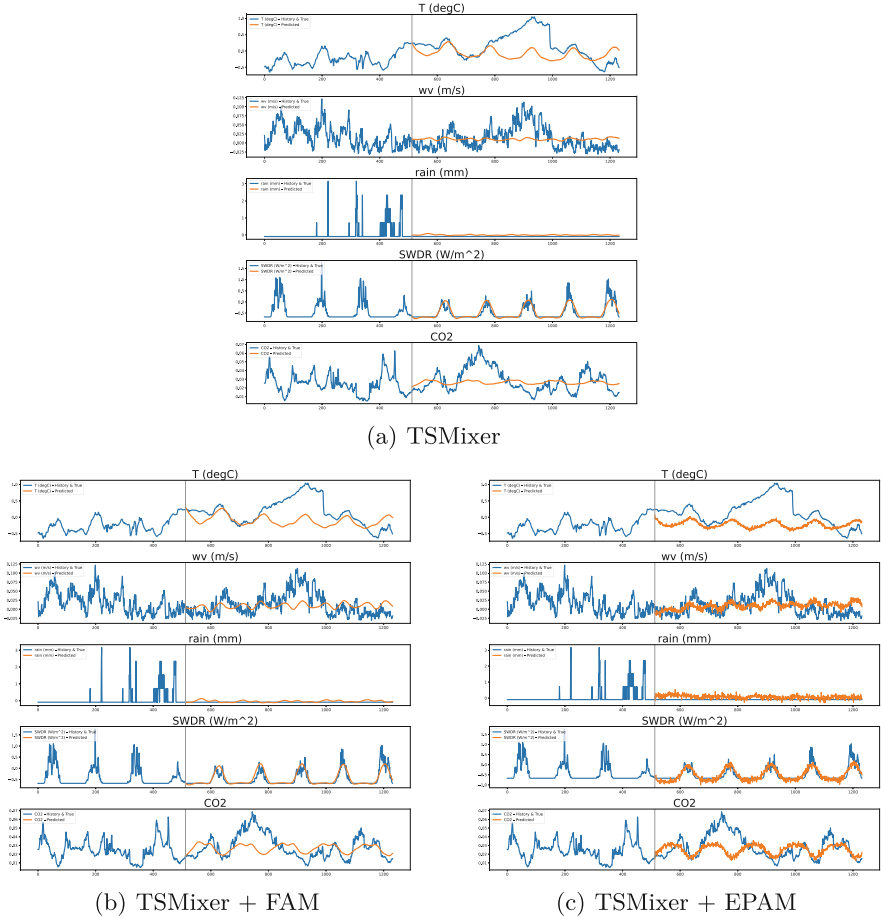


Fig. 5. Examples of ground truth (in blue) and prediction (in orange) by TSMixer, +FAM, and +EPAM models, for T (Temperature), RH (Relative Humidity), WV (Wind Velocity), SWDR (Short Wave Downward Radiation), and CO2 (CO2 concentration) variables in the test data \mathcal{D}_{tr} of Weather dataset with forecasting length $T = 720$. The history length L is fixed at 512 for all experiments, and the grey vertical line indicates the boundary date between the history and the future. (Color figure online)

forecasting involving rapid changes within short periods, such as Electricity data, EPAM’s ability to make detailed adjustments is considered effective.

From these observations, it is clear that in the domain of multivariate time series data with complex inter-feature relationships, the proposed method can enhance the ability to utilize the relationships between features for more accurate forecasting.

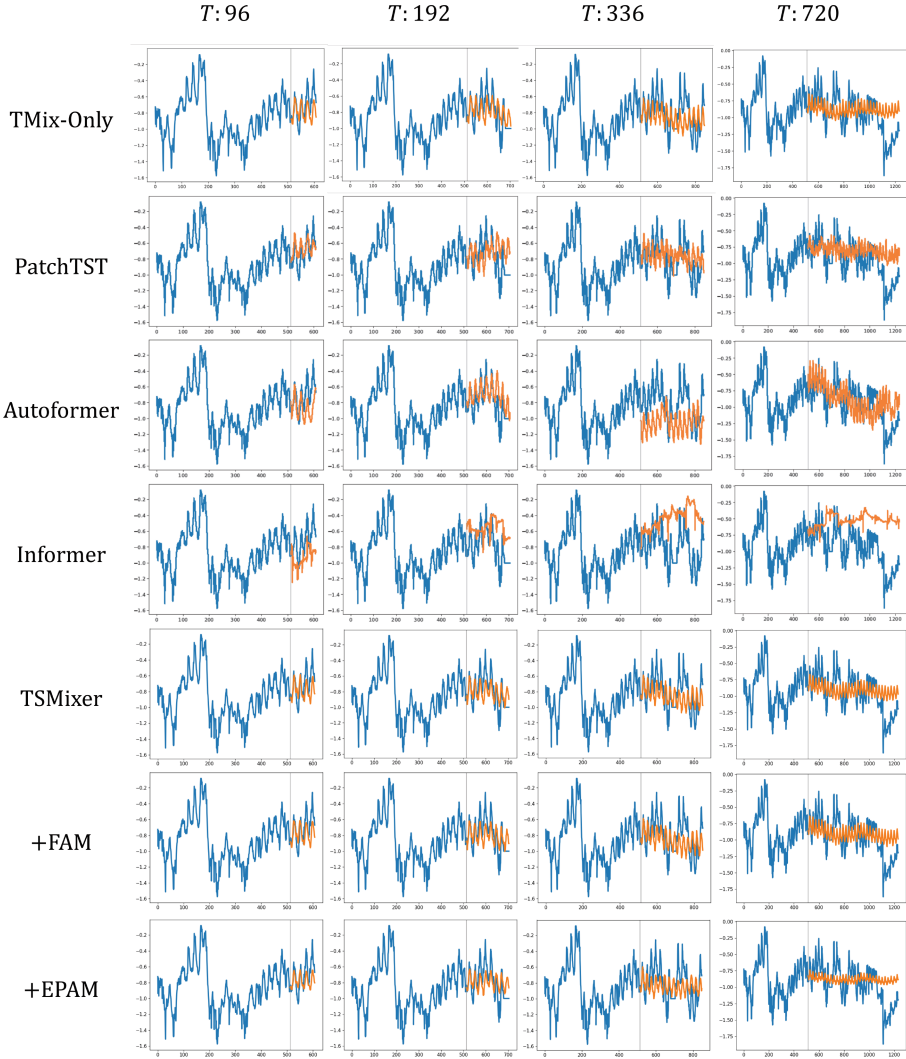


Fig. 6. Examples of ground truth (in blue) and prediction (in orange) for Oil Temperature (OT) variable in the test data \mathcal{D}_{tr} of ETTh1 dataset. Each row and column corresponds to a different model and forecast length $T \in \{96, 192, 336, 720\}$. The history length L is fixed at 512 for all experiments, and the grey vertical line indicates the boundary date between the history and the future. (Color figure online)

To further analyze the effectiveness of the proposed method, we visualized the experimental results for different forecasting lengths across all comparison methods. Fig. 6 depicts examples of forecasts for Oil Temperature (OT) variable of ETTh1 dataset with forecasting length $T \in \{96, 192, 336, 720\}$ —the history

$X_{:c}^{-t}$ and ground truth $X_{:c}^{t+}$ (Eq. 1) in blue and the prediction $\widehat{X}^{t+}_{:c}$ (in Eq. 2) in orange, where c corresponds to OT.

In contrast to nonlinear multivariate models, i.e., Autoformer and Informer, which increasingly diverge from the ground truth as the forecast horizon extends, TSMixer-based models are capable of delivering forecasts that are on par with univariate models across various forecast lengths, T . Furthermore, compared to TSMixer, FAM tends to predict with more periodic trends, while EPAM tends to predict with finer amplitudes. This likely occurs because the models have effectively adjusted the mixing features based on the overall frequency of the training data and the proximity to representative events. Consequently, these models not only capture the temporal mixing of information but also extract useful information through feature mixing.

5 Conclusion

In this study, we enhance feature mixing in the TSMixer model for multivariate time series forecasting by introducing principal frequency components through the Frequency-Aware Mixer (FAM) and incorporating distances to principal time steps with the Event Proximity-Aware Mixer (EPAM). These proposed methods enable time-dependent and adaptive feature mixing, demonstrating the utility of permutation-dependent feature mixing for dynamically capturing the relationships between features. Experimental results across various real-world datasets indicate the potential of the proposed methods for permutation-dependent feature mixing in adaptively modeling relationships between features and reveal the importance of feature mixing in multivariate time series forecasting.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. California Department of Transportation. <https://pems.dot.ca.gov/>. Accessed 23 Mar 2023
2. ElectricityLoadDiagrams. <https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014>. Accessed 23 Mar 2023
3. Max-Planck-Institut für Biogeochemie - Wetterdaten. <https://www.bgc-jena.mpg.de/wetter/>. Accessed 23 Mar 2023
4. Box, G.E.P., Jenkins, G.M.: Time Series Analysis, Forecasting and Control (1970)
5. Box, G.E., Jenkins, G.M.: Some recent advances in forecasting and control (1968)
6. Chen, S.A., Li, C.L., Yoder, N.C., Arik, S.O., Pfister, T.: Tsmixer: an all-mlp architecture for time series forecasting (2023). <https://arxiv.org/abs/2303.06053>
7. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* (1997)
8. Khabbazan, S., et al.: Crop monitoring using sentinel-1 data: a case study from The Netherlands (2019)
9. Kim, T., Kim, J., Tae, Y., Park, C., Choi, J.H., Choo, J.: Reversible instance normalization for accurate time-series forecasting against distribution shift (2022). <https://openreview.net/forum?id=cGDAkQo1C0p>

10. Lai, G., Chang, W.C., Yang, Y., Liu, H.: Modeling long- and short-term temporal patterns with deep neural networks. In: SIGIR (2018)
11. Nie, Y., Nguyen, N.H., Sinthong, P., Kalagnanam, J.: A time series is worth 64 words: long-term forecasting with transformers. In: International Conference on Learning Representations (2023)
12. Rangapuram, S.S., Seeger, M.W., Gasthaus, J., Stella, L., Wang, Y., Januschowski, T.: Deep state space models for time series forecasting. *Adv. Neural Inf. Process. Syst.* **31** (2018)
13. THUML: ETDataset: GitHub Repository. <https://github.com/zhouhaoyi/ETDataset> (2023). Accessed 22 Mar 2023
14. Vaswani, A., et al.: Attention is all you need. *Adv. Neural Inf. Process. Syst.* **30** (2017). <https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf>
15. Wen, R., Torkkola, K., Narayanaswamy, B., Madeka, D.: A multi-horizon quantile recurrent forecaster. In: NeurIPS (2017)
16. Wu, H., Xu, J., Wang, J., Long, M.: Autoformer: decomposition transformers with auto-correlation for long-term series forecasting. *Adv. Neural Inf. Process. Syst.* (2021)
17. Zeng, A., Chen, M., Zhang, L., Xu, Q.: Are transformers effective for time series forecasting? (2022). <https://arxiv.org/abs/2205.08459>
18. Zhou, H., et al.: Informer: beyond efficient transformer for long sequence time-series forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence (2021). <https://doi.org/10.48550/arXiv.2012.07436>
19. Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., Jin, R.: Fedformer: frequency enhanced decomposed transformer for long-term series forecasting (2022)



Prior Bilinear-Based Models for Knowledge Graph Completion

Jiayi Li^{1,2}, Ruilin Luo¹, Jiaqi Sun³, Jing Xiao⁴, and Yujiu Yang¹(✉)

¹ Tsinghua University, Beijing, China

{lijy20,lr123}@mails.tsinghua.edu.cn, yang.yujiu@sz.tsinghua.edu.cn

² Baidu Inc., Beijing, China

³ Carnegie Mellon University, Pittsburgh, PA, USA

jiaqisun@andrew.cmu.edu

⁴ Ping An Technology (Shenzhen) Co., Ltd., Shenzhen, China

XIAOJING661@pingan.com.cn

Abstract. Bilinear-based models are powerful and widely used approaches for Knowledge Graphs Completion (KGC). Despite the considerable progress achieved by bilinear-based models, prior research has predominantly focused on posterior properties, such as symmetry patterns, while neglecting the consideration of prior properties. In this paper, we identify a prior property known as "the law of identity" that eludes capture by bilinear-based models, thus impeding their comprehensive modeling of Knowledge Graph (KG) characteristics. To overcome this limitation, we propose a novel solution named Unit Ball Bilinear Model (UniBi). UniBi not only attains theoretical superiority but also enhances interpretability and performance by minimizing ineffective learning through minimal constraints. Experimental results demonstrate that UniBi effectively models the prior property while validating its interpretability and performance.

Keywords: Identity in KG · Bilinear-based model · Knowledge graph completion

1 Introduction

Knowledge Graphs (KGs) store human knowledge in the form of triple (h, r, t) , which represents a relation r between a head entity h and a tail entity t [13]. KGs benefit a lot of downstream tasks and applications, e.g., recommender system [41], dialogue system [10] and question answering [23]. Since actual KGs are usually incomplete, researchers are interested in predicting missing links to complete them, termed Knowledge Graph Completion (KGC).

We assume that a Knowledge Graph (KG) is a set of facts about sets of entities \mathcal{E} and relations \mathcal{R} . Each fact is stored by a triple $(e_i, r_j, e_k) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ where e_i and r_j denote the i -th entity and the j -th relation, respectively. To

Jiayi Li,Ruilin Luo: Equal contribution

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024

A. Bifet et al. (Eds.): ECML PKDD 2024, LNAI 14943, pp. 317–334, 2024.

https://doi.org/10.1007/978-3-031-70352-2_19

address the challenge of KGC, Knowledge Graph Embedding (KGE) emerges as a common solution, completing KGs by learning low-dimensional representations of entities and relations through a score function $s : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \rightarrow \mathbb{R}$.

As a prominent category within Knowledge Graph Embedding (KGE), bilinear-based models have demonstrated remarkable advances [11, 18, 25, 29]. However, existing works in this category primarily concentrate on posterior properties, which are derived from evidence in triples, such as relational patterns [18, 36] and complex relations [9, 27]. Here, a critical question arises: **Does a prior property exist?**

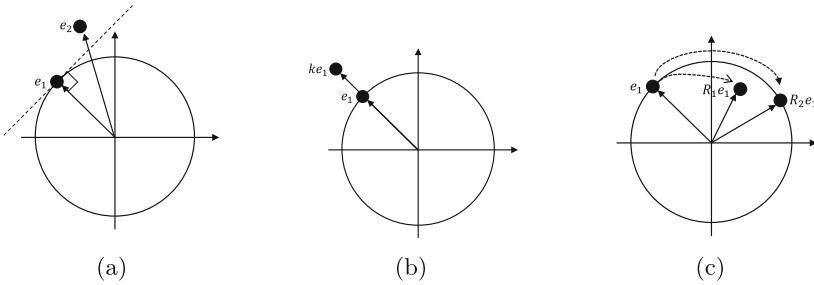


Fig. 1. The flaws of bilinear-based models and our solution in terms of modeling the uniqueness of *identity*. (a) Identity matrix fails to model *identity*. (b) Scaled identity matrix could also model *identity*. (c) An illustration of UniBi. All entities are embedded in the unit sphere and stay in the unit ball after relation-specific transformations.

Our assertion is affirmative, pointing to **the law of identity** in Logic [31]. According to this law, everything is identical to itself. In the context of Knowledge Graphs (KGs), this principle implies that not only should the representations of entities differ, but also the representation of *identity* must be unique, allowing its determination without reliance on any facts or *a priori*. However, we observe that the uniqueness of *identity* has eluded capture in previous bilinear-based models, hindering their ability to fully encapsulate the properties of KGs.

To articulate the problem more precisely, we introduce some notation. In a model with a score function denoted as $s(h, r, t)$, the capacity to capture the uniqueness of *identity* implies that $\forall h \neq t, s(h, r, h) > s(h, r, t)$ holds if and only if r represents *identity*, and its universal representation is distinctive. Additionally, the score function $s(\cdot)$ in bilinear-based models is represented as $\mathbf{h}^\top \mathbf{R} \mathbf{t}$, where $\mathbf{h}, \mathbf{R}, \mathbf{t}$ are the representations of $h, r,$ and t .

In terms of achieving such uniqueness, bilinear-based models exhibit two flaws. Firstly, as depicted in Fig. 1(a), $\mathbf{e}_1^\top \mathbf{I} \mathbf{e}_1 < \mathbf{e}_1^\top \mathbf{I} \mathbf{e}_2$, indicating that the relation matrices themselves do not perfectly model *identity*. Secondly, as illustrated in Fig. 1(b), even if a matrix, such as \mathbf{I} , does model *identity*, its scaled counterpart $k\mathbf{I}$ can also represent *identity*, thereby compromising uniqueness.

Capturing this property necessitates constraints on both entities and relations, which might reduce expressiveness. However, we mitigate this cost by

minimizing constraints, one per entity or relation, while modeling the desired property. Specifically, we normalize the vectors of entities and the spectral radius of relation matrices to 1. Given that the model captures entities within a unit ball, as shown in Fig. 1(c), we coin the model as the Unit Ball Bilinear Model (UniBi). Apart from its theoretical superiority, UniBi demonstrates increased power and interpretability. Achieving unique identity modeling requires normalizing scales that contain minimal useful knowledge. On the one hand, scale normalization prevents ineffective learning on scales, enabling UniBi to focus more on useful knowledge. On the other hand, it elucidates the relationship between the complexity of relations and the singular values in the matrices used to represent them.

We extensively validate the performance of our UniBi model on four datasets: WN18RR, FB15k-237, YAGO3-10-DR [21], and ogbl-biokg. Our results exhibit an advantage over previous structure-based models in terms of metrics such as MRR and Hits@1.

2 Related Work

Previous work mainly handle two kind of posterior properties, namely relational patterns and complex relations. On the one hand, relational patterns are the intrinsic properties of relations, and is formally introduced by ComplEx [29]. Base on this, RotatE [26] proposes composition pattern, Analogy [18] introduces analogy pattern, or commutative pattern, and Dihedral [36] adds non-commutative pattern. On the other hand, complex relations are the extrinsic properties of relation, and is introduced in TransH [35] to denote the relations that are not 1-1, or 1-N, N-1, N-N.

Distance based models choose Euclidean distance for their score functions. TransE [4] inspired by Word2Vec [22] in Natural Language Processing proposes the first distance based model, which uses translation as the linear transformation $s(h, r, t) = -\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|$. TransH [17] and TransR [17] find that TransE difficult to handle complex relations and thus apply linear projections before translation. Apart from translation, RotatE [26] first introduces rotation as the transformation. RotE [5] further combines translation and rotation. Some works also introduce hyperbolic spaces [2, 5, 33].

Beyond posterior properties, previous work of KGE can be roughly divided into the following three categories: distance, bilinear and others.

In contrast, bilinear based models have score functions in the bilinear form $s(h, r, t) = \mathbf{h}^T \mathbf{R} \mathbf{t}$. RESCAL [25] is the first bilinear based model whose relation matrices are unconstrained. Although RESCAL is expressive, it contains too many parameters and tends to overfitting. DistMult [38] simplifies these matrices into diagonal ones. ComplEx [29] further introduces complex values to model the skew-symmetry pattern. STaR [16] integrate scaling, the combination of translation and rotation. Analogy [18] uses block-diagonal to model the analogical pattern and subsumes DistMult, ComplEx, and HolE [24]. Moreover, QuatE [42] extends complex values to quaternion and GeomE [37] utilizes geometric algebra to subsume all these models.

With the rapid development of language models, many methods that focus on text and attribute information [20, 32, 34, 39] have also been proposed.

3 Methodology

3.1 Preliminary

We utilize $\hat{\mathbb{E}}$ and $\hat{\mathbb{R}}$ to denote the set of all possible representations of entities and relations. And we use $\mathbf{e} \in \hat{\mathbb{E}}$ and $\mathbf{R} \in \hat{\mathbb{R}}$ to denote the embedding vector of the entity e and the transformation matrix specific to the relation \mathbf{R} . Furthermore, we use $\|\cdot\|$ to denote the L2 norm of the vectors, $\|\cdot\|_F$ and $\rho(\cdot)$ to represent the Frobenius norm and the spectral radius of a matrix.

In this paper, we focus on n -dimensional real space \mathbb{R}^n , which means $\hat{\mathbb{E}} \subseteq \mathbb{R}^n$ and $\hat{\mathbb{R}} \subseteq \mathbb{R}^{n \times n}$. We also consider real vector spaces whose vectors are complex \mathbb{C}^n or hypercomplex space \mathbb{H}^n , since they are isomorphic to \mathbb{R}^{2n} or \mathbb{R}^{4n} .

3.2 Prior Property and Identity Relation

The law of identity, expressed as $\forall x, x = x$, prompts us to examine how entities and the *identity* relation, denoted as x and $=$, are embedded. Given that the embedding of *identity* should be specified *a priori*, we establish the following equivalent definition:

Definition 1. *A KG model can model the law of identity, which means that the embeddings of entities are different, and the embeddings of the identity relation are unique.*

While the differences in entity embeddings are effortlessly addressed, bilinear-based models struggle to capture the uniqueness of identity. We illustrate two cases where bilinear-based models fall short. Firstly, as depicted in Fig. 1(a), $\mathbf{e}_1^\top \mathbf{I} \mathbf{e}_1 < \mathbf{e}_1^\top \mathbf{I} \mathbf{e}_2$, revealing that the matrix of a relation alone does not guarantee effective modeling of *identity*. Secondly, as demonstrated in Fig. 1(b), even if a matrix, such as \mathbf{I} , does model *identity*, its scaled counterpart $k\mathbf{I}$ where $k > 0, k \neq 1$ can also represent *identity*, contradicting the quantification of uniqueness. Consequently, we introduce a formal definition, building upon Definition 1, to explore modifications to bilinear-based models for capturing this uniqueness and adhering to the law of identity.

Definition 2. *A bilinear model can model the law of identity means:*

$$\exists! \mathbf{R} \in \hat{\mathbb{R}}, \forall \mathbf{h}, \mathbf{t} \in \hat{\mathbb{E}}, \mathbf{h} \neq \mathbf{t}, \mathbf{h}^\top \mathbf{R} \mathbf{h} > \mathbf{h}^\top \mathbf{R} \mathbf{t}, \quad (1)$$

where $\exists!$ is the uniqueness quantification.

3.3 Unit Ball Bilinear Model

From the examples above, it is evident that modeling the law of identity necessitates constraints on both entities and relations, potentially reducing expressiveness. To address this trade-off, we strategically minimize the impact of constraints by imposing them minimally, with one constraint per entity or relation, while still capturing the desired property. Specifically, entity embeddings and the spectral radius of relation matrices are normalized to 1, defined as $\hat{\mathbb{E}} = \mathbf{e}, |\mathbf{e}| = 1, \mathbf{e} \in \mathbb{R}^n$ and $\hat{\mathbb{R}} = \mathbf{R}, |\rho(\mathbf{R})| = 1, \mathbf{R} \in \mathbb{R}^{n \times n}$. The proposed model is named Unit Ball Bilinear Model (UniBi), reflecting its capability to capture entities within a unit ball, as illustrated in Fig. 1(c). The score function of UniBi is expressed as:

$$s(h, r, t) = \mathbf{h}^\top \mathbf{R} \mathbf{t}, \|\mathbf{h}\|, \|\mathbf{t}\| = 1, \rho(\mathbf{R}) = 1. \tag{2}$$

We then have the following theorem.

Theorem 1. *UniBi is capable of modeling the law of identity in terms of definition 2.*

Proof. On the one hand, if $\mathbf{R} = \mathbf{I}$, it is easy to have $\forall \mathbf{h}, \mathbf{t} \in \hat{\mathbb{E}}, \mathbf{h} \neq \mathbf{t}$ that $\mathbf{h}^\top \mathbf{I} \mathbf{h} > \mathbf{h}^\top \mathbf{I} \mathbf{t}$ from the property of cosine.

On the other hand, $\forall \mathbf{R} \in \hat{\mathbb{R}}, \mathbf{R} \neq \mathbf{I}$, we can always give a counterexample. Using singular value decomposition (SVD), we have $\mathbf{R} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^\top$, where $\mathbf{\Sigma} = \text{Diag}[\sigma_1, \dots, \sigma_n]$ with $\sigma_i \geq 0$ and \mathbf{U}, \mathbf{V} are orthogonal matrices. Since $\rho(\mathbf{R}) = 1$, we have $\sigma_{max} = \max(\sigma_i) = 1$.

Besides, we notice that since \mathbf{U}, \mathbf{V} are orthogonal matrices that do not change the norm of vectors, we have $\|\mathbf{h}^\top \mathbf{U}\| = \|\mathbf{V}^\top \mathbf{t}\| = 1$ and use $\hat{\mathbf{h}}$ and $\hat{\mathbf{t}}$ to denote $\mathbf{U}^\top \mathbf{h}$ and $\mathbf{V}^\top \mathbf{t}$ for mathematical simplicity. We consider three scenarios and discuss them separately.

(1) If all singular values of \mathbf{R} are equivalent, we have $\mathbf{\Sigma} = \mathbf{I}$, and we have:

$$s(h, r, t) = \mathbf{h}^\top \mathbf{U} \mathbf{\Sigma} \mathbf{V}^\top \mathbf{t} = \hat{\mathbf{h}}^\top \mathbf{I} \hat{\mathbf{t}} = \hat{\mathbf{h}}^\top \hat{\mathbf{t}}. \tag{3}$$

It is easy to notice that the above equation just goes back to the cosine function, and it has the maximum value when $\hat{\mathbf{h}} = \hat{\mathbf{t}}$ and $\mathbf{U}^\top \mathbf{h} = \mathbf{V}^\top \mathbf{t}$. If $\mathbf{U} \mathbf{V}^\top = \mathbf{I}$, this contradicts the assumption that $\mathbf{R} \neq \mathbf{I}$. If $\mathbf{U} \mathbf{V}^\top \neq \mathbf{I}$, then we have $\mathbf{h} \neq \mathbf{t}$, since $\mathbf{h} = (\mathbf{U}^\top)^{-1} \mathbf{V}^\top \mathbf{t} = \mathbf{U} \mathbf{V}^\top \mathbf{t}$. It means $\mathbf{h}^\top \mathbf{R} \mathbf{h} < \mathbf{h}^\top \mathbf{R} \mathbf{t}$ in this situation.

(2) If not all singular values of \mathbf{R} are equivalent and $\mathbf{U} = \mathbf{V}$, there $\exists i, j \in 1, \dots, n$ that $\sigma_i \neq \sigma_j$. It may be assumed that $\sigma_i > \sigma_j$. Then we take $\forall \mathbf{h} \in \hat{\mathbb{E}}$ that has $\hat{\mathbf{h}}_j = (\frac{\sigma_i}{\sigma_j} - \epsilon) \hat{\mathbf{h}}_i$ where $\epsilon \in (0, \frac{\sigma_i}{\sigma_j} - 1)$. Then we take

$$\hat{\mathbf{t}}_k = \begin{cases} \hat{\mathbf{h}}_k, & k \neq i, j, \\ \hat{\mathbf{h}}_j, & k = i, \\ \hat{\mathbf{h}}_i, & k = j. \end{cases} \tag{4}$$

It is easy to notice that $\hat{\mathbf{h}} = \mathbf{U}^\top \mathbf{h}$, $\hat{\mathbf{t}} = \mathbf{U}^\top \mathbf{t}$ and $\mathbf{h} \neq \mathbf{t}$, then we have

$$\begin{aligned} \mathbf{h}^\top \mathbf{R} \mathbf{t} &= \mathbf{h}^\top \mathbf{U} \boldsymbol{\Sigma} \mathbf{U}^\top \mathbf{t} \\ &= \hat{\mathbf{h}}^\top \boldsymbol{\Sigma} \hat{\mathbf{t}} \\ &= \sigma_i \hat{\mathbf{h}}_i \hat{\mathbf{t}}_i + \sigma_j \hat{\mathbf{h}}_j \hat{\mathbf{t}}_j + \sum_{k \neq i, j} \sigma_k \hat{\mathbf{h}}_k \hat{\mathbf{t}}_k \\ &= \sigma_i \hat{\mathbf{h}}_i \hat{\mathbf{h}}_j + \sigma_j \hat{\mathbf{h}}_j \hat{\mathbf{h}}_i + \sum_{k \neq i, j} \sigma_k \hat{\mathbf{h}}_k^2, \end{aligned} \tag{5}$$

similarly, we have

$$\mathbf{h}^\top \mathbf{R} \mathbf{h} = \mathbf{h}^\top \mathbf{U} \boldsymbol{\Sigma} \mathbf{U}^\top \mathbf{h} = \hat{\mathbf{h}}^\top \boldsymbol{\Sigma} \hat{\mathbf{h}} = \sigma_i \hat{\mathbf{h}}_i^2 + \sigma_j \hat{\mathbf{h}}_j^2 + \sum_{k \neq i, j} \sigma_k \hat{\mathbf{h}}_k^2. \tag{6}$$

Then, we use Eq. 6 minus Eq. 5, and we have

$$\begin{aligned} &\mathbf{h}^\top \mathbf{R} \mathbf{h} - \mathbf{h}^\top \mathbf{R} \mathbf{t} \\ &= \left(\sigma_i \hat{\mathbf{h}}_i^2 + \sigma_j \hat{\mathbf{h}}_j^2 + \sum_{k \neq i, j} \sigma_k \hat{\mathbf{h}}_k^2 \right) - \left(\sigma_i \hat{\mathbf{h}}_i \hat{\mathbf{h}}_j + \sigma_j \hat{\mathbf{h}}_j \hat{\mathbf{h}}_i + \sum_{k \neq i, j} \sigma_k \hat{\mathbf{h}}_k^2 \right) \\ &= \sigma_i (\hat{\mathbf{h}}_i^2 - \hat{\mathbf{h}}_i \hat{\mathbf{h}}_j) + \sigma_j (\hat{\mathbf{h}}_j^2 - \hat{\mathbf{h}}_i \hat{\mathbf{h}}_j) \\ &= \sigma_i \hat{\mathbf{h}}_i (\hat{\mathbf{h}}_i - \hat{\mathbf{h}}_j) - \sigma_j \hat{\mathbf{h}}_j (\hat{\mathbf{h}}_i - \hat{\mathbf{h}}_j) \\ &= (\sigma_i \hat{\mathbf{h}}_i - \sigma_j \hat{\mathbf{h}}_j) (\hat{\mathbf{h}}_i - \hat{\mathbf{h}}_j) \\ &= \left(\sigma_i \hat{\mathbf{h}}_i - \sigma_j \left(\frac{\sigma_i}{\sigma_j} - \epsilon \right) \hat{\mathbf{h}}_i \right) \left(\hat{\mathbf{h}}_i - \left(\frac{\sigma_i}{\sigma_j} - \epsilon \right) \hat{\mathbf{h}}_i \right) \\ &= \left(\sigma_i \hat{\mathbf{h}}_i - \sigma_i \hat{\mathbf{h}}_i + \epsilon \sigma_j \hat{\mathbf{h}}_i \right) \left(1 - \frac{\sigma_i}{\sigma_j} + \epsilon \right) \hat{\mathbf{h}}_i \\ &= \epsilon \sigma_j \hat{\mathbf{h}}_i^2 \left(1 - \frac{\sigma_i}{\sigma_j} + \epsilon \right) \\ &< 0, \end{aligned} \tag{7}$$

which means that $\mathbf{h}^\top \mathbf{R} \mathbf{h} < \mathbf{h}^\top \mathbf{R} \mathbf{t}$ in this case.

(3) If not all singular values of \mathbf{R} are equivalent and $\mathbf{U} \neq \mathbf{V}$, there exists $k \in 1, \dots, n$ such that $\sigma_k = \sigma_{max} = 1$ since $\rho(\mathbf{R}) = 1$. Then, we take the following.

$$\hat{\mathbf{h}}_i = \hat{\mathbf{t}}_i = \begin{cases} 1, & i = k, \\ 0, & i \neq k. \end{cases} \tag{8}$$

It should be noted that $\hat{\mathbf{h}} = \hat{\mathbf{t}}$ and $\mathbf{h} \neq \mathbf{t}$ since $\mathbf{U} \neq \mathbf{V}$. Then, we have

$$\mathbf{h}^\top \mathbf{R} \mathbf{t} = \mathbf{h}^\top \mathbf{U} \boldsymbol{\Sigma} \mathbf{V}^\top \mathbf{t} = \hat{\mathbf{h}}^\top \boldsymbol{\Sigma} \hat{\mathbf{t}} = \sigma_k = 1. \tag{9}$$

Since UniBi is bounded by $[-1, 1]$, we have $\mathbf{h}^\top \mathbf{R} \mathbf{h} \leq 1 = \mathbf{h}^\top \mathbf{R} \mathbf{t}$, which means that $s(h, r, h) > s(h, r, t)$ does not always hold.

In summary, we can conclude that UniBi has $\mathbf{h} \neq \mathbf{t}$, $\mathbf{h}^\top \mathbf{R} \mathbf{h} > \mathbf{h}^\top \mathbf{R} \mathbf{t}$ iff $\mathbf{R} = \mathbf{I}$, which means that UniBi can model *identity* uniquely. Considering that entities are embedded differently, we conclude that UniBi can model the law of identity in terms of definition 2.

Although the proposed model has been proven to model the law of identity, it still has a practical disadvantage since it is difficult to directly represent all matrices whose spectral radius is 1. In addition, it is also time-consuming to calculate the spectral radius $\rho(\cdot)$ via singular values decomposition (SVD). To avoid unnecessary decomposition, we divide a relation matrix into three parts $\mathbf{R} = \mathbf{R}_h \mathbf{\Sigma} \mathbf{R}_t$ where $\mathbf{R}_h, \mathbf{R}_t$ are orthogonal matrices and $\mathbf{\Sigma} = \text{Diag}[\sigma_1, \dots, \sigma_n]$ is a positive semidefinite diagonal matrix. We maintain the independence of these three components during training. Therefore, it becomes simple to obtain matrices whose spectral radius is 1, that is, $\frac{\mathbf{R}_h \mathbf{\Sigma} \mathbf{R}_t}{\sigma_{max}}$. We transform the score function Eq. 2 into the following form.

$$s(h, r, t) = \frac{\mathbf{h}^\top \mathbf{R}_h \mathbf{\Sigma} \mathbf{R}_t \mathbf{t}}{\sigma_{max} \|\mathbf{h}\| \|\mathbf{t}\|}, \tag{10}$$

where σ_{max} is the maximum among σ_i .

In addition, we find that the calculation of the orthogonal matrix is still time-consuming [27]. To this end, we only consider the diagonal orthogonal block matrix, where each block is a low-dimensional orthogonal matrix. Specifically, we use k -dimensional rotation matrices to build \mathbf{R}_h and \mathbf{R}_t . Taking \mathbf{R}_h as an example $\mathbf{R}_h = \text{Diag}[\mathbf{SO}(k)_1, \dots, \mathbf{SO}(k)_{\frac{n}{k}}]$, where $\mathbf{SO}(k)_i$ denotes the i -th special orthogonal matrix, that is, the rotation matrix.

The rotation matrix only represents the orthogonal matrices whose determinants are 1 and does not represent the ones whose determinants are -1 . To this end, we introduce two diagonal sign matrices of n -th order $\mathbf{S}_h, \mathbf{S}_t \in \mathbb{S}$ where

$$\mathbb{S} = \{\mathbf{S} \mid \mathbf{S}_{ij} = \begin{cases} \pm 1, & \text{if } i = j, \\ 0, & \text{if } i \neq j. \end{cases}\}. \tag{11}$$

Thus, we could rewrite the score function Eq. 10 to

$$s(h, r, t) = \frac{\mathbf{h}^\top \mathbf{R}_h \mathbf{S}_h \mathbf{\Sigma} \mathbf{S}_t \mathbf{R}_t \mathbf{t}}{\sigma_{max} \|\mathbf{h}\| \|\mathbf{t}\|}. \tag{12}$$

However, the sign matrix \mathbf{S}_h and \mathbf{S}_t are discrete. To address this problem, we notice that $\mathbf{S}_h, \mathbf{\Sigma}, \mathbf{S}_t$ can be merged into a matrix $\mathbf{\Xi}$ that

$$\mathbf{\Xi}_{ij} = \begin{cases} s_i s_j \sigma_i, & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases} \tag{13}$$

where $s_i = (\mathbf{S}_h)_{ii}$, $s_j = (\mathbf{S}_t)_{jj}$, $i, j = 1, \dots, n$ and $\mathbf{\Xi} = \text{Diag}[\xi_1, \dots, \xi_n]$. Thus, we incorporate the discrete matrices $\mathbf{S}_h, \mathbf{S}_t$ into the continuous matrix $\mathbf{\Xi}$.

$$s(h, r, t) = \frac{\mathbf{h}^\top \mathbf{R}_h \mathbf{\Xi} \mathbf{R}_t \mathbf{t}}{|\xi_{max}| \|\mathbf{h}\| \|\mathbf{t}\|}, \tag{14}$$

where $|\xi_{max}|$ is the maximum among $|\xi_i|$.

4 Experiments

4.1 Experiment Setup

Table 1. Statistics of the benchmark datasets.

Dataset	$ \mathcal{E} $	$ \mathcal{R} $	Training	Validation	Test
WN18RR	40,943	11	86,835	3,034	3,134
FB15k-237	14,541	237	272,115	17,535	20,466
YAGO3-10-DR	122,873	36	732,556	3,390	3,359
ogbl-biokg	93773	51	4762678	162886	162870

Dataset. We evaluate our models on four widely adopted benchmarks: WN18RR [8], FB15k-237 [28], YAGO3-10-DR [1], and ogbl-biokg [12]. The first three datasets involve removing reciprocal triples to address data leakage issues in WN18, FB15K, and YAGO3-10, respectively. The ogbl-biokg is a KG encompassing a substantial volume of biomedical data sourced from the Open Graph Benchmark. Comprehensive statistics for these datasets are provided in Table 1.

Baseline Models. In this context, we explore two specific variants of UniBi: UniBi-O(2) and UniBi-O(3), employing rotation matrices in 2 and 3 dimensions to construct the orthogonal matrix. Specifically, we utilize the unit complex value and the unit quaternion to model 2D and 3D rotations, employing 2×2 and 4×4 matrices, respectively. UniBi is systematically compared against several bilinear models, including RESCAL [25], CP [11], ComplEx [29], and QuatE [42]. Additionally, it is juxtaposed with other models such as RotatE [26], MurE [2], RotE [5], Turcker [3], ConvE [8], PairRE [6], and TripleRE [40].

Metrics. For our evaluation, we employ Mean Reciprocal Rank (MRR) and Hits@ k ($k = 1, 3, 10$) as the key metrics. MRR provides the average inverse rank of correct entities, offering resilience to outliers. Hits@ k signifies the proportion of correct entities ranked above k .

4.2 Optimization

We adopt the reciprocal setting [15], which creates a reciprocal relation r' for each r and a new triple (e_k, r'_j, e_i) for each $(e_i, r_j, e_k) \in \mathcal{K}$. Instead of using Cross Entropy directly [15, 42, 43], we add an extra scalar $\gamma > 0$ before the softmax function. Since UniBi is bounded, it brings an upper bound to loss that makes the model difficult to optimize, as discussed by [30].

$$\mathcal{L} = - \sum_{(h,r,t) \in \mathcal{K}_{train}} \log \left(\frac{\exp(\gamma \cdot s(h, r, t))}{\sum_{t' \in \mathcal{E}} \exp(\gamma \cdot s(h, r, t'))} \right) + \lambda \cdot Reg(h, r, t) \quad (15)$$

where $Reg(h, r, t)$ is the regularization term and $\lambda > 0$ is its factor. Specifically, we only take $Reg(h, r, t)$ as DURA [43] in experiments since it significantly

outperforms other regularization terms. In addition, γ is set to 1 for previous methods and greater than 1 for UniBi. And we set the dimension n to 500.

Table 2. Hyperparameters found by grid search. λ is the regularization coefficient, γ is the scaling factor, b is the batch size.

Model	WN18RR			FB15K237			YAGO3-10-DR			ogbl-biokg		
	λ	γ	b	λ	γ	b	λ	γ	b	λ	γ	b
CP	1e-1	1	100	5e-2	1	100	5e-3	1	1000	5e-3	1	500
ComplEx	1e-1	1	100	5e-2	1	100	1e-2	1	1000	1e-3	1	500
RESCAL	1e-1	1	1000	5e-2	1	1000	5e-2	1	1000	5e-3	1	500
UniBi-O(2)	2	20	100	2	25	1000	1.5	30	1000	5e-3	1	500
UniBi-O(3)	2	15	100	1.5	20	1000	1.5	30	1000	5e-3	1	500

5 Implementation Details

We fix the dimension of all models except RESCAL on WN18RR to 500, while RESCAL on WN18RR is set to 256 following [43]. We choose Adam [14] as the optimizer and fix the learning rate at $1e-3$. We set the maximum epochs to 200 and apply the early stopping strategy.

We set the scaling factor γ to 1 for all models except UniBi. And we search γ from $\{1, 5, 10, 15, 20, 25, 30\}$ for UniBi. For the factor for regularization λ we search $\{1, 5e-1, 1e-1, 5e-2, 1e-2, 5e-3, 1e-3\}$ for all models except UniBi and $\{0.5, 1, 1.5, 2, 2.5, 3\}$ for it. The search results are listed in the Table 2. We search for the batch size from $\{100, 1000\}$. In addition, we implemented all the experiments in PyTorch with a single NVIDIA GeForce RTX 2080Ti graphics card for FB15k-237, WN18RR and YAGO-3-10DR.

For the experiments in Sect. 6.1, we set γ for UniBi without DURA in Sect. 6.1 to 10, 15, 25 for WN18RR, FB15k-237, and YAGO3-10-DR for UniBi, respectively. For the experiments on the ogbl-biokg dataset. We choose Adagrad [19] as the optimizer following [7]. We fix the learning rate at $1e-2$, γ at 1, and batch size at 500. And we search λ from $\{1e-3, 5e-3, 1e-2\}$. Additionally, we search for the embedding size from $\{1000, 2000, 3000\}$ for UniBi and other reproducible models. All experiments on ogbl-biokg can be run with a single NVIDIA GeForce RTX 4090 graphics card.

5.1 Main Results

In this section, we illustrate how constraints contribute to the improved performance of UniBi. We primarily compare our model with previous state-of-the-art (SOTA) models, including CP [11], ComplEx [29] and RESCAL [25], using

Table 3. Evaluation results on WN18RR and FB15k-237 datasets. We reimplement RotE, CP, RESCAL, and ComplEx with $n = 500$ and denoted by †, while we take other results from original papers. We also implement the results of PairRE and TripleRE on WN18RR. The best results are in **bold** while the seconds are underlined.

Model	FB15k-237				WN18RR			
	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
CP† [11]	0.361	0.266	0.387	0.551	0.457	0.414	0.469	0.549
RESCAL† [25]	0.364	<u>0.272</u>	0.396	0.547	0.495	0.452	0.491	0.575
DistMult [38]	0.241	0.155	0.263	0.419	0.430	0.390	0.440	0.490
ComplEx† [29]	0.363	0.269	0.390	<u>0.552</u>	0.487	0.445	0.497	0.571
ConvE [8]	0.325	0.237	0.356	0.501	0.430	0.400	0.440	0.520
QuatE [42]	0.348	0.248	0.382	0.550	0.488	0.438	<u>0.508</u>	<u>0.582</u>
RotatE [26]	0.338	0.241	0.375	0.533	<u>0.476</u>	0.428	0.492	0.571
MurP [2]	0.335	0.243	0.367	0.518	0.481	0.440	0.495	0.566
TuckER [3]	0.358	0.266	0.394	0.544	0.470	0.443	0.482	0.526
RotE [5]	0.346	0.251	0.381	0.538	<u>0.494</u>	0.446	0.512	0.585
PairRE [6]	0.351	0.256	0.387	0.544	0.461	0.417	0.479	0.553
TripleRE [40]	0.351	0.251	0.392	<u>0.552</u>	0.473	0.433	0.491	0.562
UniBi-O(2)	0.370	0.274	<u>0.406</u>	0.561	0.487	0.460	0.502	0.566
UniBi-O(3)	<u>0.369</u>	0.274	0.407	0.561	0.492	<u>0.452</u>	0.505	0.571

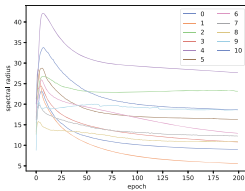
DURA regularization. Although these models have been implemented by [43], the dimensions of CP and ComplEx are excessively high and have not been tested on YAGO3-10-DR. Thus, we re-implement them in this paper for a fair comparison. We also re-implement some other advanced models, such as RotE, PairRE and TripleRE.

As shown in Table. 3, UniBi achieves comparable results to previous bilinear-based models. UniBi is only slightly and justifiably below RESCAL on WN18RR since RESCAL needs require much more time and space. Additionally, as demonstrated in Table. 4, UniBi achieves the highest or second highest performance in all metrics on the YAGO3-10-DR and large-scale ogbl-biokg¹.

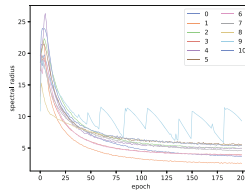
¹ Without considering ensemble learning and exogenous constraints, we achieved the best performance on OGB-biokg.

Table 4. Evaluation results on ogbl-biokg and YAGO3-10-DR datasets. We take the results of PairRE and TripleRE on ogbl-biokg from OGB leaderboard and reimplement other models. For YAGO3-10-DR, we take the results of models denoted by ‡ from [1]. The best results are in **bold** while the seconds are underlined.

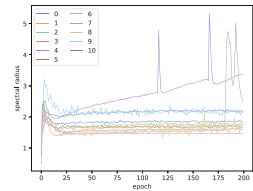
Model	ogbl-biokg				YAGO3-10-DR			
	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
CP [11]	0.838	0.774	0.883	0.951	0.241	0.175	0.253	0.370
RESCAL [25]	0.817	0.745	0.868	0.946	0.233	0.168	0.243	0.360
DistMult‡ [38]	0.825	0.756	0.874	0.948	0.192	0.133	0.208	0.307
ComplEx‡ [29]	0.846	0.787	0.887	0.952	0.238	0.174	0.253	0.360
ConvE‡ [8]	-	-	-	-	0.204	0.147	0.216	0.315
QuatE [42]	0.820	0.748	0.873	0.947	0.229	0.166	0.248	0.351
RotatE‡ [26]	0.815	0.743	0.868	0.943	0.216	0.160	0.229	0.324
MurP [2]	-	-	-	-	0.201	0.146	0.214	0.308
TuckER‡ [3]	0.817	0.745	0.870	0.946	0.207	0.148	0.221	0.320
RotE [5]	-	-	-	-	0.215	0.157	0.227	0.326
PairRE [6]	0.817	0.743	0.872	0.947	0.210	0.159	0.239	0.348
TripleRE [40]	0.835	0.768	0.885	0.952	0.204	0.154	0.233	0.346
UniBi-O(2)	<u>0.852</u>	<u>0.794</u>	<u>0.895</u>	<u>0.954</u>	0.247	<u>0.179</u>	0.268	<u>0.376</u>
UniBi-O(3)	0.856	0.798	0.897	0.956	<u>0.246</u>	0.180	<u>0.264</u>	0.377



(a) RESCAL w/o reg.



(b) RESCAL w/ Frobenius



(c) RESCAL w/ DURA

Fig. 2. Why learning on the scales are ineffective. We test RESCAL with 3 setting a) no regularization, b) use Frobenius norm as regularization, c) use DURA as regularization. We use index rather than name to denote different relations for simplicity. And we notice that 1) non-convergence exists in every case, 2) the better the result, the more the scales converge, 3) Regularization cannot stop the fluctuation of scales. (Better view in color, zoom in, note the difference in the vertical coordinates.)

6 Further Analysis

6.1 UniBi Prevents Ineffective Learning

Here, we demonstrate that ineffective learning does exist, which means the scale is not only redundant but also harmful. As shown in Fig. 2, we take RESCAL as an example to show this phenomenon. We notice that 1) non-convergence

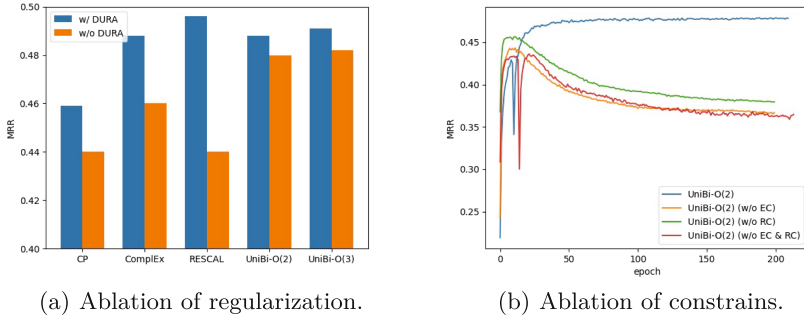


Fig. 3. UniBi benefits from preventing ineffective learning. (a) UniBi less relies on regularization and other models are not. (b) Neither entity constraint (EC) nor relation constraint (RC) alone stops the sliding of performance.

exists in every case, 2) the better the result, the more the scales converge, and 3) regularization cannot stop the fluctuation of scales.

We believe that these cases illustrate, on the one hand, the positive correlation between the constraint scale and the effect, on the other hand, the mere constraint cannot eliminate fluctuations that may interfere with the model learning. Therefore, we think scale is harmful and learning about it is ineffective, and we need a hard constraint rather than a regularization term to prevent this completely.

We further verify that the superiority of UniBi stems from its ability to prevent ineffective learning. We conduct further comparisons without regularization. In addition, we also adopt EC (entity constraint) and RC (relation constraint) to study the effect of both constraints. All experiments are implemented on WN18RR.

On the one hand, as shown in Fig. 3(a), when the regularization term is removed, UniBi’s performance decreases slightly while other models’ performance decreases significantly. This indicates that UniBi’s learning is less dependent on extra regularization, as it effectively prevents ineffective learning. On the other hand, we illustrate the MRR metric of UniBi and its ablation models on the validation set as the epoch grows in Fig. 3(b). On the other hand, as illustrated in Fig. 3(b), we observe the MRR metric of UniBi and its ablation models on the validation set as epochs increase. It is evident that either constraint alleviates overfitting to some extent but fails to prevent the downward sliding trend due to the unconstrained part potentially diverging in scale. Thus, both constraints are indispensable for preventing ineffective learning and improving the performance of UniBi.

6.2 Prior Property Modeling

In this section, we demonstrate that UniBi can effectively model the law of identity, a capability lacking in previous models. Additionally, we emphasize

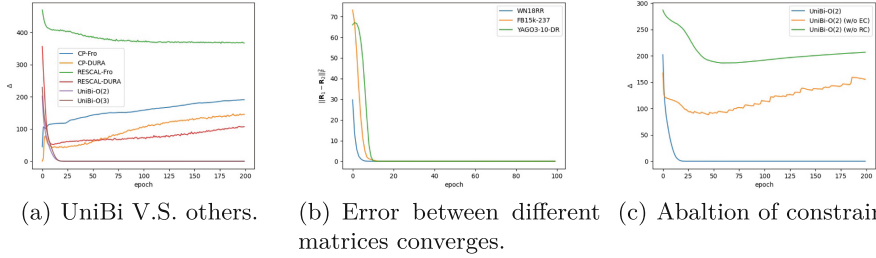


Fig. 4. UniBi is capable to uniquely model *identity*. (a) the imbalance degree (Δ) of UniBi converges to 0 while others diverge. (b) The errors between different matrices modeling *identity* converge to 0 on different datasets. (c) Both entity constrain (EC) and relation constrain (RC) are indispensable for UniBi to model *identity*.

the indispensability of constraints on both entity and relation embeddings. To achieve this, we augment benchmarks by explicitly introducing *identity* as a new relation, utilizing its corresponding matrix to assess the modeling of uniqueness. Given the distinct nature of entities, embodying the law implies capturing the uniqueness of the *identity* relation. This necessitates the convergence of the matrix associated with the *identity* relation to either the identity matrix \mathbf{I} or a scaled variant. For evaluation purposes, we introduce a novel metric, the imbalance degree,

$$\Delta = \left(\sum_i \frac{\sigma_i}{\sigma_{max}} - 1 \right)^2 \tag{16}$$

providing a quantitative measure of convergence and uniqueness modeling.

We first compare UniBi with CP [11] and RESCAL [25], representing the least and most expressive bilinear models on FB15k-237. Additionally, we apply DURA [43] to these models to investigate their ability to adhere to the law of identity under extra regularization. As depicted in Fig. 4(a), the imbalance degree Δ of UniBi converges to 0, while others fail. This observation verifies UniBi’s unique capability to model *identity* distinctly. Furthermore, although the imbalance of other models decreases to some extent when using DURA, they still fall short of achieving a unique representation of *identity*. To demonstrate UniBi’s ability to converge uniquely to *identity*, we employ two matrices, \mathbf{R}_1 and \mathbf{R}_2 , to model it independently. As illustrated in Fig. 4(b), the error between \mathbf{R}_1 and \mathbf{R}_2 also converges to 0, indicating their convergence to \mathbf{I} .

We proceed with an ablation study to ascertain the necessity of both the entity constraint (EC) and the relation constraint (RC) for the unique modeling of *identity*. The experiments reveal that employing either constraint in isolation is insufficient to capture the uniqueness of *identity*, as depicted in Fig.4(c). This finding validates the identified issues illustrated in Fig.1(a) and Fig. 1(b).

6.3 Interpretability

In addition to enhancing performance, scale normalization proves instrumental in comprehending intricate relations. We define complex relations based on whether the head per tail of a relation (hptr) or the tail per head of a relation (tphr) exceeds a specified threshold of 1.5 [35]. Subsequently, relations are categorized into four types: 1-1, 1-N, N-1, and N-N. However, we contend that this categorization is overly coarse-grained and propose a more refined, continuous metric for complexity. To illustrate this notion effectively, we provide an example in Fig. 5, along with the complexity definition.

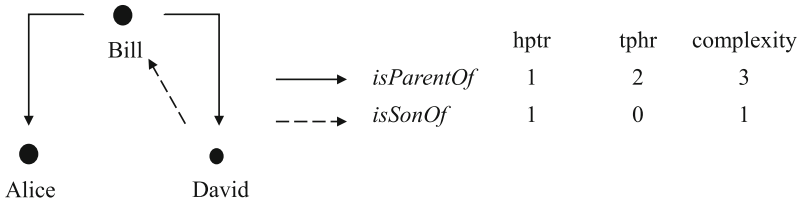


Fig. 5. A toy example to show how to calculate complexity.

Definition 3. *The complexity of a relation is the sum of its hptr and tphr.*

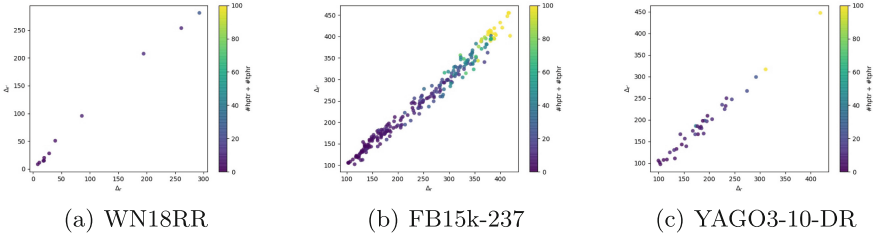


Fig. 6. The imbalance degree (Δ) and complexity ($\# \text{ hptr} + \# \text{ tphr}$) of relations in WN18RR, FB15k-237 and YAGO3-10-DR respectively. Two metrics are highly correlated, and the imbalance of a relation (Δ_r) and the imbalance of its reciprocal one ($\Delta_{r'}$) are very close.

Intuitively, the handling of complex relations involves aggregating entities through projection [17,35]. This suggests that the higher the complexity of a relation, the more pronounced its aggregation effect, and conversely, a lower complexity implies a weaker aggregation effect.

We observe that the aggregation effect can be effectively characterized by the relative ratio, or imbalance degree, of singular values in the matrices of relations.

For any relation matrix $\mathbf{R} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$, where both \mathbf{U} and \mathbf{V} are isometry, the singular values of the scaling matrix $\mathbf{\Sigma}$ exclusively contribute to the aggregation phenomenon.

Furthermore, the singular values of UniBi are constrained to be less than or equal to 1, as the spectral radius. This observation establishes a promising correlation between the singular values of our model and the aggregation, thereby indicating the complexity. Hence, we can utilize singular values to represent the complexity of relations, enhancing the interpretability of UniBi. It is noteworthy that this interpretability extends to other bilinear-based models, provided they normalize the spectral radius of their relation matrix, akin to UniBi.

To substantiate this assertion, we examine the relationship between singular values and the complexity of each relation across three benchmarks, where complexity is computed following Definition 3. Additionally, we gauge the singular values of a relation using the imbalance degree Δ . To distinguish between Δ for a relation r and its reciprocal relation r' , we denote them as Δ_r and $\Delta_{r'}$ respectively.

As illustrated in Fig. 6, we observe a correlation between singular values and the complexity of a relation. Notably, Δ_r and $\Delta_{r'}$ remain closely aligned even in cases of unbalanced relations (1-N or N-1), highlighting that complexity is managed through aggregation, irrespective of direction.

6.4 Comparison in Modeling Identity

Table 5. Ablation of incorporation of *identity* triples (it) on WN18RR dataset. The original version excludes *identity* triples.

Methods	WN18RR	
	MRR	Hits@1
CP	0.457	0.414
CP (w/ it)	0.442	0.403
ComplEx	0.487	0.445
Complex (w/ it)	0.471	0.435
RESCAL	0.495	0.452
RESCAL (w/ it)	0.480	0.438

Perhaps the simplest and most direct approach to modeling the identity attribute is directly constructing triples for the *identity* relation in the training set. We select CP, ComplEx, and RESCAL for the ablation study on the WN18RR dataset. The results are demonstrated in Table. 5. The results are demonstrated in Table. 5, indicate that directly adding identity triples to the training set generally leads to decreased performance. This decline can be attributed to the fact that learning about identity alone does not contribute to the performance of other relations and may even be detrimental to the overall results. Furthermore,

the number of identity triples significantly surpasses that of other relations. For instance, on the WN18RR and FB15k-237 datasets, the average number of triples assigned to each relation in the training set is 7894 and 1148, respectively. In contrast, the number of identity triples equals the number of entities, amounting to 40493 and 14514, respectively. This substantial difference introduces bias into the model’s learning process regarding other relations.

7 Conclusion

In this paper, we introduce a novel perspective, termed *prior property*, for analyzing and modeling Knowledge Graphs (KGs), extending beyond traditional posterior properties. Notably, we observe that existing bilinear-based models struggle to capture this property, prompting us to propose UniBi as a validated solution. Specifically, UniBi applies well-designed normalization on the embeddings of entities and relations with minimal constraints. UniBi not only addresses the challenges posed by the law of identity but also brings forth additional advantages through normalization. Firstly, the normalization safeguards against ineffective learning, thereby enhancing overall model performance. Secondly, it unveils a noteworthy insight: the relative ratio of singular values corresponds to the complexity of relations, significantly improving interpretability.

In summary, we believe the question of prior property and the paradigm of UniBi can provide interesting and useful directions for the studies of bilinear-based models.

Acknowledgements. This work was partly supported by the Shenzhen Science and Technology Program (JSGG20220831110203007) and the "Graph Neural Network Project" of Ping An Technology (Shenzhen) Co., Ltd.

References

1. Akrami, F., Saeef, M.S., Zhang, Q., Hu, W., Li, C.: Realistic re-evaluation of knowledge graph completion methods: an experimental study. In: SIGMOD, pp. 1995–2010. ACM (2020)
2. Balazevic, I., Allen, C., Hospedales, T.M.: Multi-relational poincaré graph embeddings. In: NeurIPS, pp. 4465–4475 (2019)
3. Balazevic, I., Allen, C., Hospedales, T.M.: TuckER: tensor factorization for knowledge graph completion. In: EMNLP/IJCNLP, vol. 1, pp. 5184–5193 (2019)
4. Bordes, A., Usunier, N., García-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NeurIPS, pp. 2787–2795 (2013)
5. Chami, I., Wolf, A., Juan, D., Sala, F., Ravi, S., Ré, C.: Low-dimensional hyperbolic knowledge graph embeddings. In: ACL, pp. 6901–6914 (2020)
6. Chao, L., He, J., Wang, T., Chu, W.: PairRE: knowledge graph embeddings via paired relation vectors. In: ACL/IJCNLP, vol. 1, pp. 4360–4369 (2021)
7. Chen, Y., Minervini, P., Riedel, S., Stenetorp, P.: Relation prediction as an auxiliary training objective for improving multi-relational graph representations. In: 3rd Conference on Automated Knowledge Base Construction (2021). <https://openreview.net/forum?id=Qa3uS3H7-Le>

8. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2d knowledge graph embeddings. In: AAAI, pp. 1811–1818 (2018)
9. Gao, H., Yang, K., Yang, Y., Zakari, R.Y., Owusu, J.W., Qin, K.: QuatDE: dynamic quaternion embedding for knowledge graph completion (2021)
10. He, H., Balakrishnan, A., Eric, M., Liang, P.: Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings. In: ACL, pp. 1766–1776. Association for Computational Linguistics (2017)
11. Hitchcock, F.L.: The expression of a tensor or a polyadic as a sum of products. *J. Math. Phys.* **6**(1–4), 164–189 (1927)
12. Hu, W., et al.: Open graph benchmark: datasets for machine learning on graphs. In: NeurIPS (2020)
13. Ji, S., Pan, S., Cambria, E., Marttinen, P., Yu, P.S.: A survey on knowledge graphs: representation, acquisition, and applications. *IEEE Trans. Neural Netw. Learn. Syst.* **33**(2), 1–21 (2021)
14. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: ICLR (2015)
15. Lacroix, T., Usunier, N., Obozinski, G.: Canonical tensor decomposition for knowledge base completion. In: ICML, vol. 80, pp. 2869–2878 (2018)
16. Li, J., Yang, Y.: Star: Knowledge graph embedding by scaling, translation and rotation. arXiv preprint [arXiv:2202.07130](https://arxiv.org/abs/2202.07130) (2022)
17. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: AAAI, pp. 2181–2187 (2015)
18. Liu, H., Wu, Y., Yang, Y.: Analogical inference for multi-relational embeddings. In: ICML, vol. 70, pp. 2168–2178 (2017)
19. Luo, L., Xiong, Y., Liu, Y., Sun, X.: Adaptive gradient methods with dynamic bound of learning rate. In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6–9 2019(2019). <https://openreview.net/forum?id=Bkg3g2R9FX>
20. Luo, R., Li, J., Zhang, J., Xiao, J., Yang, Y.: Prior relational schema assists effective contrastive learning for inductive knowledge graph completion. In: Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), pp. 13014–13025 (2024)
21. Mahdisoltani, F., Biega, J., Suchanek, F.M.: YAGO3: a knowledge base from multilingual wikipedias. In: CIDR (2015)
22. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NeurIPS, pp. 3111–3119 (2013)
23. Mohammed, S., Shi, P., Lin, J.: Strong baselines for simple question answering over knowledge graphs with and without neural networks. In: NAACL-HLT, vol. 2, pp. 291–296 (2018)
24. Nickel, M., Rosasco, L., Poggio, T.A.: Holographic embeddings of knowledge graphs. In: AAAI, pp. 1955–1961 (2016)
25. Nickel, M., Tresp, V., Kriegel, H.: A three-way model for collective learning on multi-relational data. In: ICML, pp. 809–816 (2011)
26. Sun, Z., Deng, Z., Nie, J., Tang, J.: ROTATE: knowledge graph embedding by relational rotation in complex space. In: ICLR (2019)
27. Tang, Y., Huang, J., Wang, G., He, X., Zhou, B.: Orthogonal relation transforms with graph context modeling for knowledge graph embedding. In: ACL, pp. 2713–2722 (2020)
28. Toutanova, K., Chen, D.: Observed versus latent features for knowledge base and text inference. In: Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality, Beijing, China, pp. 57–66 (2015)

29. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: ICML, vol. 48, pp. 2071–2080 (2016)
30. Wang, F., Xiang, X., Cheng, J., Yuille, A.L.: NormFace: L_2 hypersphere embedding for face verification. In: ACM MM, pp. 1041–1049. ACM (2017)
31. Wang, H.: From Mathematics to Philosophy, Routledge Revivals. Routledge (2016)
32. Wang, L., Zhao, W., Wei, Z., Liu, J.: SimKGC: simple contrastive knowledge graph completion with pre-trained language models. arXiv preprint [arXiv:2203.02167](https://arxiv.org/abs/2203.02167) (2022)
33. Wang, S., et al.: Mixed-curvature multi-relational graph neural network for knowledge graph completion. In: WWW, pp. 1761–1771 (2021)
34. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph and text jointly embedding. In: EMNLP, pp. 1591–1601 (2014)
35. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: AAAI, pp. 1112–1119 (2014)
36. Xu, C., Li, R.: Relation embedding with dihedral group in knowledge graph. In: ACL, vol. 1, pp. 263–272 (2019)
37. Xu, C., Nayyeri, M., Chen, Y., Lehmann, J.: Knowledge graph embeddings in geometric algebras. In: COLING, pp. 530–544 (2020)
38. Yang, B., Yih, W., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. In: ICLR (2015)
39. Yao, L., Mao, C., Luo, Y.: KG-BERT: bert for knowledge graph completion. arXiv preprint [arXiv:1909.03193](https://arxiv.org/abs/1909.03193) (2019)
40. Yu, L., Luo, Z., Liu, H., Lin, D., Li, H., Deng, Y.: TripleRE: knowledge graph embeddings via tripled relation vectors. arXiv preprint [arXiv:2209.08271](https://arxiv.org/abs/2209.08271) (2022)
41. Zhang, F., Yuan, N.J., Lian, D., Xie, X., Ma, W.: Collaborative knowledge base embedding for recommender systems. In: KDD, pp. 353–362 (2016)
42. Zhang, S., Tay, Y., Yao, L., Liu, Q.: Quaternion knowledge graph embeddings. In: NeurIPS, pp. 2731–2741 (2019)
43. Zhang, Z., Cai, J., Wang, J.: Duality-induced regularizer for tensor factorization based knowledge graph completion. In: NeurIPS (2020)



Thinking Like an Author: A Zero-Shot Learning Approach to Keyphrase Generation with Large Language Model

Siyu Wang, Shengran Dai, and Jianhui Jiang^(✉)

Gusu Laboratory of Materials, Suzhou, China
{wangsiyu2022,daishengran2021,jiangjianhui2021}@gusulab.ac.cn

Abstract. Keyphrase generation aims to automatically derive a set of phrases from a given document. Recently, automated keyphrase generation has gained prominence as a focal point of research in both industry and academia, showcasing notable advancements in performance. This paper aims to employ a Large Language Model (LLM) for a more comprehensive extraction of keyphrases from documents. We observe that many article authors adhere to a four-step process when selecting keyphrases for a document. Initially, they extract keyphrases directly from the document. In addition, some authors will also choose hypernyms of the keyphrases selected in the previous stage as new extended keyphrases. Subsequently, these authors obtain keyphrases from documents with similar topics, opting for the most appropriate ones to include. Finally, the authors organize the keyphrases obtained from the extraction, extension, and retrieval steps, discerning the final keyphrases through a ranking process. Motivated by these observations, we introduce a zero-shot learning approach for keyphrase generation using the LLM, which includes four parts. The extractor, extender, and retriever components are responsible for recalling candidate keyphrases, while the ranker component is tasked with ranking them. Experimental results demonstrate the effectiveness of our approach on various datasets. Moreover, ablation experiments shed light on the impact of each module on the model's final performance.

Keywords: Keyphrases Generation · Zero-shot Learning · Large Language Model

1 Introduction

Keyphrase generation endeavors to automatically produce a collection of phrases derived from a given document. Typically, keyphrases are composed of multiple words, serving to succinctly encapsulate the primary content of the document and highlight noteworthy topics or information. Readers can gain a general understanding of document content by utilizing keyphrases without the

S. Wang and S. Dai—Equal Contribution.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024
A. Bifet et al. (Eds.): ECML PKDD 2024, LNAI 14943, pp. 335–350, 2024.
https://doi.org/10.1007/978-3-031-70352-2_20

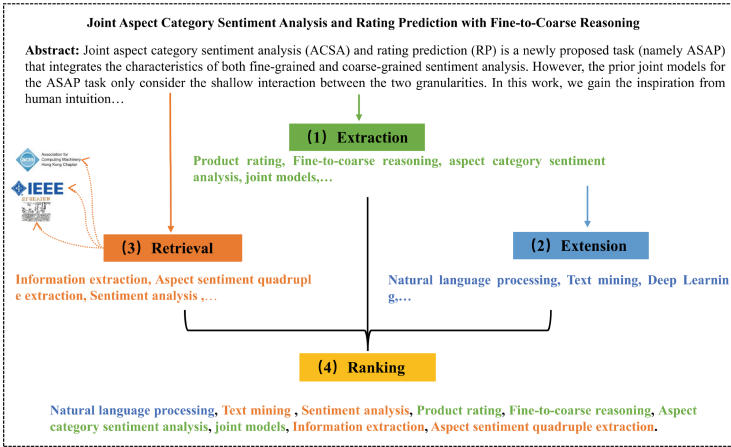


Fig. 1. Example of the article authors’ main steps for generating keyphrases. The orange color indicates keyphrases retrieved from similar documents, and the green color indicates keyphrases extracted from the source document, and the blue color indicates keyphrases extended by LLM.

need for detailed reading. Furthermore, these keyphrases can be applied in various downstream tasks, including information retrieval [16], text summarization [33], and document classification [14]. Recently, the automated generation of keyphrases has become a significant area of research attention in both industry and academia. Following the accessibility of text data in previous researches [7, 8, 19, 34], a wide range of datasets derived from scientific articles is commonly employed as benchmarks in keyphrase generation. Therefore, our study specifically concentrates on the generation of keyphrases from scientific articles.

Previous researches in keyphrase generation mainly include two types of methods: **Extraction** and **Generation**. The extraction methods [4–6, 20] mainly used graph algorithms or neural networks to extract keyphrases from documents. These methods are limited to identifying only the keyphrases that are explicitly present in the document (in this paper, we call them present keyphrases). In contrast, the generation methods [8, 9, 19, 34] mainly adopted sequence-to-sequence [29] (seq2seq) structure, these methods enable the generation of not only the keyphrases present in the document but also those absent ones (in this paper, we call it absent keyphrases), thus offering a more comprehensive range of keyphrases, and the generation methods have shown noteworthy advancements in performance. However, these methods necessitate an extensive number of document-keyphrase pairs for supervised training, acquiring such data is costly and labor-intensive [25]. Recently, LLMs, exemplified by ChatGPT, have exhibited remarkable performance across a diverse array of natural language tasks. They have achieved comparable or superior results when in contrast with their supervised counterparts potentially trained with millions of labeled examples, even in zero-shot setting [23]. [26] have conducted a prelimi-

nary evaluation of ChatGPT for the keyphrase generation task, they discovered that ChatGPT still faces challenges, particularly when it comes to generating absent keyphrases. This is mainly because many keyphrases of the document are abstract concepts, and LLMs cannot well understand these concepts from the document itself.

This paper aims to employ LLM for a more comprehensive generation of keyphrases from documents by zero-shot learning. We observe that article authors typically follow a four-step process (Extraction, Extension, Retrieval, and Ranking) when choosing keyphrases, as depicted in Fig. 1. First, they extract keyphrases from the document. At this stage, the keyphrases primarily consist of phrases present in the text. Then, some authors may choose hypernyms and synonyms of the keyphrases selected in the extraction stage as new keyphrases. This step is called Extension. For example, *Natural Language Processing* is a hypernym of *Aspect sentiment Quadruple Extraction*. In addition, article authors will also tend to refer some documents with similar contents in the same domain, and select the most suitable keyphrases to incorporate. During this phase, the identified keyphrases predominantly manifest as absent keyphrases. Ultimately, authors proceed to organize the keyphrases obtained from the previous steps and discern the final keyphrases through a ranking process.

Motivated by these observations, we propose a zero-shot learning approach with LLM for keyphrase generation, which includes four parts: **Extractor**, **Extender**, **Retriever** and **Ranker**. (1) The extractor is employed to directly extract possible keyphrases from the document, these keyphrases are called **extracted candidate keyphrases**. (2) The extender aims to obtain hypernyms and synonyms of extracted keyphrases, which is denoted as **extended candidate keyphrases**. (3) The retriever is tasked with obtaining keyphrases from documents similar to the current one, denoted as **retrieved candidate keyphrases**. (4) A novel ranker based on LLM is used to rank all candidate keyphrases and generate final document keyphrases.

Notably, each part is grounded in the zero-shot learning of LLM, thus eliminating the need for labeled data. Our contributions can be summarized into multiple aspects:

- Inspired by human behavior, we present a novel approach with LLM for keyphrase generation, which comprises four components. This method enhances keyphrase recall through a three-step process involving extraction, extension and retrieval followed by the application of a ranker to select the most pertinent keyphrases.
- In ranker, we propose a novel multi-turn selecting method based on LLM, which greatly improves the performance of ranking.
- We conduct experiments on four datasets and the experimental results show that our method outperforms all unsupervised methods as well as most supervised methods. While, the ablation experiment also shows the performance improvement brought by each component.

2 Related Work

2.1 Keyphrase Extraction

Many previous works were devoted to extracting keyphrases from documents, and most of them were unsupervised methods [2, 4–6, 20].

Specifically, these methods mainly included 2 main steps: extraction and ranking. For example, [4, 5, 20] used graph-based extraction and ranking methods. Recently, [6] rested on statistical text features extracted from single documents to select the most relevant keywords in a text.

In addition to unsupervised extraction methods, some research in recent years used supervised methods to complete keyphrases extraction tasks. [1] was approached as a sequence annotation task, where keyphrases in the document were annotated to train a neural network for keyphrase extraction. As pre-trained models have achieved great improvements in many tasks. Recently, some researchers [21, 27] have tried to use pre-trained models for keyphrase extraction. Although yielding favorable results, these methods encountered limitations in extracting absent keyphrases.

2.2 Keyphrase Generation

[19] pioneered a novel method for simultaneously generating present and absent keyphrases by integrating a sequence-to-sequence framework [29] with a copying mechanism [11]. This approach laid the foundation for many subsequent studies. For example, [8] considered that many studies neglect correlations among keyphrases, resulting in duplication and coverage issues, so they used coverage mechanisms and review mechanisms to address this problem. [34] introduced a dual copy mechanism to copy OOV words and seed words from the source text for keyphrase generation. [9] believed that titles play a very important role in keyphrase extraction, they proposed the title-guided network (TG-Net) for automatic keyphrase generation based on the encoder-decoder architecture. [7] introduced reinforcement learning based on a sequence-to-sequence architecture to encourage the generation of both sufficient and accurate keyphrases. Generally speaking, the seq2seq architecture is sensitive to the order of output keyphrases, but in practice, the output order of keyphrases does not affect its correctness, [35] proposed a novel model that utilized a fixed set of learned control codes as conditions to generate a set of keyphrases in parallel.

Besides the use of the seq2seq architecture, the research [30] in recent years has also used Adversarial Generation Networks (GAN) to generate keyphrases. Moreover, the emergence of ChatGPT has recently garnered significant attention from the computational linguistics community. [26] conducted a preliminary evaluation of ChatGPT for keyphrase generation to demonstrate its capabilities as a keyphrase generator.

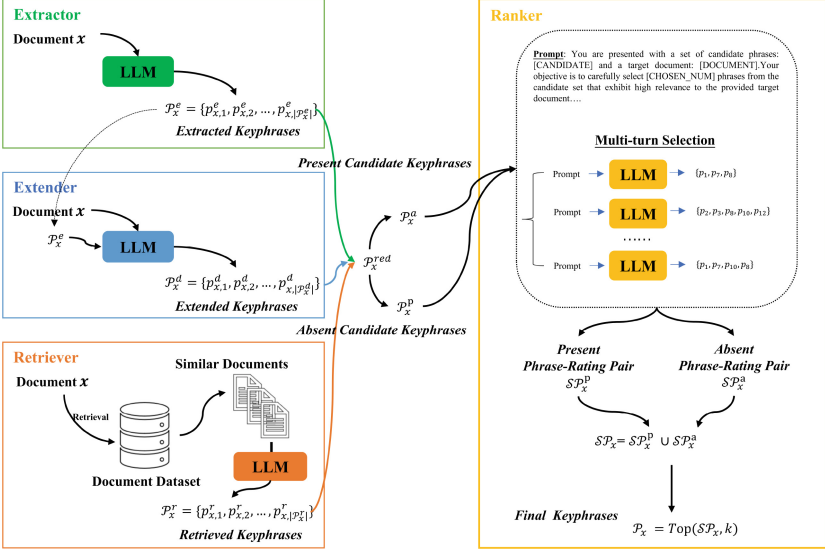


Fig. 2. Our approach for generating keyphrases.

3 Methodology

The keyphrase generation task is to generate a set of keyphrase \mathcal{P}_x from the document x . \mathcal{P}_x contains $|\mathcal{P}_x|$ keyphrases and denotes as $\mathcal{P}_x = \{p_{x,i}\}_{i=1}^{|\mathcal{P}_x|}$, where $p_{x,i}$ is one of \mathcal{P}_x .

Our method mainly consists of four parts: **Extractor**, **Extender**, **Retriever**, and **Ranker**, as shown in Fig. 2. Given a document x , we first use the extractor to obtain the extracted candidate keyphrases $\mathcal{P}_x^e = \{p_{x,i}^e\}_{i=1}^{|\mathcal{P}_x^e|}$. Then, an extender is used to obtain hypernyms and synonyms of \mathcal{P}_x^e , denoted as extended candidate keyphrases $\mathcal{P}_x^d = \{p_{x,i}^d\}_{i=1}^{|\mathcal{P}_x^d|}$. Next, the retriever is responsible for finding similar documents of x from the document database¹ \mathcal{D} and obtaining retrieved candidate keyphrases $\mathcal{P}_x^r = \{p_{x,i}^r\}_{i=1}^{|\mathcal{P}_x^r|}$. Finally, we design an LLM-based ranker to rank candidate keyphrases obtained by previous steps to get the final keyphrases \mathcal{P}_x .

3.1 Extractor

As previously discussed, authors commence the selection of keyphrases by extracting pertinent central phrases from the document. Aligned with this approach, we construct an extractor based on LLM. After inputting document x

¹ In this paper, we use the KP20k training data as the document dataset, we removed the labels for all data. KP20k dataset contains 509,818 training samples. Each sample from these datasets consists of the title, abstract, and keyphrases of a scientific article. Dataset is available at <https://huggingface.co/datasets/taln-ls2n/kp20k>.

and prompt, we use LLM to extract keyphrases, which are denoted as:

$$\mathcal{P}_x^e = LLM(x, \mathcal{TP}_1), \quad (1)$$

where \mathcal{TP}_1 is a prompt for LLM, which is shown in Table 1. \mathcal{P}_x^e is extracted candidate keyphrases, which mainly include present keyphrases and a small amount of absent keyphrases.

3.2 Extender

Through extraction, we have obtained the extracted candidate keyphrases \mathcal{P}_x^e , and then we expand the keyphrases in \mathcal{P}_x^e . As mentioned above, some authors will choose keyphrases that are hypernyms or synonyms of extracted keyphrases. Motivated by this observation, we use LLM to obtain the hyponyms or synonyms of keyphrases in \mathcal{P}_x^e , which is denoted as \mathcal{P}_x^d :

$$\mathcal{P}_x^d = LLM(x, \mathcal{P}_x^e, \mathcal{TP}_2), \quad (2)$$

where \mathcal{TP}_2 is a prompt for LLM, which is shown in Table 1.

Table 1. Three prompts are designed for chatting with LLM, where [DOCUMENT], [CANDIDATE], and [CHOSEN_NUM] are the filled contents.

Name	Prompts
\mathcal{TP}_1	Generate present and absent keywords from the following provided target document: [DOCUMENT] Each noun keyword should comprise 1 to 4 words. Ensure that these keywords are closely related to the content of the target document.
\mathcal{TP}_2	You are a hypernym and synonym keywords extender. Your task involves analyzing two sets of text: A target document, provided as: [DOCUMENT]. A set of keywords extracted from the document, provided as: [CANDIDATE]. Your task is to extend as many as possible common hypernym and synonym keywords of the extracted keywords that are not explicitly mentioned in the document text and not in the extracted keywords but closely align with its topic.
\mathcal{TP}_3	You are presented with a set of candidate phrases: [CANDIDATE] and a target document: [DOCUMENT]. Your objective is to carefully select [CHOSEN_NUM] phrases from the candidate set that exhibit high <u>relevance</u> to the provided target document.

3.3 Retriever

Authors will also tend to consult keyphrases from other documents with similar content, selecting pertinent ones to incorporate as their own. Most of these are synonyms of extracted candidate keyphrases. Given this, we built a retriever to fetch similar documents of x and obtain the keyphrases in these documents

Algorithm 1. Extracting keyphrases from retrieval documents

Input: input document x , document dataset \mathcal{D} , top k keyphrases, similarity threshold t
Output: \mathcal{P}_x^r

```

1:  $\mathbf{x} \leftarrow \text{Encoder}(x)$     ▷ In this paper, we use a pre-trained sentence transformer as
   our encoder. https://huggingface.co/sentence-transformers.
2: for  $d$  in  $\mathcal{D}$  do
3:    $\mathbf{d} \leftarrow \text{Encoder}(d)$ 
4:    $\text{sim} \leftarrow \text{cosine}(\mathbf{x}, \mathbf{d})$ 
5:   if  $\text{sim} > t$  then
6:      $\text{keyphrases} \leftarrow \text{LLM}(d, \mathcal{TP}_1)$     ▷ Extracting by equation (1).
7:     for  $k$  in  $\text{keyphrases}$  do
8:        $\mathcal{SP}_x^r[k] \leftarrow \mathcal{SP}_x^r[k] + 1$ 
9:     end for
10:  end if
11: end for
12:  $\mathcal{P}_x^r = \text{Top}(\mathcal{SP}_x^r, k)$     ▷ Selecting top  $k$  candidate keyphrases.

```

by Eq. (1). Intuitively, the significance of a keyphrase can be inferred from its frequency of occurrence in these similar documents. Therefore, we obtain the keyphrases while concurrently tallying the frequency of their occurrences and output a pair $\mathcal{SP}_x^r = \{(p_{x,i}^r, t_{x,i}^r)\}_{i=1}^{|\mathcal{SP}_x^r|}$, where $p_{x,i}^r$ is i^{th} keyphrase, and $t_{x,i}^r$ is the number of $p_{x,i}^r$ occurrences. Finally, we select the top k keyphrases from \mathcal{SP}_x^r that appear most frequently as retrieved candidate keyphrases \mathcal{P}_x^r . The specific algorithm is shown in Algorithm 1.

3.4 Ranker

Nowadays, ranking models based on LLM have been widely used in recommendation systems [23, 28], which mainly include two types: point-wise and list-wise approaches, and neither of them has yielded optimal results [12]. In contrast to the previous ones, in this paper, we introduce a relevance rating founded on multi-turn selection to assess the relevance of candidate keyphrases and documents. Finally, this relevance rating is utilized for ranking purposes, ultimately determining the final keyphrases \mathcal{P}_x .

The essence of the ranking process lies in utilizing LLM to obtain a relevance rating between the candidate keyphrase and the given document. Due to a single rating lacking objectivity and accuracy when using LLM, we propose a multi-turn selection approach based on LLM for relevance rating, details in Algorithm 2. We let LLM perform r turns selection, in each turn, LLM is tasked with selecting different numbers of keyphrases that are most relevant and irrelevant to x from the candidate set. Additionally, absent keyphrases typically represent broad and abstract terms, whereas present keyphrases are directly extracted from the document. LLM, consequently tends to focus more on present keyphrases. For a fair comparison, we separately evaluate absent and present keyphrases and subsequently aggregate the rating for the final ranking.

Algorithm 2. Relevance rating algorithm

Input: document x , candidate phrases $\mathcal{P}_x^{p/a}$ (the superscript p/a denotes the \mathcal{P}_x^p or \mathcal{P}_x^a), rank prompt \mathcal{TP}_3 , number of turns r , reward factor λ , penalty factor θ

Output: phrase-rating pair $\mathcal{SP}_x^{p/a}$

```

1: for  $i$  in  $r$  do
2:    $\text{Shuffle}(\mathcal{P}_x^{p/a})$   $\triangleright$  Shuffle the order of candidate keyphrases.
3:    $\text{RelevancePhraseList} \leftarrow \text{LLM}(x, \mathcal{P}_x^{p/a}, \text{chosen\_num}, \mathcal{TP}_3)$   $\triangleright$  Each turn, we
   vary sample numbers  $\text{chosen\_num}$ .
4:    $\text{UnrelevancePhraseList} \leftarrow \text{LLM}(x, \mathcal{P}_x^{p/a}, \text{chosen\_num}, \mathcal{TP}_3)$   $\triangleright$  Selecting the
   irrelevant keyphrases, where we replace "relevance" in  $\mathcal{TP}_3$  with "irrelevance".
5:   for  $\text{phrase}$  in  $\text{RelevancePhraseList}$  do
6:      $\mathcal{SP}_x^{p/a}[\text{phrase}] \leftarrow \mathcal{SP}_x^{p/a}[\text{phrase}] + \lambda$ 
7:   end for
8:   for  $\text{phrase}$  in  $\text{UnrelevancePhraseList}$  do
9:      $\mathcal{SP}_x^{p/a}[\text{phrase}] \leftarrow \mathcal{SP}_x^{p/a}[\text{phrase}] - \theta$   $\triangleright$   $\lambda$  and  $\theta$  are reward factor and
   penalty factor.
10:  end for
11: end for
12: for  $\text{phrase}$  in  $\mathcal{SP}_x^{p/a}$  do
13:    $\mathcal{SP}_x^{p/a}[\text{phrase}] \leftarrow \mathcal{SP}_x^{p/a}[\text{phrase}] / \text{sum}(\mathcal{SP}_x^{p/a})$ 
14: end for

```

Specifically, we first combine extracted candidate keyphrases \mathcal{P}_x^e , retrieved candidate keyphrases \mathcal{P}_x^r and extended candidate keyphrases \mathcal{P}_x^d into a new set \mathcal{P}_x^{red} . Then, we split \mathcal{P}_x^{red} into two sets \mathcal{P}_x^p and \mathcal{P}_x^a according to whether it appears in x . Next, we calculate relevance rating \mathcal{P}_x^p and \mathcal{P}_x^a separately by Algorithm 2, output phrase-rating pairs $\mathcal{SP}_x^p = \{(p_{x,i}^p, s_{x,i}^p)\}_{i=1}^{|\mathcal{SP}_x^p|}$ and $\mathcal{SP}_x^a = \{(p_{x,i}^a, s_{x,i}^a)\}_{i=1}^{|\mathcal{SP}_x^a|}$, where $p_{x,i}^{p/a}$ is i^{th} keyphrase, and $s_{x,i}^{p/a}$ is the relevance rating of $p_{x,i}^{p/a}$. Further, we softly merge the two pairs and take the top k keyphrases with the highest relevance rating as the final outputs \mathcal{P}_x , which is computed by:

$$\mathcal{P}_x = \text{Top}(\mathcal{SP}_x, k), \quad \mathcal{SP}_x = \alpha \odot \mathcal{SP}_x^p \cup \beta \odot \mathcal{SP}_x^a, \quad (3)$$

where \odot represents the weights α and β multiplied by all values of \mathcal{SP}_x^p and \mathcal{SP}_x^a , which is expressed as:

$$\alpha \odot \mathcal{SP}_x^p = \{(p_{x,i}^p, \alpha \cdot t_{x,i}^p)\}_{i=1}^{|\mathcal{SP}_x^p|}, \beta \odot \mathcal{SP}_x^a = \{(p_{x,i}^a, \beta \cdot t_{x,i}^a)\}_{i=1}^{|\mathcal{SP}_x^a|}. \quad (4)$$

Generally speaking, the number of present keyphrases in the document is greater than the absent keyphrases, to ensure better output performance. In Equation (4), we adjust the weight of the absent and present keyphrases through hyperparameter α, β . For example, if we want more keyphrases from \mathcal{SP}_x^p , we can increase the value of α .

4 Experimental Setup

4.1 Datasets

Table 2. Statistics of the testing set on four datasets. #SAM: the average number of samples, avg.#KP: the number of keyphrases, |KP|: the average length of keyphrase, %AKP: the proportion of absent keyphrases.

Dataset	#SAM	avg.#KP	KP	%AKP
NUS	211	10.81	2.22	45.36
Krapivin	400	5.83	2.21	44.33
SemEval	100	14.43	2.38	55.61
Inspec	500	9.79	2.48	26.42

We perform experiments with four datasets comprised of scientific articles, including NUS [22], Krapivin [18], Semeval [17] and Inspec [13]. Each one comprises a title, an abstract, and a set of keyphrases. The test dataset statistics are shown in Table 2.

In contrast to prior studies [7, 8, 19, 34], our approach is rooted in the LLM-based zero-shot learning method, eliminating the necessity of training data for the model.

4.2 Baselines

We choose three categories of baselines for comparative analysis: unsupervised methods, supervised methods, and LLMs-based zero-shot learning methods.

- **Unsupervised methods:** **Yake** [6] is resting on statistical text features extracted from single documents to select the most relevant keyphrase of a document. **TopicRank** [5] and **TextRank** [20] are graph-based extraction and ranking methods.
- **Supervised methods:** We choose the model **cat2seq** [19] based on seq2seq which is the first model to generate present and absent keyphrases. The model **cat2seqTG-2RF** [7] is introducing reinforcement learning based on a sequence-to-sequence architecture to encourage the generation of both sufficient and accurate keyphrases. The adversarial generative network model **GAN_{MR}** [30] and the transformer-based model **ONE2SET** [35], which are utilizing a fixed set of learned control codes as conditions to generate a set of keyphrases in parallel.
- **LLM-based zero-shot learning methods:** We select the latest large models (**ChatGPT**², **Vicuna7B** [10] and **Mistral7B** [15]) as our baselines. In these baselines, we obtain the keyphrases of the document directly extract through the prompt. For a fair comparison, all LLMs use the same prompt TP_1 .

² <https://chat.openai.com/>.

4.3 Implementation Details

In this paper, we choose Openchat 7B [31], a fine-tuned model based on Mistral 7B [15], to build our extractor, extender, retriever, and ranker. Openchat³ learns from mixed-quality data without preference labels, delivering exceptional performance on par with ChatGPT, even with a 7B model which can be run on a consumer GPU⁴

Extractor and Extender. We fill the source document into the \mathcal{TP}_1 , and make a query to the LLM to get the extracted keyphrases. We then put them together with the source document into \mathcal{TP}_2 , and make a query to LLM to get the extended keyphrases.

Retriever. We utilize the training data (excluding labels) of KP20K [19] as the document database \mathcal{D} . A Sentence-Transformer [24] is used to map each document in \mathcal{D} to a 768 dimensional representation, and then insert it into the vector database Milvus [32] along with original document. When a source document is provided, firstly, we convert it to a vector with Sentence-Transformer, then, retrieve similar documents above a threshold t in Milvus. In this paper, the t is set to 0.8. In particular, we use a filtering mechanism to prevent the retrieval of identical documents. After getting similar documents, we use the extractor to get the keyphrases from all similar documents, respectively. By counting the frequency of these keyphrases and ranking them, we choose the top $k = 10$ as the final retrieved keyphrases.

Considering the absence of retrieved similar documents in some cases, we design a polish mechanism⁵ that using LLM to rewrite the source document, thus the similar documents generated.

Ranker. In the experiment, we prompt LLM for multiple turns with randomly shuffled candidate keyphrases and varying sample numbers to select the most or least relevant phrases (we use Table 1 \mathcal{TP}_3 as positive prompt, and we replace “relevance” in \mathcal{TP}_3 with “irrelevance” as negative prompt). When a keyphrase is deemed relevant, its rating will be augmented by a reward factor. Conversely, a penalty factor will be applied if the LLM considers it irrelevant. In this paper, number of turns r , reward factor λ , and penalty factor θ are 5, 1.0, and 0.5, respectively. Finally, α and β are introduced to adjust the weights for present

³ <https://github.com/imoneoi/openchat>.

⁴ Our code is available at <https://github.com/syogo/Zero-Shot-Learning-Keyphrase-Generator>.

⁵ The prompt for polishing documents: “Below is a paragraph from an academic paper. Polish the writing to meet the academic style, and improve the spelling, grammar, clarity, concision and overall readability. When necessary, rewrite the whole sentence. Paragraph: [DOCUMENT] Note that you should use synonyms or hypernym phrases to replace something in the text. Try your best to use a variety of expressions. Please response the answer directly without any other irrelevant words”.

and absent keyphrases. For example, on Krapivin dataset, α and β are 10.0 and 1.0, respectively.

4.4 Evaluation Metrics

In alignment with prior studies [19, 34–36], we employ macro-averaged F1@5, F1@ \mathcal{M} and Recall@ \mathcal{O} metrics for the prediction of both present and absent keyphrases. When using F1@5, blank keyphrases are added to make the keyphrase number reach five if the prediction number is less than five. In F1@ \mathcal{M} , \mathcal{M} denotes the number of ground truth keyphrases. In F1@ \mathcal{O} , \mathcal{O} denotes the number of predicted keyphrases. In this case, $\mathcal{O} = |\mathcal{P}_x|$ and we simply take all the predicted phrases for evaluation without truncation.

We employ the Porter Stemmer⁶ to ascertain the identity of two keyphrases, subsequently eliminating any duplicated keyphrases post-stemming.

Table 3. * represents the baseline of our implementation by PKE [3]. † denotes zero-shot learning methods. We highlight the best performance among zero-shot learning methods in bold.

Present Keyphrases Generation								
Models	NUS		Krapivin		SemEval		Inspec	
	F1@5	F1@M	F1@5	F1@M	F1@5	F1@M	F1@5	F1@M
Yake*	0.222	0.234	0.189	0.179	0.200	0.204	0.183	0.186
TopicRank*	0.222	0.214	0.162	0.157	0.194	0.206	0.277	0.299
TextRank*	0.090	0.123	0.117	0.113	0.087	0.148	0.326	0.369
CatSeq	0.323	0.397	0.269	0.354	0.242	0.283	0.225	0.262
CatSeqTG+RF	0.375	0.433	0.300	0.369	0.287	0.329	0.253	0.301
GAM _{mr}	0.348	0.417	0.288	0.369	–	–	0.258	0.299
ONE2SET	0.406	0.450	0.326	0.364	0.331	0.357	0.285	0.324
ChatGPT†	0.196	0.248	0.222	0.211	0.168	0.242	0.334	0.384
Mistral7B †	0.211	0.255	0.223	0.212	0.173	0.246	0.305	0.373
Vicuna7B†	0.184	0.225	0.222	0.220	0.128	0.191	0.253	0.317
Ours	0.291	0.268	0.253	0.243	0.256	0.253	0.413	0.415
Absent Keyphrases Generation								
Models	NUS		Krapivin		SemEval		Inspec	
	F1@5	F1@M	F1@5	F1@M	F1@5	F1@M	F1@5	F1@M
CatSeq	0.016	0.028	0.018	0.036	0.016	0.028	0.004	0.008
CatSeqTG+RF	0.019	0.031	0.030	0.053	0.021	0.030	0.012	0.021
GAM _{mr}	0.026	0.038	0.042	0.057	–	–	0.013	0.019
ONE2SET	0.042	0.060	0.047	0.073	0.026	0.034	0.021	0.034
ChatGPT †	0	0	0.002	0.002	0.002	0.001	0.005	0.005
Mistral7B †	0	0.002	0.005	0.005	0.006	0.005	0.009	0.012
Vicuna7B †	0.002	0.006	0.003	0.002	0	0.001	0.004	0.005
Ours	0.002	0.012	0.005	0.005	0.003	0.009	0.004	0.012

⁶ <https://www.nltk.org/howto/stem.html>.

5 Results and Analysis

5.1 Present and Absent Keyphrase Generation

Table 3 presents the performance of various models in generating both present and absent keyphrases across different datasets. In the prediction of present keyphrases, compared to unsupervised methods, our model achieves substantial improvements on each dataset. On the other hand, among the LLM-based zero-shot learning methods (including ChatGPT, Mistral7B, and Vicuna7B), we also obtain the best results on four datasets. Overall, the gap between our method and supervised methods in the present keyphrases prediction task is not obvious, and on some datasets, it even exceeds all supervised methods.

In the task of predicting absent keyphrases, unsupervised methods cannot extract absent keyphrases, so we only compare with supervised methods and LLM-based zero-shot learning methods. The generation of absent keyphrases proves to be a highly challenging task, leading to suboptimal performance across all models. Our model exhibits superior performance when compared to zero-shot learning methods across all datasets. Nevertheless, it is essential to acknowledge that our model still experiences a substantial performance gap when compared to supervised models in this challenging task.

In summary, our method surpasses all unsupervised and partially supervised models, outperforming zero-shot learning methods based on LLMs across most of datasets. Compared with generating present keyphrases, generating absent keyphrases is more challenging. Experiments have proved that we have achieved better results compared to zero-shot learning methods in the task of generating absent keyphrases.

5.2 Ablation Study

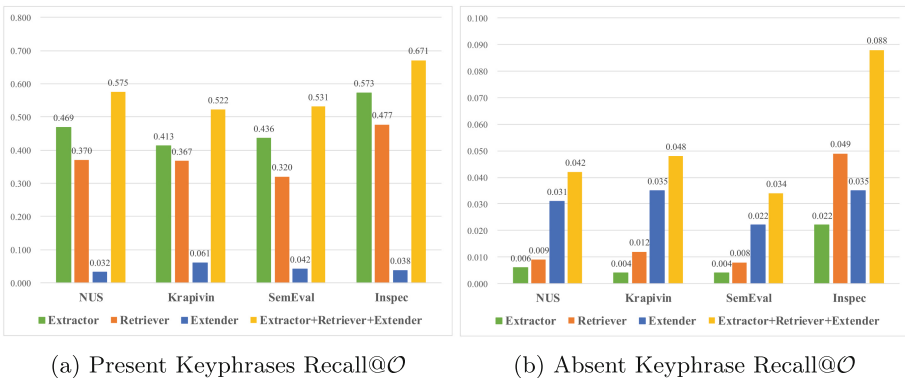


Fig. 3. The Recall@0 of the extractor, extender and retriever on different datasets.

Effect of Extractor, Extender and Retriever. We believe that the main aim of extractor, extender, and retriever is to recall keyphrases as much as possible and provide high-quality candidate phrases for subsequent ranking. The results of the ablation experiment are depicted in Fig. 3, revealing that all three components successfully have recalled a substantial number of candidate keyphrases. Notably, the extractor demonstrates exceptional performance by recalling more than half of the present keyphrases across the majority of datasets. Our observations further highlight the extractor’s proficiency in recalling present keyphrases, while the retriever and extender exhibit greater efficacy in recalling absent keyphrases. This result aligns seamlessly with our overarching goal of acquiring a diverse spectrum of candidate keyphrases through the collaborative synergy of the extractor, extender, and retriever.

Table 4. The impact of the ranker on keyphrases generation.

Present Keyphrase Generation								
Models	NUS		Krapivin		SemEval		Inspec	
	F@5	F@M	F@5	F@M	F@5	F@M	F@5	F@M
<i>w/o Ranker</i>	0.145	0.191	0.141	0.152	0.114	0.182	0.189	0.248
Ours	0.291	0.268	0.253	0.243	0.256	0.253	0.413	0.415
Absent Keyphrase Generation								
Models	NUS		Krapivin		SemEval		Inspec	
	F@5	F@M	F@5	F@M	F@5	F@M	F@5	F@M
<i>w/o Ranker</i>	0.016	0.015	0.006	0.006	0.006	0.013	0.018	0.015
Ours	0.002	0.012	0.005	0.005	0.003	0.009	0.004	0.012

Effect of Ranker. The primary role of the ranker is to prioritize the keyphrases obtained by extractor, extender and retriever according to their relevance to the source document. In this section, we compare the results before and after ranking, as shown in Table 4. F@5 and F@M metric for each dataset exhibits significant enhancement after the ranking process, particularly in the realm of present keyphrase prediction. However, in the absent keyphrases generation task, the effect after ranking has not been greatly improved, and some datasets have also declined to a certain extent. This is mainly because there are very few absent keyphrases candidates in the recall stage, so the ranker ranks more relevant present keyphrases to the front. Therefore, the performance of present keyphrases generation is greatly improved and the performance of absent keyphrases generation is reduced.

Effect of Multi-turn Selection Approach. In ranker, we first let LLM select multiple turns from the candidate keyphrases to confirm the relevance rating of

Table 5. The impact of different ranker strategies on present keyphrase generation. *w/o MS* means the multi-turn selection approach is not used in the model, namely $r = 1$. *w/o IRQ* denotes that each turn of selection in the ranker only picks the most relevant keyphrases, but the most irrelevant keyphrases are not considered.

	NUS						SemEval					
	P@5	R@5	F1@5	P@MR	MF1@M		P@5	R@5	F1@5	P@MR	MF1@M	
<i>w/o MS</i>	0.276	0.269	0.272	0.201	0.369	0.260	0.234	0.190	0.210	0.163	0.369	0.226
<i>w/o IRQ</i>	0.176	0.167	0.171	0.162	0.295	0.209	0.164	0.139	0.150	0.135	0.308	0.187
Ours	0.297	0.286	0.291	0.208	0.376	0.268	0.280	0.236	0.256	0.181	0.418	0.253
	Krapivin						Inspec					
	P@5	R@5	F1@5	P@MR	MF1@M		P@5	R@5	F1@5	P@MR	MF1@M	
<i>w/o MS</i>	0.098	0.148	0.117	0.106	0.171	0.131	0.451	0.375	0.410	0.379	0.524	0.440
<i>w/o IRQ</i>	0.147	0.238	0.182	0.141	0.243	0.178	0.267	0.225	0.244	0.253	0.348	0.293
Ours	0.207	0.328	0.254	0.193	0.329	0.243	0.454	0.379	0.413	0.358	0.493	0.415

the keyphrases and source document. To verify the effectiveness of multi-turn selection, we conduct an ablation experiment to compare the ranking effects of selecting different numbers of turn r . In experiments, we set $r = 1$ and $r = 5$. The present keyphrases generation results are shown in Table 5. It can be seen that after multi-turn selection, the model can obtain better results. On each dataset, recall and precision increase significantly through the multi-turn selection approach.

Effect of Prompt in Ranker. In each turn, LLM is tasked with selecting a list of keyphrases that are most relevant and irrelevant to the source document from the candidates. The purpose of asking questions from both relevant and irrelevant aspects is that LLM can evaluate the relevance of candidate keyphrases from two perspectives. To verify the impact of questioning from two perspectives on ranking performance, we conduct an ablation experiment. In Table 5, we report the ranking results after removing irrelevant prompts, it can be seen that the effect of rating from only one aspect is far lower than the effect of ranking from two aspects.

6 Conclusion

In this paper, we propose a zero-shot learning approach to keyphrase generation with LLM, including four parts. Extractor, extender, and retriever are responsible for recalling candidate keyphrases, and the ranker is responsible for ranking them. Experiments show that our method surpasses ChatGPT and unsupervised methods in most datasets, and even surpasses supervised methods in some datasets and achieves good results. In addition, ablation experiments also demonstrate the impact of each module on the final performance.

However, our approach also has some limitations. The prompt has a great impact on the effect of our method. In addition, we have not yet reached the performance of supervised models in some datasets.

Acknowledgments. This study was supported by the National Key Research and Development Program of China (2022YFF0707103).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Alzaidy, R., Caragea, C., Giles, C.L.: Bi-lstm-crf sequence labeling for keyphrase extraction from scholarly documents. In: WWW, pp. 2551–2557 (2019)
2. Bennani-Smires, K., Musat, C., Hossmann, A., Baeriswyl, M., Jaggi, M.: Simple unsupervised keyphrase extraction using sentence embeddings. In: CoNLL, pp. 221–229 (2018)
3. Boudin, F.: pke: an open source python-based keyphrase extraction toolkit. In: COLING, pp. 69–73 (2016)
4. Boudin, F.: Unsupervised keyphrase extraction with multipartite graphs. In: NAACL, pp. 667–672 (2018)
5. Bougouin, A., Boudin, F., Daille, B.: Topicrank: graph-based topic ranking for keyphrase extraction. In: IJCNLP, pp. 543–551 (2013)
6. Campos, R., Mangaravite, V., Pasquali, A., Jorge, A., Nunes, C., Jatowt, A.: Yake! keyword extraction from single documents using multiple local features. *Inf. Sci.* **509**, 257–289 (2020)
7. Chan, H.P., Chen, W., Wang, L., King, I.: Neural keyphrase generation via reinforcement learning with adaptive rewards. arXiv preprint [arXiv:1906.04106](https://arxiv.org/abs/1906.04106) (2019)
8. Chen, J., Zhang, X., Wu, Y., Yan, Z., Li, Z.: Keyphrase generation with correlation constraints. In: EMNLP, pp. 4057–4066 (2018)
9. Chen, W., Gao, Y., Zhang, J., King, I., Lyu, M.R.: Title-guided encoding for keyphrase generation (2019)
10. Chiang, W.L., et al.: Vicuna: an open-source chatbot impressing gpt-4 with 90%* chatgpt quality (2023). See <https://vicunalmsys.org>. Accessed 14 April 2023
11. Gu, J., Lu, Z., Li, H., Li, V.O.: Incorporating copying mechanism in sequence-to-sequence learning. In: ACL, pp. 1631–1640 (2016)
12. Hou, Y., Zhang, J., Lin, Z., Lu, H., Xie, R., McAuley, J., Zhao, W.X.: Large language models are zero-shot rankers for recommender systems. arXiv preprint [arXiv:2305.08845](https://arxiv.org/abs/2305.08845) (2023)
13. Hulth, A.: Improved automatic keyword extraction given more linguistic knowledge. In: EMNLP, pp. 216–223 (2003)
14. Hulth, A., Megyesi, B.: A study on automatically extracted keywords in text categorization. In: COLING-ACL, pp. 537–544 (2006)
15. Jiang, A.Q., et al.: Mistral 7b. arXiv preprint [arXiv:2310.06825](https://arxiv.org/abs/2310.06825) (2023)
16. Jones, S., Staveley, M.S.: Phrasier: a system for interactive document retrieval using keyphrases. In: SIGIR, pp. 160–167 (1999)
17. Kim, S.N., Medelyan, O., Kan, M.Y., Baldwin, T.: SemEval-2010 task 5: automatic keyphrase extraction from scientific articles. In: SemEval, pp. 21–26 (2010)

18. Krapivin, M., Autaeu, A., Marchese, M.: Large dataset for keyphrases extraction (2009)
19. Meng, R., Zhao, S., Han, S., He, D., Brusilovsky, P., Chi, Y.: Deep keyphrase generation. In: ACL, pp. 582–592 (2017)
20. Mihalcea, R., Tarau, P.: Textrank: bringing order into text. In: EMNLP, pp. 404–411 (2004)
21. Mu, F., et al.: Keyphrase extraction with span-based feature representations. arXiv preprint [arXiv:2002.05407](https://arxiv.org/abs/2002.05407) (2020)
22. Nguyen, T.D., Kan, M.-Y.: Keyphrase extraction in scientific publications. In: Goh, D.H.-L., Cao, T.H., Sølvberg, I.T., Rasmussen, E. (eds.) ICADL 2007. LNCS, vol. 4822, pp. 317–326. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-77094-7_41
23. Qin, Z., et al.: Large language models are effective text rankers with pairwise ranking prompting. arXiv preprint [arXiv:2306.17563](https://arxiv.org/abs/2306.17563) (2023)
24. Reimers, N., Gurevych, I.: Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In: Inui, K., Jiang, J., Ng, V., Wan, X. (eds.) EMNLP-IJCNLP, pp. 3982–3992. Association for Computational Linguistics, Hong Kong, China, November (2019)
25. Shen, X., Wang, Y., Meng, R., Shang, J.: Unsupervised deep keyphrase generation. In: AAAI, vol. 36, pp. 11303–11311 (2022)
26. Song, M., et al.: Is chatgpt a good keyphrase generator? a preliminary study. arXiv preprint [arXiv:2303.13001](https://arxiv.org/abs/2303.13001) (2023)
27. Sun, S., Liu, Z., Xiong, C., Liu, Z., Bao, J.: Capturing global informativeness in open domain keyphrase extraction. In: NLPCC. pp. 275–287. Springer (2021)
28. Sun, W., Yan, L., Ma, X., Ren, P., Yin, D., Ren, Z.: Is chatgpt good at search? investigating large language models as re-ranking agent. arXiv preprint [arXiv:2304.09542](https://arxiv.org/abs/2304.09542) (2023)
29. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. NIPS **27** (2014)
30. Swaminathan, A., Zhang, H., Mahata, D., Gosangi, R., Shah, R., Stent, A.: A preliminary exploration of gans for keyphrase generation. In: EMNLP. pp. 8021–8030 (2020)
31. Wang, G., Cheng, S., Zhan, X., Li, X., Song, S., Liu, Y.: Openchat: Advancing open-source language models with mixed-quality data. arXiv preprint [arXiv:2309.11235](https://arxiv.org/abs/2309.11235) (2023)
32. Wang, J., Yi, X., Guo, R., Jin, H., Xu, P., Li, S., Wang, X., Guo, X., Li, C., Xu, X., et al.: Milvus: A purpose-built vector data management system. In: SIGMOD. pp. 2614–2627 (2021)
33. Wang, L., Cardie, C.: Domain-independent abstract generation for focused meeting summarization. In: ACL. pp. 1395–1405 (2013)
34. Wang, S., Jiang, J., Huang, Y., Wang, Y.: Automatic keyphrase generation by incorporating dual copy mechanisms in sequence-to-sequence learning. In: COLING. pp. 2328–2338 (2022)
35. Ye, J., Gui, T., Luo, Y., Xu, Y., Zhang, Q.: One2set: Generating diverse keyphrases as a set. In: ACL-IJCNLP. pp. 4598–4608 (2021)
36. Yuan, X., Wang, T., Meng, R., Thaker, K., Brusilovsky, P., He, D., Trischler, A.: One size does not fit all: Generating and evaluating variable number of keyphrases. In: ACL. pp. 7961–7975 (2020)



Molecular Graph Representation Learning via Structural Similarity Information

Chengyu Yao^{1,2}, Hong Huang^{1,3}, Hang Gao^{1,2}, Fengge Wu^{1,2(✉)},
Haiming Chen³, and Junsuo Zhao^{1,2}

¹ University of Chinese Academy of Sciences, Beijing 100081, China

{yaochengyu2023, gaohang, fengge, junsuo}@iscas.ac.cn, huanghong@ios.ac.cn

² National Key Laboratory of Space Integrated Information System, Institute of Software Chinese Academy of Sciences, Beijing 100081, China

³ Key Laboratory of System Software (Chinese Academy of Sciences) and State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China
chm@ios.ac.cn

Abstract. Graph Neural Networks (GNNs) have been widely employed for feature representation learning in molecular graphs. Therefore, it is crucial to enhance the expressiveness of feature representation to ensure the effectiveness of GNNs. However, a significant portion of current research primarily focuses on the structural features within individual molecules, often overlooking the structural similarity between molecules, which is a crucial aspect encapsulating rich information on the relationship between molecular properties and structural characteristics. Thus, these approaches fail to capture the rich semantic information at the molecular structure level. To bridge this gap, we introduce the **Molecular Structural Similarity Motif GNN (MSSM-GNN)**, a novel molecular graph representation learning method that can capture structural similarity information among molecules from a global perspective. In particular, we propose a specially designed graph that leverages graph kernel algorithms to represent the similarity between molecules quantitatively. Subsequently, we employ GNNs to learn feature representations from molecular graphs, aiming to enhance the accuracy of property prediction by incorporating additional molecular representation information. Finally, through a series of experiments conducted on both small-scale and large-scale molecular datasets, we demonstrate that our model consistently outperforms eleven state-of-the-art baselines. The codes are available at <https://github.com/yaoyao-yaoyao-cell/MSSM-GNN>.

Keywords: Molecular property prediction · Graph neural networks · Graph representation learning · Graph kernel

C. Yao and H. Huang—These authors contributed equally to this work.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70352-2_21.

1 Introduction

Molecular Representation Learning, a critical discipline in bioinformatics and computational chemistry, has witnessed significant advancements in recent years [11, 22, 33]. Accurate prediction of molecular properties and activities is essential for drug discovery [17], toxicity assessment [46], and other biochemical applications [30]. Nowadays, molecular representation learning has been widely integrated with Graph Neural Networks (GNNs), which are powerful tools for processing graph data and have been successfully applied in the molecular domain [4, 15, 45]. However, most existing GNNs use the basic molecular graphs topology to obtain structural information through neighborhood feature aggregation and pooling methods [12, 21, 23]. This leads them to overlook the comprehensive chemical semantics.

To address this challenge, several emerging approaches have been proposed around molecular graphs. Specifically, some approaches [37, 45] take only the atom-level or motif-level information in heterogeneous molecular graphs as GNNs' input to recognize common subgraphs with special meanings. By identifying the significance of ring compounds in molecular structures, Zhu et al. [48] propose the Ring-Enhanced Graph Neural Network (\mathcal{O} -GNN). Alternatively, other methods [13, 18, 43] represent the molecular using motif-aware models that consider properties of domain-specific motifs. Furthermore, there exists a multitude of techniques [1, 25, 44] that center their focus on studying the relations among substructures to recognize critical patterns hidden in motifs and improve the reliability of molecular property prediction.

Despite the considerable progress compared to traditional GNNs, most recent studies focus only on the message passing within individual molecules. The relationships between molecular structures are often ignored, which may result in the partial loss of semantic information. Moreover, the functions and properties of chemical molecules largely depend on their structures [42]. For instance, consider examples illustrated in Fig. 1. Molecules with similar structures often have similar properties. Therefore, we need to take specific measures to represent the structural similarity between molecules, which can benefit downstream tasks such as molecular property prediction.

Based on the above-mentioned considerations, we design a **Molecular Structural Similarity Motif** (MSSM) graph that empowers GNNs to capture the rich structural and semantic information from inter-molecule. The design starts by constructing a nested motif dictionary to re-represent molecular graphs. In light of the diverse node types present in motif-based molecular graphs, we propose a Mahalanobis **W**eisfeiler-**L**ehman **S**hortest-**P**ath (MWLSP) graph kernel. This kernel is designed to assess structural similarity from both the perspectives of length and position. It overcomes the limitation of the shortest path graph kernel [2], which only retains connectivity information. By leveraging label information from different nodes and their neighbors, it provides a more granular representation of the graph, enhancing its expressiveness.

In this work, we propose a method that effectively considers inter-molecule structural similarity from a global perspective without sacrificing information

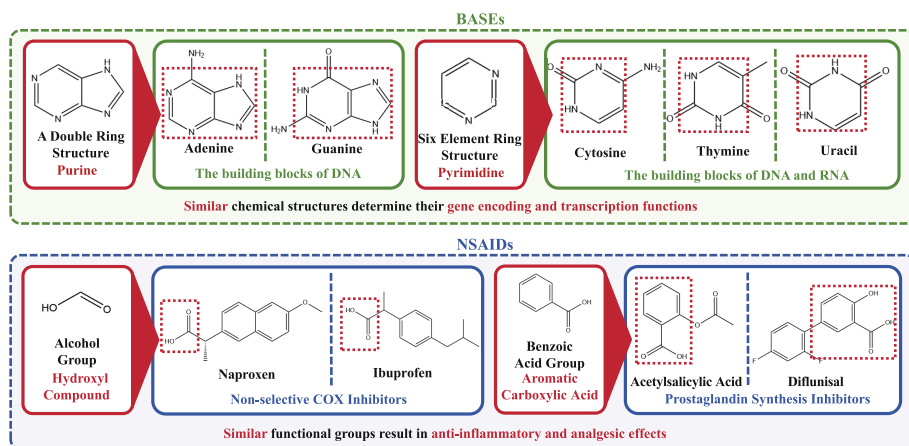


Fig. 1. Examples of molecules with similar structures often exhibit similar properties, a phenomenon observed in biological and chemical domains.

in individual molecules. The method consists of the following major components: Firstly, it extracts motifs from molecules to create the motif dictionary and represents each molecule by utilizing the dictionary. Secondly, it uses our proposed Molecular Structural Similarity Motif (MSSM) graph to exploit rich semantic information from graph motifs. Finally, it applies GNNs to learn compositional and structural feature representation for each molecular graph and their similarities based on the MSSM graph. The experimental results show that our model can significantly outperform other state-of-the-art GNN models on various molecular property prediction datasets.

To summarize, our contributions are as follows:

- Considering the actual molecule structure, we designed a novel molecular graph representation method to represent motif structural information.
- To improve the accuracy of GNNs in molecular property prediction tasks, we design a MSSM graph by employing the MWLSP graph kernel. It quantifies the similarity between molecules through graph kernel scores and obtains a more comprehensive semantic representation at the structural level.
- We show in the experiments that our model empirically outperforms state-of-the-art baselines on several benchmarks of real-world molecule datasets.

2 Related Work

2.1 Motifs in Molecular Graphs

Motif refers to the basic structure that constitutes any characteristic sequence. It can be viewed as a subgraph with a specific meaning in the molecular graph. For example, the edges in a molecular graph represent chemical bonds, and the rings

represent the molecular ring structure. Several algorithms have been introduced to leverage motifs in different applications, including contrastive learning [32], self-supervised pretraining [47], generation [19] and drug-drug interaction prediction [17]. The motif extraction techniques used in the above methods, whether relying on exact counting [5] or sampling and statistical estimation [36], have not utilized the structural similarities among motifs to enhance the expressiveness of molecular graphs.

2.2 Molecular Graph Representation Learning

DL has been widely applied to predict molecular properties. Molecules are usually represented as 1D sequences, including amino acid sequences, SMILES [41] and 2D graphs [11]. Wu et al. [38] proposed a new molecular joint representation learning framework, MMSG, based on multi-modal molecular information (from SMILES and graphs). However, these approaches cannot capture the rich information in subgraphs or graph motifs. A few works based on GNNs have been reported to leverage motif-level information. Specifically, some approaches [37, 45, 48] introduced the molecular graph representation learning method by constructing heterogeneous motif graphs from extracting different types of motifs. Alternatively, other methods [26, 48] decomposed each training molecule into fragments by breaking bonds and rings in compounds to design novel GNN variants. Although these methods obtain more expressive molecular graphs, the challenge in motif-based approaches mainly comes from the difficulty in efficiently measuring similarities between input graphs. While existing graph kernel methods [2, 7, 14, 31] can calculate scores by comparing different substructures of graphs to complete the measurement, there is currently no comparison method for motif-based molecular graphs. Therefore, our method focuses on learning motif structural information in the representation.

3 Methods

In this section, we propose a novel method to construct a Molecular Structural Similarity Motif Graph Neural Network (MSSM-GNN) (Illustrated in Fig. 2) which takes the MSSM graph as input.

Generally, the framework of the method consists of three parts: **(i)** Molecular graph representation; **(ii)** MSSM graph construction based on graph kernel; **(iii)** MSSM-GNN construction. Below, we explain in more detail about these parts.

3.1 Molecular Graph Representation

In molecular graphs, motifs are subgraphs that appear repeatedly and are statistically significant. Therefore, we propose a novel molecular graph representation method based on chemical domain knowledge and BRICS¹ algorithm to

¹ Breaking retrosynthetically interesting chemical substructures [10].

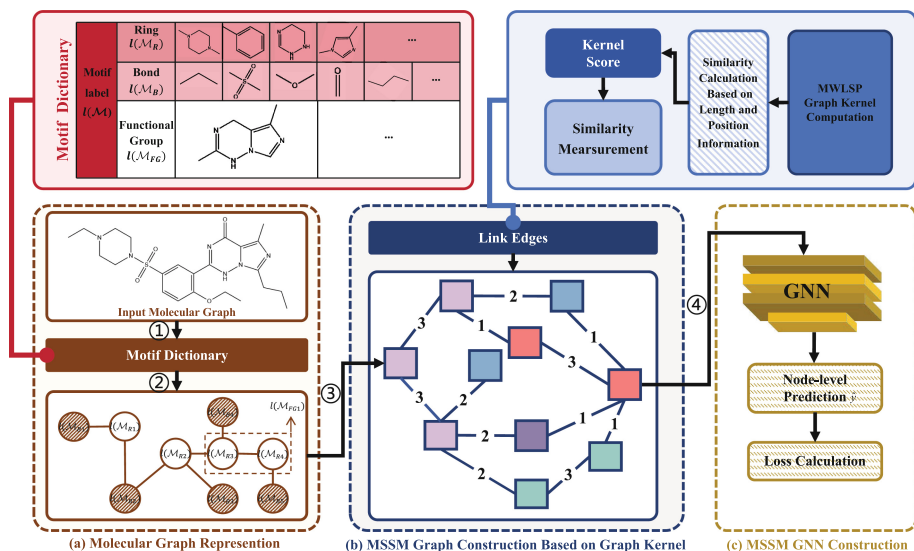


Fig. 2. The framework of our proposed Molecular Structural Similarity Motif Graph Neural Network.

represent molecular structural information better. It considers both the internal atomic structure and the overall impact of special functional groups in the molecule. Its main process consists of the following two steps: (i) Motif Dictionary; (ii) Molecular Graph Re-representation.

Motif Dictionary. Let $\mathcal{G} = (V, E)$ denote a molecular graph, where V is a set of atoms, $E \subseteq V \times V$ is a set of bonds between atoms. Generally, we denote a motif of the molecule \mathcal{G} by $\mathcal{M} = \langle \hat{V}, \hat{E} \rangle$, where \hat{V} is a subset of V and \hat{E} is the subset of E corresponding to \hat{V} , which includes all edges connecting nodes in \hat{V} . Due to the impact of ring, bond, and functional group structures on a molecule’s stability, mechanical properties, and reactivity [42], we extract these structures as three distinct types of motifs from \mathcal{G} . This extraction aims to establish a correlation between molecular structure and properties, facilitating a targeted capture of diverse chemical features within the molecule. It considers important structural components within the molecule as much as possible and can be extended to different types of molecules, making it a general approach.

To systematically organize and store the extracted motif information, we construct a motif dictionary \mathcal{D} by preprocessing all molecules in the dataset, as outlined in step ① of Fig. 2(a). The \mathcal{D} contains molecular identifiers as outer keys, each associated with nested dictionaries. These inner dictionaries categorize structural motif types with corresponding lists of extracted labels. We define the label $l(\mathcal{M})$ as the type of \mathcal{M} . The example in Fig. 2(a) illustrates that ring type

Piperazine² can be expressed as $l(\mathcal{M}_{R1})$. This organization efficiently stores and retrieves structural information within each molecule.

Molecular Graph Re-representation. Based on the motif dictionary, we traverse the structure type and their corresponding motif lists for each molecule within it. This process aims to re-represent the molecular graph by establishing connections between the motifs in molecules. We defined the graph as $\mathcal{G}_{\mathcal{M}} = (\mathcal{V}, \mathcal{E})$, where \mathcal{V} and \mathcal{E} denote a set of motifs and a set of adjacency relationship between motifs of the molecule, respectively. In the $\mathcal{G}_{\mathcal{M}}$, a motif \mathcal{M} is associated with a label $l(\mathcal{M})$ and adjacent motifs are connected by edges. As illustrated in step ②, for the drug molecule vardenafil, we can use the proposed algorithm to construct a $\mathcal{G}_{\mathcal{M}}$ from motifs out of the \mathcal{D} .

3.2 MSSM Graph Construction Based on Graph Kernel

Through the above method, we obtain a re-representation molecular graph $\mathcal{G}_{\mathcal{M}}$. To provide GNN with more information, we will construct a Molecular Structural Similarity Motif (MSSM) graph in step ③. In the MSSM graph, each node represents a $\mathcal{G}_{\mathcal{M}}$, and the edge represents two nodes $\mathcal{G}_{\mathcal{M}_1}$ and $\mathcal{G}_{\mathcal{M}_2}$ with structural similarity. We calculate the similarity between two $\mathcal{G}_{\mathcal{M}}$ by utilizing Mahalanobis Weisfeiler-Lehman Shortest-Path(MWLSP) graph kernel.

The fundamental idea of the graph kernel is to measure the similarity via the comparison of $\mathcal{G}_{\mathcal{M}}$ ' substructures. The kernel we proposed retains expressivity and is still computable in polynomial time.

As depicted in Fig. 2, MWLSP graph kernel takes $\mathcal{G}_{\mathcal{M}_1}$, $\mathcal{G}_{\mathcal{M}_2}$ as input, and its main process consists of the following steps: (i) Preprocess Input; (ii) Perform MWLSP Graph Kernel Computation; (iii) Comparison Scores of Graph-substructures. We give a pseudocode description of the MWLSP Graph Kernel in Algorithm 1.

Preprocess Input. In line 2, we utilize Floyd-transformation (For detailed explanation, see Appendix A.1) $Ft(\mathcal{G}_{\mathcal{M}})$ to convert graphs $\mathcal{G}_{\mathcal{M}_1}$ and $\mathcal{G}_{\mathcal{M}_2}$ into graphs \mathcal{G}_{F1} and \mathcal{G}_{F2} , respectively. $Ft(\mathcal{G}_{\mathcal{M}})$ generates the shortest path between all nodes in $\mathcal{G}_{\mathcal{M}}$. The shortest path between the vertex v and u is represented as (v, u) . The (v, u) is the shortest path among all paths between two nodes. \mathcal{G}_{F1} and \mathcal{G}_{F2} contain all the information regarding the shortest path substructure partitions in $\mathcal{G}_{\mathcal{M}_1}$ and $\mathcal{G}_{\mathcal{M}_2}$, respectively. Specifically, \mathcal{G}_{F1} has the same vertices as $\mathcal{G}_{\mathcal{M}_1}$, and the edge (v, u) in \mathcal{G}_{F1} represents detailed information about the shortest path in $\mathcal{G}_{\mathcal{M}_1}$.

$$\mathcal{G}_{F1} = Ft(\mathcal{G}_{\mathcal{M}_1}) \quad \mathcal{G}_{F2} = Ft(\mathcal{G}_{\mathcal{M}_2}) \quad (1)$$

² A six-membered ring compound containing two nitrogen atoms.

Perform MWLSP Graph Kernel Computation. $K_{mwlsps}(\mathcal{G}_{F1}, \mathcal{G}_{F2})$ will compute the similarity between two graphs, $\mathcal{G}_{\mathcal{M}_1}$ and $\mathcal{G}_{\mathcal{M}_2}$, by summing up $k(e_1, e_2)$ i.e., the comparison scores between substructures e_1 and e_2 in line 3–7. E'_1 is the set of all edges in \mathcal{G}_{F1} and e_1 is one of the edges in E'_1 . e_1 represents a shortest path substructure in $\mathcal{G}_{\mathcal{M}_1}$, and similarly for e_2 .

$$K_{mwlsps}(\mathcal{G}_{F1}, \mathcal{G}_{F2}) = \sum_{e_1 \in E'_1} \sum_{e_2 \in E'_2} k(e_1, e_2) \quad (2)$$

Algorithm 1: MWLSP Graph Kernel Calculation

Data: Graphs $\mathcal{G}_{\mathcal{M}_1} = (\mathcal{V}_1, \mathcal{E}_1)$, $\mathcal{G}_{\mathcal{M}_2} = (\mathcal{V}_2, \mathcal{E}_2)$, c , H
Result: Kernel Score K_{mwlsps}

- 1 **Function** MWLSPGraphKernel ($\mathcal{G}_{\mathcal{M}_1}$, $\mathcal{G}_{\mathcal{M}_2}$, c , H):
- 2 $\mathcal{G}_{F1} \leftarrow Ft(\mathcal{G}_{\mathcal{M}_1})$ $\mathcal{G}_{F2} \leftarrow Ft(\mathcal{G}_{\mathcal{M}_2})$;
- 3 $kernel_score \leftarrow 0$;
- 4 **for** e_1 in $E(\mathcal{G}_{F1})$ **do**
- 5 **for** e_2 in $E(\mathcal{G}_{F2})$ **do**
- 6 $kernel_score += k(e_1, e_2, c, H)$;
- 7 **return** $kernel_score$;
- 8 **Function** LengthSim(e_1 , e_2 , c):
- 9 $sim_1 \leftarrow \max(0, c - |length(e_1) - length(e_2)|)$;
- 10 **return** sim_1 ;
- 11 **Function** PositionSim(e_1 , e_2 , H):
- 12 Initialize labels L_1 and L_2 based on e_1 and e_2 ;
- 13 **for** h in $[0, H]$ **do**
- 14 **for** u in $V(e_1)$ **do**
- 15 $nbrs_sorted \leftarrow \text{sort}(\text{labels of neighbors of } u \text{ lexicographically})$
- 16 $L^{(h+1)}(u) \leftarrow \text{hash}(L^h(u), nbrs_sorted)$
- 17 **for** v in $V(e_2)$ **do**
- 18 Calculate $L^{(h+1)}(v)$ using the same method as above.
- 19 Calculate the Mahalanobis distance $D^{(h)}(u, v)$ between
- 20 $L^{(h)}(u)$ and $L^{(h)}(v)$ at the h -th iteration.
- 21 Sum $D(u, v)$ across all final iteration yields sim_2
- 22 **return** sim_2 ;
- 23 **Function** $k(e_1, e_2, c, H)$:
- 24 $sim_1 \leftarrow \text{LengthSim}(e_1, e_2, c)$ $sim_2 \leftarrow \text{PositionSim}(e_1, e_2, H)$;
- 25 **return** $sim_1 \times sim_2$;

Proposition 1. Let n be the average number of nodes and d be the dimensionality of the features. Each node is associated with a d -dimensional feature vector. The time complexity for the kernel given by Eq. 2 is $O(n^3 + n^4 * (1 + Hnd^3))$.

The proof is given in the **Appendix B**.

Comparison Scores of Graph-Substructures. For $k(e_1, e_2)$, we will calculate the similarity of substructures from two aspects: length and position. The calculation formulas are respectively $sim_1(e_1, e_2)$ and $sim_2(e_1, e_2)$. For the aspect of length, $sim_1(e_1, e_2)$ utilizes the Brownian bridge [6] to assess the similarity between e_1 and e_2 in line 8–10. It returns the largest value when two edges have identical lengths and 0 when the edges differ in length by more than a hyperparameter c . Furthermore, we can change the c to control the similarity threshold, thus adjusting the filtering criteria.

$$sim_1(e_1, e_2) = \max(0, c - |\text{length}(e_1) - \text{length}(e_2)|) \quad (3)$$

For the aspect of positional information, $sim_2(e_1, e_2)$ establishes a Weisfeiler-Lehman(WL) propagation scheme [31] on the graphs, iteratively comparing labels on the nodes and their neighbors via Mahalanobis Distance(MD) [8].

Specifically, we let h be the current WL iteration which ranges from 0 to H (H is the total number of iterations). $L^h(u)$ is a set of node labels, representing the positional information of node u at the current iteration h . $\mathcal{N}^h(u) = \{L^h(u_{left}), L^h(u_{right})\}$ represents the positional information of u 's neighboring nodes at the current iteration h . In the shortest path graph, u_{left} and u_{right} are the only two neighbor nodes of u . The scheme primarily consists of several steps, described in line 11–22:

Firstly, we compare two paths, e_1 and e_2 , by utilizing the motif labels to initialize the sets of all node labels on these paths in line 12.

$$L^0(u) = l(u) \quad (4)$$

Next, if identical node labels exist, further iterative evaluation is conducted. We define the iterative rule with the hash function: in each iteration, the positional information of u includes one more iteration of node connectivity compared to the previous iteration. By inputting the positional information of the current iteration's node u , i.e., $L^h(u)$ and its neighboring nodes, i.e., $\mathcal{N}^h(u)$, we use Eq. 5 to compute $L^{h+1}(u)$, i.e., the positional information of u in the next iteration $h + 1$. And $\text{sort}(\cdot)$ sorts the labels lexicographically. The specific execution process is shown in line 13–18.

$$L^{h+1}(u) = \text{hash}(L^h(u), \text{sort}((L^h(v_1), \dots, L^h(v_{|\mathcal{N}(u)|}))))), \quad (5)$$

$$v_j \in \mathcal{N}^h(u), j \in \{1, \dots, |\mathcal{N}(u)|\}.$$

Through the aforementioned process, we can represent the positional information of all nodes in e_1 and e_2 by utilizing $L^h(u)$. Furthermore, in line 20, we use MD (Please see **Appendix A.2** for explanation) to measure the similarity between nodes. $D^h(u, v)$ denotes the MD between the $L^h(u)$ and $L^h(v)$ at a specific iteration h . M^h is the covariance matrix $Cov(L^h(u), L^h(v))$. The utilization of MD considers the diverse distribution characteristics of nodes belonging to different types in the heterogeneous feature space. In the context of $\mathcal{G}_{\mathcal{M}}$, distinct types of motifs may correspond to varied structures or properties. Therefore, we

can quantify the similarity between motifs based on the distribution characteristics of each motif type.

$$D^h(u, v) = \sqrt{(L^h(u) - L^h(v))^T M^h (L^h(u) - L^h(v))} \quad (6)$$

Finally, we cumulatively aggregate the MD from the 0-th to the H -th iteration in line 21. Through a weighted synthesis, we calculate the relational similarity between u and v , considering positional information across all iterations. Therefore, we can calculate the similarity score $sim_2(e_1, e_2)$ by comparing the position similarity relationships among all nodes in e_1 and e_2 .

$$sim_2(e_1, e_2) = \sum_{u \in V(e_1)} \sum_{v \in V(e_2)} \exp\left(-\frac{1}{2} \sum_{h=0}^H D^h(u, v)\right) \quad (7)$$

For the above iterative process, we set two termination conditions:

- (i) There is no intersection in the positional information of all nodes in e_1 and e_2 within the current iteration. This condition implies that, in the next iteration, the positional information of nodes for e_1 and e_2 is dissimilar, so we can terminate early.
- (ii) We have calculated the positional information for all iterations in e_1 and e_2 , and the total number of iterations will not exceed $\min(|\text{length}(e_1), \text{length}(e_2)|)$.

In summary, the comparison score of the graph substructure can be obtained by multiplying the similarities of the above two parts in line 23–25.

$$k(e_1, e_2) = sim_1(e_1, e_2) * sim_2(e_1, e_2) \quad (8)$$

MSSM Graph Construction. We can construct the MSSM graph based on the similarity calculation result of the above MWLSP graph kernel. Since the structural similarity analysis of molecules often does not require very precise numerical values, it focuses on the relative similarity between molecules. To reduce the complexity of the comparison, we simplify the kernel score to an integer range of $[0, 3]$ by dividing it by the maximum achievable value:

$$S(Ft(\mathcal{G}_{M_i}), Ft(\mathcal{G}_{M_j})) = \left\lfloor \frac{3 \cdot K_{mwlspl}(Ft(\mathcal{G}_{M_i}), Ft(\mathcal{G}_{M_j}))}{\max(K_{mwlspl}(Ft(\mathcal{G}_{M_i}), Ft(\mathcal{G}_{M_j})))} \right\rfloor \quad (9)$$

where $\lfloor x \rfloor$ represents rounding x down to the nearest integer.

Considering the above possible calculation results, we use the similarity score $S(Ft(\mathcal{G}_{M_i}), Ft(\mathcal{G}_{M_j}))$ to represent the corresponding edge weight value A_{ij} and formally establish detailed measurement standard Sim_{ij} as follows:

$$Sim_{ij} = \begin{cases} \text{Very High Similarity} & \text{if } A_{ij} = 3, \\ \text{Relatively High Similarity} & \text{if } A_{ij} = 2, \\ \text{Average Similarity} & \text{if } A_{ij} = 1, \\ \text{Dissimilar} & \text{if } A_{ij} = 0 \end{cases} \quad (10)$$

where if $A_{ij} > 0$, \mathcal{G}_{M_i} and \mathcal{G}_{M_j} have a similar relationship, and a connecting edge with corresponding weight value needs to be established; otherwise, there is no need to perform connection processing. Figure 2(b) provides an example of MSSM graph construction.

3.3 MSSM-GNN Construction

In this part, we build an MSSM-GNN to learn graph structural feature representations of the MSSM graph. In graph learning, the input MSSM graphs can be denoted as $\mathcal{G}_{MSSM} = (\mathcal{V}_{MSSM}, \mathcal{E}_{MSSM})$, where \mathcal{V}_{MSSM} is the node set of $\mathcal{G}_{\mathcal{M}}$, and \mathcal{E}_{MSSM} is the edge set of similarity relationship between two $\mathcal{G}_{\mathcal{M}}$. And we use $y \in \mathcal{Y}$ as the node-level property label for $\mathcal{G}_{\mathcal{M}_i}$, where \mathcal{Y} represents the label space.

For graph property prediction, a predictor with the encoder-decoder architecture is trained to encode \mathcal{G}_{MSSM} into a node representation vector in the latent space and decode the representation to predict \hat{y} . Specifically, we fed the MSSM graph data into GNN to acquire \hat{y} (corresponds to step ④):

$$\hat{y} = \text{GNN}(\mathcal{G}_{MSSM}) \in \mathcal{Y}. \quad (11)$$

The loss function used in our model is the label prediction loss. The label prediction loss function \mathcal{L}_{pred} is derived similarly to existing methods:

$$\mathcal{L}_{pred} = \text{CE}(\hat{y}, y). \quad (12)$$

where \hat{y} represents the predicted value, y is the ground truth, and CE represents the Cross-Entropy loss function used in classification tasks.

In this way, we can get a more comprehensive feature representation of the entire \mathcal{G}_{MSSM} . It contains all the information on the connected motifs, retaining the atomic structure relationships and connections within the original motifs. Therefore, we can get a more accurate prediction of molecular properties based on the MSSM-GNN. The process is illustrated in Fig. 2(c).

4 Experiments

In this section, we investigate how our proposed method improves GNN performance on molecular property tasks. In our investigations, we raise the following questions: **Q1**: Compared with state-of-the-art baselines, how effective is MSSM-GNN in improving the accuracy of molecular prediction on common bioinformatics graph benchmark datasets? **Q2**: If experiments are conducted on real-world datasets, will MSSM-GNN still have an effect? **Q3**: Does feature learning of similarities between molecules play a more critical role in MSSM-GNN? **Q4**: What impact will the setting of the similarity threshold on different datasets have on the final classification results?

In response to the above problems, we conducted a series of experimental studies. Some basic settings of experiments and analysis of results are as follows:

4.1 Experimental Settings

Datasets. To verify whether MSSM-GNN provides more information conducive to accurate classification, we evaluate our model on five popular bioinformatics graph benchmark datasets from TUDataset [27], which includes four molecular datasets PTC [34], MUTAG [9], NCI1 [35], MUTAGENICITY [20] and one protein dataset PROTEINS [3].

Table 1. Graph classification accuracy (%) on various TUDataset graph classification tasks. The best performers on each dataset are shown in **bold**.

Methods	PTC	NCI1	MUTAG	PROTEINS	MUTAGENICITY
DGCNN	58.6 ± 2.5	74.4 ± 0.5	85.8 ± 1.7	75.5 ± 0.9	72.3 ± 2.6
GCN	64.2 ± 4.3	80.2 ± 2.0	85.6 ± 5.8	76.0 ± 3.2	79.8 ± 1.6
GIN	64.6 ± 7.0	82.7 ± 1.7	89.4 ± 5.6	76.2 ± 2.8	82.0 ± 0.3
PatchySAN	60.0 ± 4.8	78.6 ± 1.9	92.6 ± ± 4.2	75.9 ± 2.8	77.9 ± 1.3
GraphSAGE	63.9 ± 7.7	77.7 ± 1.5	85.1 ± 7.6	75.9 ± 3.2	78.8 ± 1.2
PPGN	66.2 ± 6.5	83.2 ± 1.1	90.6 ± 8.7	77.2 ± 4.7	78.6 ± 0.9
WEGL	64.6 ± 7.4	76.8 ± 1.7	88.3 ± 5.1	76.1 ± 3.3	80.8 ± 0.4
CapsGNN	71.2 ± 1.9	78.4 ± 1.6	86.7 ± 6.9	76.3 ± 4.6	79.5 ± 0.7
GSN	68.2 ± 7.2	83.5 ± 2.3	90.6 ± 7.5	76.6 ± 5.0	81.0 ± 1.5
HM-GNN	78.5 ± 2.6	83.6 ± 1.5	96.3 ± 2.8	79.9 ± 3.1	83.0 ± 1.1
GPNN	78.2 ± 1.2	83.1 ± 0.3	92.6 ± 1.8	76.8 ± 3.9	83.0 ± 0.4
OURS	81.1 ± 1.7	85.5 ± 0.3	97.3 ± 2.6	83.3 ± 0.4	84.0 ± 0.5

Baselines. We compare our model with eleven state-of-the-art GNN models for molecular property tasks: Deep Graph CNN (DGCNN) [29], GCN [21], GIN [40], PATCHYSAN [28], GraphSAGE [14], Provably Powerful Graph Networks (PPGN) [24], Wasserstein Embedding for Graph Learning (WEGL) [22], Capsule Graph Neural Network (CapsGNN) [39], GSN [4], HM-GNN [45], GPNN [15].

4.2 Performance Evaluation on Molecular Graph Datasets

To learn graph feature representations in our molecular structural similarity motif graphs, 3 GNN layers are applied. For a fair comparison, we evaluate all baselines using the experiment settings provided by [45]. The hyper-parameters we tune for each dataset are (1) the learning rate \in 0.01, 0.05; (2) the number of hidden units \in 16, 64, 1024; (3) the dropout ratio \in 0.2, 0.5. We set the verification method as the mean and standard deviation of the seven best validation accuracies from ten folds. We compare MSSM-GNN with the baseline approaches on the abovementioned dataset to answer **Q1**. The comparison results are summarized in Table 1. We make the following observations:

MSSM-GNN significantly outperforms baseline models on all five datasets for molecular prediction. Among them, on the PROTEINS dataset, the accuracy of MSSM-GNN increased by 3.4% compared with the best method. The superior performances on five molecular datasets demonstrate that motif substructures extracted from the motif dictionary, along with the calculated similarity relationships between molecular nodes based on it, facilitate GNNs in learning improved motif-level and molecular-level feature representations of molecular graphs.

Table 2. Graph Classification Results (%) on Open Graph Benchmark datasets.

Methods	ogbg-molhiv	ogbn-proteins	ogbg-moltoxcast	ogbg-molpcba
GCN	75.99 ± 1.19	72.51 ± 0.35	61.13 ± 0.47	24.24 ± 0.34
GIN	77.07 ± 1.49	77.68 ± 0.20	62.19 ± 0.36	27.03 ± 0.23
GSN	77.90 ± 0.10	85.80 ± 0.28	62.61 ± 0.45	27.00 ± 0.70
PNA	79.05 ± 1.32	86.82 ± 0.18	63.47 ± 0.67	25.70 ± 0.60
HM-GNN	79.03 ± 0.92	86.42 ± 0.08	64.38 ± 0.39	28.70 ± 0.26
GPNN	77.70 ± 2.30	87.74 ± 0.13	65.22 ± 0.47	28.90 ± 0.91
OURS	79.70 ± 0.03	89.17 ± 0.07	66.57 ± 1.00	30.07 ± 0.37

4.3 Performance Evaluation on Large-Scale Real-World Datasets

To answer **Q2**, we evaluate our model on four large-scale real-world datasets from the Open Graph Benchmark (OGB) [16]. They are two binary classification datasets—ogbg-molhiv, ogbn-proteins and two multiclass classification datasets—ogbg-molt oxcast, ogbg-molpcba.

In this part, we compare our model with GIN, GCN, GSN, PNA, HM-GNN and GPNN. Except that the hyperparameters we tuned for each dataset varied as (1) learning rate \in 0.01, 0.001; (2) number of hidden units \in 10, 16; (3) dropout rate \in 0.5, 0.7, 0.9; (4) the batch size \in 128, 5000, 28000, others are the same as above experiment. Table 2 shows the AP results on Ogbg-molpcba and ROC-AUC results on the other three datasets. We observe: *our method is significantly better than the other compared methods by obvious margins.* The results prove our model’s superior generalization ability on real-world datasets, which is crucial for its potential applications in various domains, including drug discovery, bioinformatics, and chemical safety assessment.

4.4 Ablation Study

To address **Q3**, we conducted ablation experiments on different components of MSSM-GNN, focusing on the motif-based molecular graph representation and the similarity calculation. The corresponding conclusions are as follows:

Effect of Motif-Dictionary Representation. As shown in Table 3, comparing task performance before and after removing the motif dictionary module yields the following observations: Performance on three graph classification datasets benefits from the module, resulting in accuracy improvements ranging from 0.8% to 2.8%. These improvements could potentially be attributed to the module’s effective learning of valuable information about the molecule’s substructure.

Table 3. Ablation studies of the motif dictionary and measurement of length and position similarity.

Datasets	PTC_MR	PTC_FR	MUTAG	PROTEINS
MSSM-GNN	81.1 ± 1.7	80.9 ± 1.5	97.3 ± 2.6	83.3 ± 0.4
w/o motif dictionary	78.3 ± 1.1	78.6 ± 1.4	96.5 ± 2.7	82.5 ± 1.2
w/o length similarity	77.9 ± 1.5	78.3 ± 1.1	94.6 ± 0.2	81.2 ± 0.9
w/ edit distance	77.1 ± 2.9	78.2 ± 1.7	93.1 ± 0.6	80.8 ± 0.4

Effect of Length-Similarity Calculation. As shown in Table 3, we observe a significant drop in performance when the length-similarity calculation is not included, amounting to an absolute drop of 2.1%–3.2%. These observations confirm that evaluating path structures from a length perspective indeed facilitates the learning of the global information and inherent connectivity relationships among motif-level substructures, thereby contributing to representing graph information more comprehensively.

Effect of Position-Similarity Calculation. In MSSM-GNN, the location similarity calculation method we designed is MWL. By replacing MWL with edit distance, we examined the impact of the graph similarity metric. As Table 3 shows, MWL offers advantages over edit distance. MWL not only quantifies structural similarity but also incorporates the type and position information of different nodes in graph modeling, thereby effectively representing the real molecular graph structure. Meanwhile, MWL becomes particularly advantageous for larger-scale graph datasets, offering significant enhancements by extracting richer structural information. For example, our model enhances PROTEINS more than MUTAG.

4.5 Sensitive Analysis

In this part, to explore Q4, we further evaluate the hyperparameter c that we introduced in our proposed similarity calculation formula. We modify the value of hyperparameter c that controls the similarity threshold and observe how the

performance changes. We perform such experiments on multiple datasets. The results are shown in Fig. 3.

As observed, the performance peaks when the value of c is 2 across all three datasets. With the increase in c , the impact of the similarity threshold on training also becomes more pronounced. It is evident that the performance of MSSM-GNN decreases as c increases from 2 to 6, indicating that the c indeed influences the representation learning capabilities of MSSM-GNN. We believe that c assists in filtering out samples with low similarity, emphasizing those contributing more significantly to the training, thereby enhancing overall performance.

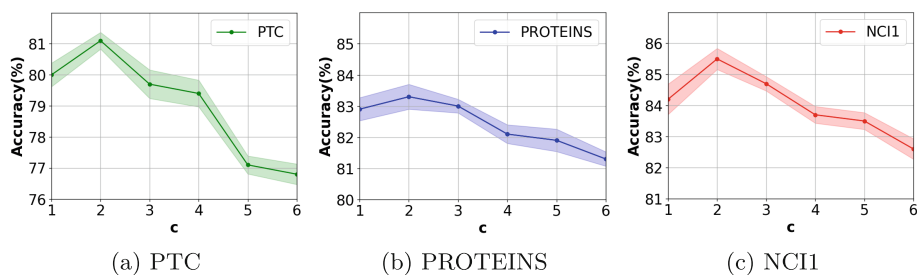


Fig. 3. Performance of MSSM-GNN on three different datasets with varying hyperparameters c .

5 Conclusions

This paper proposes an effective model for molecular graph representation learning, Molecular Structural Similarity Motif GNN (MSSM-GNN). We explicitly incorporate the similarity representations between molecules into GNN and jointly update them with motif representations. Specifically, we connect two molecules through edge weights calculated by a novel MWLSP graph kernel, enabling message passing between molecular graphs. We use the GNN model to learn the MSSM graph and get the motif-level and molecule-level graph embedding. Experiments demonstrate the superiority of our model in various datasets, which beats a group of baseline algorithms.

Acknowledgments. The authors would like to thank the editors and reviewers for their valuable comments. This work is supported by the CAS Project for Young Scientists in Basic Research (Grant No. YSBR-040) and the National Natural Science Foundation of China (Grant No. 62372439), the National Science Foundation of Beijing, China (Grant No. 4232038), the Basic Research Program of ISCAS (Grant No. ISCAS-JCMS-202307).

References



1. Atsango, A., Diamant, N.L., Lu, Z., Biancalani, T., Scalia, G., Chuang, K.V.: A 3D-shape similarity-based contrastive approach to molecular representation learning. arXiv preprint [arXiv:2211.02130](https://arxiv.org/abs/2211.02130) (2022)
2. Borgwardt, K.M., Kriegel, H.P.: Shortest-path kernels on graphs. In: Fifth IEEE International Conference on Data Mining (ICDM 2005), pp. 8–pp. IEEE (2005)
3. Borgwardt, K.M., Ong, C.S., Schönauer, S., Vishwanathan, S., Smola, A.J., Kriegel, H.P.: Protein function prediction via graph kernels. *Bioinformatics* **21**(Suppl_1), i47–i56 (2005)
4. Bouritsas, G., Frasca, F., Zafeiriou, S., Bronstein, M.M.: Improving graph neural network expressivity via subgraph isomorphism counting. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**(1), 657–668 (2022)
5. Cantoni, V., Gatti, R., Lombardi, L.: Morphological analysis of 3D proteins structure. In: International Conference on Bioinformatics Models, Methods and Algorithms, vol. 2, pp. 15–21. SciTePress (2011)
6. Chow, W.C.: Brownian bridge. *Wiley Interdiscip. Rev. Comput. Stat.* **1**(3), 325–332 (2009)
7. Dan, J., et al.: Self-supervision meets kernel graph neural models: from architecture to augmentations. In: 2023 IEEE International Conference on Data Mining Workshops (ICDMW), pp. 1076–1083. IEEE (2023)
8. De Maesschalck, R., Jouan-Rimbaud, D., Massart, D.L.: The mahalanobis distance. *Chemometr. Intell. Lab. Syst.* **50**(1), 1–18 (2000)
9. Debnath, A.K., Lopez de Compadre, R.L., Debnath, G., Shusterman, A.J., Hansch, C.: Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *J. Med. Chem.* **34**(2), 786–797 (1991)
10. Degen, J., Wegscheid-Gerlach, C., Zaliani, A., Rarey, M.: On the art of compiling and using ‘drug-like’ chemical fragment spaces. *ChemMedChem Chem. Enabling Drug Discov.* **3**(10), 1503–1507 (2008)
11. Duvenaud, D.K., et al.: Convolutional networks on graphs for learning molecular fingerprints. In: Advances in Neural Information Processing Systems, vol. 28 (2015)
12. Gao, H., Ji, S.: Graph u-nets. In: International Conference on Machine Learning, pp. 2083–2092. PMLR (2019)
13. Geng, Z., et al.: De novo molecular generation via connection-aware motif mining. arXiv preprint [arXiv:2302.01129](https://arxiv.org/abs/2302.01129) (2023)
14. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
15. Hevathige, A., Wang, Q.: Uplifting the expressive power of graph neural networks through graph partitioning. arXiv preprint [arXiv:2312.08671](https://arxiv.org/abs/2312.08671) (2023)
16. Hu, W., et al.: Open graph benchmark: datasets for machine learning on graphs. In: Advances in Neural Information Processing Systems, vol. 33, pp. 22118–22133 (2020)
17. Huang, K., Xiao, C., Hoang, T., Glass, L., Sun, J.: Caster: predicting drug interactions with chemical substructure representation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 702–709 (2020)
18. Inae, E., Liu, G., Jiang, M.: Motif-aware attribute masking for molecular graph pre-training. arXiv preprint [arXiv:2309.04589](https://arxiv.org/abs/2309.04589) (2023)
19. Jin, W., Barzilay, R., Jaakkola, T.: Hierarchical generation of molecular graphs using structural motifs. In: International Conference on Machine Learning, pp. 4839–4848. PMLR (2020)

20. Kazius, J., McGuire, R., Bursi, R.: Derivation and validation of toxicophores for mutagenicity prediction. *J. Med. Chem.* **48**(1), 312–320 (2005)
21. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: *International Conference on Learning Representations* (2016)
22. Kolouri, S., Naderializadeh, N., Rohde, G.K., Hoffmann, H.: Wasserstein embedding for graph learning. *arXiv preprint [arXiv:2006.09430](https://arxiv.org/abs/2006.09430)* (2020)
23. Liu, Y., Wang, L., Liu, M., Lin, Y., Zhang, X., Oztekin, B., Ji, S.: Spherical message passing for 3D molecular graphs. In: *International Conference on Learning Representations (ICLR)* (2022)
24. Maron, H., Ben-Hamu, H., Serviansky, H., Lipman, Y.: Provably powerful graph networks. In: *Advances in Neural Information Processing Systems*, vol. 32 (2019)
25. Maziarz, K., et al.: Learning to extend molecular scaffolds with structural motifs. In: *International Conference on Machine Learning* (2021)
26. Maziarz, K., et al.: Learning to extend molecular scaffolds with structural motifs. In: *International Conference on Learning Representations* (2022)
27. Morris, C., Kriege, N.M., Bause, F., Kersting, K., Mutzel, P., Neumann, M.: TUDataset: a collection of benchmark datasets for learning with graphs. *arXiv preprint [arXiv:2007.08663](https://arxiv.org/abs/2007.08663)* (2020)
28. Niepert, M., Ahmed, M., Kutzkov, K.: Learning convolutional neural networks for graphs. In: *International Conference on Machine Learning*, pp. 2014–2023. PMLR (2016)
29. Phan, A.V., Le Nguyen, M., Nguyen, Y.L.H., Bui, L.T.: DGCNN: a convolutional neural network over large-scale labeled graphs. *Neural Netw.* **108**, 533–543 (2018)
30. Shen, J., Nicolaou, C.A.: Molecular property prediction: recent trends in the era of artificial intelligence. *Drug Discov. Today Technol.* **32**, 29–36 (2019)
31. Shervashidze, N., Schweitzer, P., Van Leeuwen, E.J., Mehlhorn, K., Borgwardt, K.M.: Weisfeiler-Lehman graph kernels. *J. Mach. Learn. Res.* **12**(9) (2011)
32. Subramonian, A.: Motif-driven contrastive learning of graph representations. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 15980–15981 (2021)
33. Sun, M., Zhao, S., Gilvary, C., Elemento, O., Zhou, J., Wang, F.: Graph convolutional networks for computational drug development and discovery. *Brief. Bioinform.* **21**(3), 919–935 (2020)
34. Toivonen, H., Srinivasan, A., King, R.D., Kramer, S., Helma, C.: Statistical evaluation of the predictive toxicology challenge 2000–2001. *Bioinformatics* **19**(10), 1183–1193 (2003)
35. Wale, N., Watson, I.A., Karypis, G.: Comparison of descriptor spaces for chemical compound retrieval and classification. *Knowl. Inf. Syst.* **14**, 347–375 (2008)
36. Wernicke, S.: Efficient detection of network motifs. *IEEE/ACM Trans. Comput. Biol. Bioinf.* **3**(4), 347–359 (2006)
37. Wu, F., Radev, D., Li, S.Z.: Molformer: motif-based transformer on 3D heterogeneous molecular graphs. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, pp. 5312–5320 (2023)
38. Wu, T., Tang, Y., Sun, Q., Xiong, L.: Molecular joint representation learning via multi-modal information. *arXiv preprint [arXiv:2211.14042](https://arxiv.org/abs/2211.14042)* (2022)
39. Xinyi, Z., Chen, L.: Capsule graph neural network. In: *International Conference on Learning Representations* (2018)
40. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? In: *International Conference on Learning Representations* (2018)

41. Xu, Z., Wang, S., Zhu, F., Huang, J.: Seq2seq fingerprint: an unsupervised deep molecular embedding for drug discovery. In: Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics, pp. 285–294 (2017)
42. Xue, L., Bajorath, J.: Molecular descriptors in chemoinformatics, computational combinatorial chemistry, and virtual screening. *Comb. Chem. High Throughput Screening* **3**(5), 363–372 (2000)
43. Yang, N., Zeng, K., Wu, Q., Jia, X., Yan, J.: Learning substructure invariance for out-of-distribution molecular representations. In: Advances in Neural Information Processing Systems, vol. 35, pp. 12964–12978 (2022)
44. Yu, W., Chen, S., Gong, C., Niu, G., Sugiyama, M.: Atom-motif contrastive transformer for molecular property prediction. arXiv preprint [arXiv:2310.07351](https://arxiv.org/abs/2310.07351) (2023)
45. Yu, Z., Gao, H.: Molecular representation learning via heterogeneous motif graph neural networks. In: International Conference on Machine Learning, pp. 25581–25594. PMLR (2022)
46. Zhang, X., et al.: Artificial intelligence for science in quantum, atomistic, and continuum systems. arXiv preprint [arXiv:2307.08423](https://arxiv.org/abs/2307.08423) (2023)
47. Zhang, Z., Liu, Q., Wang, H., Lu, C., Lee, C.K.: Motif-based graph self-supervised learning for molecular property prediction. In: Advances in Neural Information Processing Systems, vol. 34, pp. 15870–15882 (2021)
48. Zhu, J., et al.: O-GNN: incorporating ring priors into molecular modeling. In: The Eleventh International Conference on Learning Representations (2023)



Efficient Privacy-Preserving Truth Discovery and Copy Detection in Crowdsourcing

Xiu Susie Fang¹, Xinyang Du¹ , Hao Chen¹, Ziqi Wei², Yong Zhan³,
and Guohao Sun¹ 

¹ Donghua University, Shanghai 201620, China

{xiu.fang,ghsun}@dhu.edu.cn, {2222791,2222842}@mail.dhu.edu.cn

² CAS Key Laboratory of Molecular Imaging, Institute of Automation,
Beijing 100190, China
ziqi.wei@ia.ac.cn

³ Secondary European Affiliation, Tiergartenstr. 17, 69121 Heidelberg, Germany
zhanyong@linkingfresh.com

Abstract. Researchers continue to focus on privacy-preserving truth discovery and achieve certain results with the increasingly popular trend of privacy protection. However, the common existence of copiers among workers is overlooked in existing privacy-preserving truth discovery, which causes decreased accuracy. Since methods based on encryption or perturbation may easily introduce noise and lose correlation between original data, it is challenging to detect copiers on privacy-preserving data. To address this challenge, in this paper, we propose an anti-copy iterative model based on lightweight homomorphic encryption, called CAPP-TD. First, we propose a lightweight privacy protection mechanism based on Paillier homomorphic encryption that preserves the correlation of privacy data. Compared with traditional homomorphic encryption-based algorithms, it requires less communication and computation overhead to perform truth discovery with copy detection. We then propose an iterative truth discovery method that can efficiently detect copy relationships in encrypted data and exclude copiers from truth inference to improve accuracy. Experimental results on both real-world and synthetic datasets and thorough security analysis demonstrate that CAPP-TD protects crowdsourcing systems from adversaries and enables highly accurate truth discovery.

Keywords: Crowdsourcing · Truth discovery · Copy detection · Privacy preserving

1 Introduction

Recently, the rapid development and widespread application of crowdsourcing technology have brought about privacy concerns. Malicious workers or unregulated third-party crowdsourcing servers may collect workers' personal information for profit. Malicious data requesters may attempt to attack crowdsourcing servers to obtain other requesters' task data [8,9]. To protect data privacy,

researchers have focused on privacy-preserving truth discovery algorithms and made some progress [2, 13–15]. These approaches leverage techniques like differential privacy [13, 14], anonymization [2], and homomorphic encryption [15, 16]. However, to our knowledge, these works have overlooked the existence of copiers. Crowdsourcing workers are driven by financial incentives, leading some to blindly copy answers to complete as many tasks as possible and get paid more. Copiers may directly copy answers from other workers while sitting together. Some workers may share their answers to help their friends to earn more. Moreover, malicious workers may collude and deliberately submit wrong answers to manipulate the final aggregated answers, such as Sybil attack [1]. These prevalent behaviors occur offline in crowdsourcing systems, making it challenging for the server to identify which workers have made real efforts. If these copiers happen to copy incorrect answers, the crowdsourcing server may wrongly identify them as the truth, misleading the data requesters and resulting in substantial financial losses or personal harm.

As an example, given a symptom description, workers need to choose an answer as the most likely diagnosis from five options, including A. Cardiogenic asthma, B. Chronic mouth-breathing bronchitis, C. Bronchial asthma, D. Spontaneous pneumothorax, E. hypersensitivity pneumonitis. The ground truth is bronchial asthma. Because cardiogenic asthma is similar to bronchial asthma, workers may mistake bronchial asthma for cardiogenic asthma. If copiers happen to copy the cardiogenic asthma answer, it can lead to a significant increase in the number of cardiogenic asthma supporters, ultimately treating it as the truth. Hence, copy detection is crucial for crowdsourcing servers. It can avoid the waste of rewards, ensure that workers are paid fairly, discourage opportunistic behavior, and penalize lazy copiers. Moreover, it could improve the accuracy of answer aggregation and prevent data requesters from being misled by incorrect answers. Several traditional truth discovery methods take the existence of copiers into consideration [10–12, 19, 20]. However, these methods operate on plaintext without addressing privacy protection concerns. Continuing with the previous example, the workers who answer these questions are typically sufferers of these diseases. They willingly share their experiences to assist doctors in improving diagnoses. However, it is crucial to ensure that their private information remains confidential and is not misused for harassment or any other unauthorized purposes.

To conduct privacy-preserving truth discovery while considering the existence of copiers, we are facing with three challenges: i) The privacy protection mechanism should maintain data correlation while ensuring the desensitization of sensitive information. Perturbation may blur the clues in the original plaintext that can be utilized for copy detection, enabling copiers to conceal their behavior. Anonymization directly exposes the plaintext to the malicious workers. ii) Given the large volume of data and the limited effective duration in crowdsourcing, the privacy-preserving truth discovery framework must be lightweight. Though the simple application of homomorphic encryption can effectively preserve data correlation, it leads to increased computing and communication costs. iii) Tra-

ditional truth discovery methods with copy detection are designed for plaintext and are not easily applicable to ciphertext scenarios. Developing a novel copy detection approach tailored to privacy-preserving truth discovery in crowdsourcing scenarios is significant.

To tackle the above challenges, we propose a privacy-preserving truth discovery framework called **Copy-Aware Privacy-Preserving Truth Discovery (CAPP-TD)** that takes into account the presence of copiers. For challenge i and ii, we propose a lightweight privacy protection mechanism based on Paillier homomorphic encryption that can preserve data correlation. Unlike traditional homomorphic encryption-based mechanisms, our mechanism only requires three encryption and decryption operations and two communications between servers to perform truth discovery with copy detection. For challenge iii, we propose an iterative truth discovery model based on copy detection in ciphertext scenarios. Our model first calculates the answer consistency group based on the similarity matrices. Then, it infers the estimated truth by considering the answer consistency groups and worker weights. During this process, copiers are identified and eliminated based on the similarity matrices and worker weights. Lastly, the worker weights are updated based on the estimated truth. In summary, our main contributions are as follows:

- We propose a lightweight privacy protection mechanism based on Paillier homomorphic encryption that can preserve data correlation. Compared with other homomorphic encryption mechanisms, this mechanism supports more secure truth discovery based on similarity matrices from the encrypted answers;
- We proposed the CAPP-TD framework, which enables truth discovery with copy detection in privacy-preserving crowdsourcing and eliminates copier answers to improve aggregation accuracy. To the best of our knowledge, this is the first work that conducts copy detection in crowdsourcing with privacy protection;
- We conducted extensive experiments on both real-world and synthetic datasets as well as thorough security analysis. The experimental results demonstrate the effectiveness, accuracy, and practicality of our proposed framework.

2 Related Work

Yin et al. [2] first formally defined the truth discovery problem and proposed the TruthFinder model, which uses Bayesian distribution to iteratively evaluate source weights and truth. Since then, a large number of truth discovery methods have been proposed [5]. Li et al. proposed the CRH [5] model, which uses an optimization model to process heterogeneous data. Truth discovery has subsequently been widely cited in crowdsourcing [17, 18] to obtain accurate estimated truth. Revolt [18] exploits disagreements in the crowd to identify ambiguous concepts, thereby improving aggregation accuracy; Wu et al. [17] proposed the TILCC

model, aiming to improve the quality of integrated labels for single-choice classification problems in crowdsourcing labeling tasks. However, none of these works take into account the privacy-preserving issues existing in crowdsourcing.

With the rapid development of crowdsourcing platforms, people are increasingly concerned about privacy issues. Li et al. [6] proposed that workers can use the parameters they provide to generate noise samples based on exponential distribution, and then add this unique noise to the data to form Gaussian distribution noise, thereby completing the data aggregation process; Pang et al. [7] improved the personalized classification based on privacy protection, and achieved high-precision truth discovery by adjusting the privacy level while protecting privacy; Sun et al. [8] considered the problem of providing personalized rewards to workers according to their privacy needs to satisfy the privacy preferences of different workers; Tang et al. [2] solved the user privacy problem through anonymization method and proposed a perturbation method to prevent data from being obtained by malicious third parties. Homomorphic encryption is introduced into the field of truth discovery as a traditional encryption mode. It does not modify the workers' answers, but directly transfers the workers' answers from the plaintext domain to the ciphertext domain, and then performs truth aggregation and weight update according to different homomorphic encryption algorithms. Ding et al. [9] adopted an architecture of central servers and multiple edge servers, considered the impact of the distance between tasks and workers on answers, and proposed a secure payment mechanism; Although these methods can achieve a certain degree of conflict resolution while preserving privacy, none of them consider the dependencies between workers.

Some works considered source correlation in crowdsourcing. Dong et al. [12] proposed methods to consider source correlation in truth discovery scenarios, believing that there may be dependencies between sources that always provide the same wrong answer; Jiang et al. [10] comprehensively considered the copy and incentive mechanism in crowdsourcing, and the detected copiers may not receive rewards. Wang et al. [11] focused on copiers and multi-valued objects, and proposed a comprehensive Bayesian method with fine-grained copy detection; Li et al. [19] proposed a method for copy detection taking into account the scale of big data and the number of sources. Chen et al. [20] proposed the CONAN method for maximizing the use of copier' answers to enhance aggregation accuracy. However, none of the works considered the case where the data was encrypted or perturbed.

3 Problem Statement

3.1 System Model

In this paper, we attempt to implement truth discovery with copy detection in privacy-preserving single-choice crowdsourcing tasks. Single-choice task contains a question and a set of candidate choices and asks workers to select a single choice out of the candidate choices, which is very common in crowdsourcing, such as disease identification, dog breed classification and sentiment analysis, etc. There

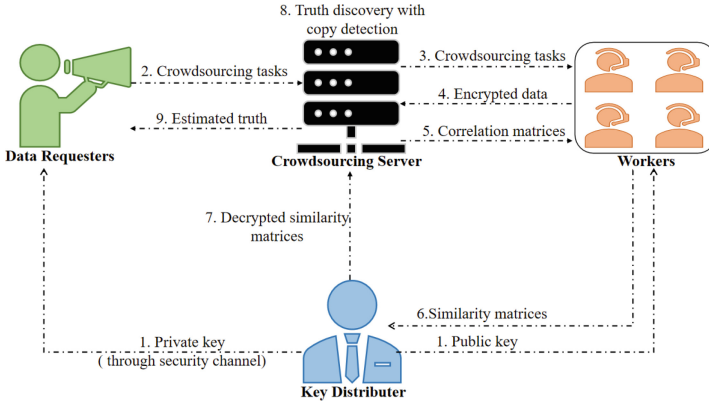


Fig. 1. Workflow of CAPP-TD.

are four entities involved in this work: the key distributor, the data requester, the worker and the crowdsourcing server. The key distributor generates a pair of Paillier homomorphic keys (p_w, s_w) for each worker, where p_w is the public key and s_w is the private key. It is responsible for key distribution and data decryption. The data requesters post tasks $t \in T$ to the crowdsourcing server. Their goal is to obtain the estimated truth $\hat{o}_t^* \in O^*$, minimizing the distance between O^* and the ground truth \hat{O} . The workers $w \in W$ work on the tasks assigned to them, their answer set is denoted as O . Given a specific worker w , $o_t^w \in O$ represents his answer on task t , r_w represents his weight. The workers encrypt their answers as $\hat{o}_t^w \in \hat{O}$ and upload them to the crowdsourcing server. The crowdsourcing server is responsible for task assignment, worker weight estimation, copy detection and truth calculation. For the sake of approach demonstration, we also denote the set of tasks assigned to worker w as T_w , the set of workers assigned task t as W_t , the potential copier set of worker w as $P_w = \{w' | T_{w'} \cap T_w \neq \emptyset\}$ (when two workers are assigned the same task, it is possible for one worker to copy from another worker). The workflow of our framework is presented as follows (as shown in Fig. 1):

Step 1: The key distributor runs the Paillier homomorphic encryption algorithm to generate the key pair (p_w, s_w) and broadcasts the public key to workers and sends the private key to data requesters over a security channel;

Step 2: The data requester posts the crowdsourcing tasks to the crowdsourcing server.

Step 3: Crowdsourcing server assigns tasks T_w to each worker;

Step 4: Each worker completes his tasks to generate the answer set O^w and encrypts O^w as \hat{O}^w with a public key p_w and then uploads it to the crowdsourcing server;

Step 5: The crowdsourcing server constructs a correlation matrix $\hat{C}_{|T_w| \times |P_w|}^{rw}$ (see Sect. 4.1) for each worker, and sends each matrix to its corresponding worker.

Step 6: Each worker performs an additive homomorphism calculation based on correlation matrix $\hat{C}_{|T_w| \times |P_w|}^w$ to construct similarity matrix $\hat{S}_{|T_w| \times |P_w|}^w$ (see Sect. 4.1), and sends $\hat{S}_{|T_w| \times |P_w|}^w$ to the key distributor (Table 1);

Table 1. Notations used in this paper

Notations	Definition
t, T	A task (resp., set of all tasks)
o_t^w, O	Worker w 's answer on task t (resp., set of all answers)
\hat{o}_t^w, \hat{O}	Worker w 's encrypted answer on task t (resp., set of all encrypted answers)
\acute{o}_t, \acute{O}	Ground truth of task t (resp., set of all ground truth)
o_t^*, O^*	Estimated truth of task t (resp., set of all estimated truth)
\hat{o}_t^*, \hat{O}^*	Encrypted estimated truth of task t (resp., set of encrypted estimated truth)
w, W	A worker (resp., set of all workers)
(p_w, s_w)	A pair of Paillier homomorphic keys for worker w , p_w/s_w is the public/private key
T_w	The set of tasks assigned to worker w
W_t	The set of workers assigned task t
P_w	The potential copier set of worker w
r_w	Weight of worker w
$\hat{C}_{ T_w \times P_w }^w$	Correlation matrix of worker w
$\hat{S}_{ T_w \times P_w }^w$	Similarity matrix of worker w
$S_{ T_w \times P_w }^w$	Decrypted similarity matrix of worker w
A_t^l	Answer consistency group of task t constructed in l -th iteration
A_t^*	Correct answer consistency group
\mathbb{A}^w	The set of answer consistency group that contains worker w

Step 7: The key distributor decrypts $\hat{S}_{|T_w| \times |P_w|}^w$ with s_w and sends $S_{|T_w| \times |P_w|}^w$ to the crowdsourcing server;

Step 8: The crowdsourcing server performs truth discovery with copy detection (see Sect. 4.3). Specifically, it iteratively updates worker weights, removes copiers, and estimates truth until satisfying the convergence criteria.

Step 9: The crowdsourcing server sends the encrypted aggregation result $\hat{o}_t^* \in \hat{O}^*$ to the data requester, who then uses the private key s_w to decrypt the aggregation result $o_t^* \in O^*$.

3.2 Security Model and Design Goals

In this section, we introduce the security model and design goals of our framework. All entities in our framework are assumed to be semi-honest, meaning they adhere to the framework but may be curious about the privacy data of others, which aligns with the existing work. For the crowdsourcing server values its reputation, it cannot collude with other entities. Furthermore, following [15], we give economic incentives to the key distributor to prevent it from colluding.

The main threats to the framework may include as follows: *Semi-honest entities* within the framework may attempt to decrypt the ciphertext and obtain the privacy information; *Copiers* may undermine the final estimated truth by blindly copying other workers' answers or engaging in Sybil attack to subvert the answers. *External adversary* has the capability to eavesdrop on the communication channel, acquiring answers from workers or even the estimated truth by the crowdsourcing server, then exploit this information for financial motives.

To effectively detect copiers while preserving privacy in truth discovery, our research focuses on the following design goals: *Privacy preservation*: protect the privacy of workers and the estimated truth by allowing only the data requester to access the estimated truth. *Accuracy*: with the help of the key distributor, crowdsourcing server enables to evaluate worker dependencies and punishes copiers to improve the accuracy of truth discovery. *Efficiency*: keep computation and communication costs acceptable.

4 The Design of CAPP-TD

4.1 Privacy Protection Mechanism

Existing privacy protection mechanism based on homomorphic encryption requires encryption and decryption operations with a time complexity of $O(n)$, along with the same cost of communications between servers during the process, which causes significant computation and communication overhead, and provides more attack opportunities for the adversary. In contrast, our privacy protection mechanism performs encryption and decryption operations only three times, i.e., the time complexity of $O(1)$. No encryption and decryption operations are needed in the truth discovery process. This highly reduces computation and communication costs, as well as the possibility of hijacking by the adversary. Our proposed privacy protection mechanism satisfies all possible threats in the security model. Detailed security analysis can be found in Sect. 6.

In our proposed privacy protection mechanism, firstly, the key publisher uses Paillier homomorphic encryption to generate a key pair (p_w, s_w) for each worker. The key distributor then broadcasts the public key p_w to each worker and transmits the private key s_w to the data requesters through a secure channel. After tasks are uploaded by data requesters and allocated to workers by crowdsourcing server, each worker provides answers to task set T_w and encrypts the answers with the public key p_w .

Based on the encrypted answer set \hat{O} uploaded by all workers, the crowdsourcing server constructs the potential copier set P_w , which includes all workers who have done at least one of the same tasks as worker w . Then, the crowdsourcing server constructs a correlation matrix $\hat{C}_{|T_w| \times |P_w|}^{w}$ for each worker, where each row represents a task $t \in T_w$, each column represents a worker $w' \in P_w$, each element $\hat{c}_t^{w,w'}$ is the encrypted answer $\hat{o}_t^{w'} = E(o_t^{w'})$ provided by w' on t . $E(\cdot)$ is the encrypt function and $D(\cdot)$ is the decrypt function. In our problem setting, workers are not assumed to answer all tasks in T . Therefore, in $\hat{C}_{|T_w| \times |P_w|}^{w}$, there may be missing values. In such case, we use -1 to fill in the missing elements.

The crowdsourcing server then sends each correlation matrix $\hat{C}_{|T_w| \times |P_w|}^w$ to the corresponding worker w . Each worker constructs a similarity matrix $\hat{S}_{|T_w| \times |P_w|}^w$ individually based on $\hat{C}_{|T_w| \times |P_w|}^w$ and O^w by applying additive homomorphic calculations. In $\hat{S}_{|T_w| \times |P_w|}^w$, each element $\hat{s}_t^{w,w'}$ is calculated by:

$$\hat{s}_t^{w,w'} = \begin{cases} \hat{c}_t^{w,w'} \frac{1}{o_t^w} = E(o_t^{w'}) \frac{1}{o_t^w}, & \text{if } \hat{c}_t^{w,w'} \neq -1 \\ -1, & \text{if } \hat{c}_t^{w,w'} = -1. \end{cases} \quad (1)$$

After the computation, each worker w uploads similarity matrix $\hat{S}_{|T_w| \times |P_w|}^w$ to the key distributor. The key distributor decrypts $\hat{S}_{|T_w| \times |P_w|}^w$ to obtain a corresponding decrypted similarity matrix $S_{|T_w| \times |P_w|}^w$ by using the private key s_w . In particular, each element $s_t^{w,w'}$ in $S_{|T_w| \times |P_w|}^w$ is calculated by:

$$s_t^{w,w'} = (D(E(o_t^{w'}) \frac{1}{o_t^w})) = \begin{cases} 1, & \text{if } \frac{o_t^{w'}}{o_t^w} = 1 \\ 0, & \text{if } \frac{o_t^{w'}}{o_t^w} \neq 1 \\ -1, & \text{if } \hat{s}_t^{w,w'} = -1. \end{cases} \quad (2)$$

Therefore, each element $s_t^{w,w'}$ represents the correlation between worker w and w' on task t . Specifically, if $s_t^{w,w'} = 1$, there may be a copy relation between worker w and w' on task t , because they select the same answer on this task. Otherwise, worker w and w' are independent in task t . In this way, our privacy-preserving mechanism preserves data correlation by capturing whether any pair of workers choose the same answer on their mutual task.

The key distributor then sends the decrypted similarity matrix $S_{|T_w| \times |P_w|}^w$ to the crowdsourcing server to help run truth discovery with copy detection. After the convergence criterion is satisfied, the crowdsourcing server sends the encrypted estimated truth \hat{O}^* to data requesters, while data requesters use private key s_w to obtain the final decrypted estimated truth O^* .

4.2 Copy Detection

Our copy detection method operates on the principle that if two workers consistently make the same mistakes, there may exist a copy relationship between them. However, unlike previous works that rely on probability-based theory, our method utilizes algebraic operations, which can be performed in decrypted data.

The crowdsourcing server constructs answer consistency groups for each task based on decrypted similarity matrices, i.e., workers who do the same answer are assigned to the same group. Specifically, we apply iterative method to construct answer consistency group. Because the server does not know the plaintext answers of each worker, we use the iteration round number to mark the answers chosen by the worker groups. We use A_t^l to depict the answer consistency group constructed in l -th iteration for task t . For each task $t \in T$, we construct a dictionary, denoted as $D_t = \{A_t^l : \hat{o}_t^w\}$, to preserve the mapping relationship between

each answer consistency group A_t^l and the answer \hat{o}_t^w selected by A_t^l . In the l -th iteration, if $W_t \neq \emptyset$, the crowdsourcing server selects a worker $w \in W_t$, checks each element in t -th row in $S_{|T_w| \times |P_w|}^w$. If $s_t^{w,w'} = 1$, add w' to A_t^l and remove w' from W_t . After all elements in t -th row are traversed, we add $\{A_t^l : \hat{o}_t^w\}$ to D_t . The server iteratively repeats the above operations until $W_t = \emptyset$. Say it takes L iterations to complete the answer consistency group construction for task t , then L groups have been constructed, and there are L pairs of $\{A_t^l : \hat{o}_t^w\}$ in D_t .

To better fit our situation, the above principle is modified: for two workers, if they appear in multiple mutual answer consistency groups, and most of these groups choose wrong answers, there may exist a copy relationship between them. Based on this, we first determine whether two workers are independent by calculating the proportion of mutual tasks they make. We believe that if two workers rarely finish the same tasks, they are likely to be independent. Obviously, copiers can not copy from someone who completes different tasks. For each worker pair (w, w') , we introduce a metric denoted as $IS(w, w')$, to measure whether worker w' and w are independent of each other, which is calculated as follows:

$$IS(w, w') = \frac{|T_w \cap T_{w'}|}{\min(|T_w|, |T_{w'}|)}. \tag{3}$$

Given a predefined threshold ε , if $IS(w, w') < \varepsilon$, we consider that w and w' are independent of each other. Then, we remove w' from P_w and w from $P_{w'}$, because w' (resp., w) is no longer a potential copier of w (resp., w'). Conversely, if $IS(w, w') > \varepsilon$, it is likely that w (resp., w') is a copier candidate of w' (resp., w). After traversing all worker pairs, each worker’s potential copied worker set P_w is updated. By introducing $IS(w, w')$, we can avoid further dependency analysis for independent workers, therefore reducing the time and space overhead of the entire copy detection. We need to further measure whether the two workers always make the same mistakes. So, for each w' remains in P_w , we introduce a metric, denoted as $DS(w, w')$, to measure the actual copy relationship between w and w' , which is calculated as follows:

$$DS(w, w') = \frac{\sum_{A_t^{l''} \in \mathbb{A}^w \cap \mathbb{A}^{w'}} f(v(D_t[A_t^{l''}]))}{\min(\sum_{A_t^l \in \mathbb{A}^w} f(v(D_t[A_t^l])), \sum_{A_t^{l'} \in \mathbb{A}^{w'}} f(v(D_t[A_t^{l'}])))}, \tag{4}$$

$$f(v(D_t[A_t^l])) = \lg(1 - v(D_t[A_t^l]) + v(D_t[A_t^*])), \tag{5}$$

where $D_t[A_t^l]$ depicts the encrypted answer selected by A_t^l , A_t^* is the correct answer consistency group, $v(\cdot)$ is a function calculates the answer veracity, which we will describe in Sect. 4.3, $f(\cdot)$ is a mapping function. It is clear that if $v(D_t[A_t^l]) = v(D_t[A_t^*])$, worker w and w' are correct on task t , so the task t is no longer a proof of the existence of a copy relationship between w and w' . Correspondingly, when $v(D_t[A_t^l]) < v(D_t[A_t^*])$, worker w and w' make the same wrong answer on task t , so it can be a proof of the existence of copy relationship between w and w' . In this case, the smaller $v(D_t[A_t^l])$ is, the greater the distance between this answer consistency group and the correct answer consistency group.

In this way, $DS(w, w')$ can measure how frequently worker w and w' make the same mistake and the severity of this mistake.

Here, we introduce another predefined parameter σ , if $DS(w, w') > \sigma$, the crowdsourcing server confirms a copy relationship between w and w' , then it compares r_w with $r_{w'}$ (we will describe worker weight calculation in Sect. 4.3). If $r_w < r_{w'}$, w is regarded as the copier, we remove w' from P_w . Otherwise, w' is the copier, we remove w from $P_{w'}$. After all workers' potential copier sets P_w are updated, we penalize the weight of each copier w' by the following equation:

$$r_w = r_w + \sum_{w' \in P_w} r_{w'} \cdot (-\lg(DS(w, w'))), \quad (6)$$

Note that it is possible for the estimated truth in the process to change along with iteration. It is also possible for the correct answer consistency group to change with iteration. Therefore, in CAPP-TD, $IS(w, w')$ only performs one operation while $DS(w, w')$ performs operations along with the iteration process.

4.3 Truth Discovery with Copy Detection

To enhance security and defend against adversary attacks in truth discovery, we propose a truth discovery method based on encrypted data. Our truth discovery method with copy detection comprises three components: worker weight update, copy detection, and encrypted truth inference. These components are iteratively executed until the convergence criterion is met.

Worker Weight Update: With fixed encrypted truth, we can identify which answer consistency group is the correct answer consistency group. r_w , is updated according to the number of correct answer consistency groups he or she is in.

$$r_w = -\log \frac{\sum_{A_t^l \in \mathbb{A}^w} \mathbb{1}(D_t[A_t^l], D_t[A_t^*])}{\sum_{w' \in W} \sum_{A_t^l \in \mathbb{A}^{w'}} \mathbb{1}(D_t[A_t^l], D_t[A_t^*])}, \quad (7)$$

among them, $\mathbb{1}$ is an Index function, which can be expressed as follows:

$$\mathbb{1}((A_t^l, A_t^*)) = \begin{cases} 0, & \text{if } D_t[A_t^l] \neq D_t[A_t^*] \\ 1, & \text{if } D_t[A_t^l] = D_t[A_t^*]. \end{cases}, \quad (8)$$

Encrypted Truth Inference: Based on fixed worker weights, the veracity of each answer is estimated by applying the following equation:

$$v(D_t[A_t^l]) = \frac{\sum_{w \in A_t^l} r_w}{\sum_{w' \in W_t} r_{w'}}. \quad (9)$$

Then for each task, we calculate $v(D_t[A_t^l])$ for each $l \in L$, and choose the answer consistency group with the highest $v(D_t[A_t^l])$ as the correct answer consistency group of task t . When the convergence criterion is met, we set

the encrypted answer of the correct answer consistency group $D_t[A_t^*]$ as the encrypted estimated truth \hat{O}^* .

$$D_t[A_t^*] = \arg \max_{l \in L} v(D_t[A_t^l]), \quad (10)$$

5 Experiment

5.1 Experimental Setup

We extensively experimented on two real-world datasets of AMT and a synthetic dataset to evaluate our proposed framework. All experiments were implemented and performed using Python version 3.9 on an AMD Ryzen 5 5600U CPU. We designated the lower-weighted worker in a pair with a copy relationship as the copier during the experiment.

Dataset: we used two public real-world crowdsourcing datasets¹, named NLP [3] and DOG [4], both were collected from AMT. Since the dataset is from real-world, there is a certain percentage of copiers naturally, which we will demonstrate in our experiments. *NLP*: the dataset comprises 1,000 tweets treated as tasks. Each tweet is evaluated by 20 workers in a crowdsourcing survey to determine its sentiment (positive or negative). This binary task involves 85

Table 2. Experimental result on two datasets.

Dataset	<i>PPTD</i>	<i>CRH</i>	<i>MV</i>	<i>TF</i>	Cosines
NLP	0.933	0.952	0.912	0.952	0.722
DOG	0.821	0.827	0.815	0.831	0.63
Dataset	<i>Bayes</i>	<i>AL</i>	<i>Sums</i>	<i>G-LCA</i>	<i>CAPP-TD</i>
NLP	0.95	0.944	0.946	0.923	0.956
DOG	0.82	0.826	0.828	0.83	0.839

workers and results in 20,000 records. *DOG*: the dataset consists of 807 tasks assigned to 109 workers. Each task involves identifying a specific dog breed from four given options. Each task is answered by 10 workers. Overall, the dataset contains 8,070 records. Then, We followed the Sybil attack [1] to inject malicious copiers into the NLP dataset to evaluate the capabilities of the algorithm, this synthetic dataset is called *NLP-S*.

Baseline Methods: In our experiments, since the novel scenario we address cannot use existing methods for copy detection, we utilized CAPP-TD without the copy detection module (PPTD) as the baseline method. Additionally,

¹ Due to economic and ethical reasons, we were unable to find datasets with private content, so we used these datasets for simulation.

we included CRH [5], TruthFinder [2] (TF), Cosines, Bayes, AverageLog (AL), Sums, GuessLCA [21] (G-LCA), Majority Voting (MV) as baseline methods.

Evaluation Metrics: *Accuracy (accu):* We used accuracy as the evaluation criterion. In the case of single-choice tasks, accuracy is defined as the ratio of estimated truth to ground truth. *Calculation cost:* We evaluated the time required for each module to run once and the number of iterations required for the framework to converge.

5.2 Comparative Studies

We conducted experiments to demonstrate the performance of CAPP-TD. We demonstrated the presence of copiers in the dataset and evaluated the effectiveness of our copy detection module. The average results from 10 experiments using two real-world datasets are presented in Table 2.

In comparing other methods on the two datasets, we found that PPTD is not the best, but CAPP-TD achieves the highest accuracy among all baseline methods. Firstly, this highlights copy relationships between workers are prevalent in the real world as the accuracy improved after incorporating the copy detection module. Secondly, our method outperformed other methods, which demonstrates that our approach preserves worker privacy, excludes copiers, and enhances accuracy. Lastly, it is worth noting that regardless of the algorithms used, the accuracy on the DOG dataset was generally lower than that on the NLP dataset, as shown in Table 2. This disparity was attributed to the sparsity of the DOG dataset, which is not our concern in this work, more attention will be paid in future work.

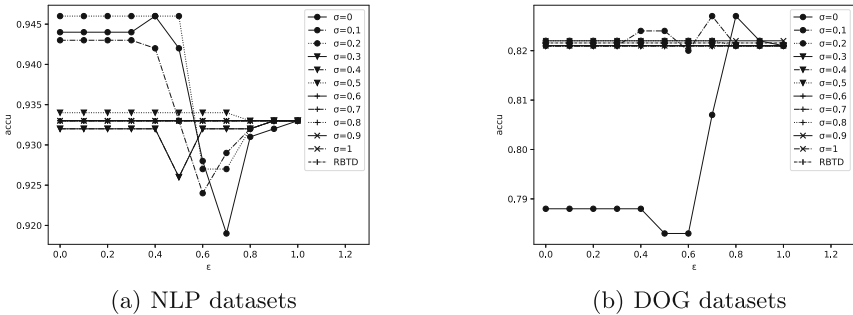


Fig. 2. Parameter impact on two real-world datasets.

5.3 Impact of Different Concerns

Accuracy Experiments: To analyze the impact of hyperparameters σ and ϵ on CAPP-TD compared to PPTD, we used PPTD as the baseline method. Each set of parameter settings ran 10 times, and the average was used as a result.

to ensure reliability. The experimental results shown in Fig. 2(a) and Fig. 2(b) represent NLP dataset and DOG dataset respectively.

Impact of ε : The variety of ε determines the proportion of worker relationships used for similarity calculation in determining copy relationships. When $\varepsilon = 0$, it meant that all relationships were included, which was unreasonable since not all workers have copy relationships. Setting ε served as a screening step to optimize the time complexity of the algorithm.

CAPP-TD exhibited better accuracy than PPTD when ε ranged from 0 to 0.6. However, accuracy decreased as ε increased. This is because the algorithm considers fewer worker relationships during the similarity calculation, potentially excluding copy evidence and leading to incorrect penalization of certain workers. When ε was between 0.8 and 1.0, the results were similar to PPTD, indicating that an excessively high value of ε caused CAPP-TD to perform similarly to PPTD. Therefore, selecting an appropriate ε value is crucial for balancing accuracy and efficiency in practical applications.

Impact of σ : The diversity of σ determined the level of penalty imposed on workers identified as copiers. $\sigma = 0$ means all copiers are penalized, while $\sigma = 1$ means no copiers are penalized.

By analyzing Fig. 2(a) and Fig. 2(b), we observed the impact of different σ on the accuracy of CAPP-TD. In NLP dataset (Fig. 2(a)), CAPP-TD improved accuracy compared to PPTD when σ ranged from 0 to 0.5, with the highest improvement at $\sigma = 0.2$. In DOG dataset (Fig. 2(b)), when $\sigma = 0$, the results were lower than the baseline due to data sparsity, resulting in unfair penalization of workers. Copy detection has a positive impact on accuracy for σ ranging from 0.1 to 0.2. However, as σ increased to 0.3 or higher, accuracy gradually decreased until it degraded to PPTD. Overall, CAPP-TD exhibited less fluctuation in accuracy on both datasets, highlighting its robustness to parameter variations. In the evaluation of copy detection, we set the parameters as $\varepsilon = 0.1$ and $\sigma = 0.2$ for the best performance in most experiments.

Table 3. Results with varying proportion of copiers.

Percentage of injection (%)	0	5	15	25	35	45	55
CRH	0.952	0.953	0.95	0.901	0.626	0.471	0.121
MV	0.912	0.906	0.877	0.735	0.659	0.453	0.121
TF	0.952	0.952	0.948	0.889	0.762	0.525	0.121
Cosines	0.722	0.722	0.721	0.708	0.665	0.552	0.272
Bayes	0.95	0.952	0.946	0.943	0.577	0.121	0.121
AL	0.944	0.947	0.936	0.797	0.595	0.46	0.121
Sums	0.946	0.95	0.938	0.849	0.626	0.471	0.121
G-LCA	0.923	0.919	0.919	0.791	0.601	0.467	0.121
CAPP-TD	0.945	0.942	0.932	0.926	0.752	0.655	0.201

Evaluation of Copy Detection: The evaluation results are presented in Table 3. We tested the resilience of CAPP-TD to malicious copiers using NLP-S dataset, with copiers injected in proportion to the total number of workers. Due to the space limit, we do not present the results of other datasets which are similar to the above.

Table 3 shows that CAPP-TD demonstrated strong resistance to high percentages of copier injection compared to other methods, emphasizing the effectiveness and efficiency of our proposed copy detection module. At low injection ratios, the accuracy of CAPP-TD can also be on average. In contrast, Cosines, despite showing slight effectiveness at a 55% injection ratio, performed significantly lower than the other methods overall, rendering it an ineffective solution to the malicious copiers problem.

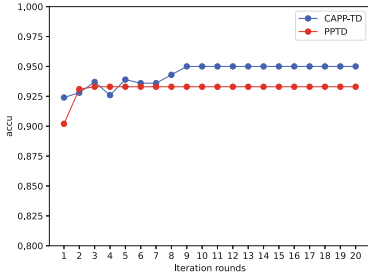
Table 4. Comparison of time cost experimental results (in seconds).

Module	CAPP-TD	PPTD
Encryption/Decryption module	0.09	0.09
Truth estimation Module	2.097	2.402
Weight update module	0.083	0.062
Copy detection module	0.302	None

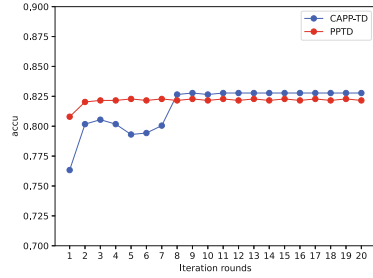
It’s worth noting that some methods show a slight increase in accuracy with a low percentage of copier injection. This suggests that better accuracy is achieved because copiers tend to copy high-quality workers, leading to a higher percentage of correct answers. This finding is also supported by research in [20].

Time Overhead: Experiments were conducted on the NLP dataset to compare CAPP-TD and PPTD. The results in Table 4 are the average values from 20 repeated runs. The results of the Dog dataset are similar. Figure 3 illustrates the evolution of the accuracy of CAPP-TD and PPTD after 20 rounds of running.

Table 4 shows no change of runtime in encryption and decryption with the copy detection module introduced. The copy detection module speeds up the truth estimation module by reducing the reliability of incorrect answer consistency groups when a copier is detected. However, weight updates take slightly longer due to considering copy relationships in the worker update module. Figure 3 shows that PPTD reached convergence with about 4 rounds on both datasets, while CAPP-TD required 6 more rounds on NLP dataset and 7 more rounds on DOG dataset to reach convergence, but the accuracy of convergence improved compared to PPTD. It can be seen that the addition of the copy detection module does not have much of an impact on the framework time overheads.



(a) NLP datasets



(b) DOG datasets

Fig. 3. Number of iteration rounds of the algorithm on two real-world datasets.

6 Security Analysis

Under the premise that the private key is transmitted through a secure channel, we can prove that our framework protects the privacy of users and estimated truth. The proof is as follows:

- i) In CAPP-TD, the decrypted similarity matrix contains only 0, 1, and -1 , which only shows the correlation of the workers on the tasks, and cannot be analyzed from this to find out what the specific answers are. Similarly, the answer consistency group contains only workers who have answered the same task, and it cannot be analyzed from this what the specific answers made by these workers are. These two data structures are the only plaintext in the framework. This proves that we do not leak user answers or user privacy while extracting data correlations from encrypted data.
- ii) In CAPP-TD, the semi-honest assumption allows entities in the framework to derive private information from the information they have, while our design ensures that all entities cannot derive. Our encryption algorithm is shown [22] to be incapable of leaking arbitrary information in the plaintext from the ciphertext, so workers and crowdsourcing server that have no secret keys get no private information. The data requester and the key distributor who has the secret key can only obtain the final estimated truth and the encrypted similarity matrix, respectively. Neither can obtain any privacy information through the information they possess. The key distributor is usually considered to be a trusted third party, typically a government agency. The data requester is usually a research organization that only needs the estimated truth. Both entities are considered to be trusted in most existing works. In this paper, we relax the restrictions on the assumptions about these two entities, thus allowing our framework to be adapted to broader scenarios.
- iii) CAPP-TD is resistant to adversaries, malicious copiers and blind copiers. The adversary, even if it intercepts all the information except the secret key, because of the Indistinguishability under chosen-plaintext attack (IND-CPA) of the encryption mechanism we employ, cannot derive the private information and the estimated truth. Malicious copiers are thought to follow the

Sybil attack to join forces to deliberately submit incorrect answers for some tasks while disguising themselves by answering randomly in other tasks. Blind copiers attempt to copy the answers of normal workers for financial gain, they naturally exist in the datasets. We have shown in our experiments that our method can find and eliminate all types of copiers, thereby improving accuracy.

7 Conclusion

To conduct copy-aware truth discovery in privacy-preserving crowdsourcing, we propose an anti-copy iterative model based on lightweight homomorphic encryption, called CAPP-TD. Firstly, we design a lightweight privacy-preserving mechanism utilizing Paillier homomorphic encryption. This mechanism preserves data correlation while minimizing communication and computation overhead. Subsequently, we propose an iterative truth discovery method capable of detecting copy relationships in encrypted data and excluding copiers during the truth inference process, leading to improved accuracy. To the best of our knowledge, CAPP-TD is the first copy-aware truth discovery under privacy-preserving crowdsourcing framework. We extensively evaluated CAPP-TD on real-world datasets focusing on single-choice tasks. The experiments confirmed the effectiveness, accuracy, and practicality of CAPP-TD.

Acknowledgments. This work was supported by Shanghai Science and Technology Commission (No. 22YF1401100), Fundamental Research Funds for the Central Universities (No. 22D111210), and National Science Fund for Young Scholars (No. 62202095, No. 62102058).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Yuan, D., Li, G., Li, Q., et al.: Sybil defense in crowdsourcing platforms. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM), Singapore, Singapore, pp. 1529–1538 (2017)
2. Tang, J., Fu, S., Liu, X., et al.: Achieving privacy-preserving and lightweight truth discovery in mobile crowdsensing. *IEEE Trans. Knowl. Data Eng.* **34**(11), 5140–5153 (2021)
3. Zheng, Y., Li, G., Li, Y., et al.: Truth inference in crowdsourcing: is the problem solved? *Proc. VLDB Endow.* **10**(5), 541–552 (2017)
4. Zhou, D., Basu, S., Mao, Y., et al.: Learning from the wisdom of crowds by minimax entropy. In: Proceedings of the 2012 Advances in Neural Information Processing Systems (NIPS), Lake Tahoe, NV, USA, pp. 2195–2203 (2012)
5. Li, Q., Li, Y., Gao, J., et al.: Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation. In: Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data (SIGMOD), Snowbird, Utah, USA, pp. 1187–1198 (2014)

6. Li, Y., Xiao, H., Qin, Z., et al.: Towards differentially private truth discovery for crowd sensing systems. In: Proceedings of the 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS), Coimbatore, Tamilnadu, India, pp. 1156–1166 (2020)
7. Pang, X., Wang, Z., Liu, D., et al.: Towards personalized privacy-preserving truth discovery over crowdsourced data streams. *IEEE/ACM Trans. Networking* **30**(1), 327–340 (2021)
8. Sun, P., Wang, Z., Wu, L., et al.: Towards personalized privacy-preserving incentive for truth discovery in mobile crowdsensing systems. *IEEE Trans. Mob. Comput.* **21**(1), 352–365 (2020)
9. Ding, X., Lv, R., Pang, X., et al.: Privacy-preserving task allocation for edge computing-based mobile crowdsensing. *Comput. Electr. Eng.* **97**, 107528 (2022)
10. Jiang, L., Niu, X., Xu, J., et al.: Incentivizing the workers for truth discovery in crowdsourcing with copiers. In: Proceedings of the 39th International Conference on Distributed Computing Systems (ICDCS), Dallas, Texas, USA, pp. 1286–1295 (2019)
11. Wang, X., Sheng, Q.Z., Fang, X.S., et al.: An integrated Bayesian approach for effective multi-truth discovery. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (CIKM), Melbourne, Australia, pp. 493–502 (2015)
12. Dong, X.L., Berti-Equille, L., Srivastava, D.: Integrating conflicting data: the role of source dependence. *Proc. VLDB Endow.* **2**(1), 550–561 (2009)
13. Dong, X.L., Berti-Equille, L., Hu, Y., Srivastava, D.: Global detection of complex copying relationships between sources. *Proc. VLDB Endow.* **3**(1–2), 1358–1369 (2010)
14. Zhang, C., Zhu, L., Xu, C., et al.: LPTD: achieving lightweight and privacy-preserving truth discovery in CIoT. *Futur. Gener. Comput. Syst.* **90**, 175–184 (2019)
15. Dong, C., Wang, Y., Aldweesh, A., et al.: Betrayal, distrust, and rationality: Smart counter-collusion contracts for verifiable cloud computing. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS), New York, NY, USA, pp. 211–227 (2017)
16. Franklin, M.J., Kossmann, D., Kraska, T., et al.: CrowdDB: answering queries with crowdsourcing. In: Proceedings of the 2011 ACM SIGMOD International Conference on Management of data (SIGMOD), New York, NY, USA, pp. 61–72 (2011)
17. Wu, G., Zhou, L., Xia, J., et al.: Crowdsourcing truth inference based on label confidence clustering. *ACM Trans. Knowl. Discov. Data* **17**(4), 1–20 (2023)
18. Chang, J.C., Amershi, S., Kamar, E.: Revolt: collaborative crowdsourcing for labeling machine learning datasets. In: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI), New York, NY, USA, pp. 2334–2346 (2017)
19. Li, X., Dong, X.L., Lyons, K.B., et al.: Scaling up copy detection. In: Proceedings of 31st International Conference on Data Engineering (ICDE), Seoul, Korea, pp. 89–100 (2015)
20. Chen, P., Sun, H., Fang, Y., et al.: CONAN: a framework for detecting and handling collusion in crowdsourcing. *Inf. Sci.* **515**, 44–63 (2020)

21. Waguih, D.A., Berti-Équille, L.: Truth Discovery Algorithms: An Experimental Evaluation. Doctoral dissertation, Qatar Foundation; QCRI (2014)
22. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_16



FCFL: A Fairness Compensation-Based Federated Learning Scheme with Accumulated Queues

Lingfu Wang¹, Zuobin Xiong², Guangchun Luo¹, Wei Li³, and Aiguo Chen⁴(✉)

¹ School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, China

² Department of Computer Science, University of Nevada-Las Vegas, Las Vegas, USA

³ Department of Computer Science, Georgia State University, Georgia, USA

⁴ Institute of Intelligent Computing, University of Electronic Science and Technology of China, Chengdu, China

agchen@uestc.edu.cn

Abstract. The surge of ubiquitous data underscores the need for Federated learning (FL), which allows distributed data entities to learn a global model orchestrally without revealing their private local data, ensuring the privacy and security of users. However, the performance of the trained global model on individual clients is impaired by the heterogeneous nature of the client's local data, exposed as the performance unfairness in FL. Such unfairness issues grab the research community's attention and a few recent works embark upon fair solutions via reweighting clients during aggregation but overlooking the impact of client selection for aggregation. To fill this gap, in this paper, a Fairness Compensation-based FL scheme (FCFL) is proposed to alleviate the unfairness amongst clients. In particular, the unfairness of each client during the FL training process is estimated as the accuracy difference between local performance and global performance, and accumulated queues are calculated for the cumulative unfairness value in each round. In addition, a fairness compensation FL method is devised, which can select participating clients dynamically and adjust the aggregation weights adaptively in each round to guarantee fairness in the training process. Specifically, the proposed FCFL scheme is a flexible framework with tunable parameters and the FedAvg algorithm is its special case when $\alpha = 0$. Finally, intensive experiments are conducted on three benchmark datasets with different settings, demonstrating that the FCFL outperforms the state-of-the-art baselines by improving the fairness metric up to 30.4% while maintaining a competitive accuracy performance. The source code is available at <https://github.com/wlffff/FCFL>.

Keywords: Federated learning · Performance fairness · Data heterogeneity · Client selection · Weighting strategy

1 Introduction

The advancement of Artificial Intelligence (AI) is driven by the ubiquitous data generated from a wealth of devices, however, conventional machine learning typically adopts a centralized mode in data collection and training, which poses

significant challenges in multiple faucets. Firstly, uploading large amounts of data incurs communication and storage costs [14,27] of burdened infrastructure. Secondly, with the ever-increasing privacy and security concerns [26], it is hard to convince unwilling data owners to share their raw data with an untrusted service provider under this centralized paradigm [24]. Besides, pressing regulations and laws are enforced by many governments on private data with stricter data management and stewardship, such as the General Data Protection Regulations (GDPR) from the European Union and Personal Information Protection and Electronic Documents Act (PIPEDA) from Canada. To solve this dilemma, Federated Learning (FL), as a promising solution is proposed by Google recently [15]. FL is a distributed machine learning paradigm consisting of a server and multiple clients, which can learn a global model on the server side without access to clients' local private data. Since the original data is kept locally rather than being sent to a remote server, the challenges of communication, storage, and privacy are somehow mitigated by FL, and thus broad application scenarios are fertilized, including medical images [9], recommendation systems [22], and the Internet of Things (IoT) [25].

Yet, FL is not the silver bullet, and some issues have emerged in recent years. In this paper, we study the fairness aspect of FL, which is one of the major concerns of FL that impedes the realistic application. As illustrated in Fig. 1, since the global model is trained based on unknown local datasets of clients, where divergence may exist amongst client local data and model. Therefore the performance of the global model may vary across the diverse clients, causing unfair performance (e.g., accuracy) as shown on the right side of Fig. 1. Specifically, though the global model performs well on average, such unfairness is manifested in those clients (referred to as vulnerable clients) who receive lower accuracy due to biased client selection or minority in data representations.

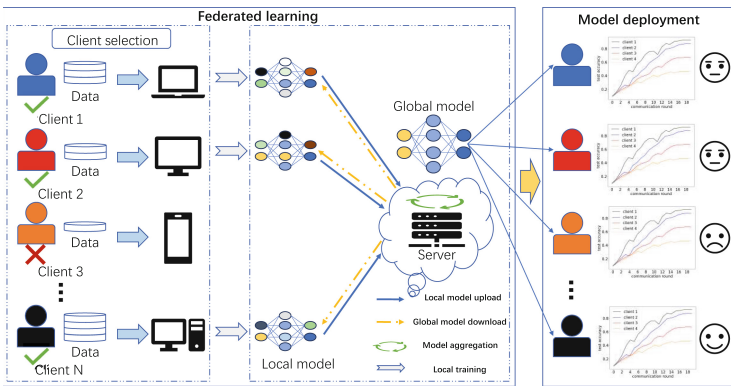


Fig. 1. (Left): Example of horizontal FL. (Right): Performance fairness of FL.

To alleviate unfairness in FL, AFL is proposed by Mohri et al. [16] to optimize the worst-performing client by minimax optimization. Inspired by fair resource

allocation in wireless networks, Li et al. [11] presented the q-FedAvg method, which introduces a parameter q to reweight the loss of different clients. Zhao et al. [30] proposed DRFL, which dynamically adjusts the weight assigned to each client and is more flexible in parameter tuning. Although these works can alleviate unfairness by adjusting weights, they often ignore the impact of client selection in fairness. In addition, the client selection in existing methods is either randomly done [10, 15] or based on the local data amount [8, 28], which may lead to poor local performance of the global model. Therefore, it is a challenge to design a fair FL scheme that handles client selection without incurring a negative impact on the client side.

In this work, we propose the **F**airness **C**ompensation-based **FL** (FCFL) to alleviate unfairness in FL. In each training round, the server updates the unfairness queue of each client, where the queues are utilized for client selection and aggregation reweighting based on cumulative unfairness value. To our best knowledge, this is the first work to harness fair client selection to achieve performance fairness of FL. The contributions of this paper are summarized as follows:

- The accumulated unfairness of clients is defined during the process of FL and unfairness queues are maintained to measure if a client is treated fairly.
- Based on the accumulated unfairness queues, a fairness compensation method FCFL is designed to balance the client selection and aggregation reweighting, which can improve the fairness of FL.
- Evaluations on three datasets are conducted to confirm the advantages of our method and compare it with state-of-the-art methods. The experimental results demonstrate the fairness and effectiveness of our proposed FCFL.

The rest of the paper is organized as follows. The related work on the fairness of FL is discussed in Sect. 2. Then, the proposed FCFL method is detailed in Sect. 3 with descriptions of client selection and aggregation reweighting. Empirical evaluation is presented in Sect. 4, and finally Sect. 5 concludes the paper.

2 Related Work

Fairness in FL can be divided into collaborative fairness [13, 29], group fairness [3, 4], selection fairness [7, 18], and performance fairness [11], as per different fairness goals. In particular, collaborative fairness means that clients should be rewarded in proportion to their contributions; group fairness aims to minimize disparities among different groups based on sensitive attributes (e.g., gender and race); selection fairness ensures that each client has a fair chance of being selected to participate in training; and performance fairness seeks to reduce the variance of global model accuracy across clients. This paper mainly focuses on performance fairness. For a comprehensive review of the fairness in FL, please refer to the survey paper [19] by Shi et al.

Table 1. Related work on performance fairness of FL.

References	Method	Dataset
AFL (2019) [16]	Minimax optimization	Fashion MNIST; Adult; Cornell movie; PTB
q-FedAvg (2019) [11]	Reweighting	Synthetic; Vehicle; Sent 140; Shakespeare
FedGini (2023) [12]	Objective function	Synthetic; CIFAR-10; Sent 140
DRFL (2022) [30]	Reweighting	Synthetic; Fashion MNIST; Adult
Ada-FFL (2023) [2]	Reweighting	Synthetic; Vehicle; Sent 140
FedFa (2022) [8]	Reweighting	MNIST; FEMNIST; Synthetic; Sent 140; Shakespeare
PG-FFL (2022) [21]	Reweighting	Fashion MNIST; CIFAR-10; CIFAR-100
FedFV (2021) [23]	Gradient projection	MNIST; Fashion MNIST; CIFAR-10
GIFAIR (2023) [28]	Reweighting; Objective function	FEMNIST; Shakespeare;
FedMGDA (2022) [6]	Multi-objective optimization	Fashion MNIST; CIFAR-10; Shakespeare; Adult
FedMDFG (2023) [17]	Multi-objective optimization	MNIST; Fashion MNIST; CIFAR-10; CIFAR-100
FairWire+ (2024) [5]	Multi-objective optimization	CIFAR-10; CIFAR-100; FEMNIST

2.1 Performance Fairness in FL

The vanilla FedAvg algorithm aggregates client local models by calculating a weighted average based on the amount of training data [15], therefore causing significant differences in model accuracy due to the data heterogeneity of different clients. As countermeasures, AFL [16] is the first approach to improve the fairness of FL, which used minimax optimization to maximize the performance of the worst-performing device. However, this method cannot guarantee generalization in large-scale settings. To improve the scalability of AFL, researchers have proposed the q-FedAvg [11] method by introducing the parameter q for clients reweighting to achieve better fairness. Since q-FedAvg, performance fairness has become a pivotal problem in FL, and many methods have been proposed to improve the fairness of q-FedAvg, including designing novel objective functions, reweighting, eliminating gradient conflicts, and multi-objective optimization.

Designing Novel Objective Functions. FedGini [12] modified the objective function to improve fairness by introducing a Gini penalty term. GIFAIR [28] achieved fairness by introducing a regularization term to penalize loss differences among client groups. Ada-FFL [2] improved the objective function of q-FedAvg by introducing regularized local loss terms and Frobenius distance to design an adaptive fair FL.

Reweighting. FedFa [8] combined the training accuracy and frequency to design an appropriate weight selection algorithm and adopted double momentum gradient optimization to accelerate the model’s convergence. PG-FFL [21] used reinforcement learning to achieve reweighting, and automatically learned strategies

through a reward function constructed based on Gini coefficients and accuracy. Building on the q-FedAvg method, DRFL [30] proposed a novel approach, which can dynamically adjust the weight assigned to clients. The regularization term in GIFAIR [28] can also be viewed as a dynamic client reweighting technique that can adaptively assign higher weights to clients with poor performance.

Eliminating Gradient Conflicts. Researchers have found that conflicting gradients are one of the reasons for unfairness in FL. To address this issue, FedFV [23] first used cosine similarity to detect gradient conflicts, and then iteratively eliminated conflicts by modifying the direction and magnitude of gradients, thereby improving the fairness of FL.

Multi-objective Optimization. FedMGDA [6] pioneered the formalization of FL into multi-objective optimization and proposed a novel fair FL scheme using a multiple gradient descent algorithm. Under the guidance of multi-objective optimization, FedMDFG [17] and FairWire+ [5] are also proposed. FedMDFG can find a fair descent direction by adding a fair-driven objective, and the line search strategy can ensure an appropriate step size. These two major designs guarantee the fairness and robustness of the scheme. FairWire+ considered the inherent noise induced by wireless channels and designed an algorithm based on noisy gradients, which can find a common descent direction for all clients. We summarize the current fair FL for performance fairness in Table 1.

2.2 Client Selection in FL

Although the above methods can relieve unfairness through various approaches, they ignore the possible impact of client selection on the fairness of FL. Client selection is also an important research topic, which can achieve different goals. For instance, Power-of-Choice [1] identified clients with the highest loss in each round and included them in training to boost model performance. GreedyFed [20] selected clients with the highest contribution based on the Shapley value, improving model accuracy and convergence speed. By utilizing Lyapunov optimization, FairFedCS [18] achieved better selection fairness. In this vein, we aim to study how client selection can improve performance fairness in this work.

3 Fairness Compensation Federated Learning (FCFL)

In this section, we present the original fair FL method, FCFL, with problem setting, proposed algorithms, and analysis.

3.1 Problem Setting

As shown in Fig. 1, general FL has the following three steps: (1) client selection, (2) local training, and (3) weighted aggregation. Since the distribution of local datasets is different, the performance of the global model varies notably across different clients. This phenomenon is referred to as performance fairness. To be precise, the fairness of the global model can be defined as follows.

Definition 1. (Fairness of performance distribution [11]). A model θ_1 is said to be fairer than θ_2 if the accuracy of θ_1 on the N clients $\{a_1, a_2, \dots, a_N\}$ is more *uniform* than that of θ_2 on the N clients.

In this work, we use the variance of the accuracy on all clients as a measure of fairness, and the goal of our work is to reduce the variance while maintaining a similar average accuracy of the global model. To achieve this goal, client selection is a crucial aspect but is overlooked by most existing works. Intuitively, FL selects clients to participate in training based on the amount of local data [8, 28], which leads to the global model being biased towards clients with more data. We visualize this issue on the MNIST dataset and the CNN model in Fig. 2. Here, the Dirichlet function is used to partition the dataset into 20 clients, where 2 clients are selected for each training round. Figure 2(a) shows the distribution of local data for each client, and it can be seen that the amount and classes of data on each client differ significantly (such as clients 3, 10, 13), which simulates the data distribution in real-world scenarios. Using the data amount-based client selection method will reduce the selected times of these vulnerable clients in training (as can be seen from the green bin in Fig. 2(b)). Random selection has an equal chance of selecting each client, but it produces a poor performance of the global model on vulnerable clients. Our unfairness-based selection prioritizes the selection of vulnerable clients and finally can achieve performance fairness. Taking client 3 as an example, it is rarely selected in the amount-based method due to the limited local data, while our unfairness-selection method selects clients based on unfairness in each round, with client 3 being selected significantly more frequently. Note that unfairness-based selection selects clients based on cumulative unfairness, so it may be possible for vulnerable clients to have fewer choices than random ones. However, our proposed weight allocation based on cumulative unfairness will give vulnerable clients more weight, thereby improving the fairness of the scheme.

3.2 Overview of FCFL

To achieve the goal of fairness, we first investigate the reasons for performance fairness in FL. As demonstrated above, due to the data heterogeneity, some vulnerable clients cannot be selected fairly, such as client 3 in Fig. 2(b). To alleviate unfairness, an intuitive approach is to compensate for these vulnerable clients based on their unfairness level. To maintain the clients' computation cost, we focus on improving the selection ratio and assigning more aggregation weights for vulnerable clients. Based on this idea, we propose FCFL which considers both client selection and aggregation reweighting in the FL progress. The framework of FCFL is depicted in Fig. 3. In each training round, all clients first upload the local performance (i.e., the accuracy of the global model on local clients) to the server, and then the server updates the accumulated unfairness queue to select clients and calculate aggregation weights. Next, the selected clients perform local training and upload the local model and accuracy. Finally, the server completes the aggregation and estimates the global performance for the next round of FL

training. Comprehensive explanations of this approach will be provided in the subsequent sections.

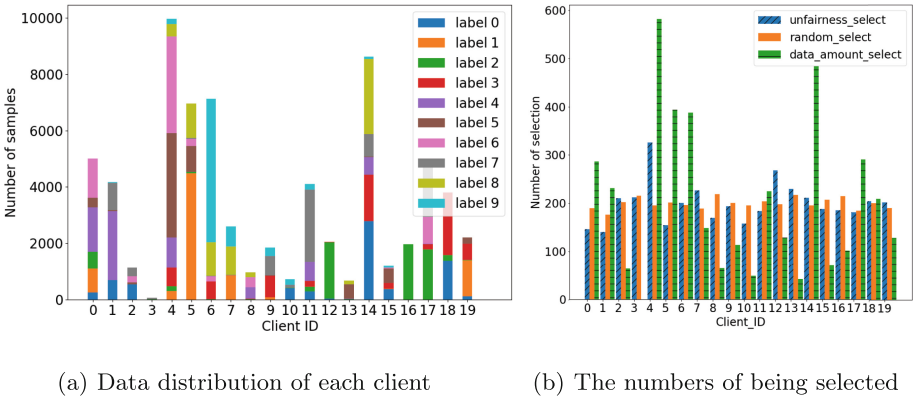


Fig. 2. The impact of data distribution on client selection methods.

3.3 Accumulated Unfairness Queues

In FL, performance fairness is evaluated by the accuracy differences of the global model on different local clients. However, the global model is not available during the training process until aggregation is performed. Therefore, to approximate the unfairness of clients in training rounds, the unfairness level is measured by the difference between the estimated global model accuracy and evaluated local accuracy, which can be calculated as follows.

$$u f_i^t = \begin{cases} \widetilde{Acc}^t - Acc_i^t, & \text{if } \widetilde{Acc}^t > Acc_i^t \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where Acc_i^t represents the evaluated local accuracy of client i in round t and $\widetilde{Acc}^t = \sum_{i=1}^m \omega_i^t \widetilde{Acc}_i^t$ is the estimated global model accuracy in round t , in which \widetilde{Acc}_i^t denotes the local training accuracy of client i in round t and m is the number of selected clients. In this paper, we assume that the server does not have access to validation data, which is realistic in the actual applications, so the performance of the global model can only be obtained through estimation. $u f_i^t$ is the unfairness level of the client i in round t , which is a cumulative value that reflects whether the client i has been treated fairly or not so far and indicates the priority of each client to be selected for training by the FCFL algorithm.

To track the cumulative unfairness of all clients during training, we introduce a queue $Q_i(t)$ to store the unfairness value of each client i in round t as described in the following formula. The intuition of this queue is to track the cumulative

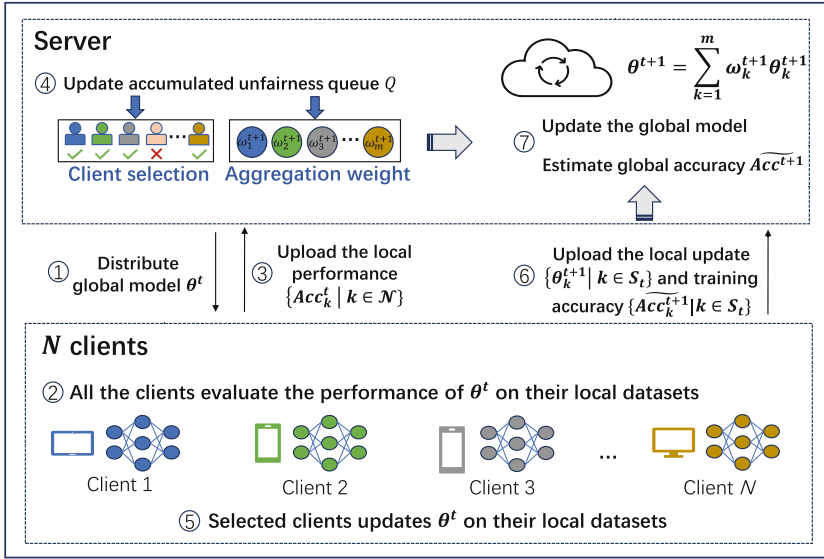


Fig. 3. The framework of FCFL.

unfairness of each participant, providing a basis for subsequent client selection and weight allocation.

$$Q_i(t) = \max \{ Q_i(t-1) + \alpha uf_i^t - \omega_i^t \cdot \mathbf{1}_{[x_i(t-1)=1]}, 0 \}, \quad (2)$$

where $x_i(t-1) \in \{0, 1\}$ indicates whether client i has been selected in the $(t-1)$ -th round (1=yes, 0=no). $\mathbf{1}_{[condition]}$ is an indicator function, which equals 1 if [condition] is true and 0 if not. α is a hyper-parameter that controls fairness. The design rationale of $Q_i(t)$ are as follows to facilitate vulnerable clients:

- For clients with **low-accuracy**, $Q_i(t)$ is a cumulative value that reflects the overall unfairness level in each round.
- For clients who are **not selected**, the indicator function $\mathbf{1}_{[condition]}$ is 0, providing more unfairness increment by αuf_i^t .
- For clients with **small weight**, the cumulative unfairness value will have a small penalty $-\omega_i^t$, resulting in a relatively high $Q_i(t)$.

Specifically, a client who has not been treated fairly (low-accuracy, not selected, or small weight) will have a higher $Q_i(t)$, and our FCFL algorithm will compensate these clients based on $Q_i(t)$ by client selection and aggregation reweighting later.

3.4 Client Selection and Aggregation Reweighting

Client selection is the first crucial step towards fair FL. After updating the accumulated unfairness queue, the server can select the top- m clients with the

highest $Q_i(t)$ from the overall clients set \mathbf{N} to perform local training. This step guarantees that the vulnerable clients with higher cumulative unfairness level will secure their chance to be selected by the FL system. After client selection and local training, the other crucial step is to allocate more weight for these clients to improve their contribution in aggregation. The adjusted aggregation weight is calculated as follows:

$$\omega_i^{t+1} = \begin{cases} \frac{n_i}{\sum_{i=1}^m n_i}, & \text{if } Q_1(t) = \dots = Q_m(t) = 0 \\ \frac{Q_i(t)}{\sum_{i=1}^m Q_i(t)}, & \text{otherwise} \end{cases} \quad (3)$$

where n_i is the data amount of client i and $Q_i(t)$ is the cumulative unfairness value. When the cumulative unfairness of all selected clients is 0, the aggregation weight is proportional to the local data amount, which is used to initialize weight in the beginning. Otherwise, the higher the client's unfairness value, the higher the aggregation weight it gets, which ensures that vulnerable clients have a greater influence on the aggregated global model.

Remark. It is important to note that selecting clients based solely on unfairness may lead to the global model being biased towards vulnerable clients with rare datasets, which can be another form of unfairness and ultimately lead to a decrease in the average accuracy of the global model across all clients. To address this problem, we introduce a hyper-parameter r to balance the two kinds of unfairness. This is achieved by using a random selection method to select a portion of r clients and using our client selection method to select the remaining. By doing so, our FCFL can proactively improve fairness while ensuring global accuracy. The influence of the hyper-parameter r is evaluated as well in Sect. 4.4.

The process of the proposed FCFL is illustrated in Algorithm 1. In each round, our FCFL selects a set of clients S_t to participate in the training through an additional communication round and determines the aggregation weights based on the unfairness queues. For each client, multiple training steps are performed, and then the updated parameters and local training accuracy are uploaded to the server. Finally, the server aggregates the parameters and estimates the performance of the global model. Note that in 0-th round, since $\{Q_i(t) = 0 | i \in \mathbf{N}\}$ is initialized fairly, selecting the top- m clients becomes a random selection method, and the aggregation weight is proportional to the amount of data.

Remark. The FedAvg algorithm can be seen as a special case of our FCFL. When α in Eq. (2) is 0, the cumulative unfairness value of all clients $Q_i(t)$ is 0 in every round. Hence, client selection is random and the aggregation weight is proportional to the amount of data. As α increases, the unfairness uf_i^t imposes more influence in $Q_i(t)$, which will have a higher chance of being selected and receiving higher aggregation weights, thus improving the fairness of FL.

Algorithm 1. Fairness Compensation Federated Learning (FCFL)**Input:** clients set \mathcal{N} ; communication rounds T ; local epochs E ; learning rate η **Output:** global model θ^{T+1} **Server executes:**initialize global model θ^0 **for** each round $t = 0, 1, 2, \dots, T$ **do**distribute global model θ^t to all clientsall client evaluate θ^t and upload local accuracy $\{Acc_i^t | i \in \mathcal{N}\}$ to the server**if** $t == 0$ **then**initialize unfairness queues: $Q_1(0) = Q_2(0) = \dots = Q_N(0) = 0$ select m clients to constitute subset S_t according to the selection methodinitialize the aggregation weight $w_i^{t+1} = n_i / \sum_{i=1}^m n_i, i \in S_t$ **else**calculate the unfairness of all clients via **Eq. (1)**update the accumulated unfairness queue via **Eq. (2)**select m clients to constitute subset S_t according to the selection methodcalculate the aggregation weight via **Eq. (3)**:**end if****for** each client $i \in S_t$ **do** $\theta_i^{t+1}, \widetilde{Acc}_i^{t+1} \leftarrow \mathbf{ClientUpdate}(i, \theta^t)$ **end for**

update global model parameters:

$$\theta^{t+1} = \sum_{i=1}^m \omega_i^{t+1} \theta_i^{t+1}$$

estimate global performance:

$$\widetilde{Acc}^{t+1} = \sum_{i=1}^m \omega_i^{t+1} \widetilde{Acc}_i^{t+1}$$

end for**ClientUpdate**(i, θ^t): // Run on client i client $i \in S_t$ updates θ^t for E epochs with step size η to obtain θ_i^{t+1} client $i \in S_t$ evaluates θ_i^{t+1} on local datasets to obtain \widetilde{Acc}_i^{t+1} client sends θ_i^{t+1} and \widetilde{Acc}_i^{t+1} to the server**3.5 Analysis of Communication and Computation Overhead**

The major bottlenecks of FL are the communication cost between server and edge devices [8] as well as the local client computation power. Compared with the FedAvg algorithm, our FCFL does not introduce too much communication overhead as analyzed below. In Algorithm 1, although we have additional communication from clients, the client only needs to upload the local accuracy Acc_i^t once, which costs 8 more bits per round. This communication is necessary to calculate the unfairness queue and select clients to participate in training. After local training on the client, each client $i \in S_t$ sends θ_i^{t+1} and \widetilde{Acc}_i^{t+1} to the server, where the accuracy \widetilde{Acc}_k^{t+1} is an extra cost for communication but with

small size. Overall, our FCFL only involves a few more bits of communication cost per round.

We then analyze the computation overhead of FCFL. On the client side, client devices require additional accuracy calculations (evaluating the performance of the global model on local datasets), and the remaining calculations are the same as in FedAvg. To save local computation cost, instead of evaluating the \widetilde{Acc}_k^{t+1} by going through the entire local dataset, we can use mini-batch samples from the local dataset to obtain an estimated accuracy. On the server side, the calculation of unfairness, cumulative unfairness, aggregated weights, and global performance estimation can be completed through simple arithmetic operations within $O(N)$. Considering that the server typically has high computing power, such computational cost is negligible and will not impact the bottleneck to FCFL. Empirical results of FCFL’s efficiency are presented in Sect. 4.3.

4 Experiments

In this section, the performance of FCFL is compared with other methods for different perspectives, including fairness 4.2, efficiency 4.3, and hyperparameter 4.4. In addition, we conduct ablation experiments in Sect. 4.5 to verify the impact of our client selection and reweighting methods.

4.1 Experimental Settings

All experiments are conducted on three public datasets: MNIST and CIFAR-10 with 100 local clients, and Shakespeare with 31 local clients. We only consider non-IID scenarios since heterogeneous non-IID data distribution is the reason for performance fairness. To simulate this scenario, we design three settings to allocate data to the clients. (1) We sort all data samples based on labels and then split them into 200 shards, where each client randomly picks 2 shards without replacement. (2) We utilize the Dirichlet function to set different levels of non-IID local clients [23]. (3) In *The Complete Works of William Shakespeare* [15], each speaking role in each play is treated as a device, We subsample 31 speaking roles following the setting in [11]. We randomly divide the data for each local client into 8 : 2 for training and testing, respectively.

Training. Three models, MLP, CNN, and RNN, are adopted for the experiments. For MNIST, we use a CNN which contains two convolutions and maximum pooling. We use an MLP which contains a hidden layer on CIFAR-10. For Shakespeare, we use an RNN model which contains an embedding layer and an LSTM layer. All the code is implemented in PyTorch to simulate a federated network with 1 server and several clients, where 10% of clients are selected for training in each round. The local batch size is 64, the local epoch is 1, the server’s momentum factor is 0.5, and the number of communication rounds for MNIST and CIFAR-10 is 2000 and for Shakespeare is 500. Note that a communication round refers to the process of completing a model update through interaction between the server and the clients.

Baselines. We compare FCFL with the classic method FedAvg [15] and various state-of-the-art fairness methods in FL, including q-FedAvg [11], FedFa [8], and GIFAIR [28]. Based on the code provided by their authors, we directly rewrite the code for comparison. The presented results are averaged from 5 runs with different random seeds.

4.2 Fairness of FCFL

We compare the proposed FCFL with four FL algorithms, FedAvg, q-FedAvg, FedFa, and GIFAIR to verify the fairness of our method. The value of q in q-FedAvg is set to $\{0.1, 0.2, 0.5, 1.0, 2.0, 5.0\}$, the value of accuracy weight and frequency weight in FedFa is set to $\{(0.4, 0.6), (0.5, 0.5), (0.6, 0.4)\}$, and the parameter λ in GIFAIR is set to $\{0.3, 0.5, 0.7\}$. We take the best performance of each method for the comparison. The derived variance and accuracy are displayed in Table 2, from which we can see that our FCFL method produces the lowest variance of **11.03** on MNIST, **114.59** on CIFAR-10, and **67.48** on Shakespeare. Moreover, taking the CIFAR-10 dataset as an example, compared with q-FedAvg($q=2.0$), FedFa, and GIFAIR, our FCFL reduces the variance by **23.4%**, **30.4%**, **27.7%**, respectively. Similar variance reductions can also be seen on the MNIST and Shakespeare datasets. Since variance is an important metric of fairness, it indicates that our FCFL can achieve the fairest performance among all baselines. In addition, the accuracy of the worst 10% client of FCFL is significantly higher than other experiments. In the CIFAR-10 dataset, compared with the second-best method q-FedAvg ($q = 2.0$) in baselines, the worst 10% performance is increased from 24.98 to **28.03**. Similarly, in the MNIST dataset, the worst 10% performance increased from 88.63 to **89.17**, and in Shakespeare, the worst 10% performance increased from 37.92 to **38.55**. This confirms our FCFL has shown great improvement in protecting unfairly treated clients. As for the global average accuracy, our FCFL can reach 96.06%, 46.12%, and 50.55% in MNIST, CIFAR-10, and Shakespeare respectively, which is very competitive (around 1% difference) to other baselines. In the best 10% accuracy, our FCFL method is a bit lower than some baselines since we emphasize more focus on vulnerable clients and deliver much better fairness.

To summarize, the FCFL method can achieve better fairness in FL while maintaining competitive average accuracy in most cases, which will attract more minority clients to participate in FL, thereby expanding FL applications.

4.3 Efficiency of FCFL

We also record the trend of loss value and the test accuracy of the global model for each round in the MNIST and CIFAR-10 datasets and plot them in Fig. 4. As depicted in Fig. 4(a), the loss of proposed FCFL reduces fast as the communication round increases, which affirms that FCFL can converge as fast as FedAvg, FedFa, and GIFAIR, and is significantly faster than q-FedAvg. In Fig. 4(b), the test accuracy of FCFL increases rapidly and reaches a convergence value after

Table 2. Statistics of the test accuracy distribution on different datasets.

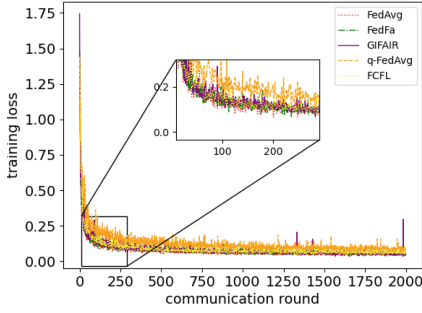
Dataset	Method	Accuracy	Best 10%	Worst 10%	Variance
MNIST	FedAvg	95.96	100.00	87.47	15.06
	q-FedAvg— $q=0.2$	96.09	100.00	88.40	12.58
	FedFa— $\alpha=0.6, \beta=0.4$	96.23	100.00	88.37	12.61
	GIFAIR— $\lambda=0.5$	96.12	100.00	88.63	12.72
	FCFL— $\alpha=0.3, r=0.4$	96.06	100.00	89.17	11.03
CIFAR-10	FedAvg	46.35	68.67	20.16	178.93
	q-FedAvg— $q=2.0$	47.14	66.81	24.98	149.50
	FedFa— $\alpha=0.6, \beta=0.4$	46.64	68.10	23.19	164.68
	GIFAIR— $\lambda=0.5$	46.61	67.02	23.18	158.36
	FCFL— $\alpha=0.3, r=0.6$	46.12	65.39	28.03	114.59
Shakespeare	FedAvg	49.16	70.65	35.34	89.87
	q-FedAvg— $q=2.0$	50.24	69.77	37.92	75.99
	FedFa— $\alpha=0.5, \beta=0.5$	49.03	69.06	36.23	79.54
	GIFAIR— $\lambda=0.3$	50.01	68.50	36.24	78.25
	FCFL— $\alpha=0.1, r=0.6$	50.55	68.74	38.55	67.48

1000 rounds. Similar results can be observed from the CIFAR-10 dataset, which states that our FCFL method has a reasonable convergence speed.

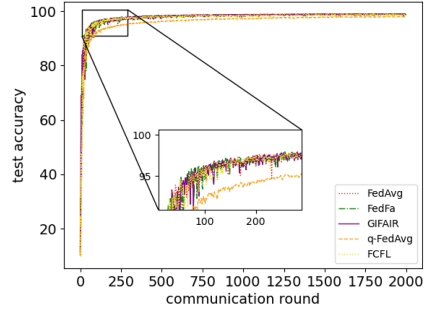
To further validate the efficiency of our FCFL, we compare the time cost for one communication round in different datasets and models. The statistics are presented in Table 3. From Table 3, it can be seen that FCFL does not introduce too much time cost. In some cases, it can achieve the same time efficiency as FedFa and q-FedAvg. Therefore, although FCFL adds one more communication, it does not consume too much time and thus can maintain time efficiency while improving fairness.

4.4 Effect of Hyper-parameter r

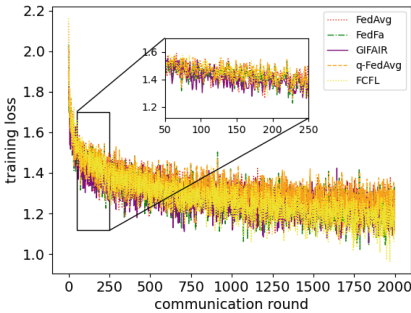
To investigate the impact of the client selection ratio r on our FCFL method, the average accuracy and variance are plotted in Fig. 5 for both datasets. The range of r is $[0, 1]$, representing the ratio of clients selected by the random selection method. Theoretically, increasing r will lead to an increase in accuracy, since greater randomness allows the global model to extract data from more clients; More randomly selected clients cause the system to ignore vulnerable clients, increasing variance among all clients. Due to the simplicity of the MNIST



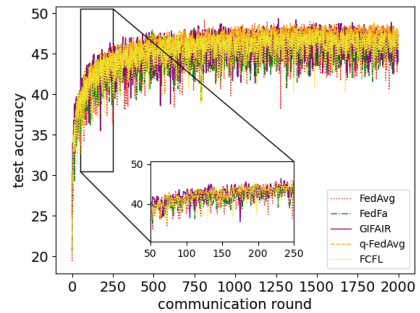
(a) MNIST training loss



(b) MNIST test accuracy



(c) CIFAR-10 training loss



(d) CIFAR-10 test accuracy

Fig. 4. The training loss (left) and test accuracy (right) of FCFL.

dataset, the performance distribution for each client is uniform, and the average precision and variance do not show significant changes with the parameter r in Fig. 5(a). In contrast, the analysis is reflected in Fig. 5(b) more obviously, where the average accuracy and variance increase along with r , which confirms that enlarging the random ratio r of selected clients will derive higher accuracy and variance of FL system. In our experiments, the parameter r is tuned by grid search, and $r = 0.6$ is selected as the best value that can reach a perfect trade-off between accuracy and variance (i.e., fairness).

4.5 Ablation Experiments

A series of ablation experiments are conducted on CIFAR-10 to validate our proposed techniques. We compare FCFL with its two variants: (i) FCFL|RS, replacing unfairness-based selection with random selection; and (ii) FCFL|DAR, replacing unfairness-based reweighting with data amount reweighting. The comparison results are presented in Table 4, which demonstrates that the complete FCFL can effectively balance fairness and performance compared to the two variants. Specifically, when comparing FCFL and FCFL|RS, it can be seen that

Table 3. The running time of one communication round.

Time(s)	Setting			
Method	MNIST(MLP)	MNIST(CNN)	CIFAR-10(MLP)	CIFAR-10(CNN)
FedAvg	0.87	1.91	0.89	1.84
q-FedAvg	0.95	1.99	0.96	1.90
FedFa	1.03	2.01	0.96	1.91
GIFAIR	0.88	1.90	0.88	1.85
FCFL	0.94	2.01	0.96	1.95

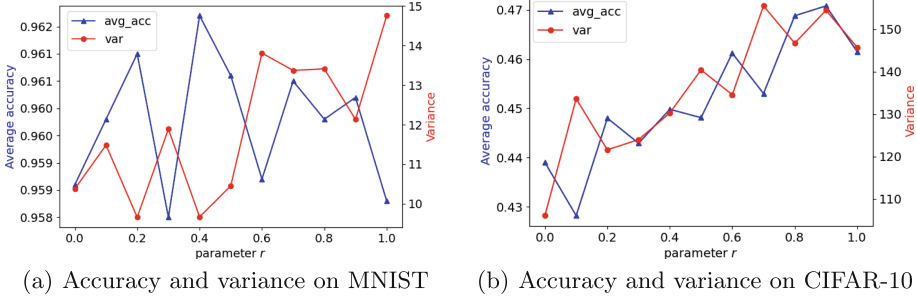


Fig. 5. Analysis of the parameter r on (a) MNIST, and (b) CIFAR-10.

unfairness-based selection can significantly improve the worst 10% performance, and reduce variance from 151.31 to 114.59. This improvement indicates that unfairness-based client selection can solve the unfairness issue effectively. Compared with FCFL|DAR, although the variance is reduced much by FCFL|DAR, it is achieved by sacrificing global accuracy and the best 10% accuracy, which states that our unfairness-based reweighting can improve accuracy effectively. The results of the ablation study indicate that the proposed FCFL takes advantage of both client selection and reweighting strategy, providing a well-justified fairness and performance in FL.

Table 4. Ablation studies of FCFL on CIFAR-10.

Method	Accuracy	Best 10%	Worst 10%	Variance
FCFL	46.12	65.39	28.03	114.59
FCFL RS	46.06	65.60	22.91	151.31
FCFL DAR	43.56	61.00	26.57	96.31

5 Conclusion

The performance fairness problem of FL is investigated in this work, where we propose FCFL, a fairness compensation-based FL algorithm to improve fairness

on vulnerable clients. The proposed FCFL considers both client selection and aggregation reweighting to compensate for unfairly treated clients by adopting accumulated unfairness queues. Through intensive experiments and comparison with the existing baselines, the proposed FCFL is demonstrated to improve fairness by 30.4% with high efficiency. In future work, we will extend the study on how to estimate unfairness accurately with approximate global performance and how to select hyper-parameters adaptively to improve overall performance. Furthermore, combining this approach with selection fairness would be an interesting idea in our future investigation.

Acknowledgments. This research was supported by the National Science Foundation grants No. 2011845 and 2343619.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Cho, Y.J., Wang, J., Joshi, G.: Client selection in federated learning: convergence analysis and power-of-choice selection strategies. arXiv preprint [arXiv:2010.01243](https://arxiv.org/abs/2010.01243) (2020)
2. Cong, Y., et al.: Ada-FFL: adaptive computing fairness federated learning. *CAAI Trans. Intell. Technol.* (2023)
3. Du, W., Xu, D., Wu, X., Tong, H.: Fairness-aware agnostic federated learning. In: *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, pp. 181–189. SIAM (2021)
4. Ezzeldin, Y.H., Yan, S., He, C., Ferrara, E., Avestimehr, A.S.: Fairfed: enabling group fairness in federated learning. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. vol. 37, pp. 7494–7502 (2023)
5. Hamidi, S.M., Damen, O.: Fair wireless federated learning through the identification of a common descent direction. *IEEE Commun. Lett.* (2024)
6. Hu, Z., Shaloudegi, K., Zhang, G., Yu, Y.: Fedmgda+: federated learning meets multi-objective optimization. *IEEE Trans. Netw. Sci. Eng.* **9**(4), 2039–2051 (2022)
7. Huang, T., Lin, W., Wu, W., He, L., Li, K., Zomaya, A.Y.: An efficiency-boosting client selection scheme for federated learning with fairness guarantee. *IEEE Trans. Parallel Distrib. Syst.* **32**(7), 1552–1564 (2020)
8. Huang, W., Li, T., Wang, D., Du, S., Zhang, J., Huang, T.: Fairness and accuracy in horizontal federated learning. *Inf. Sci.* **589**, 170–185 (2022)
9. Jiang, M., Wang, Z., Dou, Q.: Harmofl: harmonizing local and global drifts in federated learning on heterogeneous medical images. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. vol. 36, pp. 1087–1095 (2022)
10. Li, T., Hu, S., Beirami, A., Smith, V.: Ditto: Fair and robust federated learning through personalization. In: *International Conference on Machine Learning (ICML)*, pp. 6357–6368. PMLR (2021)
11. Li, T., Sanjabi, M., Beirami, A., Smith, V.: Fair resource allocation in federated learning. arXiv preprint [arXiv:1905.10497](https://arxiv.org/abs/1905.10497) (2019)
12. Li, X., Zhao, S., Chen, C., Zheng, Z.: Heterogeneity-aware fair federated learning. *Inf. Sci.* **619**, 968–986 (2023)

13. Liu, Z., Chen, Y., Yu, H., Liu, Y., Cui, L.: Gtg-Shapley: efficient and accurate participant contribution evaluation in federated learning. *ACM Trans. Intell. Syst. Technol. (TIST)* **13**(4), 1–21 (2022)
14. Luo, B., Li, X., Wang, S., Huang, J., Tassiulas, L.: Cost-effective federated learning in mobile edge networks. *IEEE J. Sel. Areas Commun.* **39**(12), 3606–3621 (2021)
15. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: *Artificial Intelligence and Statistics*, pp. 1273–1282. PMLR (2017)
16. Mohri, M., Sivek, G., Suresh, A.T.: Agnostic federated learning. In: *International Conference on Machine Learning (ICML)*, pp. 4615–4625. PMLR (2019)
17. Pan, Z., Wang, S., Li, C., Wang, H., Tang, X., Zhao, J.: Fedmdfg: federated learning with multi-gradient descent and fair guidance. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. vol. 37, pp. 9364–9371 (2023)
18. Shi, Y., Liu, Z., Shi, Z., Yu, H.: Fairness-aware client selection for federated learning. In: *2023 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 324–329. IEEE (2023)
19. Shi, Y., Yu, H., Leung, C.: Towards fairness-aware federated learning. *IEEE Trans. Neural Netw. Learn. Syst.* (2023)
20. Singhal, P., Pandey, S.R., Popovski, P.: Greedy Shapley client selection for communication-efficient federated learning. *IEEE Networking Lett.* (2024)
21. Sun, Y., Si, S., Wang, J., Dong, Y., Zhu, Z., Xiao, J.: A fair federated learning framework with reinforcement learning. In: *2022 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE (2022)
22. Wahab, O.A., Rjoub, G., Bentahar, J., Cohen, R.: Federated against the cold: a trust-based federated learning approach to counter the cold start problem in recommendation systems. *Inf. Sci.* **601**, 189–206 (2022)
23. Wang, Z., Fan, X., Qi, J., Wen, C., Wang, C., Yu, R.: Federated learning with fair averaging. *arXiv preprint [arXiv:2104.14937](https://arxiv.org/abs/2104.14937)* (2021)
24. Xiong, Z., Cai, Z., Takabi, D., Li, W.: Privacy threat and defense for federated learning with non-iid data in AIoT. *IEEE Trans. Industr. Inf.* **18**(2), 1310–1321 (2021)
25. Xiong, Z., Li, W., Cai, Z.: Federated generative model on multi-source heterogeneous data in IoT. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. vol. 37, pp. 10537–10545 (2023)
26. Xiong, Z., Li, W., Li, Y., Cai, Z.: Exact-fun: an exact and efficient federated unlearning approach. In: *2023 IEEE International Conference on Data Mining (ICDM)*, pp. 1439–1444. IEEE (2023)
27. Yang, Q., Liu, Y., Chen, T., Tong, Y.: Federated machine learning: concept and applications. *ACM Trans. Intell. Syst. Technol. (TIST)* **10**(2), 1–19 (2019)
28. Yue, X., Nouiehed, M., R, A.K.: Gifair-FL: a framework for group and individual fairness in federated learning. *INFORMS J. Data Sci.* **2**(1), 10–23 (2023)
29. Zhao, J., Zhu, X., Wang, J., Xiao, J.: Efficient client contribution evaluation for horizontal federated learning. In: *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3060–3064. IEEE (2021)
30. Zhao, Z., Joshi, G.: A dynamic reweighting strategy for fair federated learning. In: *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8772–8776. IEEE (2022)



MMDL-Based Data Augmentation with Domain Knowledge for Time Series Classification

Xiaosheng Li¹, Yifan Wu², Wei Jiang¹, Ying Li², and Jianguo Li¹(✉)

¹ Ant Group, Hangzhou, China

{lixiaosheng.lxs,shouzhi.jw}@antgroup.com, jglee@outlook.com

² Peking University, Beijing, China

{yifanwu,li.ying}@pku.edu.cn

Abstract. Plenty of time series classification methods have been proposed in the past. Most methods utilize the labeled time series instances to build classifiers, ignoring the explicit domain knowledge. However, in real-world applications, practitioners may identify domain characteristics of the time series, and build the heuristic relationship between the class labels of the time series and these domain characteristics. In this paper, we investigate the possibility of incorporating the domain knowledge into time series classification for possible performance improvement. To this end, we propose a Modified Minimum Description Length (MMDL)-based data augmentation method to inject domain knowledge into time series classification. Based on the type of domain knowledge, the proposed method applies the MMDL shapes or residuals to augment the training data. Experimental results demonstrate that the proposed method consistently improves the classification accuracy across all tested datasets and achieves better results than other time series data augmentation methods.

Keywords: Time series classification · Data augmentation · Minimum description length

1 Introduction

Over the last two decades, time series classification has been considered as one of the most challenging problems in the field of data mining [12, 48], drawing significant attention from both academic and industrial communities. Plenty of methods for time series classification have been proposed [22, 32–34, 36]. Most of these methods only leverage labeled time series datasets to train classifiers that can predict the class labels of unseen time series instances. However, in real-world applications, domain experts often possess knowledge about specific characteristics of the time series that may influence their class labels. Therefore, incorporating such domain knowledge into time series classification may improve the classification performance.

X. Li and Y. Wu—Equal contribution.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024

A. Bifet et al. (Eds.): ECML PKDD 2024, LNAI 14943, pp. 403–420, 2024.

https://doi.org/10.1007/978-3-031-70352-2_24

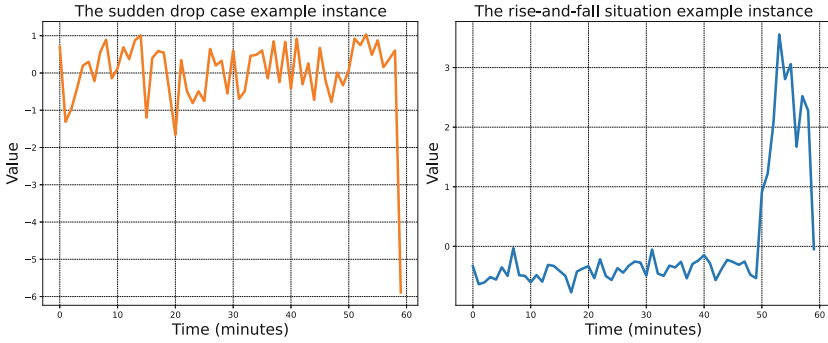


Fig. 1. Examples of a sudden drop situation requiring alerting (left) and a rise-and-fall situation without requiring alerting (right).

Figure 1 (left)¹ illustrates a real-world example of detecting system anomalies using time series data. A typical alarm rule is applied to determine whether the value of a time series drops by a certain percentage in a certain time. If this condition is met, an alarm is triggered to indicate a potential anomaly in the system. However, a common situation can lead to false alarms, as shown in Fig. 1 (right), where the value of the time series rises before falling. The alarm rule is triggered when the fall occurs. In practice, the *rise-and-fall* situation is normal and should not trigger an alarm.

To reduce the number of false alarms triggered by the aforementioned alarm rule, we need a time series classifier to identify the *rise-and-fall* situation. When the time series is classified as *rise-and-fall*, no alarm will be triggered. In this time series classification problem, we have the crucial domain knowledge that 1) the class labels are solely determined by the overall shape of the time series: the *rise-and-fall* situation presents an upside-down U or V shape while the others do not; 2) the local details in the time series do not affect the class labels. The later section of the paper shows that incorporating this knowledge into a state-of-the-art time series classifier [10] boosts its average classification accuracy from 0.765 to 0.825 on this problem.

Conversely, we also encounter real-world situations where the overall shapes of time series have no influence on their class labels. For example, there are two common types of alarm rules used to detect system anomalies using time series data: the percentage-rule and the value-rule. The alarm rule used in the *rise-and-fall* situation is a percentage-rule. In the case of the value-rule, an alarm is triggered if the value of a time series falls below or rises above a certain threshold value. Figure 2 (See footnote 1) shows examples that fit these two types of rules respectively. The time series in Fig. 2 (left) oscillates quite dramatically, making it inappropriate to apply the percentage rule. Instead, a bottom value as the threshold value for the alarm would be more suitable in this case. For the time

¹ The time series come from actual online monitored metrics in Ant Group. The series are standardized and the actual meaning is anonymized for confidentiality.

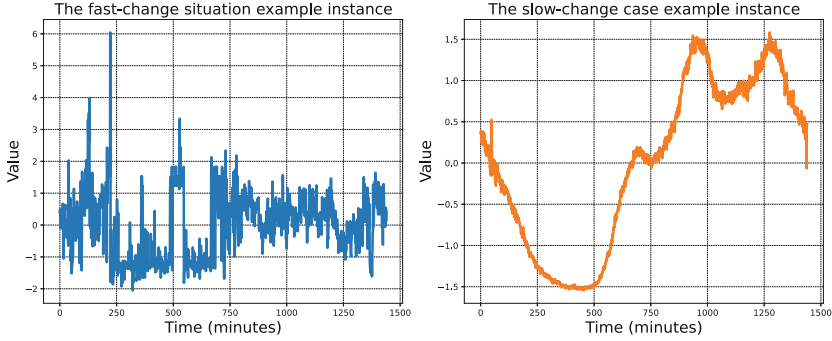


Fig. 2. Examples of a fast-change situation that should apply the value-rule (left) and a slow-change situation that needs the percentage-rule (right).

series in Fig. 2 (right), its value changes quite slowly, making it feasible to use a percentage-rule to capture the possible anomaly. Establishing a specific threshold value for the time series is not practical in this situation, as the value of the time series slowly fluctuates within a wide range.

This rule type determination can be formulated as a time series classification problem. The time series with the percentage-rule have a smooth texture, while those with the value-rule are analogous to rough surfaces. Additionally, the time series with the percentage-rule may have very different overall shapes, as long as their values change slowly and smoothly. Therefore, when handling this problem, it is important to consider the domain knowledge that 1) the overall shapes of the time series do not impact the class labels; 2) the class labels are solely determined by the texture or local details of the time series. As shown in later sections, incorporating this domain knowledge into time series classification can increase the average classification accuracy from 0.858 to 0.903 on this problem.

The above insights demonstrate the significance of leveraging domain knowledge to enhance the accuracy of time series classification in real-world applications. To incorporate the domain knowledge into existing time series classification methods, we propose a Modified Minimum Description Length (MMDL)-based time series data augmentation method. For each instance in the training set, one augmented instance is created which has the same class label as the original one. Specifically, the method utilizes either the MMDL shape or the MMDL residual to augment the training set based on the type of domain knowledge. When instances with the MMDL shape are added to the training set, classifiers will prioritize the overall shapes of the time series and pay less attention to the local details. Conversely, adding instances with the MMDL residual to the training set will lead classifiers to ignore the overall shapes of the time series and focus more on the local details.

We compare the proposed method with other time series data augmentation methods using both our domain-specific datasets and widely-used benchmark datasets. The results demonstrate that the proposed method consistently

improves the classification accuracy across all tested datasets and achieves better results than other compared methods.

The rest of the paper is organized as follows. Section 2 gives the background and related work. Section 3 details the proposed method of incorporating domain knowledge and data augmentation. Section 4 provides the experimental evaluation and comparison with other methods. Finally, Sect. 5 concludes the paper.

2 Background and Related Work

2.1 Time Series Classification

The time series classification problem differs from the general classification problems in that the time series data values have temporal correlation and the shapes of the series have a large impact on the determination of the class labels. So directly applying the standard machine learning methods to the time series data may not generate satisfactory results. Hence many time series classification approaches first transform the raw time series into flat features or representations and then apply the standard machine learning methods to the newly transformed features or representations to receive the class label prediction.

Based on the representations that the methods utilized, the state-of-the-art time series classification approaches can be categorized into distance-based, feature-based, interval-based, shapelet-based, dictionary-based, convolution-based, deep-learning-based, and hybrid approaches [36].

The distance-based methods [33,35] apply a certain distance measure to quantify the similarity between the time series instances. Most methods in this category combine the distance measures with the one-nearest neighbor classifiers to perform the tasks. The one-nearest-neighbor classifier with the Dynamic Time Warp (DTW) [43] is widely regarded as a baseline in literature.

The feature-based methods [6,34] extract global statistical measures from time series as representations, and the interval-based approaches [3,11] derive features from interval statistics. Shapelets are the snippets in the time series that can be used to best distinguish different classes [51]. The shapelet-based methods either embed the shapelet in a decision tree fashion [38,40] or use the distances from the time series instances to the shapelets as transformed representations for classification [20,22].

The dictionary-based methods [32,44] first apply a sliding window to extract all the sub-series from the time series data and then transform the sub-series into symbols. Histograms of the symbols are built as the classification representations. The convolution-based methods [9,10] adopt 1-dimension random-value kernels to perform convolution on the time series to generate features for classification. The deep-learning-based algorithms [18,24] apply neural networks on the raw time series data and the hybrid methods combine approaches from different categories together.

2.2 Time Series Data Augmentation

Data augmentation serves to increase the dataset size by generating new data instances from the original data, which aims to improve the task performance with the augmented dataset. Existing data augmentation methods for time series classification can be categorized into transformation-based, pattern-based, generative-based, decomposition-based, and automated methods [16].

Many transformation-based techniques apply straightforward operations on the time series data to create new instances. Example transformation operations are jittering, rotation, scaling, magnitude-warping, permutation, and so on [46]. The transformation can be carried out in time domain [29, 41], frequency domain [19, 49], or both [30].

The pattern-based methods [1, 26] work by extracting and utilizing specific patterns or components in the time series to generate synthetic data. The components can be from the time realm [15, 25] or the frequency realm [7, 39]. The generative-based approaches first fit the generative models using the available training instances, then create new augmented instances using these models. Based on the type of the generative model, this category can be further divided into statistical-model-based methods [4, 27] and deep-learning-based methods [37, 50].

The decomposition-based approaches [31, 47] decompose the time series into different components and augment each component separately. Then the components are assembled to form new synthetic instances. As for the automated data augment methods [5, 14], adaptive augmentation strategies are used to form an optimal data augmentation process according to the data at hand.

The proposed method differs from the above techniques in that it takes advantage of the generalized domain knowledge to guarantee the correctness of the augmented instances' class labels.

2.3 Minimum Description Length

The MDL principle [21] is a theory for model selection and can be summarized as "Choose the model that gives the shortest description of data" [42]. In the realm of time series, this principle is used to find the intrinsic cardinality and dimensionality [23], to discover the intrinsic patterns [45], or to perform semi-supervised classification [2]. In these applications, the description length is defined as following [23]:

$$DL(T, M) = DL(M) + DL(T|M) \quad (1)$$

where $DL(T, M)$ is the description length of the time series T given the model M . $DL(M)$ is the number of bits needed to be used to store the model. $DL(T|M)$ is the number of bits required to be utilized to rebuild the time series T given the model M . $DL(T|M)$ is calculated as $DL(T - M)$, where $T - M$ denotes the difference between the time series T and model M , i.e., one can store the difference values between the model values and the real series values. The difference values vector is encoded using Huffman coding and the coding length is $DL(T - M)$.

The idea behind MDL is to balance the model complexity (first term in Eq. 1, called model cost) and the model fidelity (second term in the equation, known as correction cost). The model that has the minimum description length is considered as the best model or representation for the data according to the principle.

3 The Proposed Method

The idea behind the proposed method is to incorporate the domain knowledge into the time series classification process to enhance its accuracy. Experience inspires us to generalize our domain knowledge in many situations into two types. Type 1 indicates the case where the class labels of the time series are solely determined by the overall shapes of the time series and are not affected by the local details. Type 2 is the case where only the texture of the time series decides the class labels and the shapes of the time series do not influence the class labels. Examples of these two types have been shown in Sect. 1.

If the domain knowledge is Type 1, the proposed method will extract the essential shapes of the training series and add back them to the training set for the following classification. It is safe to assign the shape instances the same class labels as their respective original time series. These augmented series can guide the classifiers to focus more on the overall shapes of the time series and ignore the local details, thus helping to improve the final classification results.

On the other hand, if the domain knowledge is Type 2, the difference between the training series and their respective essential shapes will be used to augment the training set. The residual is used to create the pure texture series, which has the same class labels as the original series. Adding these residual series to the training set can lead the classifiers to ignore the shapes and focus more on the local texture of the series.

To extract the essential shapes from the time series objectively and in the hope of not adding extra parameters to tune, we adapt the MDL method in the process. Regarding the shapes as models to approximate the time series, the modified MDL procedure could generate proper shapes to capture the overall outline of the time series and ignore the local details.

Algorithm 1 gives the pseudo-code for the proposed method. The input of the algorithm is the time series data set D and the domain knowledge type P (1 stands for Type 1 and 2 for Type 2 respectively). The algorithm returns the augmented dataset A as output.

In Line 1–2, each instance I is taken out from D and the time series and class labels are assigned to T and C respectively. Line 3 transforms the time series T into segments denoted as *Segs*. In our implementation, every consecutive 3 points are combined to become a segment (So the series indices for the segments are $[0 - 2]$, $[3 - 5]$, $[6 - 8]$, \dots). The mean value of the time series values in each segment is the respective segment value.

In Line 4, the segment values are discretized to a cardinality of 256. Previous research [23] suggests that using the 256-cardinality version in fact does not have much difference from using the original data on classification tasks.

Algorithm 1. MMDL-based Data Augmentation**Input:**

D : time series dataset
 P : domain knowledge type

Output:

A : augmented dataset

```

1: for each  $I \in D$  do
2:    $[T, C] = I$ 
3:    $Segs = SeriesToSeg(T)$ 
4:    $Segs = Discretization(Segs, 256)$ 
5:    $segsLen = length(Segs)$ 
6:    $bestCost = inf$ 
7:    $BestModel = Segs$ 
8:   for  $numSeg = segLen - 1$  to 1 do
9:      $Segs = MergeSegs(Segs)$ 
10:     $modelCost = ModelCost(segLen, numSeg)$ 
11:     $correctCost = CorrectCost(T, Segs)$ 
12:     $totalCost = modelCost + correctCost$ 
13:    if  $totalCost < bestCost$  then
14:       $bestCost = totalCost$ 
15:       $BestModel = Segs$ 
16:    end if
17:  end for
18:   $Shape = SegToSeries(BestModel)$ 
19:  if  $P == 1$  then
20:     $A.add([Shape, C])$ 
21:  else if  $P == 2$  then
22:     $A.add([T - Shape, C])$ 
23:  end if
24: end for

```

Line 5 receives the number of segments in $Segs$ and Line 6–7 initialize the best total cost and best model. Here, the segments with their respective values are regarded as the model to approximate the original time series.

Line 8–17 find the best model under the MDL principle. In Line 8–9, each time two nearby segments are selected to merge. The series indices of the two segments are merged together and the larger one of the two segment values become the new value of the merged segment. All the nearby segment pairs in $Segs$ are checked and the pair that generates the minimum merge-error is selected to merge. The merge-error is calculated as $|DV| \times LIS$, where DV is the difference between the two segment values and LIS denotes the length of indices of the segment with the smaller segment value.

Line 10 calculates the model cost for the current $Segs$. The model cost $modelCost$ can be received using the following equation:

$$modelCost = (\log_2(256) + \lceil \log_2(segLen) \rceil) \times numSeg \quad (2)$$

For each of the num_seg segments, the number of bits that one needs to use to represent the possible 256 segment values are $\log_2(256)$. $\lceil \log_2(segLen) \rceil$ is used to represent the information needed to record the positions of the segments.

Line 11 delivers the correction cost for Segs. Previous works surveyed in Sect. 2 use the Huffman coding length of the difference vector between the original time series values and the model predicting values as the correction cost. We modified the MDL procedure by proposing a different method to calculate the correction cost as displayed in the following equation:

$$correctCost = \sum \log_2(1 + |T - SegToSeries(Segs)|) \quad (3)$$

Essentially the method encodes each element of the difference vector ($T - SegsToSeries(Segs)$) individually and sums up the overall bits as the correction cost.

The idea is to encourage the algorithm to find models that have segment values approximate to the original time series values. For example, two difference vectors $[0, 0, 0, 1, 1, 1]$ and $[0, 0, 0, 3, 3, 3]$ have the same Huffman coding length, while the first one has a smaller correction cost according to Eq. 3 and is thus more preferred.

During the loop, Line 13–16 find the best segment model for the given data, and Line 18 transforms the segments to series values as the shape of the original series. If the case is Type 1, then the shape will be added to the output dataset (Line 19–20). If the situation is Type 2, then the residual series (difference vector) between the original series and the shape series will instead go into the augmented dataset C (Line 21–22).

Figures 3 and 4 show the augmented instances for the examples of the two types introduced in Sect. 1 respectively.

Note that the proposed method is designed to incorporate the generalized domain knowledge into the classification process so it is suitable for the situations where the domain knowledge is available. Also, the two types of domain knowledge do not cover all the possible situations. For example, there are situations where both the overall shapes and the local details of the series may have an impact on the class labels. In these cases, other data augmentation methods surveyed in Sect. 2 can be used to enhance the classification performance.

4 Experimental Evaluation

4.1 Experimental Setup

To evaluate the effectiveness of the proposed method, we conduct experiments on four scenarios. Among these, two are derived from our domain of expertise (i.e., false alarm elimination and alarm rule type selection), while the other two are from widely-used benchmark datasets (i.e., signal type classification and appliance recognition). There are some other datasets available, but we are not very familiar with their underlying mechanism, and not sure if the class labels are mainly affected by the overall shapes or local details or both, so we choose to

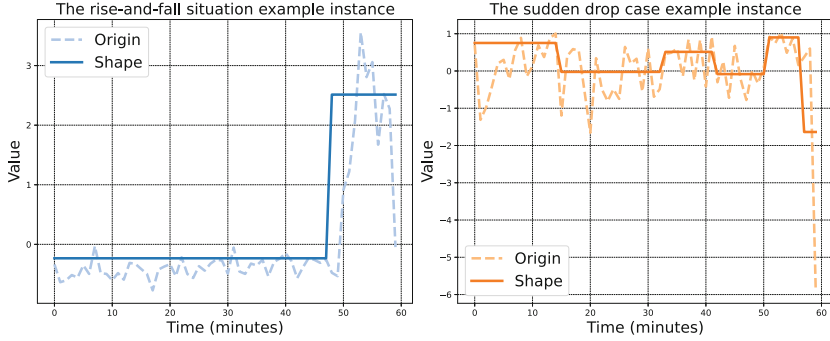


Fig. 3. Augmented instances for the rise-and-fall situation.

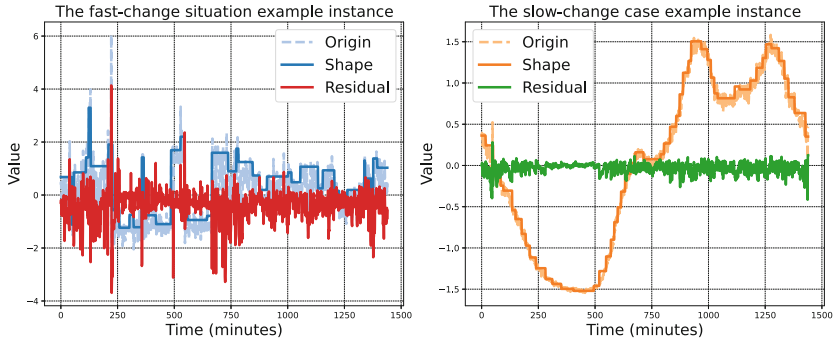


Fig. 4. Augmented instances for the slow-change situation.

focus on the cases where we have domain knowledge. The mechanism behind the tested scenarios will be analyzed and the labeling of the classes will be discussed.

We compare the proposed method (denoted as MMDL) with all the evaluated time series data augmentation methods in a recent survey [16], which include Jittering [46], Rotation [46], Scaling [46], Magnitude Warping [46], Permutation [46], Random Permutation [46], Time Warping [46], Window Slicing [29], Window Warping [29], SPAWNER [26], wDBA [15], RGW [25], RGWs [25], DGW [25], DGWs [25], and TimeGAN [52]. We also include the version using the original MDL procedure (denoted as MDL) in the comparison.

Middlehurst et al. [36] conducts an extensive experimental evaluation on the time series classification methods including all the categories surveyed in Sect. 2. The results indicate that Hydra-MultiROCKET [10] is one of the best methods both in terms of accuracy and speed. Thus, we adopt this classifier to work with the data augmentation methods under comparison.

The dataset for each scenario contains a training set and a testing set. The data augmentation methods will be applied on the training set to generate new data instances. The new instances are combined with the training set to form an augmented set. The Hydra-MultiROCKET classifier will be trained on the

augmented set and predict the class labels of the instances in the testing set. The only parameter for the proposed method is the domain knowledge type, which will be set according to the domain knowledge in each scenario. The data augmentation methods under comparison and the classifiers will use the default parameter settings. All the reported accuracy results are the average values of ten different runs with different random seeds. The source code of the proposed method and our domain-specific datasets are available online².

4.2 False Alarm Elimination

As introduced in Sect. 1, we need to distinguish between the time series that contain drops caused by system anomalies and those that belong to the rise-and-fall situation. The latter will also trigger the alarm rule to produce false alarms. Thus, a time series classifier is used to judge the type of the time series. If the type belongs to the rise-and-fall situation, no alarm will be triggered. Examples of these two types of time series have been shown in Fig. 1.

From the analysis, one learns that the class labels are determined solely by the overall shape of the time series. The rise-and-fall series contain an upside-down U or V shape while the series from the other class does not. The local details do not affect the class labels. Thus, the domain knowledge type of the proposed method in this application is set to 1.

The RiseFall dataset contains the time series we collected in this scenario. It contains 75 time series of length 60 and their respective class labels. Class label 0 denotes the rise-and-fall instances and class label 1 indicates the instances that should be alarmed. There are 45 instances in the class label 0 and the rest are in the class label 1. The series are standardized to have zero mean and one standard deviation. We randomly select one third of the instances to form the training set and the remains become the testing set.

The proposed method as well as the compared methods are run on this dataset, and the results are listed in Table 1. The table shows the average classification accuracy and the ranks of MMDL and the compared methods on the testing set. The None method in the table denotes the results without data augmentation. Comparing the results between None and MMDL, one observes that MMDL improves the average classification accuracy from 0.765 to 0.825. Also, MDL has a higher test accuracy than None, but are inferior to MMDL. Furthermore, from the rank columns one observes that MDL achieves the best classification accuracy among the methods under comparison in this scenario. These results demonstrate the effectiveness of the proposed method.

² <https://github.com/alipay/MMDL-based-Data-Augmentation-with-Domain-Knowledge-for-Time-Series-Classification>.

Table 1. Experimental results on the RiseFall dataset

Method	Acc.	Rank	Method	Acc.	Rank	Method	Acc.	Rank
dgw	0.767	12	rgws	0.747	16	windowwarp	0.769	10
dgws	0.784	4	rotation	0.755	15	timeGAN	0.725	18
jitter	0.780	7	scaling	0.771	9	None	0.765	13
magwarp	0.765	13	spawner	0.780	7	MDL	0.784	4
permutation	0.792	3	timewarp	0.696	19	MMDL	0.825	1
randomperm	0.806	2	wdba	0.782	6			
rgw	0.743	17	windowslice	0.769	10			

4.3 Alarm Rule Type Selection

There are two common types of alarm rules for different kinds of time series: the percentage-rule and the value-rule, as introduced in Sect. 1. The percentage-rule is fit for the slowly changing time series with smooth texture, while the value-rule is proper for the time series with sudden changes with rough surfaces. Figure 2 gives two examples suitable for the two types of alarm rules respectively. We utilize a time series classifier to select the type of alarm rules automatically according to the time series.

From the analysis, one can see that the overall shapes of the series do not influence the class labels, while the class labels are solely determined by the texture of the series. Thus, the domain knowledge type of the proposed method is set to 2.

The RuleTypes dataset contains the time series instances we collected in this scenario. There are 239 time series instances of length 1440 with their respective class labels. Class label 1 denotes the percentage-rule and class label 0 indicates the value-rule. 140 out of the 239 instances are in class label 0 and the rest fall in class label 1. One third of the instances are randomly selected to form the training set and the rest become the testing set.

Table 2 shows the experimental results of MMDL and compared methods on this dataset. One observes that MMDL improves the average classification accuracy from 0.858 to 0.903, which is also the best result among the methods under comparison. These results demonstrate the effectiveness of the proposed methods.

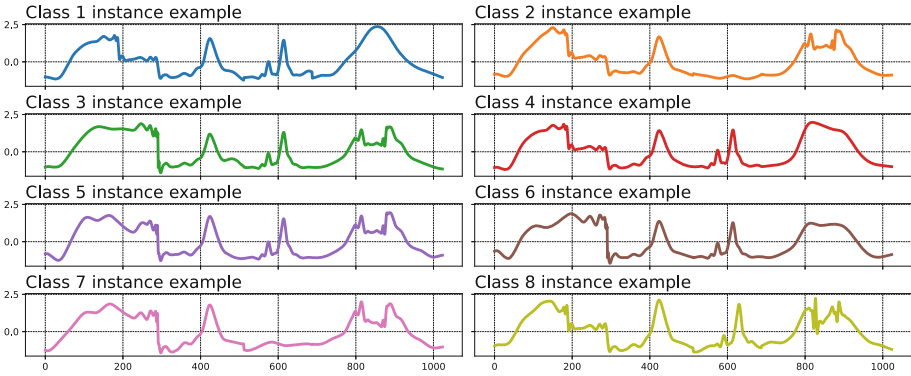
4.4 Signal Type Classification

In this scenario, we evaluate MMDL and the methods under comparison on the Mallat dataset from the UCR time series classification archive [8]. The Mallat dataset contains 55 signal instances in the training set and 2345 signal instances in the testing set. There are 8 different classes (Class 1–8) in this dataset and Fig. 5 shows the example instances for each class respectively.

The instances between different classes have different overall shapes. For instance, the instance belonging to Class 2 has a sunken area in the rightmost

Table 2. Experimental results on the RuleTypes dataset

Method	Acc.	Rank	Method	Acc.	Rank	Method	Acc.	Rank
dgw	0.851	17	rgws	0.865	9	windowwarp	0.856	15
dgws	0.858	12	rotation	0.837	19	timeGAN	0.857	14
jitter	0.875	6	scaling	0.866	8	None	0.858	12
magwarp	0.868	7	spawner	0.853	16	MDL	0.893	2
permutation	0.880	4	timewarp	0.842	18	MMDL	0.903	1
randomperm	0.876	5	wdba	0.860	11			
rgw	0.892	3	windowslice	0.864	10			

**Fig. 5.** Example instances of different classes from the Mallat dataset.

part of the series, which is not present in the instance of Class 1. Additionally, the instances inside the same class have the same overall shape and differ in local details. So, in this case, we set the domain knowledge type of the proposed method to 1.

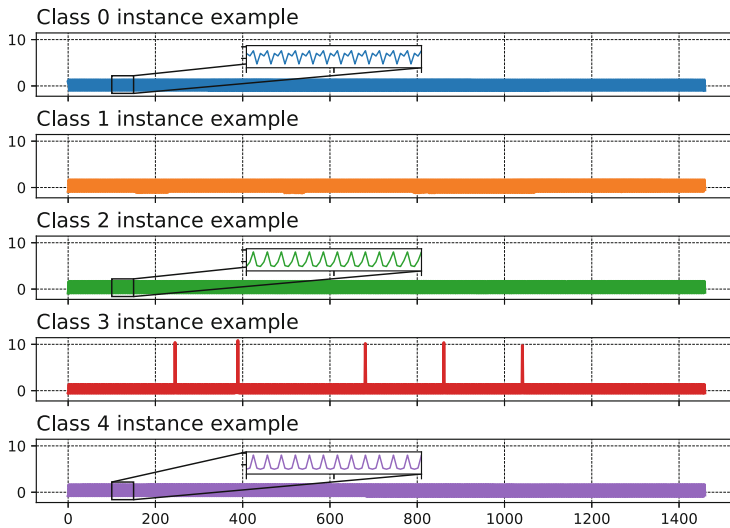
The experimental results on the Mallat dataset are given in Table 3. One observes that the proposed method achieves the best result among all the compared methods.

4.5 Appliance Recognition

In this scenario, we evaluate MMDL and the compared methods on the ACSF1 dataset from the UCR time series archive [8], which contains the power consumption record of different classes of appliances. The classes correspond to 10 categories of home appliances: mobile phones (via chargers), coffee machines, computer stations (including monitor), fridges and freezers, Hi-Fi systems (CD players), lamps (CFL), laptops (via chargers), microwave oven, printers, and televisions (LCD or LED) [17]. There are 100 instances in the training set and 100 instances in the testing set. Figure 6 and Fig. 7 depict the example instances from each of the 10 classes.

Table 3. Experimental results on the Mallat dataset

Method	Acc.	Rank	Method	Acc.	Rank	Method	Acc.	Rank
dgw	0.933	5	rgws	0.923	12	windowwarp	0.912	17
dgws	0.915	13	rotation	0.936	4	timeGAN	0.884	19
jitter	0.937	3	scaling	0.924	10	None	0.927	8
magwarp	0.913	16	spawner	0.923	11	MDL	0.911	18
permutation	0.913	15	timewarp	0.937	2	MMDL	0.941	1
randomperm	0.914	14	wdba	0.932	7			
rgw	0.925	9	windowslice	0.933	6			

**Fig. 6.** Example instances of Class 0–4 from the ACSF1 dataset.

From the data one can see that the instances from different classes have the same oscillating outlines. The difference between the classes lies in the local details, as demonstrated in the zoomed-in zone in the given figures. Thus, the domain knowledge type of the proposed method is set to 2.

Table 4 lists the experimental results on the ACSF1 dataset. One can observe that the proposed method also manages to improve the average classification accuracy and is the second-best method among the compared methods.

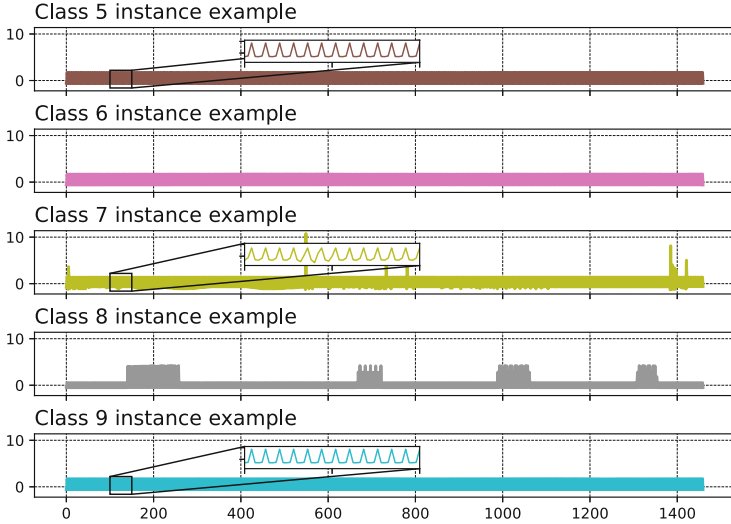


Fig. 7. Example instances of Class 5-9 from the ACSF1 dataset.

Table 4. Experimental results on the ACSF1 dataset

Method	Acc.	Rank	Method	Acc.	Rank	Method	Acc.	Rank
dgw	0.885	11	rgws	0.888	9	windowwarp	0.877	17
dgws	0.884	12	rotation	0.904	6	timeGAN	0.880	15
jitter	0.875	19	scaling	0.884	12	None	0.892	8
magwarp	0.899	7	spawner	0.876	18	MDL	0.910	2
permutation	0.907	4	timewarp	0.881	14	MMDL	0.910	2
randomperm	0.915	1	wdba	0.906	5			
rgw	0.886	10	windowslice	0.880	15			

Figure 8 gives a summary of the average rank of the compared methods. The horizontal bar lengths stand for the average rank of the respective data augmentation methods on the four scenarios, and the methods are sorted in ascending order based on the average rank. One can see that the proposed MMDL-based method achieves the best overall performance among the compared methods on the four scenarios. MMDL has better or equal results on all the tested cases than MDL, demonstrating the effectiveness of the proposed correction cost calculation method on the tested scenarios.

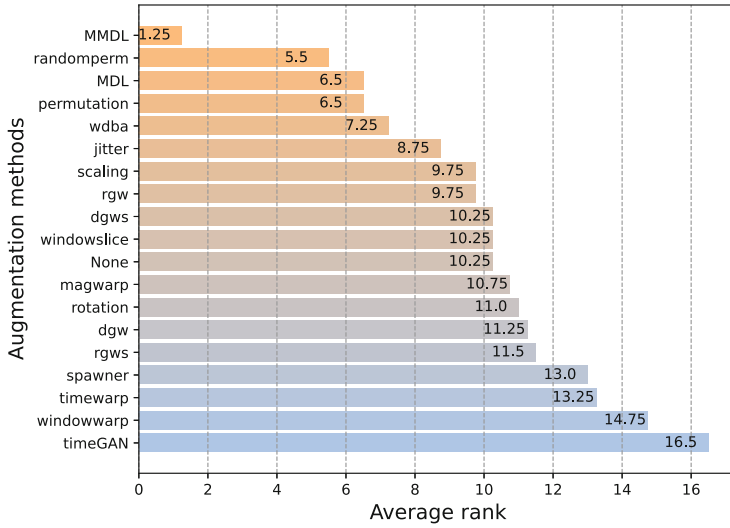


Fig. 8. Average rank of the compared methods.

5 Conclusion and Future Work

This paper presents a data augmentation method to incorporate domain knowledge into the time series classification process to enhance the classification accuracy. The proposed method uses an MMDL-based procedure to extract the essential shapes of the time series data. According to the type of domain knowledge, either the shapes or the residual of the shapes will be used to augment the training data. Experimental evaluation demonstrates that the proposed method consistently improves the classification accuracy across all tested cases and achieves better results than other time series data augmentation methods.

The experimental evaluation adopts Hydra-MultiROCKET [10] as the classifier. Combining other classifiers with the proposed method could be a future research direction. Applying other techniques like APCA [28] and DFT [13] to extract the shapes of time series in the procedure could be another future research direction.

Acknowledgments. Ying Li and Yifan Wu were supported by Ant Group Research Fund.

References

1. Aboussalah, A.M., Kwon, M., Patel, R.G., Chi, C., Lee, C.G.: Recursive time series data augmentation. In: The Eleventh International Conference on Learning Representations (2022)
2. Begum, N., Hu, B., Rakthanmanon, T., Keogh, E.: A minimum description length technique for semi-supervised time series classification. Integration of reusable systems, pp. 171–192 (2014)
3. Cabello, N., Naghizade, E., Qi, J., Kulik, L.: Fast, accurate and interpretable time series classification through randomization. arXiv preprint [arXiv:2105.14876](https://arxiv.org/abs/2105.14876) (2021)
4. Cao, H., Tan, V.Y., Pang, J.Z.: A parsimonious mixture of gaussian trees model for oversampling in imbalanced and multimodal time-series classification. IEEE Trans. Neural Networks Learn. Syst. **25**(12), 2226–2239 (2014)
5. Cheung, T.H., Yeung, D.Y.: Modals: modality-agnostic automated data augmentation in the latent space. In: International Conference on Learning Representations (2020)
6. Christ, M., Braun, N., Neuffer, J., Kempa-Liehr, A.W.: Time series feature extraction on basis of scalable hypothesis tests (tsfresh-a python package). Neurocomputing **307**, 72–77 (2018)
7. Cui, X., Goel, V., Kingsbury, B.: Data augmentation for deep neural network acoustic modeling. IEEE/ACM Trans. Audio Speech Lang. Process. **23**(9), 1469–1477 (2015)
8. Dau, H.A., et al.: The ucr time series classification archive, October 2018. https://www.cs.ucr.edu/~eamonn/time_series_data_2018/
9. Dempster, A., Petitjean, F., Webb, G.I.: Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. Data Min. Knowl. Disc. **34**(5), 1454–1495 (2020)
10. Dempster, A., Schmidt, D.F., Webb, G.I.: Hydra: competing convolutional kernels for fast and accurate time series classification. Data Mining and Knowledge Discovery, pp. 1–27 (2023)
11. Deng, H., Runger, G., Tuv, E., Vladimir, M.: A time series forest for classification and feature extraction. Inf. Sci. **239**, 142–153 (2013)
12. Esling, P., Agon, C.: Time-series data mining. ACM Comput. Surv. (CSUR) **45**(1), 1–34 (2012)
13. Faloutsos, C., Ranganathan, M., Manolopoulos, Y.: Fast subsequence matching in time-series databases. ACM SIGMOD Rec. **23**(2), 419–429 (1994)
14. Fons, E., Dawson, P., Zeng, X.j., Keane, J., Iosifidis, A.: Adaptive weighting scheme for automatic time-series data augmentation. arXiv preprint [arXiv:2102.08310](https://arxiv.org/abs/2102.08310) (2021)
15. Forestier, G., Petitjean, F., Dau, H.A., Webb, G.I., Keogh, E.: Generating synthetic time series to augment sparse datasets. In: 2017 IEEE International Conference on Data Mining (ICDM), pp. 865–870. IEEE (2017)
16. Gao, Z., Li, L., Xu, T.: Data augmentation for time-series classification: an extensive empirical study and comprehensive survey. arXiv preprint [arXiv:2310.10060](https://arxiv.org/abs/2310.10060) (2023)
17. Gisler, C., Ridi, A., Zufferey, D., Abou Khaled, O., Hennebert, J.: Appliance consumption signature database and recognition test protocols. In: 2013 8th International Workshop on Systems, Signal Processing and their Applications (WoSSPA), pp. 336–341. IEEE (2013)

18. Gong, X., Si, Y.W., Tian, Y., Lin, C., Zhang, X., Liu, X.: Kdctime: knowledge distillation with calibration on inceptiontime for time-series classification. *Inf. Sci.* **613**, 184–203 (2022)
19. Goubeaud, M., Gmyrek, N., Ghorban, F., Schelkes, L., Kummert, A.: Random noise boxes: data augmentation for spectrograms. In: 2021 IEEE International Conference on Progress in Informatics and Computing (PIC), pp. 24–28. IEEE (2021)
20. Guillaume, A., Vrain, C., Elloumi, W.: Random dilated shapelet transform: a new approach for time series shapelets. In: International Conference on Pattern Recognition and Artificial Intelligence, pp. 653–664. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-09037-0_53
21. Hansen, M.H., Yu, B.: Model selection and the principle of minimum description length. *J. Am. Stat. Assoc.* **96**(454), 746–774 (2001)
22. Hills, J., Lines, J., Baranauskas, E., Mapp, J., Bagnall, A.: Classification of time series by shapelet transformation. *Data Min. Knowl. Disc.* **28**, 851–881 (2014)
23. Hu, B., Rakthanmanon, T., Hao, Y., Evans, S., Lonardi, S., Keogh, E.: Discovering the intrinsic cardinality and dimensionality of time series using mdl. In: 2011 IEEE 11th International Conference on Data Mining, pp. 1086–1091. IEEE (2011)
24. Ismail Fawaz, H., Lucas, B., Forestier, G., Pelletier, C., Schmidt, D.F., Weber, J., Webb, G.I., Idoumghar, L., Muller, P.A., Petitjean, F.: Inceptiontime: Finding alexnet for time series classification. *Data Min. Knowl. Disc.* **34**(6), 1936–1962 (2020)
25. Iwana, B.K., Uchida, S.: Time series data augmentation for neural networks by time warping with a discriminative teacher. In: 2020 25th International Conference on Pattern Recognition (ICPR), pp. 3558–3565. IEEE (2021)
26. Kamycki, K., Kapuscinski, T., Oszust, M.: Data augmentation with suboptimal warping for time-series classification. *Sensors* **20**(1), 98 (2019)
27. Kang, Y., Hyndman, R.J., Li, F.: Gratis: generating time series with diverse and controllable characteristics. *Stat. Anal. Data Mining ASA Data Sci. J.* **13**(4), 354–376 (2020)
28. Keogh, E., Chakrabarti, K., Pazzani, M., Mehrotra, S.: Locally adaptive dimensionality reduction for indexing large time series databases. In: Proceedings of the 2001 ACM SIGMOD international conference on Management of data. pp. 151–162 (2001)
29. Le Guennec, A., Malinowski, S., Tavenard, R.: Data augmentation for time series classification using convolutional neural networks. In: ECML/PKDD workshop on advanced analytics and learning on temporal data (2016)
30. Lee, T.E.K., Kuah, Y., Leo, K.H., Sanei, S., Chew, E., Zhao, L.: Surrogate rehabilitative time series data for image-based deep learning. In: 2019 27th European Signal Processing Conference (EUSIPCO), pp. 1–5. IEEE (2019)
31. Li, C., Yang, H., Cheng, L., Huang, F.: A time-series augmentation method based on empirical mode decomposition and integrated lstm neural network. In: 2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), pp. 333–336. IEEE (2022)
32. Lin, J., Khade, R., Li, Y.: Rotation-invariant similarity in time series using bag-of-patterns representation. *J. Intell. Inf. Syst.* **39**, 287–315 (2012)
33. Lines, J., Bagnall, A.: Time series classification with ensembles of elastic distance measures. *Data Min. Knowl. Disc.* **29**, 565–592 (2015)
34. Lubba, C.H., Sethi, S.S., Knaute, P., Schultz, S.R., Fulcher, B.D., Jones, N.S.: catch22: canonical time-series characteristics: Selected through highly comparative time-series analysis. *Data Min. Knowl. Disc.* **33**(6), 1821–1852 (2019)

35. Lucas, B., et al.: Proximity forest: an effective and scalable distance-based classifier for time series. *Data Min. Knowl. Disc.* **33**(3), 607–635 (2019)
36. Middlehurst, M., Schäfer, P., Bagnall, A.: Bake off redux: a review and experimental evaluation of recent time series classification algorithms. arXiv preprint [arXiv:2304.13029](https://arxiv.org/abs/2304.13029) (2023)
37. Moreno-Barea, F.J., Jerez, J.M., Franco, L.: Improving classification accuracy using data augmentation on small data sets. *Expert Syst. Appl.* **161**, 113696 (2020)
38. Mueen, A., Keogh, E., Young, N.: Logical-shapelets: an expressive primitive for time series classification. In: *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1154–1162 (2011)
39. Nanni, L., Maguolo, G., Paci, M.: Data augmentation approaches for improving animal audio classification. *Eco. Inform.* **57**, 101084 (2020)
40. Rakthanmanon, T., Keogh, E.: Fast shapelets: A scalable algorithm for discovering time series shapelets. In: *proceedings of the 2013 SIAM International Conference on Data Mining*, pp. 668–676. SIAM (2013)
41. Rashid, K.M., Louis, J.: Window-warping: a time series data augmentation of imu data for construction equipment activity identification. In: *ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction*, vol. 36, pp. 651–657. IAARC Publications (2019)
42. Rissanen, J.: Modeling by shortest data description. *Automatica* **14**(5), 465–471 (1978)
43. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans. Acoust. Speech Signal Process.* **26**(1), 43–49 (1978)
44. Schäfer, P.: The boss is concerned with time series classification in the presence of noise. *Data Min. Knowl. Disc.* **29**, 1505–1530 (2015)
45. Schweier, A., Höppner, F.: Finding the intrinsic patterns in a collection of time series. In: Blockeel, H., van Leeuwen, M., Vinciotti, V. (eds.) *IDA 2014. LNCS*, vol. 8819, pp. 286–297. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-12571-8_25
46. Um, T.T., et al.: Data augmentation of wearable sensor data for Parkinson’s disease monitoring using convolutional neural networks. In: *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, pp. 216–220 (2017)
47. Wen, Q., Gao, J., Song, X., Sun, L., Xu, H., Zhu, S.: Robuststl: a robust seasonal-trend decomposition algorithm for long time series. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 5409–5416 (2019)
48. Yang, Q., Wu, X.: 10 challenging problems in data mining research. *Int. J. Inf. Technol. Decision Making* **5**(04), 597–604 (2006)
49. Yang, W., Yuan, J., Wang, X.: Sfcc: data augmentation with stratified fourier coefficients combination for time series classification. *Neural Process. Lett.* **55**(2), 1833–1846 (2023)
50. Yang, Z., Li, Y., Zhou, G.: Ts-gan: time-series gan for sensor-based health data augmentation. *ACM Trans. Comput. Healthcare* **4**(2), 1–21 (2023)
51. Ye, L., Keogh, E.: Time series shapelets: a new primitive for data mining. In: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 947–956 (2009)
52. Yoon, J., Jarrett, D., Van der Schaar, M.: Time-series generative adversarial networks. *Advances in neural information processing systems* **32** (2019)



FALCUN: A Simple and Efficient Deep Active Learning Strategy

Sandra Gilhuber^{1,2(✉)}, Anna Beer³, Yunpu Ma¹, and Thomas Seidl^{1,2}

¹ LMU Munich, Munich, Germany
{gilhuber, seidl}@dbs.ifi.lmu.de

² Munich Center for Machine Learning (MCML), Munich, Germany

³ University of Vienna, Vienna, Austria
anna.beer@univie.ac.at

Abstract. We propose FALCUN, a novel deep batch active learning method that is label- and time-efficient. Our proposed acquisition uses a natural, self-adjusting balance of uncertainty and diversity: It slowly transitions from emphasizing uncertain instances at the decision boundary to emphasizing batch diversity. In contrast, established deep active learning methods often have a fixed weighting of uncertainty and diversity, limiting their effectiveness over diverse data sets exhibiting different characteristics. Moreover, to increase diversity, most methods demand intensive search through a deep neural network’s high-dimensional latent embedding space. This leads to high acquisition times when experts are idle while waiting for the next batch for annotation. We overcome this structural problem by exclusively operating on the low-dimensional probability space, yielding much faster acquisition times without sacrificing label efficiency. In extensive experiments, we show FALCUN’s suitability for diverse use cases, including medical images and tabular data. Compared to state-of-the-art methods like BADGE, CLUE, and AlfaMix, FALCUN consistently excels in quality and speed: while FALCUN is among the fastest methods, it has the highest average label efficiency.

Keywords: Deep Active Learning · Supervised Learning · Diversity and Uncertainty Sampling

1 Introduction

Deep neural networks have proven their worth in various fields and are widely used for solving complex tasks. Their great success depends largely on the availability of labeled data. However, while large volumes of unlabeled data are often easily accessible, the labeling process remains time-consuming and costly, particularly in domains like medicine and industry, where experts are essential.

Active learning (AL) strategies mitigate annotation efforts by iteratively selecting and labeling the most informative instances to enhance model performance. However, the batch setting in deep AL, where multiple instances are

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70352-2_25.

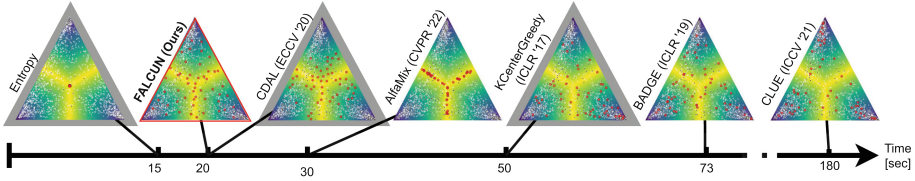


Fig. 1. Each simplex illustrates the probability space of a three-class subset of MNIST. The highest probabilities are in the corners (implied by darker colors). Small black and white dots are objects in \mathcal{L} and \mathcal{U} , respectively. Red dots are instances selected by an AL method. FALCUN acquires objects very fast and returns a meaningful selection: gray borders imply worse quality than FALCUN.

sent to the annotator simultaneously to meet the higher data demands of deep learning and reduce re-training times, poses new challenges [2]. Specifically, the question of how to select the most informative instances while minimizing redundancy is an ongoing research topic.

To assess *diversity* and *uncertainty*, established approaches often treat the probability and latent spaces separately [14, 15], requiring an additional step to merge the extracted information into a coherent acquisition. However, achieving a smooth combination of these disparate aspects can be difficult, potentially overemphasizing either uncertainty or diversity. Furthermore, a subsequent combination may rely on additional parameters [25] that are hard to select in advance. As a result, such methods might not outperform random sampling consistently, which is crucial for active learning approaches. Lastly, merging information from distinct spaces may result in highly complex methodologies, undermining their practical applicability in active learning contexts.

Moreover, using the latent representations of a deep neural network to measure diversity [2, 15, 18, 25] can be computationally intensive due to the high dimensionality of learned features. E.g., the dimensionality of the last hidden layer for commonly used architectures (see [2, 10, 14]) is 512 in ResNet18, 2048 in ResNet50, and 4096 in VGG16. Thus, searching the feature space can be very time-intensive, leading to acquisition times of up to several days. Starting the labeling process on multiple days instead of requiring only one session can drive up costs immensely, e.g., if domain experts or laboratory equipment are required. Unnecessarily long computation times are also prohibitive from an ecological point of view.

We address these challenges and propose FALCUN (**F**ast **A**ctive Learning by **C**ontrastive **U**Ncertainty). As illustrated in Fig. 1, FALCUN queries instances that yield high-quality results for deep learning while also being faster than comparative methods. Our method exclusively operates on the output probabilities to calculate uncertainty and batch diversity. In a unified and coherent acquisition, FALCUN begins by proposing instances around the decision boundary and gradually shifts focus to diverse areas as regions of high uncertainty are increasingly explored.

The main benefits of FALCUN are:

- Label efficiency and robustness: Across varying datasets, AL settings, and model architectures, FALCUN is always among the most label-efficient methods. Among all experiments, FALCUN outperforms random sampling most often ($> 70\%$) while never performing statistically worse.
- Speed and scalability: Among competitors reaching similar accuracy, FALCUN is the fastest. FALCUN is more scalable than methods operating on the latent embeddings of a neural network.
- Diversity: Even on high-redundancy data sets, FALCUN finds a **diverse** set of instances.
- Explainability and simplicity: FALCUN is **easy** to understand and implement and, therefore, attractive for practitioners and researchers. Our code is available under <https://github.com/sobermeier/falcun>.

2 Related Work

AL techniques can be grouped into the following categories.

Uncertainty-based methods estimate the informativeness of an instance based on the model’s predictive ambiguity. Common uncertainty estimates are margin uncertainty [16], entropy [20] or least confidence [19]. Labeling such instances should help to effectively refine the decision boundary and enhance generalization performance if included in the training [19]. Uncertainty-based sampling is widely used for its simplicity and effectiveness, especially when querying single instances or small batches at once. However, in the batch setting common for deep AL, where multiple instances are queried simultaneously, simple rank-based techniques become less label-efficient since they tend to select redundant instances. E.g., in Fig. 1, Entropy [22] as a non-diversity aware method selects highly repetitive instances.

Query-by-committee (QBC) refers to using a committee of classifiers and calculating statistical information over the varying outputs [4]. Due to the need for multiple classifiers, QBC approaches have a computational overhead and are less attractive for deep neural networks and big datasets. Deep Bayesian AL methods can be seen as a more elegant way to imitate a QBC. By using stochasticity in the prediction of a network, diverse outputs can be produced and used to calculate variations in the differing predictions for the same input. For instance, BALD [5] uses Monte-Carlo Dropout over multiple inference steps and calculates mutual information to assess the worthiness of an object. Still, such an approach requires multiple forward passes, which do not scale well to large unlabeled pools. Moreover, QBC methods also suffer from problems similar to uncertainty-based sampling in batch-setting.

Diversity-based techniques [18,21] minimize the information overlap within a batch. KCENTERGREEDY [18] iteratively selects the sample with the largest minimum distance to any labeled instance in the latent space to achieve decent coverage over the data space. However, only focusing on coverage can lead to selecting outliers or uninteresting instances that do not improve the performance.

Lastly, *hybrid* approaches [2, 10, 15] combine paradigms to overcome the challenges of solely uncertainty or diversity-based methods. Many methods perform a thorough search in the latent feature space to determine a sufficiently diverse set. E.g., BADGE [2] performs k -Means++ sampling on so-called gradient embeddings where large gradients indicate uncertainty. However, these gradient embeddings depend on the number of classes and the hidden dimensionality of the penultimate layer and thus get very high-dimensional. Other methods perform weighted k -Means clustering on the latent representations [15, 25] where the weights are an uncertainty estimate and select the most central point from each cluster for annotation. Due to the repeated clustering, these methods are also computationally expensive.

AlfaMix [14] also performs k -means clustering on latent representations. In contrast to other methods, only clusters on a candidate pool determined by interpolating features in the latent space are considered. Depending on the size of the candidate pool, this increases the computational efficiency. However, as shown in Fig. 1, AlfaMix has a strong emphasis on the decision boundary, which can be problematic for highly repetitive datasets.

CDAL [1] uses a similar approach as KCenterGreedy but works on the output probabilities. It selects instances where the predicted probability is furthest away from already labeled instances. However, a problem is that some concepts in the data might be harder to learn than others. If instances get labeled, but the model needs more information in such a region, CDAL would not choose instances in the region. Task-specific hard-to-learn concepts might be ignored.

BatchBALD [10] extends BALD to the batch-setting, but has exponential time-complexity [17], making it unsuitable for our setting. Sampling from the power distribution of an uncertainty score [3, 9] instead of a deterministic top k selection to increase diversity is a faster alternative. However, finding the optimal power value is hard. Small values are close to random sampling and too large values lead to a redundant selection. Thus, these methods are highly dependent on a good parameter choice.

In contrast, FALCUN uses the powering method to stay close to the original distribution instead of increasing diversity in general: it uses a dedicated diversity mechanism to be robust against parameter selection.

In summary, the main direction of deep AL research focuses on hybrid methods in the practically relevant batch setting, finding a set of informative instances with small information overlap. However, *how* to best combine uncertainty and diversity is an ongoing challenge.

3 Methodology of FALCUN

3.1 Notation

Our task is multi-class classification on an input space \mathcal{X} of size N and a set of labels $\mathcal{Y} = \{1, \dots, C\}$ for C classes. We consider pool-based AL, where a small initial labeled set $\mathcal{L} \subset \mathcal{X}$ is uniformly drawn from the unlabeled data distribution. The remaining data objects belong to the unlabeled set $\mathcal{U} = \mathcal{X} \setminus \mathcal{L}$ of

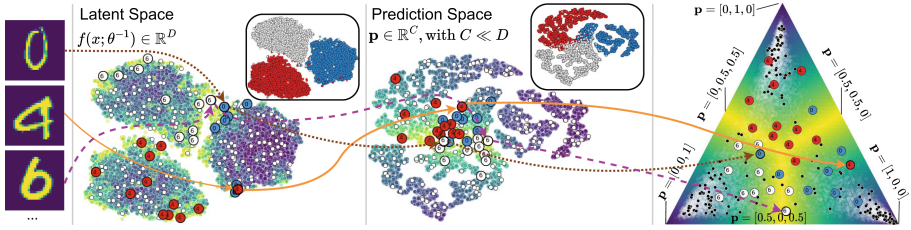


Fig. 2. FALCUN selects diverse and uncertain instances (colored circles) in the probability space (see 3-class simplex on the right). In the latent space on the left, they cover the most informative regions (yellow) while being highly diverse and stemming from different clusters. Red, white, blue imply ground truth classes. (Color figure online)

size N_u . At each AL round, Q samples are selected for annotation and retraining of the model. A classification model $f(x; \theta) \rightarrow \mathbb{R}^C$ with parameters θ maps a given input $x \in \mathcal{X}$ to a C -dimensional vector. Correspondingly, $f(x; \theta^{-1}) \rightarrow \mathbb{R}^D$ denotes the D -dimensional latent representation w.r.t. the penultimate layer of the classifier. The softmax function applied on the model output given by $f(x; \theta)$ for an object x returns the output probability vector $\mathbf{p}(x) \in [0, 1]^C$. We use a standard cross-entropy loss to optimize the parameters over the labeled pool, denoted by $\mathbb{E}_{\mathcal{L}}[l_{ce}(f(x; \theta), y)]$.

3.2 Overview

Figure 2 gives an overview of FALCUN. Instead of exploiting the latent space for diversity and the probability space for uncertainty independently, FALCUN directly uses the probabilities to select diverse *and* uncertain instances. The original data inputs (left) are forwarded through the network. The second and third columns visualize the latent and the probability space in a 2D t-SNE visualization. The colors indicate uncertainty, with yellow, lighter regions indicating higher uncertainty. On the right, the 3-dimensional simplex S is given by $S = \{(p_1, p_2, p_3) | p_i \geq 0, p_1 + p_2 + p_3 = 1\}$, where p_1, p_2, p_3 denote the posterior probability for classes 1, 2, and 3, respectively. The corners indicate a high confidence for a certain class, as reflected by a darker color. The center corresponds to a uniform posterior distribution over all classes. Small black and white dots indicate objects in \mathcal{L} and \mathbf{U} , respectively. Larger blue, red, and white circles indicate instances selected by FALCUN: they are prevalent in very informative regions in the latent space while being highly diverse.

3.3 Acquisition

Uncertainty Component. For uncertainty, we use the margin uncertainty, i.e., the difference between the probabilities of its two most probable classes:

$$u(x) := 1 - (\mathbf{p}(x)[c_1] - \mathbf{p}(x)[c_2]) \in [0, 1], \quad (1)$$

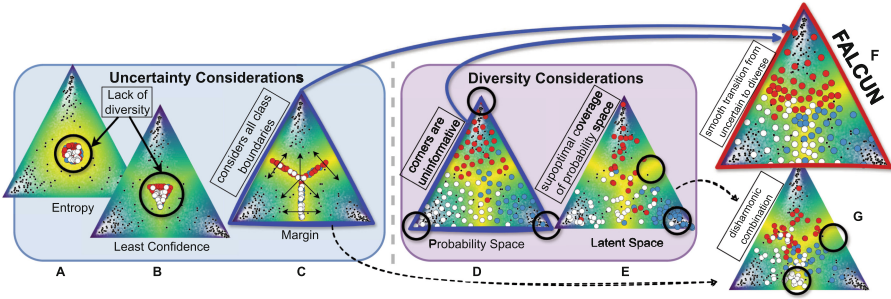


Fig. 3. Uncertainty Considerations (Left): In contrast to least confidence and entropy, the margin estimate focuses on the class boundaries between all class pairs, covering a more diverse spectrum. **Diversity Considerations** (Middle): Maximizing diversity in the probability space automatically covers diverse and uncertain regions, whereas using latent features for diversity makes a harmonic combination with uncertainty harder. **Final** (Right): **FALCUN** prefers instances at the decision boundary with a smooth transition to diverse regions.

where $0 \leq u(x) \leq 1$. Margin is a common choice for uncertainty [3, 8, 16] and naturally captures class boundaries. As illustrated in Fig. 3, margin (C) emphasizes diverse regions to be of equal interest and naturally captures more dissimilar concepts than comparable other uncertainty estimates such as entropy (A) or least confidence (B) [19]. The reason is that the margin’s extremal function has no global optimum, but its optima lie on the pairwise class boundaries in the probability space. Thus, margin uncertainty is powerful [3, 8, 25] and allows an intuitive combination with diversity, as we show in the following.

Diversity Component. To estimate diversity, we follow a similar notion as [1], measuring class-wise, contextual diversity in the probability space rather than feature-wise diversity in the possibly very high-dimensional embedding space where we might run into curse-of-dimensionality issues or computational overhead. More precisely, we measure the distances between two instances x_1 and x_2 based on their probabilities using the L1 norm $\|\cdot\|_1$:

$$dist(\mathbf{p}(x_1), \mathbf{p}(x_2)) := \|\mathbf{p}(x_1) - \mathbf{p}(x_2)\|_1 = \sum_{i=1}^C |p_i(x_1) - p_i(x_2)| \quad (2)$$

Calculating distance in the probability space accelerates computation without neglecting generalization performance [6]. Moreover, maximizing diversity in the probability space as visualized in Fig. 3 - D, automatically covers diverse and uncertain regions. In contrast, using latent features for diversity makes a harmonic combination with uncertainty harder, potentially resulting in suboptimal coverage of the probability space (see Fig. 3 E and G).

However, without careful initialization, which is hard when the query batch is still empty, maximizing diversity in the probability space also targets uninfor-

Algorithm 1. Our AL Algorithm FALCUN

Input: Unlabeled data pool \mathcal{U} , initially labeled data pool \mathcal{L} , number of acquisition rounds R , query-size Q , model $f(x; \theta)$, relevance factor γ

- 1: Train initial weights θ_0 on \mathcal{L} by minimizing $\mathbb{E}_{\mathcal{L}}[l_{ce}(f(x; \theta), y)]$
- 2: **for** $r = 1, 2, \dots, R$ **do**
- 3: Initialize empty query set: $\mathcal{Q} = \{\}$
- 4: $\forall x \in \mathcal{U}$: Compute class probabilities $\mathbf{p}(x)$
- 5: $\forall x \in \mathcal{U}$: Initialize $u(x)$ and $d(x)$ with Equations (1) and (3)
- 6: **for** $q = 1, \dots, Q$ **do**
- 7: $\forall x \in \mathcal{U}$: Calculate relevance score $r(x)$ with Equation (5)
- 8: Sample according to Equation (6)
- 9: $\mathcal{Q} = \mathcal{Q} \cup x_q$
- 10: $\forall x \in \mathcal{U}$: Update diversity values $d(x)$ using Equation (4)
- 11: **end for**
- 12: Receive new labels from oracle for instances in \mathcal{Q}
- 13: $\mathcal{L} = \mathcal{L} \cup \mathcal{Q}, \mathcal{U} = \mathcal{U} \setminus \mathcal{Q}$
- 14: Train new model θ_r from scratch on \mathcal{L} by minimizing $\mathbb{E}_{\mathcal{L}}[l_{ce}(f(x; \theta), y)]$
- 15: **end for**
- 16: **return** Final parameters θ_R obtained in round R

mative samples in the class corners. A good starting point is to focus on instances that provide different context-specific information to already well-distinguishable concepts. This can be seen as a way of diversity to the confident class corners in the simplex. The margin estimate gives us a good starting point for such diversity. Instances that receive the highest scores are (1) farthest away from the highly confident corners and (2) close to other classes. Without the second proximity consideration, focusing solely on maximizing distance to corners could bias towards the central region where all classes are equally probable (Revisit A, B, and C in Fig. 3). Margin *uncertainty* is high for instances from concepts that are *diverse* from concepts that the model can already classify confidently and, thus, naturally incorporates a diversity aspect.

Further details on the correlation between margin uncertainty and the distance to confident classes can be found in the supplementary material. Thus, we initialize the diversity score with the pre-calculated margin uncertainty and iteratively update it with each selected sample x_q :

$$d'_{init}(x) := u(x) \quad (3) \quad d'(x) \leftarrow \min(d'(x), \text{dist}(\mathbf{p}(x), \mathbf{p}(x_q))) \quad (4)$$

As diversity values can only decrease, the initialization in Eq. (3) ensures that the closer objects are to the confident corners, the less likely they will be selected. By updating the diversity score using Eq. (4), instances near objects in the current query batch receive lower scores and are less likely to be selected. Finally, we linearly normalize the values to $[0, 1]$ to align them with the uncertainty scores using min-max-normalization.

Final Relevance Score. For every point x , we calculate a relevance score $0 \leq r(x) \leq 2$, which changes over the course of each AL round. We combine the uncertainty and the diversity component by defining $r(x)$ as the sum of the uncertainty $u(x)$ and the normalized adaptive diversity score $d(x)$:

$$r(x) := u(x) + d(x). \tag{5}$$

Note that the values in $u(x)$ are static within one acquisition, but the diversity scores $d(x)$ are updated with every chosen query instance. Thus, the diversity slightly overshadows when the regions with the highest uncertainty are exhausted. When there is decent coverage in the probability space and diversity scores denote a uniform distribution, the focus is more on uncertainty. Hence, there is always a natural balance between uncertain and diverse selection depending on the current query batch. We choose x as the next query sample x_q with probability

$$x_q \sim \frac{r(x)^\gamma}{\sum_{x \in \mathcal{U}} r(x)^\gamma}, \tag{6}$$

where γ is a parameter that controls the influence of the relevance scores. $\gamma = 0$ corresponds to a uniform selection, and larger values for γ result in a stronger focus on the calculated relevance scores getting more and more deterministic (rich values get richer). Thus, γ controls the trade-off between exploration (more randomness) and exploitation (more focus on larger values in $r(x)$). See also Fig. 4, which shows the selection probabilities of points depending on their relevance scores for different values of γ . Note that we do not need γ to ensure diversity as in [3]. We use it to reduce the risk of an overly biased selection. We analyze the effect of γ and show the importance of a dedicated diversity scheme in our ablation study in Fig. 13. By combining uncertainty and diversity with our initialization, we can exploit the probability space in a harmonic way as shown in Fig. 3 F. One AL round stops when the batch \mathcal{Q} contains B samples and returns the query batch \mathcal{Q} , which will be sent to the oracle for annotation. The pseudo-code is shown in Algorithm 1.

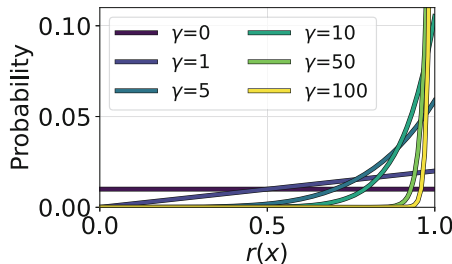


Fig. 4. Selection probability of an instance x for different γ values as a function of its relevance score $r(x)$.

Table 1. Data set properties: number of points N , number of classes C , and number of input features F .

Type	Data set	N	C	F
Image (Gray)	MNIST	60,000	10	28x28
	RMNIST	60,000	10	28x28
	FashionMNIST	60,000	10	28x28
	EMNIST	131,600	47	28x28
Image (Color)	SVHN	73,257	10	32x32x3
	BloodMNIST	11,959	8	28x28x3
	DermaMNIST	7,007	7	28x28x3
	CIFAR.10	60,000	10	28x28x3
	Tabular	OpenML-6	16,000	26
	OpenML-156	800,000	5	11
	OpenML-155	829,201	10	11

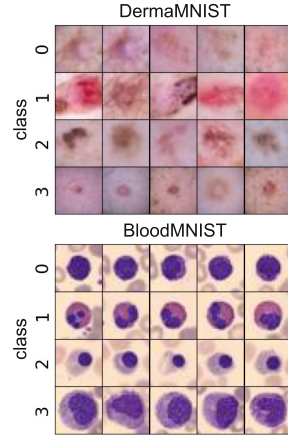


Fig. 5. Exemplary images of the two medical datasets.

4 Experiments

We evaluate the effectiveness of established AL methods and FALCUN regarding quality and acquisition runtime in isolation as well as in combination to get a complete picture. We use a broad range of datasets including grayscale images (MNIST [12], FashionMNIST [23], and EMNIST), colored images (CIFAR10 [11], SVHN [13], BloodMNIST, DermaMNIST [24]), and tabular datasets from the OpenML benchmark¹ suite (Ids: 6, 155, 156). BloodMNIST and DermaMNIST are challenging medical image datasets showcasing a task where labeling experts are limited and costly. Figure 5 shows some examples. Within a class, images can be very similar, s.t. their information is redundant. A good AL strategy should avoid selecting such repetitive instances to optimize label efficiency. To further assess the capabilities to sample a diverse subset, we include redundant versions of MNIST named RMNIST containing duplicate images (comparable to [10]). We randomly keep 10% unique original images and fill the rest with duplicated versions with added Gaussian noise. We vary the redundancy ratio in an extra experiment. Table 1 summarizes the data properties. For grayscale data we use a LeNet, a learning rate of 0.01 and train for 20 epochs. For colored data we use pre-trained Resnet18, and ResNet50, a learning rate of 0.001 and stop when a training accuracy of 99% is reached. We investigate whether the results are similar without pre-trained weights and when initializing the model with the weights from the previous round as proposed in [14]. For tabular data we use a simple multi-layer-perceptron (MLP) with two layers as proposed in [2] (hidden

¹ <https://www.openml.org/>.

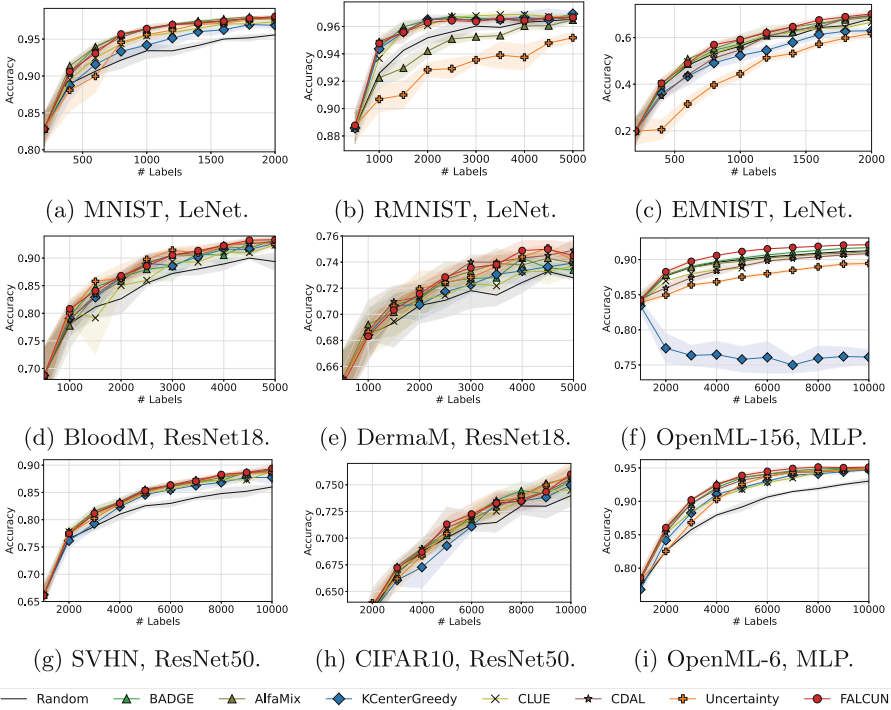


Fig. 6. Average test accuracy vs labeling budget for all active learning methods evaluated on greyscale (a, d), RGB (b, e) and tabular data (c, f).

dimensionality 1024), a learning rate of 0.0001 and use early stopping when a training accuracy of 99% is reached. We use an Adam optimizer. All experiments are performed five times with different seeds. We compare to state-of-the-art hybrid methods: BADGE [2], CDAL [1], CLUE [15], and ALFAMIX [14]. We include a diversity baseline: KCENTERGREEDY [18], an uncertainty baseline: ENTROPY sampling [19], and the passive baseline RANDOM sampling. For FALCUN, we set $\gamma = 10$. Further details are given in the publicly available code base.

4.1 Label Efficiency

Figure 6 shows the learning curves of diverse architectures and query sizes for evaluated datasets. The x-axis depicts the labeling budget, and the y-axis gives the average accuracy for varying AL methods. We see that FALCUN is among the best-performing methods for varying query sizes, data types, and model architectures. FALCUN also yields the strongest results on the tabular data: in contrast to all other competitors, it consistently outperforms random sampling on the Openml-156 dataset. Note that the ranking of the best-performing meth-

Table 2. Avg. Accuracy on CIFAR10 with varying architectures and settings. BB = backbone model, P = Pre-trained weights are used, Ctl = Continual setting where weights are not reset after each AL round, B=Labeling budget. FALCUN has most often **best** (bold) or second best performance (underlined).

BB	CtlP B	CLUE	BADGE	CDAL	AlfaMix	Random	FALCUN
Resnet50	✓ 6000	71.7	72.1	71.9	71.8	71.3	72.3
	✓ 10000	74.5	75.3	75.5	75.6	74.0	76.0
	✓ 6000	52.0	51.9	51.4	51.6	51.1	52.0
	✓ 10000	57.5	<u>58.6</u>	59.3	58.3	57.4	58.5
Resnet18	✓ 6000	<u>70.1</u>	69.9	69.8	69.9	69.4	70.2
	✓ 10000	73.6	74.0	73.5	73.6	72.3	<u>73.5</u>
	✓ 6000	54.8	55.6	55.2	55.8	54.7	55.9
	✓ 10000	60.5	60.7	61.0	60.5	59.2	<u>60.9</u>

Table 3. Avg. Accuracy on CIFAR10 with pre-trained Resnet50 using initial pool sizes (I) and query sizes (QS). We report budgets (B) after the first and last acquisition. FALCUN performs well with varying AL settings.

I	QS	B	CLUE	BADGE	CDAL	AlfaMix	Random	FALCUN
1000	1000	2000	63.4	63.4	63.4	62.9	63.2	63.7
		10000	74.5	75.3	75.5	75.6	74.0	76.0
2000	2000	4000	68.4	68.4	68.5	<u>69.3</u>	69.5	68.8
		10000	74.9	<u>75.3</u>	75.0	75.7	71.7	75.0
5000	5000	10000	74.6	75.0	74.2	75.0	74.0	75.3
		20000	78.6	78.8	79.3	79.3	76.9	79.3
5000	7500	12500	75.2	75.9	75.2	76.4	75.1	76.4
		22500	78.0	<u>79.6</u>	79.8	79.3	77.9	<u>79.6</u>

ods is not the same over varying settings. E.g., Entropy, an only uncertainty-based technique, yields good results on BloodMNIST but underperforms on certain other datasets such as EMNIST, RMNIST or Openml-156. In contrast, KCenterGreedy, a solely diversity-based approach, only yields fairly good results on the highly redundant dataset RMNIST but performs poorly on Openml-156. Not surprisingly, some datasets and settings benefit more from uncertainty, and others might work better with diversity. Table 2 show results on CIFAR10 when varying the backbone (BB), using pre-training or not (P) and using continual training instead of starting from scratch after every AL round (Ctl) for varying budgets (B). Most often, FALCUN yields best or second best results. Table 3 shows results when varying the initial pool size (I) and query size (QS) for different Budgets (B). Again, FALCUN yields best or second best results frequently. All in all, FALCUN is robust across varying settings.

Dealing with Redundancy. We especially want to emphasize that though only operating on the output probabilities, FALCUN’s success is not diminished on RMNIST. Figure 8 shows how the performance of all AL methods drops for varying redundancy ratios of the RMNIST dataset. Besides Entropy sampling, AlfaMix’s quality decreases rapidly for highly redundant datasets. We hypothesize this is due to oversampling the decision boundary, as visualized in Fig. 1. We provide all learning curves in the supplementary materials.

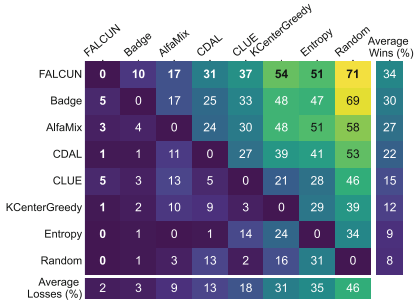


Fig. 7. Dueling matrix: The last column gives the percentage of wins of the respective method. The last row gives the percentage of losses.

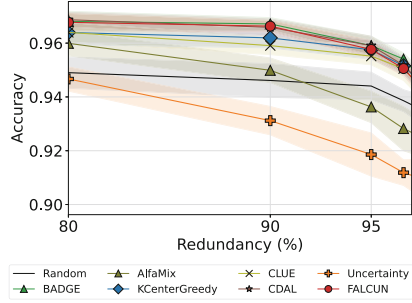


Fig. 8. Final average test accuracy for varying redundancy ratios.

Dueling Matrix Over All Experiments. Designing a robust method is hard when the characteristics of a dataset are unknown in advance. Moreover, in AL, it is hard to compare all learning curves from all experiments, and sometimes, a clear winner is hard to find. Hence, similar to previous works [2, 7, 14], we provide a dueling matrix for a comprehensive analysis of the methods’ overall performance. The column-wise entries in the matrix in Fig. 7 show the amount of **losses**, and the row-wise entries indicate the amount of **wins** against each other method (in %). A win means that for a specific experimental setting, i.e., a specific dataset, acquisition round, query size, and model architecture, comparing the results of 5 runs, a method has statistically better accuracy than the other method (with p-value=0.05).

A loss is defined analogously. Losses and wins do not necessarily sum up to 100% as the two methods can perform comparably well with no statistical difference. When discussing the quality of an AL method, it is hence important to evaluate the wins *and* losses. The bottom row and the rightmost column denote the average losses and wins over all experiments compared to all other AL methods. FALCUN is consistently strong over a wide range of datasets, as the dueling matrix in Fig. 7 shows. FALCUN has the most wins (highest numbers in every column) compared to every other method and the most wins over random

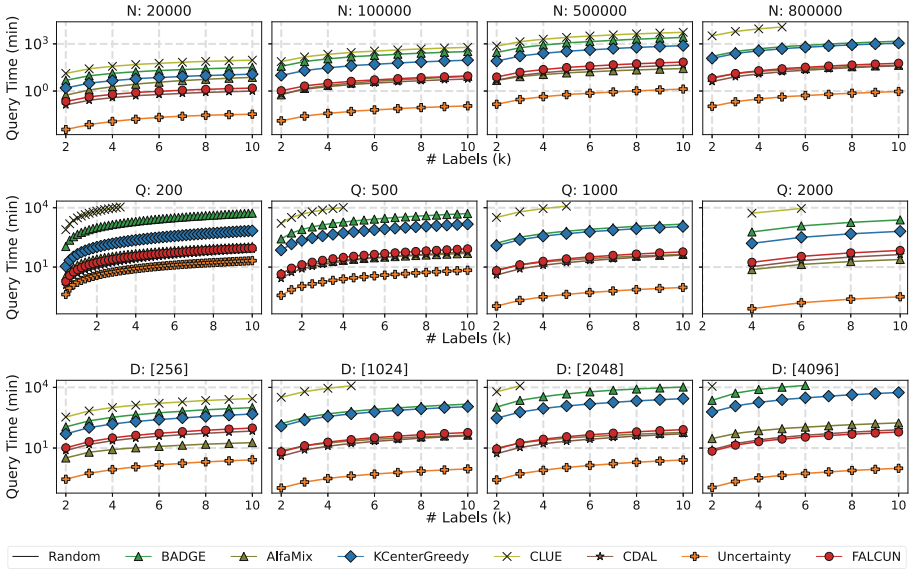


Fig. 9. Average cumulated acquisition times (y-axis) on a log-scale vs. annotated samples (x-axis) over varying unlabeled pool sizes N (first row), query sizes Q (second row), and dimensionality of the penultimate layer D (third row).

sampling. Simultaneously, it has the fewest losses. Only FALCUN is *never worse than random sampling*, one of the most important criteria for successful AL methods.

4.2 Query Time Efficiency

The training for the grayscale image datasets and tabular datasets is arguably fast (around 1 min for the last AL round). For the colored image data, training takes around 75 min in the last round. In such situations, the limiting factor for the overall runtime is the query time. We systematically analyzed the scalability of all tested methods by varying dataset size, query size, and hidden dimensionality of the multilayer perceptron evaluated for the largest of all datasets (i.e., Openml-156). We stopped each experiment after ten days (e.g. CLUE). The results are shown in Fig. 9. FALCUN denotes fast and robust runtimes over varying settings, being comparably fast as CDAL and particularly robust to varying hidden dimensionality. We summarize these extensive experiments by giving the smallest and largest average query times among the scalability analysis in Table 4. Moreover we provide runtime complexities for all methods. Note that the runtime complexity of our acquisition is dependent on the size of the unlabeled pool, the query size, and the number of classes ($\mathcal{O}(Q \cdot N_u \cdot C)$) but not on the hidden dimensionality D . BADGE, one of the strongest competitors regarding label efficiency, has a worse runtime complexity with $\mathcal{O}(Q \cdot N_u \cdot C) \in \mathcal{O}(Q \cdot N_u \cdot C \cdot D)$.

Table 4. Time Complexity w.r.t. query size Q , Dimensionality of latent features D , unlabeled pool size N_u , number of classes C , labeled pool size N_l , number of cluster rounds i , and a method-specific candidate pool in AlfaMix N_{cp} with $N_{cp} \leq N_u$, final min. and max. average cumulated query time among the scalability analysis.

AL Strategy	Time Complexity	min	max
Entropy	$\mathcal{O}(N_u)$	1.8 sec	21 min
CDAL	$\mathcal{O}(N_l \cdot N_u \cdot C + Q \cdot N_u)$	1 min	80 min
FALCUN	$\mathcal{O}(Q \cdot N_u \cdot C)$	1.5 min	97 min
AlfaMix	$\mathcal{O}(Q \cdot N_{cp} \cdot i \cdot D)$	7.3 min	175 min
KCenterGreedy	$\mathcal{O}(N_l \cdot N_u \cdot D + Q \cdot N_u)$	11.8 min	25 h
BADGE	$\mathcal{O}(Q \cdot N_u \cdot C \cdot D)$	31.5 min	208 h
CLUE	$\mathcal{O}(Q \cdot N_u \cdot i \cdot D)$	92 min	>227 h

That leads to multiple times higher run times compared to FALCUN (208hrs in the worst case for BADGE vs 97minutes for FALCUN). CDAL, followed closely by FALCUN, is the fastest among all tested methods. In the fastest setting, when the unlabeled pool contains 20,000 objects, FALCUN is only half a minute slower than CDAL. In the most challenging setting with a latent dimension of 4096, FALCUN is only 17% slower.

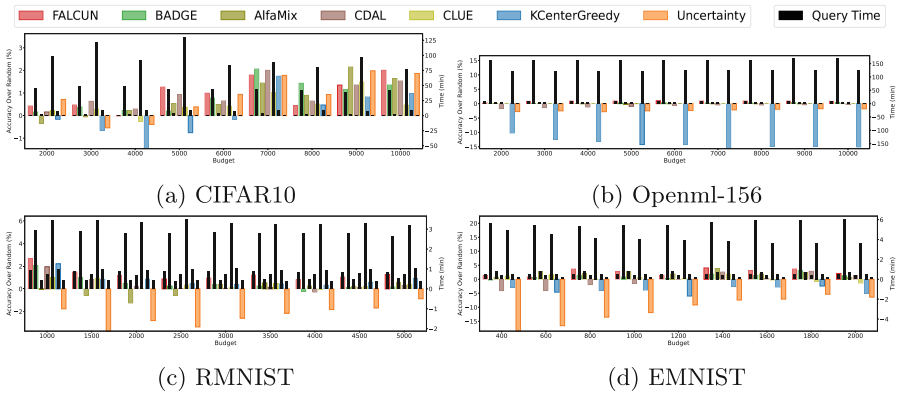


Fig. 10. Runtimes (black bars, smaller is better) and improvement over random sampling in average test accuracies (colored bars, larger is better) for all acquisition rounds for tabular data (Openml-155 and Openml-156) and grayscale data (RMNIST, EMNIST).

Considering quality and runtime together, Fig.10 shows the improvement over random sampling in terms of average accuracy per method (colored bars) and the corresponding query time in minutes in a certain acquisition round (black thin bars) for all tested methods. *Large accuracy bars are better* whereas

smaller time bars are better. FALCUN (red bars) has strong performance on all datasets and never has worse average accuracy than random sampling (i.e., values smaller than zero). CLUE and especially BADGE perform on par in some settings, but their query times are much higher, in some cases up to > 200 hours. AlfaMix is fast and has good quality on Openml-155 and decent performance on EMNIST. However, AlfaMix is prone to duplicates: it performs even worse than random sampling on RMNIST in many acquisition rounds. CDAL is quite fast but performs worse than random sampling more often, especially for small budgets on EMNIST and Openml-156. Entropy is fast, but not label-efficient. KCenterGreedy is fast for smaller datasets (e.g., RMNIST and EMNIST) but does not scale well to larger datasets (see Openml-156) and is only comparably label-efficient for the redundant dataset RMNIST because it has the strongest emphasis on maximizing diversity. FALCUN has a robust performance across all datasets and low query times (never above 10 min).

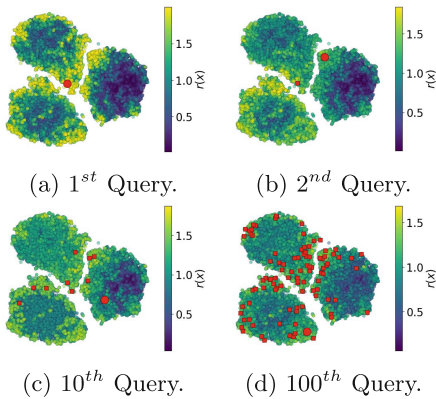


Fig. 11. Exemplary course of relevance scores $r(x)$ and their dependency of selected queries (red) on 3-class MNIST, t-SNE visualization. (Color figure online)

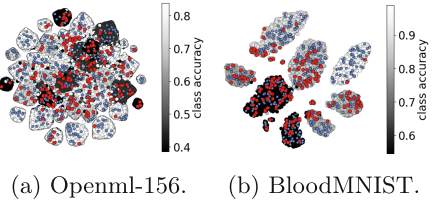


Fig. 12. Hue in the t-SNE visualizations indicates the predictive accuracy of the model on the respective class. Initially sampled objects are blue, samples chosen by FALCUN in the first acquisition round are red. FALCUN selects diverse instances favoring classes that are harder to distinguish by the current model: “darker” classes contain more red dots. (Color figure online)

4.3 Qualitative Evaluation

Figure 11 illustrates the selection of instances and the course of FALCUN’s relevance scores $r(x)$ over one acquisition round on a 3-class MNIST task (also

used for the visualization in Fig. 2) for better interpretability. Yellow regions indicate a high relevance score promoting regions of high interest. Initially, all instances with high uncertainty, primarily located at the decision boundary, receive higher scores (see Fig. 11a). The score in the surrounding of the selected instance (red circle) gets darker as the objects located close to it receive a smaller diversity score (see Fig. 11b). In the first iterations, uncertain, but still diverse instances are preferred. In Fig. 11d we derive a diverse set located in all three clusters mainly consisting of objects from uncertain areas.

In Fig. 12, we analyze FALCUN’s selection on Openml-156 (Fig. 12a) and BloodMNIST (Fig. 12b). It effectively finds instances majorly located in regions where the classifier has more confusion (darker areas) while still enhancing diversity and not oversampling certain regions. E.g., on the right, most instances are chosen from the two most uncertain classes ($\sim 55\%$ accuracy). In contrast, only two objects are selected from the most confident class where the model already achieves $\sim 99\%$ accuracy.

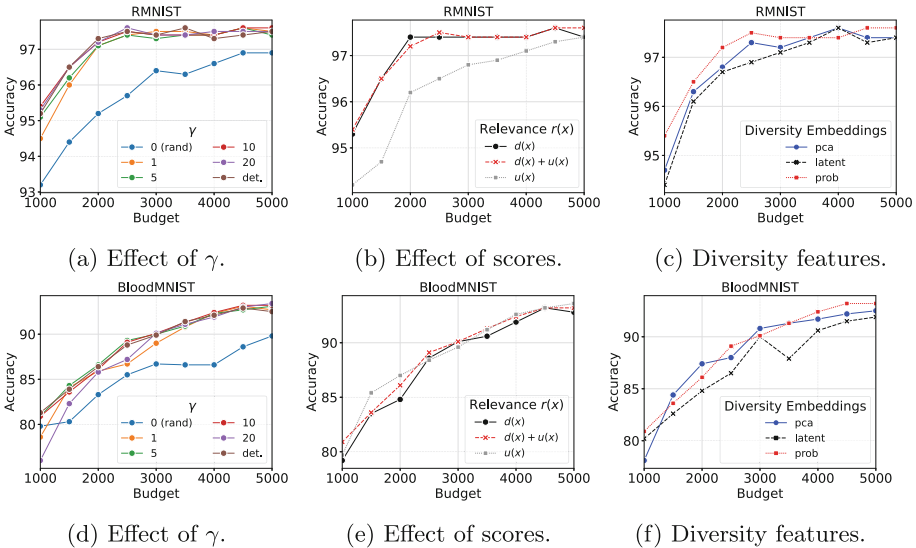


Fig. 13. Ablation Study on RMNIST (top row) and BloodMNIST (bottom row).

4.4 Ablation

Effect of γ . In Figs. 13a and 13d, we vary γ , where smaller values lean towards uniform selection and larger values lean towards deterministic selection, including a completely deterministic selection (det.). While a random selection ($\gamma = 0$, blue line) is always worst, we see that the exact choice of γ does not largely affect the performance. Having a value between 5 and 20 yields very robust

and consistent results. A deterministic selection seems similarly strong despite a few fluctuations. However, we argue that we should stick to our probabilistic selection so as not to end up in a failure mode due to highly biased selection.

Effect of Scores. Figures 13b and 13e show the results when switching off either the uncertainty or the diversity component to calculate the final relevance score. For RMNIST, considering uncertainty without diversity yields the worst results. Hence, powering similar to [3] without a dedicated diversity function is less effective for highly redundant datasets. BloodMNIST benefit more from uncertainty than from diversity. In general, our experiments show that sometimes uncertainty and sometimes diversity are more important. However, knowing which type is needed in a real-world scenario is notoriously hard when there is almost no information. In contrast, our combined score is always among the best, and due to the robustness across datasets, it is a highly attractive choice.

Effect of Diversity Features. Lastly, we investigate the performance when calculating diversity on the latent embeddings instead of the final output probabilities. As a simple baseline we also perform PCA on the latent features and use the result as input for the diversity component (see Figs. 13c and 13f). Interestingly, using latent features is worst in many situations. We assume this is due to curse-of-dimensionality issues. Furthermore, using the probability vector is almost always the best method. We hypothesize that using the probability space for uncertainty and diversity leads to a more harmonized selection. Our diversity in the probability space also indirectly covers uncertain regions, and the margin uncertainty function indirectly covers diverse concepts. Combining two isolated scores can be tricky since it could unintentionally set a too strong focus on one or the other component.

5 Conclusion

We introduced FALCUN, a novel deep AL method that employs a natural transition from emphasizing uncertain instances at the decision boundary towards enhancing more batch diversity. This natural balance ensures robust label efficiency on varying datasets, query sizes, and architectures, even on highly redundant datasets. As FALCUN only operates on the output probability vectors, it achieves faster acquisition times than many established methods performing a search through the high-dimensional embedding space of a neural network.

References

1. Agarwal, S., Arora, H., Anand, S., Arora, C.: Contextual diversity for active learning. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12361, pp. 137–153. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58517-4_9
2. Ash, J.T., Zhang, C., Krishnamurthy, A., Langford, J., Agarwal, A.: Deep batch active learning by diverse, uncertain gradient lower bounds. In: ICLR (2020)

3. Bahri, D., Jiang, H., Schuster, T., Rostamizadeh, A.: Is margin all you need? an extensive empirical study of active learning on tabular data. arXiv preprint [arXiv:2210.03822](https://arxiv.org/abs/2210.03822) (2022)
4. Dagan, I., Engelson, S.P.: Committee-based sampling for training probabilistic classifiers. In: Machine Learning Proceedings 1995, pp. 150–157 (1995)
5. Gal, Y., Islam, R., Ghahramani, Z.: Deep bayesian active learning with image data. In: ICML, pp. 1183–1192 (2017)
6. Gilhuber, S., Berrendorf, M., Ma, Y., Seidl, T.: Accelerating diversity sampling for deep active learning by low-dimensional representations. In: Kottke, D., Krempl, G., Holzinger, A., Hammer, B. (eds.) Proceedings of the Workshop on Interactive Adaptive Learning co-located with European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2022), Grenoble, France, September 23, 2022. CEUR Workshop Proceedings, vol. 3259, pp. 43–48 (2022)
7. Gilhuber, S., Busch, J., Rotthues, D., Frey, C.M., Seidl, T.: Diffusal: Coupling active learning with graph diffusion for label-efficient node classification. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 75–91. Springer (2023)
8. Jiang, H., Gupta, M.R.: Bootstrapping for batch active sampling. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining pp. 3086–3096 (2021)
9. Kirsch, A., Farquhar, S., Atighehchian, P., Jesson, A., Branchaud-Charron, F., Gal, Y.: Stochastic batch acquisition for deep active learning. arXiv preprint [arXiv:2106.12059](https://arxiv.org/abs/2106.12059) (2021)
10. Kirsch, A., Van Amersfoort, J., Gal, Y.: Batchbald: efficient and diverse batch acquisition for deep bayesian active learning. NeuRIPS, pp. 7026–7037 (2019)
11. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
12. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
13. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning. Neural Information Processing Systems (NIPS) (2011)
14. Parvaneh, A., Abbasnejad, E., Teney, D., Haffari, G.R., Van Den Hengel, A., Shi, J.Q.: Active learning by feature mixing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12237–12246 (2022)
15. Prabhu, V., Chandrasekaran, A., Saenko, K., Hoffman, J.: Active domain adaptation via clustering uncertainty-weighted embeddings. In: ICCV, pp. 8505–8514 (2021)
16. Roth, D., Small, K.: Margin-based active learning for structured output spaces. In: European Conference on Machine Learning, pp. 413–424 (2006)
17. Rubashevskii, A., Kotova, D., Panov, M.: Scalable batch acquisition for deep bayesian active learning. In: Proceedings of the 2023 SIAM International Conference on Data Mining (SDM), pp. 739–747. SIAM (2023)
18. Sener, O., Savarese, S.: Active learning for convolutional neural networks: a core-set approach. In: ICLR (2018)
19. Settles, B.: Active learning literature survey (2009)
20. Shannon, C.E.: A mathematical theory of communication. ACM SIGMOBILE mobile computing and communications review **5**(1), 3–55 (2001)
21. Sinha, S., Ebrahimi, S., Darrell, T.: Variational adversarial active learning. In: ICCV, pp. 5972–5981 (2019)

22. Wang, D., Shang, Y.: A new active labeling method for deep learning. In: 2014 International Joint Conference on Neural Networks (IJCNN), pp. 112–119. IEEE (2014)
23. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. [arXiv:1708.07747](https://arxiv.org/abs/1708.07747) (2017)
24. Yang, J., Shi, R., Wei, D., Liu, Z., Zhao, L., Ke, B., Pfister, H., Ni, B.: Medmnist v2- a large-scale lightweight benchmark for 2d and 3d biomedical image classification. *Sci. Data* **10**(1), 41 (2023)
25. Zhdanov, F.: Diverse mini-batch active learning. [arXiv:1901.05954](https://arxiv.org/abs/1901.05954) (2019)



Semi-supervised Heterogeneous Domain Adaptation via Disentanglement and Pseudo-labelling

Cassio F. Dantas^{1,3(✉)}, Raffaele Gaetano^{2,3}, and Dino Ienco^{1,3,4}

¹ INRAE, UMR TETIS, Univ. Montpellier, Montpellier, France
{cassio.fraga-dantas,dino.ienco}@inrae.fr

² CIRAD, UMR TETIS, Univ. Montpellier, Montpellier, France
raffaele.gaetano@cirad.fr

³ INRIA, Univ. Montpellier, Montpellier, France

⁴ LIRMM, Univ. Montpellier, CNRS, Montpellier, France

Abstract. Semi-supervised domain adaptation methods leverage information from a source labelled domain with the goal of generalizing over a scarcely labelled target domain. While this setting already poses challenges due to potential distribution shifts between domains, an even more complex scenario arises when source and target data differs in modality representation (e.g. they are acquired by sensors with different characteristics). For instance, in remote sensing, images may be collected via various acquisition modes (e.g. optical or radar), different spectral characteristics (e.g. RGB or multi-spectral) and spatial resolutions. Such a setting is denoted as Semi-Supervised Heterogeneous Domain Adaptation (SSHDA) and it exhibits an even more severe distribution shift due to modality heterogeneity across domains.

To cope with the challenging SSHDA setting, here we introduce *SHeDD* (Semi-supervised Heterogeneous Domain Adaptation via Disentanglement) an end-to-end neural framework tailored to learning a target domain classifier by leveraging both labelled and unlabelled data from heterogeneous data sources. *SHeDD* is designed to effectively disentangle domain-invariant representations, relevant for the downstream task, from domain-specific information, that can hinder the cross-modality transfer. Additionally, *SHeDD* adopts an augmentation-based consistency regularization mechanism that takes advantages of reliable pseudo-labels on the unlabelled target samples to further boost its generalization ability on the target domain. Empirical evaluations on two remote sensing benchmarks, encompassing heterogeneous data in terms of acquisition modes and spectral/spatial resolutions, demonstrate the quality of *SHeDD* compared to both baseline and state-of-the-art competing approaches. Our code is publicly available [here](#).

Keywords: Domain Adaptation · Heterogeneous data · Feature disentanglement · Pseudo-labeling · Consistency regularization

1 Introduction

When it comes to real-world applications of machine learning, disposing of a vast amount of labelled samples remains a major issue in many domains, especially those featured by costly and time-consuming labelling processes. Consequently, make value of already available data, covering similar downstream tasks, is of paramount importance to enhance the classification performances on target domains where labelled data are scarce. Nonetheless, this process is not straightforward due to potential differences or shifts in their underlying data distributions between a rich source labelled domain and the target one [26]. To cope with data distribution shifts between source and target domains, Domain Adaptation (DA) techniques have been proposed [21]. The main objective of this family of machine learning methods is to learn a classification model across different domains, generally sharing the same set of classes, with the aim to transfer information from a source to a target one.

Many research efforts have focused on addressing situations wherein the target domain lacks completely of associated labels, while only the source domain disposes of labelled information [21]. This scenario is commonly referred as Unsupervised Domain Adaptation (UDA). However, a more practical assumption for real-world applications is to have access to a small amount of labelled information from the target domain, enabling the simultaneous exploitation of abundant labelled samples from the source domain and limited labelled samples from the target domain. Such a setting is generally termed as Semi-Supervised Domain Adaptation [15], and existing literature has highlighted that directly using UDA approaches fails to exploit the label information associated with the target domain, thus requiring tailored solutions for this setting [16].

Nevertheless, most of the aforementioned research strategies rely on the strong assumption that data coming from source and target domains share a similar (homogeneous) data modality representation. However, in real-world applications data can be collected by means of heterogeneous sensors, as it is the case for remote sensing imagery exhibiting differences in acquisition modes (e.g. optical and radar), spectral characteristics (RGB, multi-/hyper-spectral), and spatial resolution. Consequently, it is increasingly common to encounter label-abundant source domains and label-scarce target domains that are heterogeneous in terms of data modality representation, further exacerbating potential data distribution shifts between domains. To address this challenging scenario, Semi-Supervised Heterogeneous Domain Adaptation (SSHDA) methods are gaining increasing attention within the research community [2]. However, the majority of the proposed approaches rely on pre-trained deep learning models that are only available for standard modalities (e.g. RGB imagery, text data), limiting their applicability in scenarios involving non-standard sensor data, such as those used in the medical [3] and remote sensing [13] fields.

With the aim to address the challenging SSHDA setting, in this research work we introduce a new end-to-end deep learning framework especially tailored to learn a target domain classifier by leveraging both labelled and unlabelled data from heterogeneous data sources. Our framework, *SHeDD* (Semi-supervised

Heterogeneous Domain Adaptation via Disentanglement), tackles data modality heterogeneity by extracting, via a feature disentanglement approach, domain-invariant representations, relevant for the downstream task, and domain-specific information, that can prevent cross-modality transfer. To this end, invariant and domain specific features are enforced to be orthogonal to each other with the latter carrying domain-discriminant information. Furthermore, *SHeDD* harnesses unlabelled target data to enhance its generalization ability, aiming to transfer discriminative information from the labelled source domain to the scarcely-labelled target domain. This last point is achieved via consistency learning where confident pseudo-labels are derived on the target domain by the classification model and subsequently exploited in the training process. Empirical evaluations on two remote sensing benchmarks, encompassing heterogeneous data domains in terms of acquisition modes and spectral/spatial characteristics/resolutions, clearly demonstrate the quality of *SHeDD* compared to both baseline and state-of-the-art competing methods.

This paper is organized as follows: related works are discussed in Sect. 2; the proposed method is described Sect. 3; experimental results are presented and discussed in Sect. 4, followed by concluding remarks in Sect. 5.

2 Related Works

Domain adaption [21] (DA) methods belong to the family of transfer learning approaches [26] which have the main objective to transfer a model trained on a labelled source domain to a target domain. When the target domain is completely unlabelled, Unsupervised Domain Adaptation (UDA) strategies are designed in order to align domains through data transformation and/or extract domain-invariant features to reduce the distribution gap between the labelled source and the unlabelled target domain [10].

When, for the target domain, a limited amount of labelled samples are available, Semi Supervised Domain Adaptation (SSDA) strategies have been proposed to combine both labelled (source and target) with unlabelled (target) information [5, 8, 9, 14, 16, 22–24]. In [16], a framework based on Minimax Entropy is introduced to exploit the available target supervision. The same research work clearly illustrates that directly use UDA methods performs poorly when small amount of labeled samples are accessible from the target domain, thus emphasizing the need to design specialized approaches for the SSDA setting. In [14], an adversarial learning paradigm is leveraged in order to obtain two contradictory classifiers (source and target), enforcing well-scattered source features and compact target features respectively. A slightly different approach is proposed in [8], where a cross-domain adaptive clustering algorithm is presented to achieve cluster-wise feature alignment across domains, still employing an adversarial learning strategy. In [5], additional adversarial examples are introduced to fill the gap between domains and model robustness. The work in [23] decomposes SSDA into an SSL problem within the target domain coupled with an inter-domain UDA problem, then optimizes both tasks simultaneously using co-training. Moving away from the adversarial paradigm, [17] proposes a contrastive

learning framework operating both at a class level to reduce inter-domain gap and at the instance level with strong augmentations to minimize intra-domain discrepancy. Mitigating discrepancy within the target domain is also the main goal in [6], which proposes a feature alignment approach for achieving it.

Despite the effectiveness demonstrated by these methods, they are especially designed for managing homogeneous domains since they capitalize on the fact that source and the target domains share a similar modality representation (e.g., both involving RGB images). Therefore, the direct extension of these approaches to handle a heterogeneous setting, where source and target data differ in modality representation, is challenging.

In recent years, research efforts have been devoted towards addressing DA in an heterogeneous setting [1]. These efforts primarily focus on aligning the source and target domains through heterogeneous feature transformations. For instance, [25] learns feature transformations to map source and target data into a common latent space, where both marginal and class-conditional distributions are matched. In addition, self-training via pseudo labelling is used to update the target label set. Another approach proposed in [2] introduces a joint mean embedding alignment method where a neural network based approach aligns source and target data distribution via domain discrepancy minimization. However, these methods rely on features derived either through hand-crafted processes or from modality/domain specific pre-trained models (e.g. RGB pre-trained model) thus lacking end-to-end behaviour. Such reliance can prevent their applicability in scenarios involving data beyond the standard RGB (three channels) format. Recently, [12] has introduced an end-to-end SSHDA method, addressing the aforementioned limitations. This method adopts per-domain encoders sequentially connected to a shared backbone, with a classification head used for the final decision. During training, the neural network is optimized for simultaneously classifying and align the embedding representations coming from the different heterogeneous domains with standard cross entropy and domain critic discrimination based on wasserstein distance, respectively.

In this work we propose a different framework for SSHDA based on feature *disentanglement*, intended as the capacity of a network to identify domain-invariant representations, relevant for the downstream task, by explicitly seeking in parallel to isolate domain-specific information which may hinder the cross-modality transfer.

3 Proposed Method

The proposed architecture, summarized in Fig. 1, is given by two independent encoder branches with specialized backbones (one dedicated to the source data modality and another to the target data), followed by two parallel classifiers (a task classifier and a domain classifier).

A given input data \mathbf{x} is firstly encoded by its matching backbone and the obtained embedding vector $\mathbf{z} = g(\mathbf{x}) \in \mathbb{R}^{2D}$ is then split into two equal parts: $\mathbf{z}^{spe} \in \mathbb{R}^D$ and $\mathbf{z}^{inv} \in \mathbb{R}^D$. While the former vector is fed into the domain

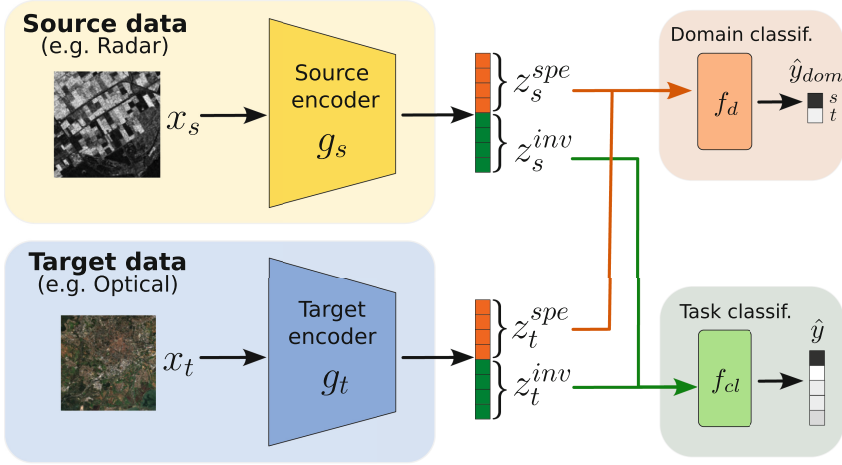


Fig. 1. Schematic view of the proposed method architecture with a separate encoder for each of the data modalities (source and target). Feature disentanglement enables domain-specific and domain-invariant information to be encoded separately into each half of the generated embedding vectors (depicted in orange and green respectively). The domain-invariant information (z^{inv}) is used by the task classifier, while the domain-specific information (z^{spe}) is used by the domain classifier. At inference time, only the bottom part of the architecture is used, the top part being instrumental in the training stage to enable the feature disentanglement procedure. (Color figure online)

classifier f_d , a binary classifier that tries to predict from which branch (source or target) the sample originates, the latter one is sent to the task classifier f_{cl} that outputs class probabilities $\hat{y} = f_{cl}(z^{inv}) \in \mathbb{R}^C$ for the C existing classes.

At training time, guided by the losses described in Sect. 3.2, the weights of these four modules —source and target encoders, task and domain classifiers— are optimized on the available supporting data composed of the following sets:

$$\begin{aligned}
 S &:= \{(\mathbf{x}_s, y_s)^{(i)}\}_{i=1}^{N_s} && \text{labelled source data.} \\
 T &:= \{(\mathbf{x}_t, y_t)^{(i)}\}_{i=1}^{N_t} && \text{labelled target data.} \\
 U &:= \{\mathbf{x}_u^{(i)}\}_{i=1}^{N_u} && \text{unlabelled target data.}
 \end{aligned}$$

From each unlabeled target sample \mathbf{x}_u , we generate a corresponding augmented counterpart (see details in Sect. 3.2) denoted $\mathbf{x}_{\hat{u}}$ that form the set below:

$$\hat{U} := \{\mathbf{x}_{\hat{u}}^{(i)}\}_{i=1}^{N_u} \quad \text{augmented unlabelled target data.}$$

where we denote N_s , N_t and N_u the corresponding dataset sizes.

At inference time, only the target encoder is retained. Similarly, only the task classifier is required. The two dropped modules, however, are crucial as supporting elements during training in order to fully guide the network’s ability to effectively disentangle domain-invariant from domain-specific information.

This ability, acquired during training and carried over to inference time in the two retained modules, helps enhancing the generalization capabilities of the final network.

3.1 Training Losses

In case of a labeled training sample, from either source or target domain, the output of the task classifier $f_{cl}(\mathbf{z}^{inv})$ is compared to its ground-truth annotation y in the following cross-entropy classification loss:

$$\mathcal{L}_{cl} = \text{CE}(f_{cl}(\mathbf{z}^{inv}), y). \tag{1}$$

Because the provenance domain $y_{dom} \in \{s, t\}$ of any given data sample is always known (even for unlabeled target samples), the domain classifier prediction $f_d(\mathbf{z}^{spe})$ can be systematically taken into account in the following cross-entropy loss:

$$\mathcal{L}_{dom} = \text{CE}(f_d(\mathbf{z}^{spe}), y_{dom}). \tag{2}$$

To further enforce disentanglement between domain-invariant and domain-specific information, we enforce orthogonality between the two embedding types for any given input sample (source and target, labelled and unlabelled):

$$\mathcal{L}_{\perp} = \frac{\langle \mathbf{z}^{inv}, \mathbf{z}^{spe} \rangle}{\|\mathbf{z}^{inv}\|_2 \|\mathbf{z}^{spe}\|_2}. \tag{3}$$

To fully exploit the available target unlabelled data (set U), for each sample $\mathbf{x}_u \in U$ we first generate an associated augmented sample $\mathbf{x}_{\hat{u}} = \text{Augment}(\mathbf{x}_u)$ (see details in Sect. 3.2) and then employ an unsupervised loss *à la* FixMatch [18] that enforces consistency between predictions obtained from the unlabeled sample \mathbf{x}_u and its augmentation $\mathbf{x}_{\hat{u}}$ via pseudo-labelling procedure:

$$\mathcal{L}_{pl} = m_u^\tau \text{CE}(f(\mathbf{z}_{\hat{u}}^{inv}), y_{\hat{u}}) \tag{4}$$

where pseudo-labels $y_{\hat{u}} := \text{argmax}(f_{cl}(\mathbf{z}_u^{inv}))$, given by the classifier predictions on the unlabelled target data $x_u \in U$, are used as ground-truth for the corresponding augmentation $x_{\hat{u}}$. Only a subset of the pseudo-labels (those with higher confidence) are retained. This is expressed through the multiplying binary factor

$$m_u^\tau := \mathbb{1}(\max(f_{cl}(\mathbf{z}_u^{inv})) > \tau) \tag{5}$$

where $\tau \in [0, 1]$ is a scalar hyper-parameter denoting the confidence threshold and $\mathbb{1}(\text{condition})$ denotes the indicator function, which is equal to 1 if *condition* holds and 0 otherwise.

3.2 Training Procedure

The proposed training scheme is summarized in Fig. 2, where we show the different input data paths through the network during training as well as the

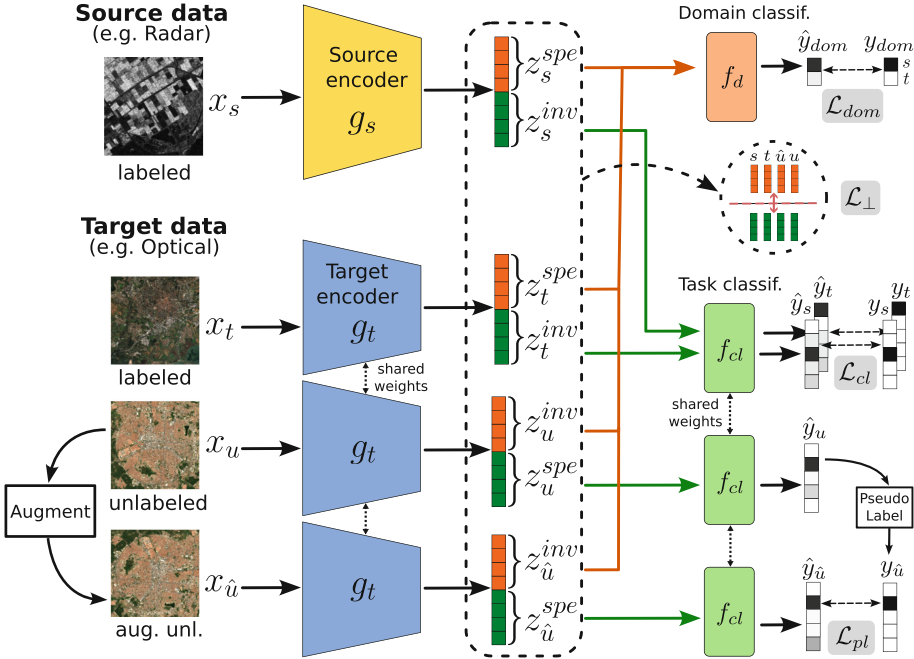


Fig. 2. Schematic view of the data flow during the training phase. The four proposed loss terms (framed in grey) are illustrated with their corresponding inputs. (Color figure online)

inputs used by each of the four proposed losses. A more detailed and formalized description of the training procedure is given in Algorithm 1.

For each epoch, we go through the source dataset sequentially (as it is usually the dataset with the highest number of samples $N_s > N_u > N_t$). This is done by batches in practice, even if in Algorithm 1 we illustrate the sample-wise case (unitary batch) for simplicity¹. At each iteration, we sample uniformly at random the same number of samples (batch size) from the set labeled and unlabeled target data —lines 4 and 5. Each sample is then passed through their matching encoder at lines 6–9 (note that the target encoder g_t is used not only for the labeled target samples \mathbf{x}_t with matching subscript, but also for the unlabeled samples \mathbf{x}_u and $\mathbf{x}_{\hat{u}}$). Finally, in lines 10–16, each of the loss terms defined in the previous section are computed with respect to the all relevant input data and, subsequently (line 17), backpropagated through the entire architecture to update its composing modules (g_s, g_t, f_{cl}, f_d) weights.

For convenience, we introduce superscripts on a loss term, say \mathcal{L}^V , to specify its application on input data coming from a certain dataset $V \in \{S, T, U, \hat{U}\}$ (or several datasets in case of multiple superscripts). For instance, we denote

¹ The generic minibatch version of Algorithm 1 is obtained simply by additionally averaging each of the loss terms over the batch dimension.

Algorithm 1. *SHeDD* Train procedure

Require: Datasets S, T, U ; Pseudo-labeling threshold τ .

- 1: **for** epoch $\in \{1, \dots, N_{ep}\}$ **do**
- 2: **for all** $(\mathbf{x}_s, y_s) \in S$ **do**
- 3: $(\mathbf{x}_t, y_t) \sim \mathcal{U}(T)$
- 4: $\mathbf{x}_u \sim \mathcal{U}(U)$
- 5: $x_{\hat{u}} = \text{Augment}(x_u)$
- 6: $\mathbf{z}_s^{inv}, \mathbf{z}_s^{spe} = g_s(x_s)$
- 7: $\mathbf{z}_t^{inv}, \mathbf{z}_t^{spe} = g_t(x_t)$
- 8: $\mathbf{z}_u^{inv}, \mathbf{z}_u^{spe} = g_t(x_u)$
- 9: $\mathbf{z}_{\hat{u}}^{inv}, \mathbf{z}_{\hat{u}}^{spe} = g_t(x_{\hat{u}})$
- 10: $\mathcal{L}_{cl}^{S,T} = \frac{1}{2} \sum_{v \in \{s,t\}} CE(f_{cl}(\mathbf{z}_v^{inv}), y_v)$
- 11: $\mathcal{L}_{dom}^{S,T} = \frac{1}{2} \sum_{v \in \{s,t\}} CE(f_d(\mathbf{z}_v^{spe}), v)$
- 12: $\mathcal{L}_{dom}^{U,\hat{U}} = \frac{1}{2} \sum_{v \in \{u,\hat{u}\}} CE(f_d(\mathbf{z}_v^{spe}), t)$
- 13: $\mathcal{L}_{\perp}^{S,T} = \frac{1}{2} \sum_{v \in \{s,t\}} \frac{\langle \mathbf{z}_v^{inv}, \mathbf{z}_v^{spe} \rangle}{\|\mathbf{z}_v^{inv}\|_2 \|\mathbf{z}_v^{spe}\|_2}$
- 14: $\mathcal{L}_{\perp}^{U,\hat{U}} = \frac{1}{2} \sum_{v \in \{u,\hat{u}\}} \frac{\langle \mathbf{z}_v^{inv}, \mathbf{z}_v^{spe} \rangle}{\|\mathbf{z}_v^{inv}\|_2 \|\mathbf{z}_v^{spe}\|_2}$
- 15: $y_{\hat{u}}, m_u^{\tau} = \text{PseudoLabel}(f_{cl}(\mathbf{z}_u^{inv}), \tau)$ // cf. equations (4) and (5)
- 16: $\mathcal{L}_{pl}^{\hat{U}} = m_u^{\tau} \cdot CE(f_{cl}(\mathbf{z}_{\hat{u}}^{inv}), y_{\hat{u}})$
- 17: Update weights of (g_s, g_t, f_{cl}, f_d) by back-propagating the loss:
 $\mathcal{L}_{cl}^{S,T} + \mathcal{L}_{dom}^{S,T} + \mathcal{L}_{dom}^{U,\hat{U}} + \mathcal{L}_{\perp}^{S,T} + \mathcal{L}_{\perp}^{U,\hat{U}} + \mathcal{L}_{pl}^{\hat{U}}$
- 18: **end for**
- 19: **end for**
- 20: **return** g_T, f_{cl}

by $\mathcal{L}_{cl}^{S,T}$ the classification loss defined in Eq. (1) applied on (and averaged over) samples from labeled source and target datasets. This notation has the merit of making more explicit to which dataset each loss applies and will prove particularly useful for our ablation analysis in Table 6.

Hence, while the classification loss \mathcal{L}_{cl} naturally applies only to labelled data (S, T), the domain classification \mathcal{L}_{dom} and orthogonality \mathcal{L}_{\perp} losses can be evaluated for both labeled (S, T) and unlabelled data (U, \hat{U}). Finally, the Fix-Match loss \mathcal{L}_{pl} applies to the augmented unsupervised data (\hat{U}) while leveraging pseudo-labels obtained for the corresponding non-augmented samples (in U). These multiple data paths involved in the proposed training scheme are summarized in Fig. 2. In the figure, we replicate the encoder and classification modules to properly outline each separate data flow, but we emphasize that these modules are characterized by an unique set of shared weights.

Data Augmentation: The employed augmentation operation $\text{Augment}(\cdot)$ (line 5 in Algorithm 1) consists of a series of possible transformations with 50% of occurrence probability each, among the following: horizontal flip; vertical flip;

rotation with random angle on the set $\{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$; color jitter (random changes in the image brightness, contrast, saturation and hue)².

4 Experiments

In order to assess the performance of *SHeDD*, we consider two different benchmarks covering heterogeneous data coming from the remote sensing field.

Table 1. Benchmark statistics and description. Each benchmark covers two heterogeneous domains. *EuroSat-MS-SAR* involves MS and SAR images, both with a spatial resolution of 10 m, for a classification task with 10 classes. *RESISC45-Euro* includes RGB and MS images, with varying spatial resolutions, for a classification task with 8 classes. The **Volume** column reports per-domain statistics in the format (# images) \times (# channels) \times (image height) \times (image width).

Benchmark	Volume	Modality	Spatial Res.	# Classes
RESISC45-Euro	5 600 \times 3 \times 256 \times 256	RGB	0.2 m–30 m	8
	24 000 \times 13 \times 64 \times 64	MS	10 m	
EuroSat-MS-SAR	27 000 \times 13 \times 64 \times 64	MS	10 m	10
	27 000 \times 2 \times 64 \times 64	SAR	10 m	

As our first dataset, we adopt the *RESISC45-Euro* benchmark previously introduced in [12]. This dataset contains 5 600 RGB images at different spatial resolutions and 24 000 multispectral (MS) images, with 13 channels, spanning eight different land cover classes. Here, the heterogeneity is related to domains covering imagery with different spatial and spectral resolutions. As our second dataset, we use the *EuroSat-MS-SAR* benchmark [20]. This dataset contains 54 000 pairs of MS and synthetic aperture radar (SAR) images, with 13 and 2 channels respectively. With the aim to avoid possible data biases and spurious correlations, for each sample of the dataset we only retain one of the two modalities. This leads to a benchmark including 27 000 MS and 27 000 SAR unaligned images over the set of ten land cover classes. Here, the heterogeneity corresponds to imagery collected via different acquisition modes (optical and radar). Details about benchmarks are reported in Table 1. For each benchmark we set up two transfer tasks where each transfer task is denoted as $(\mathcal{D}_s \rightarrow \mathcal{D}_t)$ where the right arrow indicates the transfer direction from the fully labelled source domain (\mathcal{D}_s) to the scarce labelled target domain (\mathcal{D}_t) .

Considering the competing approaches, we include in our experimental evaluation a fully supervised baseline that only exploits available target labelled data, referred as *Target Only*. As a state-of-the-art semi-supervised framework that exploits both labelled and unlabelled target samples in order to leverage

² For this transformation, we used the PyTorch implementation `torchvision.transforms.ColorJitter()` with default parameters.

the full amount of available target data, here we adopt the well-known *FixMatch* framework [18]. Finally, according to SSHDA literature, we include the *SS-HIDA* approach recently introduced in [12].

For all the competing approaches, as well as our proposed *SHeDD*, to set up a fair comparison, we adopt the same backbone architecture, ResNet-18 [4]. In the particular case of our proposed *SHeDD*, the final fully-connected layer (with softmax activation) of ResNet-18 is employed as our task classifier module and the same structure is used for the domain classifier. For *FixMatch* and *SHeDD* we fix the pseudo-labeling threshold τ to 0.95 and we use as weak augmentation the identity function and as strong augmentation a random combination of geometrical (flipping and rotation) and radiometric (color jitter) transformations. For the *SS-HIDA*, according to the original work, we used half of the backbone network as specific per-domain encoder and the rest of the backbone as shared encoder. For all the competing approaches we adopt a number of training epochs equal to 300, a batch size of 128, AdamW [11] as parameter optimizer with a learning rate of 10^{-4} . Additionally, based on recent literature [7], for all the methods we adopted an exponential moving average (EMA) of the weight parameters, with momentum equals to 0.95, since we experimentally observed that all the approaches took advantage from it.

For the experimental assessment, we set up two different transfer tasks for each benchmark, considering each of the available domains firstly as source and then as target. While for the source domain all the available data are labelled, for the target domain we varied the amount of available supervision, ranging in the set $\{25, 50, 100, 200\}$ samples per class. This means that, for instance, if the supervision value is equal to 25, then 25 labelled samples are accessible per class for the target domain. The rest of the target samples constitute the test set, which is also assumed to be available at training time as additional unlabelled target data. The assessment of the models performance, on the test set, is done considering the weighted F1-Score, subsequently referred simply as F1-Score. We repeat each experiment five times and report average and standard deviation results.

All the methods are implemented in Pytorch and available [here](#). Experiments are carried out on a workstation equipped with an Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz, with 377Gb of RAM and four RTX3090 GPU. All the methods require only one GPU for training.

4.1 Results

Tables 2, 3, 4 and 5 report the results of all the competing methods, in terms of F1-Score, varying the amount of labelled target sample in the set $\{25, 50, 100, 200\}$ for the *RESISC45-Euro* and *EuroSat-MS-SAR* benchmarks, respectively.

Concerning the *RESISC45-Euro* benchmark, we evaluate two transfer tasks: (RGB \rightarrow MS) and (MS \rightarrow RGB). Here, the two domains differ in terms of radiometric content (imagery with 3 or 13 channels) and spatial resolution as outlined in Table 1. For the first transfer task (RGB \rightarrow MS) the results are

presented in Table 2. Notably, *SHeDD* systematically outperforms all the competing approaches. Although *SS-HIDA* also exhibits improvements over baseline approaches, it achieves lower performances compared to our method.

In the second transfer task (MS \rightarrow RGB), as illustrated in Table 3, our method continues to outperform competing approaches in the majority of cases, exception made for the case with 200 labelled target samples per class where our proposed approach, nonetheless, still achieves comparable performance to *SS-HIDA*. Generally, the use of the source data clearly enables our framework to achieve a gain of over 2 points in terms of F1-Score, regardless of the amount of target labelled samples, compared to strategies relying solely on target information (*Target Only* and *FixMatch*).

Table 2. Average and standard deviation F1-Score results, over 5 runs, on *RESISC45-Euro* with RGB as source and MS as target domain (RGB \rightarrow MS) varying the amount of per-class target supervision in the range {25, 50, 100, 200}.

Method	25	50	100	200
Target Only	79.48 \pm 1.34	85.05 \pm 1.01	88.99 \pm 0.77	92.34 \pm 0.52
FixMatch	81.74 \pm 1.38	85.60 \pm 0.37	89.37 \pm 0.55	92.57 \pm 0.79
SS-HIDA	82.29 \pm 0.68	88.81 \pm 0.95	91.64 \pm 1.67	93.59 \pm 1.39
SHeDD	84.06 \pm 0.73	89.12 \pm 0.84	92.84 \pm 0.18	95.29 \pm 0.38

Table 3. Average and standard deviation F1-Score results, over 5 runs, on *RESISC45-Euro* with MS as source and RGB as target domain (MS \rightarrow RGB) varying the amount of per-class target supervision in the range {25, 50, 100, 200}.

Method	25	50	100	200
Target Only	75.19 \pm 1.67	82.52 \pm 0.91	87.45 \pm 0.82	91.74 \pm 0.65
FixMatch	77.17 \pm 1.29	82.80 \pm 0.81	87.86 \pm 0.56	91.93 \pm 0.53
SS-HIDA	79.78 \pm 1.06	85.00 \pm 1.07	89.56 \pm 2.34	93.83 \pm 0.18
SHeDD	81.72 \pm 1.93	86.65 \pm 0.82	91.00 \pm 0.55	93.79 \pm 0.32

Regarding the *EuroSat-MS-SAR* benchmark, we consider the transfer tasks: (MS \rightarrow SAR) and (SAR \rightarrow MS). Here, the two domains differ in terms of acquisition modes (Optical vs. Radar), thus providing a more challenging transfer scenario in term of source/target domain heterogeneity. The results for the first transfer task (MS \rightarrow SAR) are reported in Table 4 while the results for the second transfer task (SAR \rightarrow MS) are outlined in Table 5. Irrespective of the amount of labeled samples in the target domain, *SHeDD* consistently outperforms all the competing approaches by a notable margin. Differences are generally more

Table 4. Average and standard deviation F1-Score results, over 5 runs, on *EuroSat-MS-SAR* with MS as source and SAR as target domain (MS \rightarrow SAR) varying the amount of per-class target supervision in the range {25, 50, 100, 200}.

Method	25	50	100	200
Target Only	60.08 \pm 1.25	62.52 \pm 0.38	64.93 \pm 0.31	67.80 \pm 0.65
FixMatch	59.07 \pm 1.29	64.45 \pm 0.35	67.26 \pm 0.95	70.38 \pm 0.59
SS-HIDA	60.24 \pm 1.65	62.96 \pm 0.86	66.63 \pm 1.00	70.40 \pm 0.87
SHeDD	63.66 \pm 1.53	67.91 \pm 1.83	70.64 \pm 1.50	73.97 \pm 0.67

Table 5. Average and standard deviation F1-Score results, over 5 runs, on *EuroSat-MS-SAR* with SAR as source and MS as target domain (SAR \rightarrow MS) varying the amount of per-class target supervision in the range {25, 50, 100, 200}.

Method	25	50	100	200
Target Only	75.85 \pm 0.28	82.94 \pm 0.45	87.08 \pm 0.83	90.92 \pm 0.23
FixMatch	76.87 \pm 1.32	83.25 \pm 0.65	87.67 \pm 0.57	91.74 \pm 0.39
SS-HIDA	76.49 \pm 0.81	80.52 \pm 1.49	85.33 \pm 0.73	89.38 \pm 0.52
	82.30 \pm 1.12	88.16 \pm 0.85	91.67 \pm 0.23	94.52 \pm 0.14

pronounced for low amount of target labelled samples. For instance, when only 25 target labeled samples per-class are considered for the transfer task (SAR \rightarrow MS), *SHeDD* demonstrates nearly a 6-point increase in F1-Score over the second-best competitor.

It is worth noting that, differently from the case of *RESISC45-Euro* benchmark, here *SS-HIDA* only performs on-par with the baseline methods (*Target Only* and *FixMatch*). This point can be partly related to the architectural structure of *SS-HIDA*. While *SHeDD* employs distinct per-domain encoders, *SS-HIDA* shared a portion of its encoder between the two domains.

If on the one hand this architectural choice can prove advantageous in scenarios where domains exhibit a limited degree of heterogeneity (e.g. transferring between RGB and MS data, where one modality can be considered as a subset or superset of the other), on the other hand it may hinder transfer performance in more challenging scenarios characterized by a high degree of heterogeneity, as for the *EuroSat-MS-SAR* benchmark. Consequently, it may fail to establish an effective strategy for general heterogeneous domain adaptation. This result further supports the flexibility of our method in modeling a wide range of heterogeneous data transfer scenarios owing to its inherent structural design.

Ablation Analysis: Table 6 reports the ablation analysis of *SHeDD* on the *EuroSat-MS-SAR* benchmark where MS images serve as source domain and SAR images as target domain. Here we consider the case in which 50 labelled samples per class are available from the target domain. Six different ablations were

devised from the complete model to comprehensively assess the various components upon which *SHeDD* relies. Firstly, we observe a clear positive impact of enforcing disentanglement between domain-invariant and domain-specific features ($L_{\perp}^{S,T}$ and $L_{dom}^{S,T}$) over the scenario where only the supervised classification loss is optimized (*Abla*₁ vs *Abla*₂). Secondly, we can underline that the use of unlabelled target data, through the $L_{\perp}^{U,\hat{U}}$, $L_{dom}^{U,\hat{U}}$ and $L_{pl}^{\hat{U}}$ losses, systematically enhances the performances compared to using the labelled information alone (*Abla*₁, *Abla*₂ vs *Abla*₃, *Abla*₄, *Abla*₅ and *Abla*₆). Thirdly, the highest performances are generally attained when consistency regularization, through pseudo-labelling, is considered (*Abla*₄, *Abla*₅ and *Abla*₆). Fourthly, when either $L_{\perp}^{U,\hat{U}}$ and $L_{dom}^{U,\hat{U}}$ or $L_{pl}^{\hat{U}}$ are employed separately (*Abla*₃ and *Abla*₄), performances are still far from the ones achieved by the whole framework. This indicates that the combined use of these three losses, to leverage unlabelled target data, synergistically enhances the final outcome. Finally, the performed ablations indicate that *SHeDD* clearly benefits from all the components it is built on, thus exhibiting the best performance overall in terms of F1-Score.

Table 6. Ablation study of *SHeDD* on the *EuroSat-MS-SAR* benchmark with MS as source and SAR as target domain when 50 samples per class are considered as labelled target data. F1-Score results, in terms of mean and standard deviation over 5 runs, are reported.

Ablation	$L_{clf}^{S,T}$	$L_{\perp}^{S,T}$	$L_{dom}^{S,T}$	$L_{\perp}^{U,\hat{U}}$	$L_{dom}^{U,\hat{U}}$	$L_{pl}^{\hat{U}}$	F1-score
<i>Abla</i> ₁	✓						63.84 ± 0.34
<i>Abla</i> ₂	✓	✓	✓				64.58 ± 0.85
<i>Abla</i> ₃	✓	✓	✓	✓	✓		65.04 ± 0.74
<i>Abla</i> ₄	✓	✓	✓			✓	66.00 ± 0.87
<i>Abla</i> ₅	✓		✓		✓	✓	66.54 ± 0.77
<i>Abla</i> ₆	✓	✓		✓		✓	67.47 ± 1.63
<i>SHeDD</i>	✓	✓	✓	✓	✓	✓	67.91 ± 1.83

Visual Inspection of Learnt Representations: Figure 3 visually depicts the internal representation learnt by the different competing methods on the *RESISC45-Euro* benchmark for the transfer task (RGB → MS) when only 25 labelled samples per class for the target domain are considered. To this end, we randomly chose 50 samples per class on the target domain and we extracted the corresponding feature representation per method, that is, the embedding vector used as input to the classifier module—in the case our proposed approach, notably, the domain-invariant embeddings \mathbf{z}^{inv} are used. Subsequently, we applied t-SNE [19] to reduce the feature dimensionality for visualisation purposes.

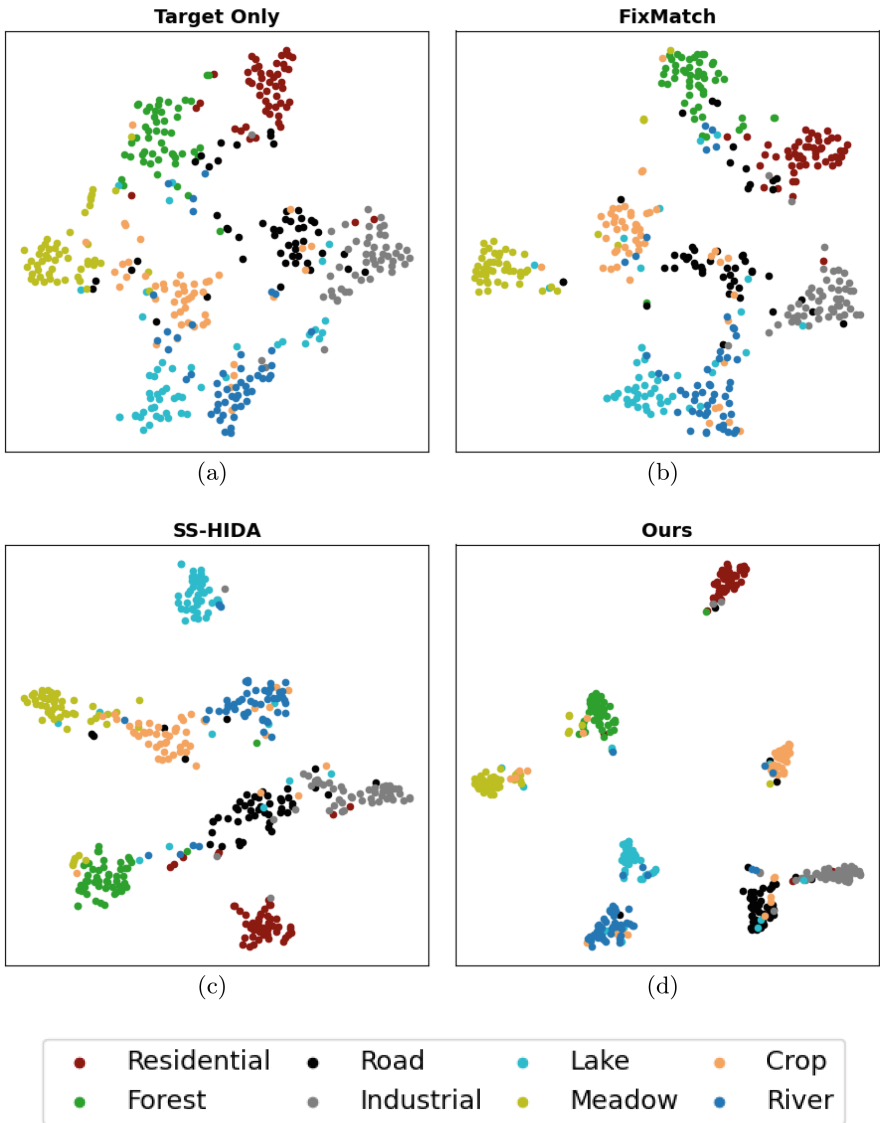


Fig. 3. Visualization of the embeddings extracted from the different competing approaches: (a) *Target Only* (b) *FixMatch* (c) *SS-HIDA* and (d) *SHeDD* when trained on the *RESISC45-Euro* benchmark with RGB as source and MS as target domain (RGB \rightarrow MS) and only 25 labelled samples per class are considered for the target domain. For this visual inspection, 50 random samples per class from the test set (coming from the target domain) are sampled. The two dimensional representation is obtained via the T-SNE algorithm [19].

When only a limited amount of labelled target data is employed to learn the underlying classification models, as for *Target Only* and *FixMatch* methods, the 2D spatial arrangement of the generated embeddings demonstrates evident visual cluttering, with samples coming from different classes overlapping. Although this phenomenon is partially alleviated on *SS-HIDA* embeddings, the resulting manifold still struggles to accurately recover the underlying eight-cluster structure. In contrast, *SHeDD* produces embeddings that depict a more distinct class-aware manifold, visually aligning better with the underlying data distribution compared to competing approaches.

Overall, the visualisation of internal features representation confirms the quantitative findings we previously discussed.

5 Conclusions and Perspectives

In this paper we have presented *SHeDD*, a deep learning based framework to cope with the challenging scenario of semi-supervised domain adaptation when source and target data are heterogeneous in terms of modality representation. Our end-to-end framework has the objective to learn a target domain classifier by leveraging labelled and unlabelled data from both source and target domain via consistency regularized pseudo-labelling and disentanglement learning. While the former mechanism allows to fully leverage the available unlabelled data, the latter allows to simultaneously extract domain-invariant representations, relevant for the downstream task, while retrieving domain-specific information, that can hinder the cross-modality transfer.

The evaluation on two real-world benchmarks, spanning different degrees of source/target domain heterogeneity, has demonstrated the effectiveness of *SHeDD* compared to baselines and recent competing approaches.

While the proposed experimental evaluation clearly demonstrates the effectiveness of *SHeDD* on challenging remote sensing benchmarks, further assessment on general computer vision tasks involving heterogeneous data sources, such as RGB/Depth, RGB/Thermal, or RGB/LIDAR data, still represents a concrete opportunity. Additional evaluations on these benchmarks could further emphasize the value of *SHeDD* in the broader field of computer vision.

In the short term, we aim to enhance the quality of *SHeDD* by drawing inspiration from recent semi-supervised learning strategies, such as *FlexMatch*, and by exploring the impact of various augmentation techniques on consistency regularization and pseudo-labeling to improve the model's performance in data-scarce environments. In the medium term, we plan to extend our framework towards a multi-source domain adaptation setting, enabling the use of multiple heterogeneous domains as source data. This could lead to a more robust classifier and potentially improved performance on the target domain. Additionally, further exploration could involve adapting the proposed framework to more structured classification tasks, such as semantic segmentation or object recognition, where data spanning heterogeneous modalities are abundant.

Acknowledgment. This work was supported by the French National Research Agency under the grant ANR-23-IAS1-0002 (ANR GEO ReSeT).

References

1. Day, O., Khoshgoftaar, T.M.: A survey on heterogeneous transfer learning. *J. Big Data* **4**, 29 (2017)
2. Fang, Z., Jie, L., Liu, F., Zhang, G.: Semi-supervised heterogeneous domain adaptation: Theory and algorithms. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**(1), 1087–1105 (2023)
3. Guan, H., Liu, M.: Domain adaptation for medical image analysis: a survey. *IEEE Trans. Biomed. Eng.* **69**(3), 1173–1185 (2022)
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27–30, 2016, pp. 770–778. IEEE Computer Society, 2016
5. Jiang, P., Wu, A., Han, Y., Shao, Y., Qi, M., Li, B.: Bidirectional adversarial training for semi-supervised domain adaptation. In: IJCAI, pp. 934–940 (2020)
6. Kim, T., Kim, C.: Attract, perturb, and explore: learning a feature alignment network for semi-supervised domain adaptation. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12359, pp. 591–607. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58568-6_35
7. Lakkapragada, A., Sleiman, E., Surabhi, S., Wall, D.P.: Mitigating negative transfer in multi-task learning with exponential moving average loss weighting strategies (student abstract). In: AAAI, pp. 16246–16247 (2023)
8. Li, J., Li, G., Shi, Y., Yu, Y.: Cross-domain adaptive clustering for semi-supervised domain adaptation. In: CVPR, pp. 2505–2514 (2021)
9. Li, J., Li, G., Yizhou, Yu.: Inter-domain mixup for semi-supervised domain adaptation. *Pattern Recognit.* **146**, 110023 (2024)
10. Liu, Y., Zhou, Z., Sun, B.: COT: unsupervised domain adaptation with clustering and optimal transport. In: CVPR, pp. 19998–20007. IEEE (2023)
11. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: ICLR. OpenReview.net (2019)
12. Obrenovic, M., Lampert, T.A., Ivanovic, M.R., Gançarski, P.: Learning domain invariant representations of heterogeneous image data. *Mach. Learn.* **112**(10), 3659–3684 (2023)
13. Peng, J., Huang, Y., Sun, W., Chen, N., Ning, Y., Du, Q.: Domain adaptation in remote sensing image classification: a survey. *IEEE J. Sel. Top. Appl. Earth Obs. Remote. Sens.* **15**, 9842–9859 (2022)
14. Qin, C., Wang, L., Ma, Q., Yin, Y., Wang, H., Fu, Y.: Contradictory structure learning for semi-supervised domain adaptation. In: SDM, pp. 576–584 (2021)
15. Qin, C., Wang, L., Qianqian Ma, Yu., Yin, H.W., Yun, F.: Semi-supervised domain adaptive structure learning. *IEEE Trans. Image Process.* **31**, 7179–7190 (2022)
16. Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. In *ICCV*, pages 8049–8057. IEEE, 2019
17. Singh, A.: CLDA: contrastive learning for semi-supervised domain adaptation. In: Beygelzimer, A., Dauphin, Y., Liang, P., Wortman Vaughan, J. (eds.) *Advances in Neural Information Processing Systems* (2021)
18. Sohn, K., et al.: Simplifying semi-supervised learning with consistency and confidence. In: *NeurIPS, Fixmatch* (2020)

19. van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**, 2579–2605 (2008)
20. Wang, Y., Hernández Hernández, H., Albrecht, C.M., Zhu, X.X.: Feature guided masked autoencoder for self-supervised learning in remote sensing. *CoRR*, abs/2310.18653 (2023)
21. Wilson, G., Cook, D.J.: A survey of unsupervised deep domain adaptation. *ACM Trans. Intell. Syst. Technol.* **11**(5), 51:1–51:46 (2020)
22. Yan, Z., Wu, Y., Li, G., Qin, Y., Han, X., Cui, S.: Multi-level consistency learning for semi-supervised domain adaptation. In: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, (IJCAI-22)*, pp. 1530–1536, July 2022
23. Yang, L., et al.: Deep co-training with task decomposition for semi-supervised domain adaptation. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8906–8916 (2021)
24. Yao, T., Pan, Y., Ngo, C.-W., Li, H., Mei, T.: Semi-supervised domain adaptation with subspace learning for visual recognition. In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2142–2150 (2015)
25. Yao, Y., Zhang, Y., Li, X., Ye, Y.: Heterogeneous domain adaptation via soft transfer network. In: *Multimedia*, pp. 1578–1586. ACM (2019)
26. Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., He, Q.: A comprehensive survey on transfer learning. *Proc. IEEE* **109**(1), 43–76 (2021)

Author Index

A

Ali, Sahara 213

B

Banerjee, Sourasekhar 58

Beer, Anna 421

Bo, Hongbo 143

C

Cao, Yushi 21

Chakraborty, Shayok 178

Chen, Aiguo 386

Chen, Haiming 351

Chen, Hao 368

Chen, Lei 250

D

d'Alché-Buc, Florence 93

Dadras, Ali 58

Dai, Shengran 335

Dantas, Cassio F. 440

de Campos, Cassio 161

Du, Xinyang 368

E

El Ahmad, Tamim 93

En, Qing 267

F

Fang, Xiu Susie 368

Faruque, Omar 213

Feng, Shanshan 284

G

Gaetano, Raffaele 440

Gao, Hang 351

Gao, Shihong 250

Gao, Zhangyang 3

Ghandi, Soroush 161

Gilhuber, Sandra 421

Gu, Xizhi 250

Guo, Ruifeng 3

Guo, Yuhong 267

H

Hachiya, Hirotake 301

Han, Jiashu 284

He, Tieke 76

Henriksson, Aron 197

Hong, Jun 143

Hooi, Bryan 21

Hu, Hengchang 21

Huang, Chao 21

Huang, Hong 351

Huang, Xiaowei 231

I

Ienco, Dino 440

J

Jiang, Chutian 178

Jiang, Jianhui 335

Jiang, Wei 403

Jin, Shi 231

K

Kragic, Danica 111

Kravberg, Alexander 111

L

Laforgue, Pierre 93

Li, Hongzheng 250

Li, Jianguo 403

Li, Jiayi 317

Li, Stan 3

Li, Wei 386

Li, Xiaosheng 403

Li, Xutao 284

Li, Ying 403

Li, Yuhui 39

Liang, Yuxuan 21
 Liu, Jiayu 231
 Liu, Weiru 143
 Liu, Xu 21
 Liu, Yunhui 76
 Luo, Guangchun 386
 Luo, Ruilin 317

M

Ma, Yunpu 421
 Marchetti, Giovanni Luca 111
 Medbouhi, Aniss Aiman 111
 Mu, Kedian 143

O

Ong, Yew Soon 284

P

Poklukar, Petra 111
 Polianskii, Vladislav 111
 Prakhya, Karthik 58

Q

Quost, Benjamin 161

S

Seidl, Thomas 421
 Shao, Yingxia 250
 Sun, Guohao 368
 Sun, Jiaqi 317
 Sun, Linzhuang 3

T

Tan, Cheng 3

V

Varava, Anastasia 111

W

Wang, Haitao 127
 Wang, Jianwu 213

Wang, Lingfu 386
 Wang, Pengxiang 143
 Wang, Siyu 335
 Wei, Jingxuan 3
 Wei, Ziqi 368
 Wu, Fengge 351
 Wu, Hejun 127
 Wu, Sihao 231
 Wu, Yifan 403
 Wu, Yongchao 197
 Wu, Zejia 39

X

Xiao, Jing 317
 Xiong, Zuobin 386

Y

Yamazono, Rikuto 301
 Yang, Junjie 93
 Yang, Yujiu 317
 Yao, Chengyu 351
 Yi, Xinpeng 231
 Yin, Xiangyu 231
 Yu, Bihui 3
 Yurtsever, Alp 58

Z

Zhan, Yong 368
 Zhang, Chao 39
 Zhang, Hongyang 39
 Zhang, Huaisong 76
 Zhang, Tianle 231
 Zhang, Xiaonan 178
 Zhang, Xinyan 250
 Zhang, Xinyu 284
 Zhao, Jianhua 76
 Zhao, Junsuo 351
 Zheng, Tao 76
 Zhou, Hansong 178
 Zhou, Min 284
 Zimmermann, Roger 21